Автономная некоммерческая профессиональная образовательная организация
«МЕЖДУНАРОДНЫЙ ВОСТОЧНО-ЕВРОПЕЙСКИЙ КОЛЛЕДЖ»
Пушкинская ул., д. 268, 426008, г. Ижевск. Тел.: (3412) 77-68-24. E-mail: mveu@mveu.ru, www. mveu.ru
ИНН 1831200089. ОГРН 1201800020641

20.02.2026 г.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

по выполнению практических работ

при изучении профессионального модуля

ПМ.03 ОБУЧЕНИЕ ГОТОВЫХ МОДЕЛЕЙ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

по специальности

09.02.13 Интеграция решений с применением технологий искусственного интеллекта

2026

Практическая работа – небольшой научный отчет, обобщающий проведенную учащимся работу, которую представляют для защиты преподавателю.

В процессе практического занятия учащиеся выполняют одну или несколько практических работ (заданий) под руководством преподавателя в соответствии с изучаемым содержанием учебного материала.

Состав и содержание практических занятий направлены на реализацию Государственных требований.

Наряду с формированием умений и навыков в процессе практических занятий обобщаются, систематизируются, углубляются и конкретизируются теоретические знания, вырабатывается способность и готовность использовать теоретические знания на практике, развиваются интеллектуальные умения.

Практические занятия проводятся в форме практической подготовки в виде работ, связанных с будущей профессиональной деятельностью.

К практическим работам предъявляется ряд требований, основным из которых является полное, исчерпывающее описание всей проделанной работы, позволяющее судить о полученных результатах, степени выполнения заданий и профессиональной подготовке учащихся.

I. Практические работы:

МДК 03.01 Разработка сценариев обучения готовых моделей

Тема практической работы №1. Исследование простых моделей искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять полный цикл работы с простой моделью ИИ: подготовка данных, обучение модели, оценка качества, визуализация результатов и сохранение артефактов.

Задание(я):

1. Установить необходимые библиотеки (scikit-learn, pandas, matplotlib, joblib).

2. Загрузить открытый датасет Iris, выполнить предварительный анализ (просмотр первых строк, статистика).

3. Разделить данные на обучающую и тестовую выборки, обучить классификатор k-Nearest Neighbors.

4. Оценить модель на тестовой выборке (точность, матрица ошибок), сохранить метрики в CSV-файл.

5. Построить график зависимости точности от значения параметра k, сохранить график в PNG-файл.

6. Сохранить обученную модель в файл с помощью joblib, подготовить скрипт-файл (main.py) с выполнением всех шагов.

7. Оформить отчёт, включив титульный лист, описание выполненных шагов, скриншоты вывода, ссылки на сохранённые файлы.

Методические указания по ходу выполнения работы:

Для установки библиотек используйте команду pip install scikit-learn pandas matplotlib joblib.

Данные Iris загружайте через функцию загрузки из scikit-learn, сохраняйте исходный CSV в папку data.

Разделение выборок делайте функцией train_test_split, сохраняйте индексы в отдельный CSV-файл.

Метрики (accuracy, confusion matrix) экспортируйте в файл metrics.csv в папку results.

График сохраняйте командой plt.savefig('accuracy_vs_k.png') в папку results.

Модель сохраняйте командой joblib.dump(model, 'knn_model.joblib') в папку models.

Все скрипты (main.py) и вспомогательные файлы размещайте в корневой папке проекта.

Отчёт оформляйте в ODT или PDF, включайте титульный лист, список заданий, описание выполнения, скриншоты терминала и графика, выводы.

Для сдачи упакуйте всю структуру проекта (папки data, results, models, main.py, отчёт) в ZIP-архив.

Отчёт: ODT или PDF, титульный лист, краткое описание каждого задания, скриншоты вывода консоли и графика, таблица метрик из CSV, выводы о качестве модели.

Сдача: ZIP-архив с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №2. Создание простого алгоритма машинного обучения, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного выполнения полного цикла разработки простого алгоритма машинного обучения: загрузка данных, предобработка, обучение модели, оценка качества и сохранение артефактов.

Задание(я):

1. Подготовка окружения и загрузка датасета Iris.

2. Предобработка данных: разделение на обучающую и тестовую выборки, масштабирование признаков.

3. Обучение модели классификации (логистическая регрессия) и сохранение обученного объекта.

4. Оценка качества модели: построение матрицы ошибок и ROC-кривой, сохранение графиков в файлы.

5. Формирование отчёта и упаковка всех артефактов в архив.

Методические указания по ходу выполнения работы:

В задании 1 установите Python и необходимые библиотеки (numpy, pandas, scikit-learn, matplotlib, joblib) с помощью pip install, создайте рабочую папку проекта и сохраните скрипт загрузки данных в файл load_data.py.

В задании 2 в скрипте preprocessing.py выполните разделение датасета на train и test (соотношение 80/20) и примените стандартизацию признаков; сохраняйте полученные массивы в файлы train.npy, test.npy и scaler.joblib.

В задании 3 в скрипте train_model.py обучите модель логистической регрессии на подготовленных данных, оцените её на тестовой выборке и сохраните обученный объект в файл model.joblib.

В задании 4 в скрипте evaluate.py постройте матрицу ошибок и ROC-кривую, отобразите их с помощью matplotlib и сохраните графики как confusion_matrix.png и roc_curve.png; также запишите метрики (accuracy, precision, recall, f1) в файл metrics.csv.

В задании 5 подготовьте отчёт в формате ODT или PDF, включив титульный лист, краткое описание каждого задания, скриншоты кода, результаты графиков и таблицу метрик; упакуйте в ZIP архив папку проекта, содержащую все .py файлы, сохранённые артефакты и отчёт.

Формат сдачи: один ZIP-архив, содержащий папку с кодом, данными, графиками, моделью и отчётом.

Отчёт должен включать: титульный лист (название работы, ФИО, группа, дата), список заданий с описанием выполненных действий, скриншоты кода (по одному для каждого скрипта), результаты графиков (вставленные изображения confusion_matrix.png и roc_curve.png), таблицу метрик из metrics.csv и выводы о качестве модели. Формат отчёта – ODT или PDF.

Сдача: отправить подготовленный ZIP-архив по электронной почте преподавателю не позднее конца текущего занятия.

Тема практической работы №3. Сравнение моделей искусственного интеллекта на основе готовых решений, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки сравнения готовых моделей искусственного интеллекта на примере открытого датасета, включая загрузку моделей, оценку их качества и визуализацию результатов.

Задание(я):

1. Подготовка окружения: установить Python и необходимые библиотеки (scikit-learn, pandas, matplotlib, joblib). Сохранить список установленных пакетов в файл requirements.txt.

2. Загрузка датасета Iris, сохранение его в CSV-файл и подготовка данных (разделение на признаки и метку, разбиение на обучающую и тестовую выборки).

3. Загрузка четырёх готовых моделей из scikit-learn (Logistic Regression, Decision Tree, Random Forest, Support Vector Machine) без их обучения с нуля, а используя их предобученные гиперпараметры. Сохранить каждую модель в отдельный файл с помощью joblib.

4. Оценка моделей на тестовой выборке: вычислить метрики точности, полноты, F1-score для каждой модели. Сохранить результаты в таблицу CSV.

5. Построение сравнительного графика метрик (bar-chart) для всех моделей. Сохранить график в файл PNG.

6. Подготовка отчёта: оформить титульный лист, описание выполненных шагов, таблицу метрик, скриншот кода и графика, выводы о сравнении моделей.

7. Сбор всех файлов (скрипты .py, requirements.txt, CSV-данные, сохранённые модели, PNG-график, отчёт) в один архив ZIP.

Методические указания по ходу выполнения работы:

В задании 1 используйте pip install для установки пакетов; после установки выполните pip freeze > requirements.txt и сохраните файл в рабочую папку.

В задании 2 загрузите датасет Iris через библиотеку sklearn.datasets, экспортируйте его в CSV (pd.DataFrame.to_csv) и разместите файл в подпапке data.

В задании 3 импортируйте классы моделей из sklearn, создайте их с параметрами по умолчанию, выполните fit на обучающей части, затем сохраните каждую модель функцией joblib.dump в подпапку models.

В задании 4 примените модели к тестовой части, рассчитайте метрики с помощью sklearn.metrics, сформируйте таблицу pandas и экспортируйте её в CSV в подпапку results.

В задании 5 постройте столбчатый график с метриками (accuracy, precision, recall, f1) для всех моделей, сохраните рисунок функцией plt.savefig('comparison.png') в подпапку results.

В задании 6 оформите отчёт в формате ODT или PDF: титульный лист, цель работы, описание каждого задания, таблицу результатов, скриншоты кода (файлы .py) и графика, выводы о лучшей модели.

В задании 7 создайте архив ZIP, включающий папки data, models, results, scripts (файлы .py), requirements.txt и готовый отчёт.

Формат сдачи: один архив ZIP, именуемый <фамилия_имя>.zip, отправить по e-mail преподавателю до конца занятия.

Отчёт: ODT или PDF, структура: титульный лист (название работы, ФИО, группа), цель работы, последовательность выполненных заданий с описанием действий, таблица метрик (CSV-файл вставить как изображение), скриншоты кода (каждый скрипт – отдельный скриншот), график сравнения (PNG), выводы и рекомендации.

Сдача: архив ZIP, содержащий все исходные файлы, данные, модели, графики и отчёт, отправить по e-mail преподавателю до конца занятия.

Тема практической работы №4. Анализ результатов работы простого алгоритма искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять полный цикл анализа результатов простого алгоритма машинного обучения: подготовка данных, обучение модели, оценка качества и визуализация метрик.

Задание(я):

1. Подготовить данные (загрузить датасет Iris, выполнить базовый анализ, сохранить предварительный набор в CSV).

2. Обучить простую модель (Logistic Regression) на подготовленных данных, сохранить обученную модель в файл.

3. Оценить качество модели (точность, метрики классификации), построить и сохранить график матрицы ошибок.

4. Оформить результаты: собрать скрипт, CSV-данные, модель, график и текстовый файл с метриками в один архив.

Методические указания по ходу выполнения работы:

Для всех заданий используйте язык Python и библиотеки scikit-learn, pandas, matplotlib, joblib (установите через pip).

Задание 1: создайте скрипт, который загружает датасет Iris из scikit-learn, выводит первые пять строк и сохраняет весь набор в файл data.csv.

Задание 2: в том же или отдельном скрипте разделите данные на обучающую и тестовую выборки, обучите Logistic Regression, сохраните модель командой joblib.dump в файл model.joblib.

Задание 3: с помощью обученной модели предскажите метки тестовой выборки, вычислите точность и классификационный отчёт, запишите их в

файл metrics.txt; постройте матрицу ошибок, сохраните график как confusion.png (используйте plt.savefig).

Задание 4: поместите все созданные файлы (data.csv, model.joblib, metrics.txt, confusion.png, скрипты .py) в один каталог, упакуйте его в архив ZIP.

Формат сдачи: архив ZIP, содержащий код, данные и отчёт в формате ODT или PDF.

Отчёт ODT или PDF: титульный лист, краткое описание каждого задания, скриншоты вывода (таблица первых строк, метрики, график confusion matrix), список файлов в архиве, выводы о качестве модели.

Сдача: отправить готовый архив ZIP по email преподавателю до конца занятия.

Тема практической работы №5. Эксперимент с настройками модели искусственного интеллекта для решения задачи, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно настраивать гиперпараметры модели машинного обучения, оценивать её качество и фиксировать результаты экспериментов.

Задание(я):

1. Подготовить данные: загрузить датасет Iris, разделить на обучающую и тестовую выборки, сохранить полученные CSV-файлы.

2. Обучить базовую модель (LogisticRegression) с параметрами по умолчанию, сохранить обученную модель в файл, вывести метрики точности на тесте.

3. Провести серию экспериментов, меняя один гиперпараметр (например, C, solver, max_iter). Для каждого варианта обучить модель,

оценить метрику, построить график зависимости точности от значения гиперпараметра, сохранить график в PNG.

4. Сохранить все модели и графики, собрать исходный код в отдельные .py файлы.

5. Подготовить отчёт, включив титульный лист, описание каждого задания, скриншоты вывода, сохранённые графики и выводы.

Методические указания по ходу выполнения работы:

Для задания 1 используйте библиотеку pandas и функцию train_test_split из sklearn.model_selection; экспортируйте наборы в CSV (train.csv, test.csv).

В задании 2 обучите модель, сохраните её через joblib.dump (model_base.joblib) и запишите точность в текстовый файл (metrics.txt).

В задании 3 создайте цикл по выбранным значениям гиперпараметра, для каждого сохраняйте модель (model_<value>.joblib), метрику (metrics_<value>.txt) и график (plot_<value>.png). График сохраняйте командой plt.savefig.

Все .py скрипты разместите в отдельной папке src, CSV-файлы – в data, модели – в models, графики – в plots.

Отчёт оформите в ODT или PDF, включив: титульный лист, список заданий, описание проведённых экспериментов, таблицу с результатами, скриншоты вывода терминала, изображения графиков, выводы.

Формат сдачи: один ZIP-архив, содержащий папки data, models, plots, src и файл отчёт (ODT или PDF).

Отчёт: ODT или PDF, титульный лист, описание заданий, таблица результатов, скриншоты консоли, изображения графиков, выводы.

Сдать ZIP-архив с указанными папками и отчётом по e-mail преподавателя не позднее конца занятия.

Тема практической работы №6. Написание отчета по базовым алгоритмам искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного выполнения полного цикла работы с базовыми алгоритмами искусственного интеллекта: от загрузки данных и обучения моделей до их оценки, визуализации результатов и оформления отчёта.

Задание(я):

1. Подготовка рабочего окружения. Установить Python, создать виртуальное окружение, установить пакеты: pip install scikit-learn matplotlib pandas joblib.

2. Загрузка и подготовка датасета Iris. С помощью sklearn.datasets загрузить датасет, сохранить исходные данные в CSV-файл.

3. Обучение трёх базовых моделей. На обучающей части датасета построить DecisionTree, KNeighbors и LogisticRegression, сохранить обученные модели в файлы при помощи joblib.

4. Оценка моделей и визуализация. Рассчитать метрики точности, построить графики зависимости точности от гиперпараметра (например, глубина дерева, число соседей) и сохранить их в PNG-файлы, экспортировать метрики в CSV-файл.

5. Подготовка отчёта. Оформить документ ODT/PDF с титульным листом, описанием выполненных шагов, скриншотами кода, таблицами метрик и вставленными графиками, сделать выводы о сравнение моделей.

Методические указания по ходу выполнения работы:

В задании 1 создайте отдельную папку проекта, внутри неё подпапки: data, models, plots, report. Сохраните весь написанный код в файлы с расширением .py в корневой папке.

В задании 2 используйте функцию load_iris из sklearn.datasets, преобразуйте данные в DataFrame pandas и сохраните файл data/iris.csv.

В задании 3 для каждой модели выполните обучение на обучающем наборе, затем сохраните объект модели в файл models/<имя_модели>.joblib.

В задании 4 рассчитайте метрику accuracy на тестовом наборе, сформируйте таблицу с результатами и сохраните её как data/metrics.csv. Постройте графики (например, accuracy vs. параметр) и сохраните их в папку plots с понятными именами файлов.

В задании 5 сформируйте отчёт, включив в него: титульный лист, цель работы, описание каждого задания, скриншоты кода (можно сделать снимки экрана), таблицу метрик, графики из папки plots, выводы. Отчёт сохраните как report/report.odt (или .pdf).

Формат сдачи: один ZIP-архив, содержащий все папки проекта (data, models, plots, report) и файл отчёта. Архив назовите <фамилия_имя>_PW6.zip.

Отчёт оформляется в ODT или PDF. Структура: титульный лист, цель работы, описание выполненных заданий, скриншоты кода, таблица метрик (CSV), графики (PNG), выводы и рекомендации.

Сдача: упаковать все материалы в ZIP-архив и отправить по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №7. Импорт и очистка данных для обучения модели, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного импорта данных, базовой их очистки и подготовки к обучению модели искусственного интеллекта.

Задание(я):

1. Скачать открытый датасет Iris в формате CSV и разместить его в рабочей папке проекта.

2. Импортировать данные из CSV в pandas DataFrame; вывести первые пять строк и сохранить их в файл preview.csv.

3. Провести базовую очистку: проверить наличие пропущенных значений, типы столбцов, удалить дублирующие записи; сохранить очищенный DataFrame в файл cleaned.csv.

4. Построить гистограммы распределения каждого признака, сохранить графики в отдельные PNG-файлы (feature_sepal_length.png, feature_sepal_width.png и др.).

5. Сохранить весь написанный код в файл data_preprocess.py.

6. Подготовить отчет о выполненной работе, включив скриншоты таблиц, описания проведённых операций и сохранённые графики.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 используйте любой веб-браузер, найдите датасет Iris (CSV) в открытом доступе и скачайте его в папку проекта.

В задании 2 откройте Python-интерпретатор или IDE, импортируйте pandas, загрузите CSV-файл в DataFrame, выведите первые пять строк и экспортируйте их в preview.csv с помощью функции to_csv.

В задании 3 проведите проверку на пропуски (isnull), определите типы столбцов (dtypes), удалите дубли (drop_duplicates) и сохраните результат в cleaned.csv.

В задании 4 с помощью библиотеки matplotlib постройте гистограммы для каждого числового признака, сохраните каждый график функцией savefig с указанием имени PNG-файла.

В задании 5 соберите весь написанный код в один файл data_preprocess.py, разместив его в корневой директории проекта.

В задании 6 создайте документ отчёта в формате ODT или PDF: титульный лист, цель работы, описание каждого задания, скриншоты вывода DataFrame (preview и cleaned), вставьте сохранённые PNG-графики, сделайте выводы о проведённой очистке.

Формат сдачи: архив ZIP, содержащий папку с кодом (data_preprocess.py), исходный и очищенный CSV-файлы, PNG-графики, файл отчёта.

Отчёт: ODT или PDF, титульный лист, цель работы, последовательное описание всех заданий, скриншоты вывода таблиц (preview.csv, cleaned.csv), вставленные PNG-графики, выводы и комментарии. Объём отчёта – 2-3 страницы.

Сдача: упаковать все материалы в архив ZIP и отправить по email преподавателю до окончания занятия.

Тема практической работы №8. Подготовка данных для работы с алгоритмом машинного обучения, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно подготавливать данные для обучения моделей машинного обучения: загружать набор, проводить первичный анализ, выполнять очистку и преобразование признаков, сохранять готовый датасет и визуализации.

Задание(я):

1. Загрузить открытый датасет Iris в Python, сохранить исходный CSV файл.

2. Выполнить исследовательский анализ: вывести статистику, построить гистограммы и графики разброса, сохранить графики в PNG.

3. Провести предобработку: проверить и удалить/заполнить пропущенные значения, закодировать категориальные признаки, масштабировать числовые признаки, сохранить преобразованный набор в CSV.

4. Оформить результаты: собрать скрипты в .py файлы, подготовить отчёт с описанием выполненных шагов, скриншотами графиков и выводами.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки командой pip install pandas scikit-learn matplotlib joblib.

Для задания 1 создайте скрипт, который загружает датасет Iris из библиотеки scikit-learn и сохраняет его в файл iris_raw.csv.

В задании 2 проведите описательную статистику (среднее, минимум, максимум, количество уникальных значений) с помощью pandas, постройте гистограммы распределения каждого числового признака и график разброса пар признаков; каждый график сохраните функцией plt.savefig('название.png').

В задании 3 проверьте наличие пропусков; при их наличии заполните средними значениями или удалите строки. Закодируйте целевой класс (если требуется) в числовой формат, примените стандартизацию (StandardScaler) к числовым признакам. Сохраните полученный набор в файл iris_preprocessed.csv.

В задании 4 разместите все .py файлы в отдельную папку, подготовьте отчёт в формате ODT или PDF, включив титульный лист, описание каждого задания, скриншоты сохранённых графиков, таблицы статистики и комментарии по проведённой предобработке.

Формат сдачи: упаковать папку с кодом, CSV-файлами и отчётом в архив ZIP.

Отчёт: ODT или PDF, титульный лист (название работы, ФИО, группа, дата), содержание, отдельный раздел для каждого задания с описанием действий, скриншотами графиков (PNG), таблицами статистики, выводами о качестве данных; в конце указать список использованных библиотек и их версии.

Сдать архив ZIP, содержащий папку с .py скриптами, CSV-файлы (iris_raw.csv, iris_preprocessed.csv), PNG-графики и готовый отчёт, по электронной почте преподавателя не позднее окончания двухчасового занятия.

Тема практической работы №9. Нормализация и стандартизация данных, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного выполнения предобработки данных: применение методов нормализации и стандартизации, визуализация их влияния и оформление результатов.

Задание(я):

1. Загрузка и первичный анализ датасета (пример: датасет Iris). Результат: файл data_raw.csv и скриншот таблицы в отчёте.

2. Применение Min-Max нормализации к числовым признакам. Результат: файл data_normalized.csv, график распределения до и после нормализации (saved_plot_norm.png).

3. Применение Z-score стандартизации к тем же признакам. Результат: файл data_standardized.csv, график распределения до и после стандартизации (saved_plot_std.png).

4. Сравнительный анализ влияния обеих техник на модель классификации (например, Logistic Regression). Результат: файл model_accuracy.txt, график сравнения метрик (saved_comparison.png).

Методические указания по ходу выполнения работы:

Для всех шагов используйте Python и библиотеки pandas, scikit-learn, matplotlib, joblib (pip install pandas scikit-learn matplotlib joblib).

Создайте отдельный .py файл для каждого задания (task1.py, task2.py, task3.py, task4.py).

Сохраняйте полученные CSV-файлы в папку results/.

Графики сохраняйте функцией plt.savefig('имя.png') в папку results/.

Модель и её метрики сохраняйте с помощью joblib.dump в папку results/.

Для отчёта подготовьте скриншоты таблиц и графиков, разместив их в соответствующих разделах.

Отчёт оформите в ODT или PDF: титульный лист, цель работы, описание каждого задания, скриншоты, выводы.

Все файлы (исходный код, результаты, отчёт) упакуйте в один ZIP-архив.

Отчёт: ODT или PDF, титульный лист, цель, пошаговое описание заданий, скриншоты таблиц и графиков, таблица сравнения метрик, выводы. Включите список файлов, помещённых в архив.

Сдача: архив ZIP со всем содержимым (код, CSV, PNG, txt, отчёт) отправить по e-mail преподавателю до окончания занятия.

Тема практической работы №10. Создание набора данных для обучения и тестирования модели, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно создавать, визуализировать и сохранять набор данных для обучения и тестирования модели машинного обучения.

Задание(я):

1. Выбрать открытый датасет (Iris) и загрузить его в Python.

2. Исследовать структуру данных: вывести первые строки, типы колонок, базовую статистику.

3. Разделить данные на обучающую (70 %) и тестовую (30 %) выборки случайным образом.

4. Сохранить обе выборки в CSV-файлы (train.csv, test.csv).

5. Построить гистограмму распределения классов в обучающей выборке и сохранить её как plot.png. 25 минут.

6. Сохранить скрипт с выполненными шагами в файл dataset_creation.py. 10 минут.

Методические указания по ходу выполнения работы:

Для выполнения всех пунктов используйте Python 3, библиотеки pandas, scikit-learn, matplotlib и joblib (установить через pip install).

Загрузку датасета осуществите через функцию загрузки из sklearn.datasets, преобразуйте в DataFrame и сохраните в переменную.

Для изучения структуры данных применяйте методы head(), info() и describe() без вывода кода в отчёт; сделайте скриншоты результатов.

Разделение данных выполните функцией train_test_split с параметром random_state для воспроизводимости.

Сохраните полученные DataFrame в CSV с помощью метода to_csv, указав кодировку UTF-8 и без индекса.

Для построения гистограммы используйте функцию hist() из matplotlib, задав подписи осей и заголовок; сохраните график командой plt.savefig('plot.png').

Все файлы (dataset_creation.py, train.csv, test.csv, plot.png) упакуйте в один ZIP-архив.

Отчёт оформите в ODT или PDF: титульный лист, описание выполненных заданий, скриншоты вывода таблиц и графика, выводы о полученных данных.

Отчёт – ODT или PDF, содержит титульный лист, список выполненных заданий, скриншоты вывода DataFrame и гистограммы, короткие выводы о полученных наборах данных.

Сдать ZIP-архив, включающий код, CSV-файлы, график и отчёт, по электронной почте преподавателя до конца занятия.

Тема практической работы №11. Визуализация данных для анализа перед обучением, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного выполнения визуализации данных для предварительного анализа перед обучением модели искусственного интеллекта

Задание(я):

1. Установить необходимые библиотеки (pandas, matplotlib, seaborn, scikit-learn). Сохранить список установленных пакетов в файл requirements.txt.

2. Скачать и загрузить в рабочую папку открытый датасет Iris. Сохранить исходный CSV-файл в подпапку data/.

3. Выполнить предварительный анализ: вывести первые 5 строк, описательные статистики, пропущенные значения. Сохранить полученные таблицы в файлы CSV (preview.csv, stats.csv) в папку results/.

4. Построить графики распределения признаков (гистограммы, box-plot) и матрицу корреляций (heatmap). Сохранить каждый график в отдельный PNG-файл в папку figs/.

5. Сформировать скрипт Python, объединяющий все операции (загрузка, анализ, визуализация). Сохранить скрипт как visualisation.py в корневой каталог.

6. Подготовить отчёт, включающий описание выполненных шагов, скриншоты полученных таблиц и графиков, выводы о качестве данных. Сохранить отчёт в формате ODT или PDF под именем report.odt (или report.pdf).

Методические указания по ходу выполнения работы:

Создайте рабочую директорию проекта и внутри неё подпапки data, results, figs.

Выполните команду pip install pandas matplotlib seaborn scikit-learn и зафиксируйте версии в requirements.txt.

Сохраните датасет Iris в формате CSV в папку data.

С помощью pandas загрузите данные, выведите первые строки и базовые статистики, проверьте наличие пропусков. Сохраните результаты в CSV-файлы в папку results.

Для визуализации используйте matplotlib и seaborn: постройте гистограммы и box-plot для каждого признака, а также тепловую карту корреляций. Сохраняйте каждый график командой plt.savefig('имя.png') в папку figs.

Объедините весь код в один файл visualisation.py, чтобы его можно было запускать без дополнительных правок.

Отчёт должен содержать титульный лист, цель работы, пошаговое описание, вставленные скриншоты таблиц и графиков, а также выводы о данных. Приложите файл requirements.txt, visualisation.py, все CSV-файлы и PNG-изображения.

Для сдачи упакуйте все перечисленные файлы в один ZIP-архив.

Отчёт оформляется в ODT или PDF: титульный лист (название работы, ФИО, группа, дата), цель работы, описание каждого задания, скриншоты таблиц и графиков, выводы по качеству данных, список использованных библиотек (requirements.txt).

Сдать архив ZIP, содержащий код, данные, графики, CSV-файлы и отчёт, по указанному email преподавателя до конца занятия.

Тема практической работы №12. Обработка пропущенных значений в данных, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно выполнять предобработку данных, выявлять и обрабатывать пропущенные значения, сохранять полученные наборы и визуализировать влияние обработки на качество модели.

Задание(я):

1. Загрузка открытого датасета (например, Iris) в Python и сохранение исходного CSV-файла (15 минут).

2. Выявление пропущенных значений: построить таблицу с количеством NaN по колонкам и сохранить её в CSV.

3. Применение трех простых стратегий заполнения: (a) удаление строк с NaN, (b) заполнение средним значением, (c) заполнение медианой; сохранить каждый полученный набор в отдельный CSV-файл.

4. Обучить базовую модель классификации (LogisticRegression) на каждом из трёх обработанных наборов, сохранить обученные модели (joblib) и метрики точности в CSV.

5. Построить график сравнения метрик точности для трёх стратегий, сохранить его в файл PNG.

Методические указания по ходу выполнения работы:

Для всех пунктов используйте Python 3 и библиотеки pandas, scikit-learn, matplotlib, joblib (pip install pandas scikit-learn matplotlib joblib).

Каждый промежуточный результат (CSV-файлы, PNG-график, .pkl/.joblib модели) сохраняйте в отдельную папку проекта с понятными именами (raw.csv, missing_report.csv, cleaned_drop.csv, cleaned_mean.csv, cleaned_median.csv, model_drop.joblib и т.д.).

Код каждого шага сохраняйте в отдельный .py файл (load_data.py, detect_missing.py, impute_drop.py, impute_mean.py, impute_median.py, train_models.py, plot_metrics.py).

Все файлы проекта упакуйте в один ZIP-архив.

Сдача: подготовьте отчёт в формате ODT или PDF, включающий титульный лист, описание выполненных заданий, скриншоты кода и результатов (таблицы, график), выводы о влиянии разных стратегий обработки пропусков.

Отчёт и архив с проектом отправьте преподавателю в указанный срок.

Отчёт ODT/PDF: титульный лист, цель практики, список выполненных заданий, описание каждой стратегии обработки, таблицы с количествами пропусков и метриками точности, скриншот графика сравнения, выводы о предпочтительной стратегии.

Сдача: архив ZIP, содержащий все .py-файлы, CSV-данные, PNG-график, сохранённые модели и отчёт ODT/PDF, отправить по e-mail преподавателю до конца занятия.

Тема практической работы №13. Создание отчета по обработке данных, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного сбора, предобработки и визуализации данных, а также формирования итогового отчёта в виде ODT/PDF

Задание(я):

1. Выбрать открытый датасет (например, Iris) и загрузить его в Python; сохранить исходный CSV как data_raw.csv .

2. Выполнить предобработку: очистка от пропусков, кодирование категориальных признаков, масштабирование числовых столбцов; результат сохранить в data_preprocessed.csv.

3. Построить два графика (распределение одного числового признака и парный график признаков); каждый график сохранить в PNG-файлы (plot_hist.png, plot_pair.png).

4. Сформировать отчёт, включив титульный лист, описание выполненных шагов, таблицы с примерами данных, сохранённые графики и выводы; сохранить как report.odt.

Методические указания по ходу выполнения работы:

Для загрузки датасета используйте pandas.read_csv; файл сохраняйте в текущую рабочую папку.

При предобработке проверьте наличие NaN, замените их медианой, преобразуйте строковые метки в числовые с помощью pandas.Categorical, выполните стандартизацию через sklearn.preprocessing.StandardScaler; результат запишите функцией pandas.DataFrame.to_csv без индекса.

Для построения графиков примените matplotlib.pyplot: первый график – гистограмма выбранного признака, второй – scatter-matrix (pairplot) выбранных признаков; каждый график сохраняйте командой plt.savefig('имя.png').

Отчёт оформляйте в LibreOffice Writer: титульный лист (название работы, ФИО, группа, дата), разделы «Описание данных», «Предобработка», «Визуализация», «Выводы». Вставьте скриншоты таблиц (скопируйте из CSV) и PNG-графики, подпишите их. Сохраните файл как report.odt.

Все полученные файлы (data_raw.csv, data_preprocessed.csv, plot_hist.png, plot_pair.png, report.odt) упакуйте в один ZIP-архив.

Отчёт: ODT или PDF, титульный лист, список выполненных заданий, описание каждого шага, таблицы-выдержки из CSV, скриншоты графиков (PNG), выводы. Объём – не менее 2 страниц текста.

Сдача: ZIP-архив, содержащий код .py, все CSV-файлы, PNG-графики и готовый отчёт, отправить по email преподавателю не позднее окончания занятия.

Тема практической работы №14. Объединение данных из разных источников для модели, объем часов 4

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного объединения данных из разных источников, подготовки обучающего набора и обучения простой модели машинного обучения.

Задание(я):

1. Подготовка окружения (установить необходимые библиотеки). Результат: файл requirements.txt.

2. Получение и сохранение исходных наборов данных (например, датасет Iris и дополнительный CSV-файл с признаками). Результат: два файла .csv в папке data/.

3. Загрузка наборов в pandas, предварительный анализ и очистка. Результат: скриншоты статистики, сохранённые в отчёт.

4. Объединение наборов по общему признаку, сохранение объединённого набора. Результат: merged.csv в папке data/.

5. Разделение данных на обучающую и тестовую выборки, обучение модели (например, RandomForest). Результат: файл модели model.joblib.

6. Оценка модели, построение графика метрик (ROC-кривая или матрица ошибок) и их сохранение. Результат: metrics.png и metrics.csv.

7. Подготовка отчёта и упаковка всех файлов в архив. Результат: архив project.zip.

Методические указания по ходу выполнения работы:

Для установки библиотек создайте файл requirements.txt со списком pandas, scikit-learn, matplotlib, joblib и выполните pip install -r requirements.txt.

Сохраните полученные CSV-файлы в каталог data/. Не меняйте их имена.

При загрузке данных используйте pandas.read_csv, проведите проверку на пропуски и типы столбцов, результаты фиксируйте в отчёте.

Объедините таблицы методом merge, укажите ключ объединения, сохраните результат как merged.csv в папке data/ (plt.savefig не требуется).

Разделите merged.csv на обучающую и тестовую части (например, 80/20), обучите модель, сохраните её командой joblib.dump в файл model.joblib.

Оцените модель (accuracy, precision, recall), постройте график метрик с помощью matplotlib и сохраните как metrics.png; экспортируйте численные метрики в metrics.csv.

Все скрипты (.py), данные (.csv), модели (.joblib), графики (.png) и отчёт (ODT или PDF) упакуйте в один ZIP-архив для сдачи.

Отчёт оформляется в ODT или PDF, включает титульный лист, цель работы, описание каждого задания, скриншоты анализа данных, таблицы метрик, график metrics.png и выводы. Объём отчёта – 5-7 страниц.

Сдача: архив ZIP (project.zip) отправить по email преподавателю до конца занятия.

Тема практической работы №15. Реализация задачи классификации с обучением с учителем, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно выполнять процесс обучения модели классификации с учителем: предобработка данных, обучение, оценка качества и сохранение модели.

Задание(я):

1. Подготовить данные (загрузить датасет Iris, разделить на обучающую и тестовую выборки, сохранить полученные CSV-файлы).

2. Обучить классификатор (использовать LogisticRegression из scikit-learn, обучить на подготовленных данных, сохранить обученную модель в файл).

3. Оценить модель (рассчитать метрики точности, построить матрицу ошибок, сохранить график и таблицу метрик, добавить скриншоты в отчёт).

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для чтения датасета Iris, выполните случайное разбиение (train_test_split) с фиксированным seed, сохраните train.csv и test.csv через to_csv.

В задании 2 импортируйте LogisticRegression, обучите модель на train.csv, сохраните её с помощью joblib.dump('model.joblib').

В задании 3 вычислите accuracy_score и confusion_matrix, постройте тепловую карту матрицы ошибок с помощью matplotlib, сохраните график plt.savefig('confusion.png'), экспортируйте метрики в CSV (metrics.csv).

Все скрипты разместите в отдельных файлах: prepare_data.py, train_model.py, evaluate.py.

Формат сдачи: ZIP-архив, содержащий папку с кодом (.py), папку data с CSV-файлами, папку results с моделью, графиками и метриками, а также отчёт в формате ODT или PDF.

Отчёт ODT/PDF: титульный лист, цель работы, описание каждого задания, скриншоты вывода скриптов, вставка сохранённого графика confusion.png, таблица метрик из metrics.csv, выводы и выводы о качестве модели.

Сдача: подготовить ZIP-архив согласно пункту "Формат сдачи" и отправить по email преподавателю до конца занятия.

Тема практической работы №16. Обучение модели для задачи регрессии, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно обучать регрессионную модель на открытом датасете, проводить её оценку и сохранять артефакты проекта.

Задание(я):

1. Установить необходимые библиотеки (pip install scikit-learn pandas matplotlib joblib).

2. Загрузить датасет Iris (использовать только числовые признаки) и сохранить его в CSV.

3. Выполнить предобработку: разделить данные на обучающую и тестовую выборки, масштабировать признаки.

4. Обучить линейную регрессию (LinearRegression) на обучающем наборе.

5. Оценить модель: вычислить MAE и R^2 на тестовой выборке, построить график предсказанных vs реальных значений, сохранить график.

6. Сохранить обученную модель в файл (joblib.dump) и экспортировать метрики в CSV.

7. Подготовить отчёт с описанием выполненных шагов, скриншотами кода и графика, выводами.

Методические указания по ходу выполнения работы:

В задании 1 откройте терминал, выполните указанные pip install команды, проверьте их успешность.

В задании 2 загрузите датасет Iris через pandas, оставив только числовые столбцы, сохраните полученный DataFrame в файл data.csv.

В задании 3 используйте функцию train_test_split из scikit-learn, задав соотношение 80/20, примените StandardScaler к признакам; результаты сохраните в файлы X_train.npy, X_test.npy, y_train.npy, y_test.npy.

В задании 4 обучите модель LinearRegression на X_train и y_train; сохраните код в файле train_model.py.

В задании 5 предскажите значения на X_test, вычислите MAE и R^2, постройте scatter-plot (реальные vs предсказанные) и сохраните его как plot.png; метрики запишите в metrics.csv.

В задании 6 сохраните модель в файл model.joblib с помощью joblib.dump; убедитесь, что все артефакты (CSV, PNG, NPY, PY) находятся в одной папке.

В задании 7 создайте отчёт ODT или PDF: титульный лист, краткое описание каждого задания, скриншоты кода, вставьте plot.png, таблицу метрик из metrics.csv, сделайте выводы о качестве модели.

Формат сдачи: архив ZIP, содержащий папку с кодом (.py), данными (.csv, .npy), моделью (.joblib), графиком (.png) и отчётом.

Отчёт ODT или PDF: титульный лист, список заданий, описание выполненных действий, скриншоты кода, график plot.png, таблица метрик из metrics.csv, выводы и заключения.

Сдача: упаковать всё в архив ZIP и отправить по email преподавателю до конца занятия.

Тема практической работы №17. Обучение модели без учителя на основе кластеризации, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно обучать модель без учителя на основе кластеризации, проводить предобработку данных, сохранять полученную модель и визуализировать результаты.

Задание(я):

1. Загрузка и подготовка открытого датасета Iris. Сохранить исходный CSV-файл в папку data/.

2. Выполнение предобработки: выделить числовые признаки, выполнить стандартизацию. Сохранить полученный массив признаков в файл data/processed.npy.

3. Обучение модели K-Means (k=3) на подготовленных данных. Сохранить обученную модель в файл models/kmeans.joblib.

4. Визуализация полученных кластеров на графике «два главных компонента». Сохранить график в файл results/clusters.png и экспортировать таблицу с метками кластеров в CSV (results/cluster_labels.csv).

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для загрузки датасета Iris из встроенных ресурсов scikit-learn; сохраните полученный DataFrame в CSV-файл в папку data/.

В задании 2 выберите только числовые столбцы, выполните стандартизацию с помощью StandardScaler; результат сохраните в бинарный файл (numpy) в папку data/.

В задании 3 импортируйте класс KMeans из scikit-learn, задайте количество кластеров 3 и обучите модель на подготовленных данных; сохраните объект модели с помощью joblib в папку models/.

В задании 4 примените метод PCA для уменьшения размерности до 2, постройте scatter-plot, раскрасив точки в соответствии с предсказанными метками кластеров; сохраните график в results/clusters.png, а таблицу с исходными объектами и их метками – в results/cluster_labels.csv.

Формат сдачи: один архив ZIP, содержащий папки data/, models/, results/ и отчёт в формате ODT или PDF.

Отчёт ODT или PDF должен включать: титульный лист (название работы, ФИО, группа, дата), краткое описание каждого задания, скриншоты кода (без полного текста), изображение сохранённого графика clusters.png, таблицу с метками (привести фрагмент CSV), выводы о качестве кластеризации.

Сдача: упаковать всё в ZIP-архив и отправить по e-mail преподавателю не позднее окончания занятия.

Тема практической работы №18. Оптимизация гиперпараметров модели с помощью Grid Search, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного подбора гиперпараметров модели машинного обучения с использованием Grid Search и оценить влияние параметров на качество модели

Задание(я):

1. Подготовка данных: загрузить датасет Iris, выполнить предобработку и разделить на обучающую и тестовую выборки. Сохранить разбиения в файлы train.csv и test.csv.

2. Обучение базовой модели: построить базовый классификатор (например, RandomForest) без настройки гиперпараметров, оценить точность на тесте и сохранить результаты в файл baseline_metrics.csv.

3. Настройка Grid Search: задать сетку гиперпараметров, запустить GridSearchCV, получить лучшую модель и таблицу результатов. Сохранить таблицу в файл grid_results.csv.

4. Оценка оптимизированной модели: измерить точность лучшей модели на тестовой выборке, сравнить с базовой, сохранить метрики в файл tuned_metrics.csv.

5. Визуализация влияния параметров: построить график зависимости точности от параметров (например, тепловая карта), сохранить как heatmap.png.

6. Сохранение артефактов: сохранить обученную лучшую модель в файл model.joblib, собрать весь код в файл main.py.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки: pip install scikit-learn pandas matplotlib joblib.

Для загрузки датасета используйте встроенные функции scikit-learn.

Разделите данные функцией train_test_split с фиксированным random_state.

Обучите базовую модель методом fit и оцените предсказания функцией predict.

Для Grid Search используйте класс GridSearchCV, укажите параметр cv=5.

Сохраните полученную лучшую модель с помощью joblib.dump.

Экспортируйте все таблицы в CSV функцией pandas.DataFrame.to_csv.

График сохраняйте командой plt.savefig('heatmap.png').

Все скрипты разместите в одной папке, назовите их согласно заданиям.

Подготовьте отчёт, включив титульный лист, описание каждого задания, скриншоты вывода (таблицы, графика) и выводы о влиянии гиперпараметров.

Отчёт в формате ODT или PDF. Структура: титульный лист; цель работы; описание выполнения каждого задания; скриншоты CSV-таблиц и графика; сравнение метрик baseline и tuned; выводы. Приложения: main.py, model.joblib, CSV-файлы, heatmap.png.

Сформировать ZIP-архив, содержащий отчёт, код (main.py), сохранённую модель, CSV-файлы и график. Отправить архив по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №19. Настройка гиперпараметров для улучшения качества модели, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного подбора гиперпараметров модели машинного обучения и оценки их влияния на качество предсказаний.

Задание(я):

1. Подготовить рабочее окружение, установить необходимые библиотеки и загрузить открытый датасет Iris. Сохранить исходный набор данных в файл data.csv.

2. Обучить базовую модель (логистическая регрессия) без настройки гиперпараметров, сохранить обученную модель в файл baseline_model.joblib и результаты предсказаний в файл baseline_predictions.csv.

3. Сформировать сетку гиперпараметров для логистической регрессии (например, параметр регуляризации C и тип штрафа). Выполнить поиск оптимальных параметров с помощью GridSearchCV, сохранить найденые лучшие параметры в файл best_params.txt.

4. Обучить финальную модель с найденными гиперпараметрами, сохранить её в файл tuned_model.joblib, а предсказания – в файл tuned_predictions.csv.

5. Оценить качество обеих моделей (accuracy, precision, recall) и построить график зависимости метрики от значения гиперпараметра C. Сохранить график в файл metric_plot.png и таблицу сравнения метрик в файл metrics_comparison.csv.

Методические указания по ходу выполнения работы:

В задании 1 создайте отдельную папку проекта, откройте терминал и выполните команды pip install scikit-learn pandas matplotlib joblib. Скачайте датасет Iris из библиотеки scikit-learn и экспортируйте его в CSV-файл data.csv в корневой каталог проекта.

В задании 2 загрузите data.csv, разделите его на обучающую и тестовую выборки (например, 70/30), обучите базовую модель логистической регрессии без изменения параметров, сохраните объект модели с помощью joblib.dump в файл baseline_model.joblib и экспортируйте предсказания на тестовом наборе в baseline_predictions.csv.

В задании 3 определите диапазон значений для гиперпараметра C (например, 0.01, 0.1, 1, 10, 100) и варианты штрафа (l1, l2). Сформируйте словарь параметров, передайте его в GridSearchCV, укажите кросс-валидацию 5-fold, запустите поиск, после завершения запишите лучшие найденные параметры в файл best_params.txt.

В задании 4 используя параметры из best_params.txt обучите новую модель, сохраните её в tuned_model.joblib и предсказания на тестовом наборе в tuned_predictions.csv.

В задании 5 вычислите метрики точности (accuracy), точности (precision) и полноты (recall) для обеих моделей, создайте таблицу сравнения и сохраните её в metrics_comparison.csv. Постройте график зависимости accuracy от значения C, подпишите оси, добавьте легенду и сохраните изображение в metric_plot.png.

Формат сдачи: подготовьте архив ZIP, в котором должны находиться: папка с исходным кодом (script.py), все CSV-файлы, файлы моделей (baseline_model.joblib, tuned_model.joblib), файл best_params.txt, график metric_plot.png и отчёт.

Формат отчёта: ODT или PDF. Титульный лист (название практики, ФИО, группа, дата). Далее – описание каждого задания, скриншоты консольных выводов, таблица metrics_comparison.csv, изображение metric_plot.png, выводы о влиянии гиперпараметров на качество модели.

Сдача: упаковать все материалы в архив ZIP и отправить по электронной почте преподавателю до окончания занятия (2 академических часа).

Тема практической работы №20. Применение метода кросс-валидации, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного применения кросс-валидации для оценки качества моделей машинного обучения.

Задание(я):

1. Подготовка окружения и загрузка датасета Iris. Результат: файл data.csv.

2. Предобработка данных: разделение признаков и целевой переменной, масштабирование. Результат: файлы X.npy и y.npy.

3. Обучение базовой модели (LogisticRegression) с использованием 5-кратной кросс-валидации. Результат: файл cv_scores.npy и график распределения метрик (cv_plot.png).

4. Сохранение обученной модели и выводов. Результат: файл model.joblib, файл metrics.csv.

5. Оформление отчёта.

Методические указания по ходу выполнения работы:

В задании 1 установите необходимые библиотеки командой pip install scikit-learn pandas matplotlib joblib. С помощью pandas загрузите датасет Iris и сохраните его в CSV-файл data.csv.

В задании 2 разделите таблицу на матрицу признаков X и вектор меток y, выполните стандартизацию признаков (StandardScaler). Сохраните полученные массивы в файлы X.npy и y.npy.

В задании 3 создайте объект модели LogisticRegression. С помощью функции cross_val_score выполните 5-кратную кросс-валидацию, получив массив оценок точности. Сохраните массив в файл cv_scores.npy. Постройте гистограмму распределения оценок, сохраните её в файл cv_plot.png (используйте plt.savefig).

В задании 4 обучите модель на всем наборе данных (fit). Сохраните обученный объект с помощью joblib.dump в файл model.joblib. Вычислите среднее и стандартное отклонение оценок кросс-валидации, запишите результаты в CSV-файл metrics.csv.

В задании 5 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание выполненных заданий,

скриншоты кода и результатов (таблицы, графики), выводы. Включите ссылки на файлы data.csv, X.npy, y.npy, cv_scores.npy, cv_plot.png, model.joblib, metrics.csv.

Формат сдачи: упакуйте все файлы (исходный код .py, CSV, NPY, PNG, JOBLIB, отчёт) в один архив ZIP.

Отчёт ODT или PDF: титульный лист; цель работы; пошаговое описание заданий; скриншоты кода и результатов (таблицы, графики); таблица метрик; выводы и выводы о качестве модели.

Сдача: архив ZIP с кодом, данными, моделью и отчётом отправить по email преподавателю не позднее окончания 2-х академических часов.

Тема практической работы №21. Оценка производительности модели после настройки, объем часов 2

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно оценивать производительность модели машинного обучения после её настройки

Задание(я):

1. Подготовить данные: загрузить датасет Iris, выполнить разбиение train/test, сохранить разбиения в CSV.

2. Обучить базовую модель: построить LogisticRegression с параметрами по умолчанию, сохранить модель.

3. Настроить гиперпараметры: выполнить GridSearchCV по параметрам C и penalty, сохранить лучшую модель.

4. Оценить качество: рассчитать accuracy, precision, recall, построить матрицу ошибок, сохранить метрики в CSV.

5. Визуализировать результаты: построить график зависимости accuracy от параметра C, сохранить как PNG.

6. Оформить отчёт: включить описание, скриншоты, таблицы, графики, сохранить в ODT/PDF.

Методические указания по ходу выполнения работы:

Для задания 1 используйте pandas и scikit-learn, результаты сохраните в data/train.csv и data/test.csv.

Для задания 2 обучите модель на train-данных, сохраните её с помощью joblib в model/baseline.pkl.

Для задания 3 выполните перебор гиперпараметров, сохраните лучшую модель в model/best.pkl и параметры в results/best_params.csv.

Для задания 4 рассчитайте метрики, запишите их в results/metrics.csv, постройте confusion matrix и сохраните как results/confusion.png.

Для задания 5 постройте график validation-curve и сохраните его как results/validation.png.

Формат сдачи: ZIP-архив, содержащий папки code, data, model, results и файл отчёт.odt/pdf.

Отчёт ODT или PDF: титульный лист, цель, описание выполненных шагов, скриншоты кода и графиков, таблица метрик, выводы.

Сдача: ZIP-архив отправить преподавателю по email до конца занятия.

Тема практической работы №22. Использование различных моделей для решения задачи классификации, объем часов 4

У2. Разрабатывать сценарии обучения, определять параметры обучения для различных типов моделей ИИ.

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного применения готовых моделей машинного обучения для решения задачи классификации, включая подготовку данных, обучение, оценку качества и сохранение артефактов.

Задание(я):

1. Загрузка и исследование датасета Iris. Сохранить первые 5 строк в файл data_sample.csv.

2. Разделение данных на обучающую и тестовую выборки. Сохранить индексы разбиения в файл split_indices.npy.

3. Обучение трёх моделей (Logistic Regression, Decision Tree, Random Forest) на обучающей выборке. Сохранить каждую модель в файл model_<name>.joblib.

4. Оценка моделей на тестовой выборке: точность, полнота, F1-score. Сохранить таблицу метрик в файл metrics.csv.

5. Визуализация результатов: построить график сравнения точности моделей и матрицы ошибок для каждой модели. Сохранить графики в файлы accuracy.png, cm_<name>.png.

6. Подготовка отчёта: оформить результаты, включить скриншоты графиков, выводы и список использованных команд.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки: pip install scikit-learn pandas matplotlib seaborn joblib numpy (один раз в начале).

Для задания 1 откройте Python-скрипт, загрузите датасет Iris из библиотеки sklearn, выведите первые пять строк и экспортируйте их в CSV файл data_sample.csv.

Для задания 2 используйте функцию train_test_split, задав фиксированный random_state, сохраните массивы индексов обучающей и тестовой выборок в файл split_indices.npy с помощью numpy.save.

Для задания 3 последовательно создайте и обучите модели LogisticRegression, DecisionTreeClassifier, RandomForestClassifier, используя обучающую выборку; после обучения сохраните каждую модель функцией joblib.dump в файлы model_logreg.joblib, model_tree.joblib, model_forest.joblib.

Для задания 4 предскажите классы на тестовой выборке каждой моделью, вычислите метрики точности, полноты и F1-score (sklearn.metrics), сформируйте таблицу с колонками Model, Accuracy, Precision, Recall, F1 и запишите её в CSV файл metrics.csv.

Для задания 5 постройте столбчатую диаграмму сравнения точности моделей (matplotlib/ seaborn) и сохраните её как accuracy.png. Для каждой модели постройте матрицу ошибок (confusion_matrix) и визуализируйте её как тепловую карту, сохранив файлы cm_logreg.png, cm_tree.png, cm_forest.png.

Для задания 6 создайте документ ODT или PDF, включив титульный лист, краткое описание каждого задания, скриншоты сохранённых графиков, таблицу метрик из metrics.csv и выводы о лучшей модели. Приложите все .py скрипты, .csv, .png и .joblib файлы.

Все файлы упакуйте в один ZIP-архив для сдачи.

Отчёт: ODT или PDF, титульный лист, описание практики, список выполненных заданий, скриншоты графиков (accuracy.png, cm_*.png), таблица metrics.csv, выводы о сравнении моделей. В конце – перечень файлов архива.

Сдача: ZIP-архив с кодом, данными, моделями, графиками и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №23. Расчет метрик точности для модели, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно рассчитывать метрики точности (precision, recall, F1) для готовой модели искусственного интеллекта и оформлять результаты в виде отчёта.

Задание(я):

1. Подготовка окружения и загрузка готовой модели.

2. Загрузка тестового набора данных (датасет Iris).

3. Получение предсказаний модели на тестовых данных.

4. Вычисление метрик точности (precision, recall, F1) для каждой категории и в среднем.

5. Визуализация метрик в виде бар-графика и сохранение графика в файл.

6. Сохранение вычисленных метрик в CSV-файл и модели в файл.

7. Оформление отчёта с описанием выполненных шагов, скриншотами результатов и выводами.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки командой: pip install scikit-learn pandas matplotlib joblib.

Создайте рабочую папку проекта и в ней подпапки: data, models, results, report.

В подпапке models разместите готовую модель (файл .pkl или .joblib).

В подпапке data загрузите набор данных Iris (можно использовать функцию загрузки из scikit-learn и сохранить его в CSV).

Выполните предсказание модели для всех объектов тестового набора и сохраните полученные метки в переменную.

С помощью функций из scikit-learn рассчитайте precision, recall и F1-score для каждого класса и их среднее (macro). Сохраните таблицу метрик в файл CSV в папке results (например, metrics.csv).

Постройте бар-график, где по оси X будут названия метрик, а по оси Y – их значения. Сохраните график в файл PNG в папке results (metrics.png).

Сохраните модель в файл с помощью joblib.dump в папке models (если модель была изменена).

Подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание каждого задания, скриншоты кода и результатов, таблица метрик, изображение графика, выводы.

Все файлы (исходный код .py, CSV, PNG, модель, отчёт) упакуйте в один ZIP-архив.

Формат сдачи: архив ZIP, отправить по email преподавателю до конца занятия.

Отчёт должен включать титульный лист, цель работы, пошаговое описание выполнения заданий, скриншоты консольных выводов и графика, таблицу метрик (вставку из CSV), выводы о качестве модели. Оформление – ODT или PDF.

Сдача: архив ZIP с кодом, данными, результатами и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №24. Оценка точности модели на новых данных, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельной оценки точности готовой модели искусственного интеллекта на новых данных, включая загрузку модели, подготовку тестового набора, вычисление метрик и оформление результатов.

Задание(я):

1. Подготовка рабочей среды (установить Python, необходимые библиотеки, создать проект).

2. Загрузка открытого датасета (например, Iris) и сохранение его в файл CSV.

3. Загрузка готовой предобученной модели (например, модель классификации, сохранённую в формате joblib).

4. Применение модели к тестовым данным, получение предсказаний.

5. Вычисление метрик точности (accuracy, precision, recall, f1-score) и построение графика зависимости метрик от классов.

6. Сохранение полученных метрик в CSV, графика в PNG и модели (при необходимости) в файл.

7. Оформление отчёта с описанием проделанных шагов, скриншотами результатов и выводами.

Методические указания по ходу выполнения работы:

Для выполнения задания используйте Python 3 и менеджер пакетов pip; установите библиотеки pandas, scikit-learn, matplotlib, joblib (pip install pandas scikit-learn matplotlib joblib).

Создайте отдельную папку проекта, в ней подпапки data, model, results, report.

Сохраните исходный датасет в папку data, а предобученную модель – в папку model.

После получения предсказаний запишите их вместе с истинными метками в файл results/predictions.csv.

Метрики сохраняйте в файл results/metrics.csv, а график – в файл results/metrics.png (используйте plt.savefig).

Код всех скриптов сохраняйте в файлы .py в корневой директории проекта (например, evaluate.py).

Подготовьте отчёт в формате ODT или PDF, включив титульный лист, описание каждого задания, скриншоты вывода консоли и графика, а также краткие выводы о качестве модели.

Соберите все материалы (папку проекта, скрипты, CSV-файлы, PNG-график, отчёт) в один ZIP-архив для сдачи.

Отчёт должен содержать: титульный лист (название работы, ФИО, группа), краткое описание цели, последовательность выполненных заданий,

скриншоты вывода метрик и графика, таблицу с метриками из CSV, выводы о точности модели. Формат – ODT или PDF.

Сдача: упаковать все файлы проекта и отчёт в ZIP-архив и отправить по e-mail преподавателю до конца занятия.

Тема практической работы №25. Применение F1-score для анализа эффективности модели, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить вычисление и интерпретацию метрики F1-score для оценки качества бинарных классификаторов.

Задание(я):

1. Подготовить данные: загрузить открытый датасет Iris, отобрать два класса, выполнить разбиение на обучающую и тестовую выборки; сохранить полученные наборы в файлы train.csv и test.csv.

2. Обучить простую модель: с помощью библиотеки scikit-learn построить логистическую регрессию на обучающем наборе; сохранить обученную модель в файл model.joblib.

3. Оценить модель: предсказать метки на тестовом наборе, вычислить precision, recall и F1-score; построить график зависимости порога классификации от F1-score и сохранить его в файл f1_curve.png; сохранить таблицу метрик в файл metrics.csv.

Методические указания по ходу выполнения работы:

Для всех задач используйте Python 3, создайте отдельный .py файл для каждого задания (prepare_data.py, train_model.py, evaluate_model.py).

Установите необходимые библиотеки командой: pip install scikit-learn matplotlib pandas joblib.

В задании 1 загрузите датасет Iris через pandas или scikit-learn, отфильтруйте два требуемых класса, выполните разбиение с помощью train_test_split, сохраните результаты в CSV-файлы в папку data/.

В задании 2 импортируйте модель LogisticRegression, обучите её на файле train.csv, сохраните объект модели с помощью joblib.dump в папку models/.

В задании 3 загрузите модель и тестовый набор, получите вероятности предсказаний, пройдитесь по диапазону порогов от 0.0 до 1.0 с шагом 0.01, для каждого порога вычисляйте precision, recall и F1-score, постройте график (ось X – порог, ось Y – F1-score) и сохраните его функцией plt.savefig('f1_curve.png') в папку results/. Сохраните итоговые метрики (лучший порог, соответствующие precision, recall, F1) в файл metrics.csv в той же папке.

После выполнения всех заданий упакуйте в один архив ZIP папки data/, models/, results/ и файл отчёта.

Формат сдачи: архив ZIP, содержащий весь исходный код, CSV-файлы, сохранённую модель, график и отчёт.

Отчёт оформляется в формате ODT или PDF и включает: титульный лист (название практической работы, ФИО студента, группа, дата); краткое описание каждого задания; скриншоты вывода консоли для каждого .py файла; таблицу метрик из файла metrics.csv; график f1_curve.png; выводы о влиянии порога на F1-score и общие выводы по работе.

Сдача: подготовьте ZIP-архив с указанными файлами и отправьте его по электронной почте преподавателю не позднее конца текущего занятия.

Тема практической работы №26. Сравнение нескольких моделей по различным метрикам, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно сравнивать готовые модели искусственного интеллекта по набору метрик, оформлять результаты в виде таблиц и графиков, сохранять модели и готовый аналитический отчёт.

Задание(я):

1. Выбор и загрузка открытого датасета (например, Iris). Подготовка обучающего и тестового наборов, сохранение разбиения в CSV-файлы.

2. Загрузка трёх готовых моделей из библиотеки scikit-learn (Logistic Regression, Decision Tree, K-Nearest Neighbors). Обучение моделей на подготовленном наборе, сохранение каждой модели в файл с помощью joblib.

3. Вычисление основных метрик качества (accuracy, precision, recall, f1-score) для каждой модели на тестовом наборе. Сохранение результатов в таблицу CSV.

4. Построение сравнительного графика метрик (например, столбчатая диаграмма). Сохранение графика в файл PNG.

5. Оформление отчёта: титульный лист, описание выполненных шагов, таблица метрик, скриншот кода (фрагменты) и графика, выводы о предпочтительной модели.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки одной командой: pip install scikit-learn pandas matplotlib joblib.

Создайте отдельную папку проекта, в ней подпапки data, models, figures и report.

В задаче 1 сохраните обучающий и тестовый набор в data/train.csv и data/test.csv.

В задаче 2 после обучения каждой модели сохраните её в models/ с понятными названиями, например, logistic.joblib, tree.joblib, knn.joblib.

В задаче 3 сформируйте таблицу metrics.csv в папке data, где строки – модели, столбцы – метрики.

В задаче 4 сохраните построенный график в figures/comparison.png, используя plt.savefig().

В задаче 5 подготовьте документ отчёта в формате ODT или PDF, разместив в нём: титульный лист, краткое описание каждого задания, таблицу metrics.csv (скриншот или вставка), изображение figures/comparison.png, выводы.

Для сдачи упакуйте всё содержимое проекта (папки data, models, figures, report) в один ZIP-архив.

Формат сдачи: архив ZIP, содержащий код .py файлов, сохранённые CSV, PNG и готовый отчёт ODT/PDF.

Отчёт должен включать: 1) титульный лист (название практики, ФИО, группа, дата); 2) цель и краткое описание работы; 3) пошаговое описание выполненных заданий; 4) таблицу метрик (скриншот или вставка CSV); 5) график сравнения метрик (скриншот PNG); 6) выводы о лучшей модели и рекомендациях. Оформление – стандартные стили LibreOffice: шрифт Times New Roman, размер 12, межстрочный интервал 1,5.

Сдача: упаковать все файлы проекта в ZIP-архив и отправить по электронной почте преподавателю не позже конца двух академических часов.

Тема практической работы №27. Построение ROC-кривой для анализа модели, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно строить ROC-кривую и оценивать качество бинарного классификатора.

Задание(я):

1. Подготовить набор данных (например, Iris, выбрать два класса).

2. Обучить простой бинарный классификатор (LogisticRegression). Время – 30 минут.

3. Вычислить вероятности, построить ROC-кривую и рассчитать AUC.

4. Сохранить график ROC в файл, оформить отчёт.

Методические указания по ходу выполнения работы:

В задании 1 загрузите выбранный датасет, отфильтруйте два класса и сохраните полученные признаки и метки в CSV (data.csv).

В задании 2 реализуйте обучение модели, сохраните обученный объект с помощью joblib (model.joblib).

В задании 3 с помощью функций sklearn.metrics вычислите false positive rate, true positive rate и AUC, постройте график с помощью matplotlib и сохраните его как roc.png (plt.savefig('roc.png')).

В задании 4 подготовьте отчёт в ODT или PDF: титульный лист, описание выполненных шагов, скриншоты кода и выводов, вставьте изображение roc.png, сделайте выводы о качестве модели.

Формат сдачи: архив ZIP, содержащий файл кода (script.py), data.csv, model.joblib, roc.png и готовый отчёт.

Отчёт: ODT или PDF, титульный лист, список задач, описание выполнения, скриншоты экранов, график ROC (roc.png), выводы о AUC.

Сдать: ZIP-архив с указанными файлами, отправить по email преподавателю до конца занятия.

Тема практической работы №28. Визуализация результатов модели с помощью confusion matrix, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно визуализировать результаты классификационной модели с помощью матрицы ошибок (confusion matrix)

Задание(я):

1. Подготовить рабочее окружение: установить Python, pip, библиотеки pandas, scikit-learn, matplotlib (pip install pandas scikit-learn matplotlib). Скачать открытый датасет Iris и сохранить его в файл data.csv. Создать скрипт train_model.py, в котором загрузить данные, разделить их на обучающую и тестовую выборки и обучить простую модель (например, LogisticRegression). Сохранить обученную модель в файл model.joblib (joblib.dump).

2. Оценить модель: в отдельном скрипте evaluate.py загрузить модель, выполнить предсказания на тестовом наборе, вычислить матрицу ошибок (confusion_matrix) и сохранить её в файл confusion_matrix.csv. Также сохранить в файл predictions.csv реальные и предсказанные метки.

3. Визуализировать матрицу ошибок: в скрипте plot_confusion.py загрузить матрицу из файла confusion_matrix.csv, построить тепловую карту с помощью matplotlib, добавить подписи классов и значения ячеек, сохранить график как confusion_matrix.png (plt.savefig).

4. Оформить отчет: создать документ отчёта в ODT или PDF, включить титульный лист, описание выполненных заданий, скриншоты кода (из файлов .py), таблицы результатов (CSV) и изображение confusion_matrix.png. Сохранить весь проект (скрипты, данные, модель, график, отчёт) в архив ZIP.

Методические указания по ходу выполнения работы:

Для каждого задания используйте отдельный .py файл, названия указаны в описании заданий.

Все промежуточные и итоговые данные (CSV, PNG, joblib) сохраняйте в каталог results внутри проекта.

Не забудьте добавить комментарии в код для пояснения шагов (без примеров кода).

Отчёт оформляйте согласно шаблону: титульный лист → цель работы → описание выполнения → результаты (таблицы, графики) → выводы.

Перед сдачей проверьте, что в архиве присутствуют: train_model.py, evaluate.py, plot_confusion.py, data.csv, model.joblib, predictions.csv, confusion_matrix.csv, confusion_matrix.png, отчёт ODT/PDF.

Отчёт в формате ODT или PDF. Структура: 1) Титульный лист (название работы, ФИО, группа); 2) Цель практической работы; 3) Описание каждого задания; 4) Скриншоты кода и выводов; 5) Таблицы результатов (CSV); 6) График confusion_matrix.png; 7) Выводы и оценка проделанной работы.

Сдать архив ZIP, содержащий весь проект и отчёт, по электронной почте преподавателю не позднее окончания занятия (2 академических часа).

Тема практической работы №29. Оптимизация модели на основе полученных метрик, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельной оптимизации готовой модели машинного обучения на основе полученных метрик

Задание(я):

1. Загрузить готовую модель и набор данных (датасет Iris). Оценить базовые метрики (accuracy, precision, recall). Сохранить метрики в CSV (baseline_metrics.csv).

2. Провести подбор гиперпараметров с помощью GridSearchCV (поиск оптимального количества деревьев и глубины для RandomForest). Сохранить результаты поиска в CSV (grid_search_results.csv).

3. Обучить модель с найденными лучшими параметрами. Оценить улучшенные метрики, сравнить с базовыми. Сохранить новые метрики в CSV (optimized_metrics.csv).

4. Визуализировать сравнение метрик (график accuracy_before_vs_after). Сохранить график в PNG (metrics_comparison.png).

5. Сохранить оптимизированную модель (model_optimized.pkl) и весь исходный код в отдельные .py файлы. Подготовить отчёт.

Методические указания по ходу выполнения работы:

Для задания 1 используйте pandas для загрузки и сохранения CSV, scikit-learn для расчёта метрик. Файл CSV разместите в папке results.

Для задания 2 настройте GridSearchCV, укажите диапазон параметров, выполните поиск. Сохраните таблицу результатов в results/grid_search_results.csv.

Для задания 3 обучите модель с лучшими параметрами, рассчитайте метрики, сохраните их в results/optimized_metrics.csv.

Для задания 4 постройте график сравнения метрик (matplotlib), сохраните как results/metrics_comparison.png.

Для задания 5 сохраните модель с помощью joblib в models/model_optimized.pkl, код разместите в src/. Оформите отчёт в ODT или PDF, включив титульный лист, описание каждого задания, скриншоты вывода и графика.

Формат сдачи: архив ZIP, содержащий папки src (исходный код), models (модель), results (CSV и PNG), report (отчёт).

Отчёт ODT или PDF: титульный лист, цель работы, описание каждого задания, таблицы метрик (baseline и optimized), график сравнения, выводы и рекомендации.

Сдать ZIP-архив с указанной структурой по email преподавателя до конца занятия.

Тема практической работы №30. Оценка модели с использованием метрик precision и recall, объем часов 4

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно оценивать готовую модель ИИ с помощью метрик precision и recall, визуализировать результаты и оформлять отчёт.

Задание(я):

1. Подготовка среды: установить необходимые библиотеки, загрузить готовую модель (joblib) и тестовый набор данных (CSV). Сохранить модель в файл model.pkl, данные в test_data.csv.

2. Получение предсказаний: применить загруженную модель к тестовому набору, сохранить предсказанные метки в файл predictions.csv.

3. Расчёт метрик и визуализация: вычислить precision, recall и матрицу ошибок, построить график зависимости precision/recall от порога классификации, сохранить график в precision_recall.png, метрики в metrics.csv.

4. Оформление отчёта и упаковка: подготовить отчёт в ODT/PDF, включить титульный лист, описание выполненных шагов, скриншоты вывода, график и таблицы метрик, упаковать всё в ZIP-архив.

Методические указания по ходу выполнения работы:

Установите библиотеки: pip install scikit-learn matplotlib pandas joblib.

Создайте рабочую папку, поместите в неё файл модели и набор тестовых данных.

В задании 1 загрузите модель с помощью joblib.load и данные через pandas.read_csv; сохраните их в указанные файлы.

В задании 2 с помощью метода predict модели получите предсказания; экспортируйте их в predictions.csv.

В задании 3 используйте функции precision_score, recall_score, confusion_matrix из scikit-learn; постройте график precision-recall с помощью matplotlib, сохраните его как precision_recall.png; метрики запишите в metrics.csv.

В задании 4 подготовьте отчёт: титульный лист, цель, описание каждого задания, скриншоты результатов (вывод консоли, график), таблицы из CSV; сохраните документ в ODT или PDF. Упакуйте код (.py), модель, CSV-файлы, график и отчёт в один ZIP-архив.

Формат сдачи: архив ZIP, отправить по email преподавателю до конца занятия.

Отчёт в ODT или PDF: титульный лист, цель работы, пошаговое описание заданий, скриншоты вывода, график precision_recall.png, таблицы metrics.csv и predictions.csv, выводы.

Сдача: ZIP-архив с кодом, моделью, данными, графиками и отчётом отправить преподавателю по указанному email до конца занятия.

Тема практической работы №31. Создание отчета по результатам оценки модели, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно выполнять оценку готовой модели искусственного интеллекта и оформлять результаты в виде отчёта.

Задание(я):

1. Подготовить рабочую папку, разместить в ней файл модели (model.joblib) и тестовый набор данных (test.csv).

2. Загрузить модель и тестовые данные, разделить данные на признаки и целевую переменную.

3. Выполнить предсказание модели на тестовом наборе, вычислить метрики точности, полноты, F1-score и ROC-AUC. Сохранить метрики в файл metrics.csv.

4. Построить графики: кривую ROC и матрицу ошибок (confusion matrix). Сохранить графики как roc.png и cm.png.

5. Сформировать отчёт, включив описание задачи, используемые данные, методы оценки, таблицу метрик и сохранённые графики.

Методические указания по ходу выполнения работы:

Создайте отдельную папку проекта и поместите в неё файлы model.joblib и test.csv.

Для загрузки модели используйте библиотеку joblib, а для чтения CSV – pandas.

Разделите таблицу test.csv на признаки (X) и целевую переменную (y).

С помощью методов scikit-learn рассчитайте требуемые метрики и запишите их в CSV-файл metrics.csv (заголовки: metric,value).

Для построения ROC-кривой и матрицы ошибок используйте matplotlib; сохраните графики функцией plt.savefig('имя.png').

В отчёте опишите каждый шаг, добавьте скриншоты таблицы метрик и графиков, сформируйте выводы о качестве модели.

Все файлы (исходный код .py, model.joblib, test.csv, metrics.csv, roc.png, cm.png, отчёт) упакуйте в один ZIP-архив.

Отчёт в формате ODT или PDF: титульный лист, цель работы, описание данных, методика оценки, таблица метрик, изображения графиков (roc.png, cm.png), выводы и список приложений.

Сдача: ZIP-архив с кодом, данными, результатами и отчётом отправить по e-mail преподавателю не позднее конца занятия.

Тема практической работы №32. Проектирование системы с интеграцией искусственного интеллекта, объем часов 4

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного создания, обучения и сохранения простой модели машинного обучения и подготовки её к дальнейшей интеграции в программные решения.

Задание(я):

1. Подготовка рабочего окружения и установка необходимых библиотек (pip install scikit-learn matplotlib pandas joblib). Сохранить список установленных пакетов в файл requirements.txt.

2. Загрузка открытого датасета Iris, предварительный анализ данных и разделение на обучающую и тестовую выборки. Сохранить полученные CSV-файлы train.csv и test.csv.

3. Обучение классификатора (например, LogisticRegression) на обучающей выборке. Сохранить обученный объект модели в файл model.joblib.

4. Оценка качества модели: вычисление метрик точности, построение матрицы ошибок и графика зависимости точности от параметра C. Сохранить метрики в файл metrics.csv и график в файл accuracy_plot.png.

5. Экспорт предсказаний модели для тестовой выборки в CSV-файл predictions.csv.

6. Оформление отчёта о выполненной работе и упаковка всех артефактов (исходный код .py, CSV-файлы, графики, модель, requirements.txt) в архив.

Методические указания по ходу выполнения работы:

Для задания 1 создайте виртуальное окружение, активируйте его и выполните установку перечисленных пакетов. Сохраните список пакетов, выполнив команду pip freeze > requirements.txt.

В задании 2 загрузите датасет Iris с помощью библиотеки scikit-learn, преобразуйте его в DataFrame pandas, проведите базовый анализ (просмотр первых строк, описательную статистику). Разделите данные на обучающую (80%) и тестовую (20%) части, сохраните каждую часть в отдельный CSV-файл (train.csv, test.csv).

В задании 3 обучите выбранный классификатор на train.csv, используя стандартные параметры. После обучения сохраните модель в файл model.joblib с помощью функции joblib.dump.

В задании 4 вычислите метрику точности на тестовой выборке, постройте матрицу ошибок и график зависимости точности от значения гиперпараметра C (проведите несколько запусков с разными C). Сохраните метрики в metrics.csv, а график — в accuracy_plot.png (используйте plt.savefig).

В задании 5 примените сохранённую модель к данным test.csv, получите предсказанные классы и сохраните их вместе с реальными метками в predictions.csv.

В задании 6 подготовьте отчёт, включив титульный лист, цель работы, описание каждого задания, скриншоты кода и результатов (графика, таблиц). Сохраните отчёт в формате ODT или PDF. Упакуйте в ZIP-архив все файлы: .py скрипт, requirements.txt, train.csv, test.csv, model.joblib, metrics.csv, predictions.csv, accuracy_plot.png и отчёт.

Отчёт: ODT или PDF, структура: титульный лист; цель практической работы; последовательное описание выполненных заданий; скриншоты кода и результатов (графики, таблицы); выводы о качестве модели и возможных путях интеграции в систему.

Сдача: подготовьте ZIP-архив со всеми перечисленными файлами и отправьте его по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №33. Создание интерфейса для работы с моделью искусственного интеллекта, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного создания пользовательского интерфейса для взаимодействия с готовой моделью искусственного интеллекта

Задание(я):

1. Подготовка окружения: установить Python, необходимые библиотеки (pip install scikit-learn, pip install pandas, pip install matplotlib, pip install tkinter-dnd2)

2. Загрузка готовой модели: загрузить предобученную модель (например, классификатор Iris, сохранённый в файле model.joblib) и проверить её работу в консоли

3. Разработка простого графического интерфейса: создать форму с полями ввода признаков, кнопкой «Получить предсказание» и областью вывода результата; реализовать обработку нажатия кнопки, вызов модели и вывод результата

4. Сохранение артефактов и подготовка отчёта: сохранить код в файл interface.py, сохранить модель (model.joblib), сохранить скриншот окна интерфейса (screenshot.png), экспортировать любые промежуточные данные в CSV, собрать всё в ZIP-архив

Методические указания по ходу выполнения работы:

В задании 1 используйте командную строку для установки пакетов; проверьте их наличие командой python -c "import sklearn" и т.п.

В задании 2 загрузите файл модели с помощью joblib.load и выполните предсказание на тестовом наборе, результаты выведите в консоль.

В задании 3 создайте GUI-приложение, разместив элементы управления (Entry, Button, Label) в окне; привяжите кнопку к функции, которая считывает ввод, преобразует его в массив и передаёт модели; результат отобразите в метке.

В задании 4 сохраните все файлы (interface.py, model.joblib, screenshot.png, any_data.csv) в одну папку, упакуйте её в архив ZIP. В отчёте укажите структуру папки, краткое описание выполненных шагов, скриншот интерфейса и выводы.

Формат сдачи: архив ZIP, содержащий код, модель, скриншоты и отчёт в формате ODT или PDF.

Отчёт должен включать титульный лист, цель работы, описание каждого задания, скриншоты интерфейса, выводы о работе модели и списка файлов в архиве. Оформление – ODT или PDF.

Сдать архив ZIP по электронной почте преподавателя не позднее конца учебного дня.

Тема практической работы №34. Взаимодействие модели искусственного интеллекта с базой данных информационной системы, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно интегрировать готовую модель искусственного интеллекта с реляционной базой данных, выполнять предсказания на данных из БД и сохранять результаты.

Задание(я):

1. Подготовка окружения и установка необходимых пакетов.

2. Создание и заполнение SQLite-базы примерными данными

3. Загрузка готовой модели (датасет Iris) и её сохранение в файл.

4. Реализация скрипта, читающего данные из БД, применяющего модель и записывающего предсказания в новую таблицу.

5. Построение и сохранение графика распределения предсказаний.

6. Подготовка отчёта и упаковка материалов.

Методические указания по ходу выполнения работы:

В задании 1 установите Python 3.x, создайте виртуальное окружение и выполните pip install scikit-learn, pandas, matplotlib, joblib, sqlite3 (встроен в стандартную библиотеку). Сохраните список установленных пакетов в файл requirements.txt.

В задании 2 используйте утилиту sqlite3 для создания файла базы data.db. Создайте таблицу raw_data с колонками, соответствующими признакам датасета Iris, и заполните её несколькими записями (можно импортировать CSV-файл с несколькими строками). Сохраните CSV-файл с исходными данными в папку data/.

В задании 3 загрузите предобученную модель классификации (например, LogisticRegression, обученную на Iris) с помощью scikit-learn, сохраните её в файл model.joblib при помощи joblib.dump. Поместите файл модели в папку models/.

В задании 4 создайте Python-скрипт integrate.py, который открывает data.db, читает данные из raw_data в DataFrame, применяет загруженную модель к этим данным, формирует предсказания и записывает их в новую таблицу predictions (колонки id и predicted_class). Сохраните скрипт в корневой каталог проекта.

В задании 5 в том же скрипте (или отдельном файле plot.py) постройте гистограмму распределения предсказаний с помощью matplotlib и сохраните её в файл predictions.png (plt.savefig('predictions.png')).

В задании 6 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание выполненных заданий, скриншоты кода и вывода, описание полученных файлов, выводы. Все файлы (скрипты, база, модель, график, requirements.txt, отчёт) упакуйте в архив ZIP.

Отчёт: ODT или PDF, титульный лист, цель, пошаговое описание заданий, скриншоты выполнения (консоль, результаты запросов, график), список файлов и выводы.

Сдача: архив ZIP с кодом, базой, моделью, графиком, requirements.txt и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №35. Тестирование взаимодействия компонентов информационной системы с моделью искусственного интеллекта, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного тестирования взаимодействия компонентов информационной системы с готовой моделью искусственного интеллекта.

Задание(я):

1. Подготовить простой сервис (Flask) – загрузить готовую модель (датасет Iris) и реализовать эндпойнт /predict, принимающий параметры признаков и возвращающий предсказание. Сохранить код в файл api.py.

2. Создать клиентский скрипт, который формирует запрос к эндпойнту /predict, получает ответ и сохраняет полученные предсказания в CSV-файл results.csv. Сохранить код в файл client.py.

3. Написать набор автоматических тестов (pytest) для проверки корректности работы эндпойнта: статус-код, тип возвращаемого значения, соответствие ожидаемому предсказанию для известного входа. Сохранить тесты в файл test_api.py и выполнить их, получив файл отчёта test_report.txt.

4. Подготовить отчёт о проделанной работе, включив скриншоты работы сервиса, примеры запросов/ответов, результаты тестов и выводы. Сохранить отчёт в формате ODT или PDF.

Методические указания по ходу выполнения работы:

Для реализации сервиса установите необходимые библиотеки: pip install flask scikit-learn joblib.

В api.py реализуйте загрузку модели из файла model.joblib (созданной заранее) и настройку маршрута /predict, который принимает параметры через GET-запрос.

В client.py импортируйте библиотеку requests, сформируйте запрос к http://127.0.0.1:5000/predict с необходимыми параметрами, полученный JSON-ответ запишите в results.csv (используйте pandas или csv).

Для тестов установите pytest: pip install pytest. В test_api.py опишите функции проверки статуса, структуры ответа и корректности предсказания для фиксированного набора входных данных.

Запустите тесты командой pytest > test_report.txt, убедитесь, что все тесты прошли успешно.

Отчёт оформите согласно шаблону: титульный лист, цель, список заданий, скриншоты (рабочего окна Flask, вывода client.py, фрагмент test_report.txt), выводы.

Соберите все файлы (api.py, client.py, test_api.py, model.joblib, results.csv, test_report.txt, отчёт) в один ZIP-архив.

Отчёт: ODT или PDF, титульный лист, цель работы, описание выполненных заданий, скриншоты работы сервиса и клиента, таблица результатов (results.csv), фрагмент отчёта тестов, выводы и оценка выполненных задач.

Сдача: ZIP-архив с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №36. Настройка API для работы с моделью искусственного интеллекта в информационной системе, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного развертывания простого веб-API на Flask для обращения к предобученной модели искусственного интеллекта и интеграции её в информационную систему.

Задание(я):

1. Установить необходимые библиотеки (Flask, scikit-learn, joblib). Сохранить список зависимостей в файл requirements.txt.

2. Сгенерировать и сохранить предобученную модель (например, классификатор на датасете Iris) в файл model.joblib в папку models/.

3. Создать Flask-приложение с endpoint /predict, которое загружает модель из model.joblib, принимает POST-запрос с JSON-данными и возвращает предсказание. Сохранить код в файл app.py.

4. Запустить сервер, выполнить тестовые запросы к API с помощью curl или аналогичного инструмента, сохранить скриншоты запросов/ответов и лог работы в файлы requests.txt и responses.txt.

5. Сформировать архив с исходным кодом, моделью, зависимостями, логами и скриншотами.

Методические указания по ходу выполнения работы:

Для задания 1 используйте команду pip install flask scikit-learn joblib и запишите её в requirements.txt.

Для задания 2 создайте скрипт train_model.py, обучите модель на датасете Iris и сохраните её через joblib.dump в models/model.joblib.

Для задания 3 в файле app.py реализуйте Flask-приложение, загрузите модель из models/model.joblib, определите маршрут /predict, обработайте входные данные в формате JSON и верните предсказание в виде JSON.

Для задания 4 запустите приложение командой python app.py, выполните несколько запросов curl -X POST -H "Content-Type: application/json" -d '{"features": [5.1, 3.5, 1.4, 0.2]}' http://127.0.0.1:5000/predict, сохраните выводы в файлы requests.txt и responses.txt, сделайте скриншоты терминала.

Для задания 5 упакуйте все файлы (requirements.txt, train_model.py, app.py, models/, requests.txt, responses.txt, скриншоты) в один ZIP-архив.

Формат сдачи: архив ZIP, содержащий код, модель, зависимости, логи и скриншоты, а также отчёт в формате ODT или PDF.

Отчёт оформляется в ODT или PDF, содержит титульный лист, цель работы, список выполненных заданий, описания выполненных шагов, скриншоты запросов/ответов, выводы о работе API. В конце указывается ссылка на архив с материалами.

Сдача: отправить готовый ZIP-архив по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №37. Интеграция модели искусственного интеллекта в информационную систему с веб-интерфейсом, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно интегрировать готовую модель искусственного интеллекта в веб-приложение и запускать её через локальный веб-интерфейс.

Задание(я):

1. Подготовка рабочего окружения. Установить Python, pip, библиотеки Flask, scikit-learn, joblib, pandas, matplotlib. Сохранить список установленных пакетов в файл requirements.txt.

2. Загрузка предобученной модели и создание API. Скачать готовый файл модели (например, model.joblib) из учебных материалов, разместить в папке project/. Написать скрипт api.py, реализующий endpoint /predict, который принимает JSON-запрос, применяет модель и возвращает результат. Сохранить скрипт в проекте.

3. Разработка простого веб-интерфейса и проверка. Создать HTML-страницу index.html с формой ввода параметров, отправляющей запрос к /predict через JavaScript (fetch). При получении ответа вывести предсказание на страницу. Сохранить страницу в папке project/. Сделать скриншот работающего интерфейса и сохранить как screenshot.png.

Методические указания по ходу выполнения работы:

В задании 1 используйте pip install для установки требуемых библиотек и запишите их версии в requirements.txt.

В задании 2 разместите файл модели в корне проекта, реализуйте API в отдельном .py файле, проверьте его запуск командой python api.py и убедитесь, что endpoint отвечает статусом 200.

В задании 3 разместите HTML-файл и связанные скрипты в той же папке, откройте его в браузере, выполните запрос к локальному серверу и сделайте скриншот результата.

Все результаты (requirements.txt, model.joblib, api.py, index.html, screenshot.png) упакуйте в один каталог project/.

Формат сдачи: ZIP-архив, содержащий каталог project/ и отчёт в формате ODT или PDF.

Отчёт ODT или PDF: титульный лист, цель работы, описание каждого задания, пути и имена сохранённых файлов, скриншот работы веб-интерфейса, выводы.

Сдать ZIP-архив с проектом и отчётом по электронной почте преподавателя не позднее конца занятия.

Тема практической работы №38. Оптимизация взаимодействия информационной системы с моделью искусственного интеллекта для обработки данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно разрабатывать и оптимизировать взаимодействие информационной системы с моделью искусственного интеллекта, включая обучение модели, создание API-сервиса и измерение производительности.

Задание(я):

1. Подготовить набор данных Iris, обучить простую модель классификации (LogisticRegression), сохранить модель в файл model.joblib.

2. Реализовать Flask-API с эндпоинтом /predict, который загружает сохранённую модель и возвращает предсказание для полученного JSON-запроса. Сохранить код в файле api.py.

3. Провести измерения времени отклика API для одиночного и пакетного запросов (по 10 запросов), сохранить результаты в CSV (metrics.csv) и построить график зависимости времени от количества запросов, сохранить как plot.png.

4. Оформить отчёт с титульным листом, описанием выполненных шагов, скриншотами работы API (Postman или curl) и вставкой графика plot.png.

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку scikit-learn и функцию train_test_split для разделения данных, обучите модель и сохраните её командой joblib.dump.

В задании 2 создайте виртуальное окружение, установите Flask (pip install flask) и необходимые пакеты, напишите скрипт, который при старте загружает model.joblib и реализует POST-метод /predict, принимающий массив признаков в формате JSON.

В задании 3 с помощью утилиты curl отправьте запросы к локальному серверу, измеряя время выполнения (опция -w "%{time_total}"). Запишите результаты в CSV, постройте график с помощью matplotlib и сохраните его функцией plt.savefig('plot.png').

В задании 4 подготовьте отчёт в формате ODT или PDF: титульный лист, цель, пошаговое описание, скриншоты запросов и ответов, график plot.png, выводы о производительности.

Формат сдачи: архив ZIP, содержащий директорию с кодом (api.py, train.py), файл модели model.joblib, CSV-файл metrics.csv, изображение plot.png и готовый отчёт.

Отчёт: ODT или PDF, титульный лист, цель работы, описание каждого задания, скриншоты выполнения запросов (curl/Postman), график plot.png, таблица с метриками, выводы и рекомендации по дальнейшей оптимизации.

Сдача: собрать все файлы в архив ZIP и отправить по электронной почте преподавателю не позднее окончания занятия.

Тема практической работы №39. Автоматизация бизнес-процессов с использованием искусственного интеллекта в информационной системе, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного применения предобученной модели искусственного интеллекта для автоматизации простого бизнес-процесса, включая подготовку данных, интеграцию модели, сохранение результатов и документирование работы.

Задание(я):

1. Подготовка рабочего окружения и загрузка тестового набора данных (пример: датасет Iris). Результат: файл data.csv.

2. Загрузка предобученной модели (например, LogisticRegression из scikit-learn) и её применение к загруженным данным. Результат: файл predictions.csv.

3. Автоматизация бизнес-процесса: написать скрипт, который читает data.csv, использует модель для предсказания и сохраняет результаты вместе с исходными данными. Результат: файл output.csv.

4. Визуализация результатов: построить график распределения предсказаний (например, столбчатая диаграмма классов) и сохранить его как PNG. Результат: файл predictions.png.

5. Сохранение артефактов: сохранить модель с помощью joblib, собрать весь код в отдельные .py файлы и упаковать их вместе с полученными данными и графиками в ZIP-архив.

Методические указания по ходу выполнения работы:

Для выполнения всех заданий используйте Python 3 и библиотеку scikit-learn; установите необходимые пакеты командой pip install scikit-learn pandas matplotlib joblib.

Задание 1: создайте папку project, в ней файл data.csv, в котором разместите набор признаков из выбранного датасета. Сохраните файл в формате CSV.

Задание 2: в отдельном файле model_load.py импортируйте модель LogisticRegression, загрузите её (можно воспользоваться готовой предобученной моделью из scikit-learn) и выполните предсказание по данным из data.csv. Сохраните предсказания в файл predictions.csv.

Задание 3: в файле automate_process.py реализуйте последовательность: чтение data.csv → предобучение модели (если необходимо) → предсказание → объединение исходных данных и предсказаний → запись в output.csv.

Задание 4: в файле plot_results.py постройте столбчатую диаграмму количества объектов в каждом предсказанном классе, используйте matplotlib, сохраните график командой plt.savefig('predictions.png'). Добавьте полученный PNG в отчёт.

Задание 5: сохраните обученную (или предобученную) модель в файл model.joblib с помощью joblib.dump. Скопируйте все .py файлы, CSV-файлы и PNG-изображение в одну папку и упакуйте её в архив ZIP.

Формат сдачи: архив ZIP, содержащий папку с кодом (.py), данными (.csv), моделью (.joblib), графиком (.png) и отчётом (ODT или PDF).

Формат отчёта: ODT или PDF. Титульный лист (название работы, ФИО, группа, дата). Далее – описание каждого задания, скриншоты выполнения (например, вывод терминала, содержимое CSV, график predictions.png). В конце – выводы о полученных результатах и о процессе автоматизации.

Сдача: отправить готовый ZIP-архив по электронной почте преподавателю не позднее окончания занятия.

Тема практической работы №40. Тестирование модели искусственного интеллекта в реальном времени в информационной системе, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного тестирования готовой модели искусственного интеллекта в реальном времени через веб-интерфейс информационной системы.

Задание(я):

1. Подготовить простой веб-сервис (Flask) для обслуживания запросов к предобученной модели. Результат: файл app.py и сохранённая модель (model.joblib).

2. Реализовать клиентский скрипт, отправляющий запросы к сервису, измеряющий время отклика, сохраняющий результаты в CSV и визуализирующий их графиком. Результат: файл client.py, файл latency.csv, график latency.png.

Методические указания по ходу выполнения работы:

В задании 1 установите необходимые пакеты (pip install flask joblib). Создайте проектную папку, в ней файл app.py. В файле импортируйте Flask, загрузите готовую модель (например, модель классификации Iris, сохранённую в формате joblib). Определите эндпоинт /predict, принимающий POST-запрос с JSON-данными и возвращающий предсказание. Запустите приложение в режиме отладки на локальном порту 5000. Сохраните файл app.py и модель (model.joblib) в корень проекта.

В задании 2 установите пакеты (pip install requests pandas matplotlib). Создайте файл client.py. В нём реализуйте цикл из 20 запросов к локальному эндпоинту /predict, передавая случайные входные данные из датасета Iris. Для каждого запроса фиксируйте время начала и окончания, вычисляйте

задержку в миллисекундах. Сохраняйте результаты (номер запроса, задержка) в CSV-файл latency.csv. После завершения постройте линейный график зависимости номера запроса от задержки, подпишите оси, укажите заголовок, и сохраните график в файл latency.png. Поместите client.py, latency.csv и latency.png в проектную папку.

Формат сдачи: архив ZIP, содержащий все исходные файлы (app.py, model.joblib, client.py, latency.csv, latency.png) и отчёт.

Отчёт оформляете в ODT или PDF: титульный лист, цель работы, описание выполненных заданий, скриншоты работы сервера и вывода клиентского скрипта, график latency.png, выводы о производительности.

Отчёт: ODT или PDF, титульный лист, цель, пошаговое описание заданий, скриншоты консольного вывода сервера и клиента, график latency.png, таблица результатов из CSV, выводы и рекомендации.

Сдача: упаковать всё в ZIP-архив и отправить по email преподавателю не позже конца занятия.

МДК 03.02 Интеграция искусственного интеллекта в информационные системы

Тема практической работы №1. Проектирование информационной системы с интеграцией искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного проектирования простой информационной системы с интеграцией готовой модели искусственного интеллекта, включая подготовку данных, обучение модели, её сохранение и создание API-интерфейса для получения предсказаний.

Задание(я):

1. Определить требования к системе и составить схему её компонентов (30 мин). Результат оформить в виде блок-схемы и сохранить как diagram.png.

2. Подготовить данные (датасет Iris), выполнить их загрузку и базовую предобработку. Сохранить полученный набор в CSV-файл data_preprocessed.csv.

3. Обучить простую модель (например, DecisionTreeClassifier), оценить её точность и сохранить обученную модель. Сохранить модель в файл model.joblib, график зависимости точности от параметра max_depth – в файл accuracy.png.

4. Реализовать небольшое веб-приложение на Flask, которое загружает сохранённую модель и предоставляет эндпоинт /predict для получения предсказаний. Код разместить в файле app.py.

5. Протестировать API с помощью curl, зафиксировать ответы и сохранить их в файл predictions.csv.

Методические указания по ходу выполнения работы:

В задании 1 используйте любой графический редактор (LibreOffice Draw, draw.io) для построения схемы компонентов: клиент, сервер API, модуль модели, хранилище данных.

В задании 2 загрузите датасет Iris с помощью pandas, выполните базовую очистку (удаление пропусков) и разделите на признаки и целевую переменную; сохраните полученный CSV-файл в папку data/.

В задании 3 установите необходимые библиотеки командой pip install scikit-learn joblib matplotlib; обучите модель, оцените её точность на тестовой части, постройте график зависимости точности от параметра max_depth и сохраните его через plt.savefig('accuracy.png'); сохраните модель с помощью joblib.dump в файл model.joblib.

В задании 4 установите Flask (pip install flask); создайте файл app.py, в котором реализуйте загрузку модели из model.joblib, определите маршрут /predict, принимающий параметры через GET-запрос и возвращающий предсказание в формате JSON; запустите приложение локально.

В задании 5 выполните запрос к API с помощью команды curl, получив предсказание для произвольного набора признаков; результат сохраните в файл predictions.csv, включая входные данные и полученный класс; сделайте скриншот вывода curl и включите его в отчёт.

Формат сдачи: архив ZIP, содержащий папку src/ с исходными .py-файлами, папку data/ с CSV-файлами, папку results/ с изображениями (diagram.png, accuracy.png, скриншот curl), файл report.odt (или pdf) и файл requirements.txt с перечнем установленных пакетов.

Формат отчёта: ODT или PDF. Структура отчёта – титульный лист (название работы, ФИО, группа), краткое описание целей, последовательность выполненных заданий с указанием времени, блоки с результатами: схемы, графики, скриншоты запросов, таблицы предсказаний, выводы о работе системы.

Сдача: собрать указанные файлы в архив ZIP и отправить по email преподавателя до окончания занятия.

Тема практической работы №2. Построение модели информационной системы с интеграцией искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного построения, обучения и сохранения простой модели искусственного интеллекта, а также её интеграции в информационную систему для получения предсказаний.

Задание(я):

1. Подготовить набор данных Iris, разделить его на обучающую и тестовую части. Сохранить полученные CSV-файлы (train.csv, test.csv).

2. Обучить классификационную модель (Logistic Regression) на обучающем наборе. Сохранить обученную модель в файл model.joblib.

3. Оценить качество модели на тестовой части, построить график зависимости точности от параметра regularization и сохранить его как accuracy.png.

4. Реализовать простой скрипт-интерфейс (CLI) для загрузки модели и получения предсказания по введённым пользователем значениям признаков. Сохранить скрипт как predict.py.

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для чтения исходного набора Iris и функции train_test_split из scikit-learn. Экспортируйте полученные части при помощи DataFrame.to_csv('имя.csv', index=False).

В задании 2 импортируйте LogisticRegression из scikit-learn, обучите её на данных из train.csv и сохраните модель с помощью joblib.dump(model, 'model.joblib').

В задании 3 рассчитайте метрику accuracy на тестовом наборе, создайте график (например, matplotlib) зависимости accuracy от значения C (параметра регуляризации) и сохраните изображение командой plt.savefig('accuracy.png').

В задании 4 создайте скрипт predict.py, который загружает модель из model.joblib, принимает от пользователя значения четырёх признаков (через input) и выводит предсказанный класс. Сохраните файл в корневой директории проекта.

Формат сдачи: один ZIP-архив, содержащий папку с кодом (.py файлы), CSV-файлы, сохранённую модель, график, а также отчёт в формате ODT или PDF.

Отчёт должен включать: титульный лист, цель работы, описание каждого задания, скриншоты выполнения (например, вывод терминала predict.py, график accuracy.png), выводы о качестве модели. Оформление – ODT или PDF.

Сдача: упаковать все материалы в ZIP-архив и отправить по электронной почте преподавателю не позже конца занятия.

Тема практической работы №3. Тестирование взаимодействия компонентов информационной системы с искусственным интеллектом, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного тестирования взаимодействия компонентов информационной системы с готовой моделью искусственного интеллекта.

Задание(я):

1. Подготовка рабочего окружения и установка необходимых пакетов (Flask, scikit-learn, pytest, joblib). Сохранить список пакетов в файл requirements.txt.

2. Создание простого Flask-сервиса, который загружает готовую модель (например, модель классификации Iris) и предоставляет эндпоинт /predict, принимающий JSON-запрос и возвращающий предсказание. Сохранить код в файл api.py.

3. Реализация клиентского скрипта, который формирует запрос к эндпоинту /predict и выводит полученный результат. Сохранить код в файл client.py.

4. Написание набора тестов с использованием pytest: проверка корректности ответа сервиса, проверка обработки некорректных данных, проверка доступности эндпоинта. Сохранить тесты в файл test_api.py.

5. Запуск тестов, сохранение результатов выполнения (например, в файл test_results.xml) и построение простого графика покрытия тестов (сохранить как coverage.png).

6. Подготовка отчёта: титульный лист, описание выполненных заданий, скриншоты работы сервиса и клиента, выводы, ссылки на файлы результатов. Сохранить отчёт в формате ODT или PDF.

7. Упаковка всех файлов (api.py, client.py, test_api.py, requirements.txt, test_results.xml, coverage.png, отчёт) в архив ZIP для сдачи.

Методические указания по ходу выполнения работы:

Для всех заданий используйте Python 3.x и свободно доступные библиотеки, устанавливаемые через pip.

Код каждого задания сохраняйте в отдельный .py файл с указанным именем.

Графики сохраняйте функцией plt.savefig('coverage.png').

Модель сохраняйте и загружайте через joblib.dump и joblib.load.

Тестовые результаты сохраняйте командой pytest --junitxml=test_results.xml.

Отчёт оформляйте в LibreOffice Writer, включайте скриншоты выполнения API и клиента, а также изображение графика покрытия.

В конце сформируйте ZIP-архив, содержащий все .py файлы, requirements.txt, test_results.xml, coverage.png и готовый отчёт.

Сдача: отправьте полученный ZIP-архив преподавателю по указанному email до конца занятия.

Отчёт в формате ODT или PDF: титульный лист, краткое описание каждого задания, скриншоты работы API и клиентского скрипта, изображение графика coverage.png, выводы о результатах тестирования.

Сдать: архив ZIP с кодом, зависимостями, результатами тестов, графиками и отчётом отправить по email преподавателя до конца занятия.

Тема практической работы №4. Настройка связей между базой данных и моделью искусственного интеллекта в информационной системе, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного связывания базы данных с обученной моделью ИИ в простом приложении

Задание(я):

1. Установить Python и необходимые библиотеки (pandas, scikit-learn, sqlalchemy). Создать SQLite-базу и таблицу для входных данных. Сохранить скрипт установки в файл setup.py.

2. Обучить простую модель (например, классификатор на датасете Iris) с использованием scikit-learn. Сохранить обученный объект модели в файл model.joblib.

3. Реализовать скрипт, который читает данные из SQLite, делает предсказание моделью и записывает результат в отдельную таблицу БД. Сохранить скрипт в файл predict.py и экспортировать таблицу с результатами в CSV.

4. Сформировать отчёт: титульный лист, описание выполненных шагов, скриншоты терминала и содержимого файлов, таблицу результатов. Сохранить отчёт в формате ODT или PDF.

Методические указания по ходу выполнения работы:

В задании 1 выполните установку через pip, создайте файл базы data.db и таблицы input_data, predictions; код сохраните в setup.py.

В задании 2 обучите модель на датасете Iris, сохраните её с помощью joblib.dump; файл модели назовите model.joblib.

В задании 3 скрипт predict.py должен подключаться к data.db, извлекать строки из input_data, получать предсказания, записывать их в predictions и экспортировать эту таблицу в results.csv.

В задании 4 подготовьте отчёт согласно структуре: титульный лист → цель работы → описание каждого задания → скриншоты → выводы. Сохраните в ODT или PDF.

Все файлы (setup.py, predict.py, model.joblib, results.csv, отчёт) упакуйте в один ZIP-архив для сдачи.

Отчёт в ODT или PDF: титульный лист, цель, пошаговое описание заданий, скриншоты терминала/IDE, таблица результатов, выводы.

Сдать ZIP-архив с кодом, моделью, данными и отчётом по электронной почте преподавателя до конца занятия.

Тема практической работы №5. Оптимизация работы искусственного интеллекта в структуре информационной системы, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного оптимизирования простых моделей машинного обучения и их интеграции в информационную систему.

Задание(я):

1. Подготовить данные: загрузить открытый датасет Iris, разделить его на обучающую и тестовую выборки. Сохранить полученные CSV-файлы (train.csv, test.csv).

2. Обучить базовую модель классификации (например, LogisticRegression) на обучающих данных, оценить её точность на тестовой выборке и сохранить метрики в файл metrics.txt.

3. Провести простую оптимизацию гиперпараметров модели с помощью GridSearchCV, выбрать лучшую конфигурацию, переобучить модель, сохранить оптимизированную модель в файл model.joblib и построить график зависимости точности от значений гиперпараметра; график сохранить в файл accuracy.png.

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для чтения датасета и функции train_test_split из scikit-learn; результат сохраните в указанные CSV-файлы в папке data.

В задании 2 импортируйте нужный классификатор из scikit-learn, обучите его на train.csv, предскажите метки для test.csv, вычислите точность (accuracy) и запишите значение в metrics.txt в папке results.

В задании 3 определите диапазон значений гиперпараметра (например, параметр C для LogisticRegression), запустите GridSearchCV, выберите лучшую модель, переобучите её на всех данных, сохраните объект модели с помощью joblib в файл model.joblib в папке models, постройте график зависимости точности от значения C с помощью matplotlib и сохраните его как accuracy.png в папке results.

Все исходные файлы кода (.py) разместите в корневой директории проекта, структуру папок оформите так: data/, models/, results/, scripts/.

Формат сдачи: архив ZIP, содержащий папки с данными, моделями, результатами, скриптами и отчёт.

Формат отчёта: ODT или PDF. Титульный лист (наименование работы, ФИО, группа, дата). Описание каждого задания, скриншоты кода и результатов, таблица с метриками, график accuracy.png, выводы о проведённой оптимизации.

Сдача: архив ZIP отправить по email преподавателю до конца занятия.

Тема практической работы №6. Визуализация взаимодействия элементов информационной системы с искусственным интеллектом, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно визуализировать взаимодействие компонентов информационной системы с моделью искусственного интеллекта, используя Python и открытые библиотеки визуализации.

Задание(я):

1. Сформировать таблицу взаимодействий (CSV) между элементами системы и ИИ-моделью.

2. Реализовать скрипт на Python, который читает CSV, строит граф взаимосвязей (networkx) и отображает его (matplotlib).

3. Сохранить полученный граф в файл PNG, экспортировать построенный граф в формат GraphML, оформить отчёт с описанием выполнения и скриншотами.

Методические указания по ходу выполнения работы:

В задании 1 создайте файл interactions.csv, где каждая строка содержит: source, target, тип_взаимодействия. Сохраните файл в папку data/.

В задании 2 создайте скрипт visualize.py. В начале укажите необходимые библиотеки (pip install networkx matplotlib pandas). Считайте CSV из папки data/, постройте неориентированный граф, задайте разные цвета/стили для типов взаимодействий, отобразите подписи узлов. Сохраните граф в файл diagram.png (plt.savefig('diagram.png')). Сохраните структуру графа в файл interactions.graphml (networkx.write_graphml).

В задании 3 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание входных данных, шаги выполнения, скриншот сохранённого графика (diagram.png), выводы. В

отдельный архив включите: папку data/ с CSV, файл visualize.py, diagram.png, interactions.graphml и готовый отчёт.

Формат сдачи: ZIP-архив, содержащий всё перечисленное, отправить преподавателю по указанному email до конца занятия.

Отчёт: ODT или PDF, титульный лист, цель, описание данных, последовательность шагов, скриншот diagram.png, выводы и список приложенных файлов.

Сдача: ZIP-архив с кодом, данными и отчётом, отправить по email преподавателя до конца занятия.

Тема практической работы №7. Обучение моделей искусственного интеллекта для обработки данных в информационной системе, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного обучения, оценки и сохранения простой модели машинного обучения на открытом датасете, а также визуализации метрик и подготовки отчёта.

Задание(я):

1. Подготовка окружения и загрузка данных.

2. Предобработка данных и разбиение на обучающую и тестовую выборки.

3. Обучение модели, оценка качества и сохранение артефактов.

4. Визуализация результатов, экспорт предсказаний и оформление отчёта.

Методические указания по ходу выполнения работы:

Для задания 1 установите Python 3, библиотеки pandas, scikit-learn, matplotlib и joblib (pip install pandas scikit-learn matplotlib joblib). Сохраните весь написанный код в файл train_model.py.

В файле train_model.py импортируйте необходимые модули, загрузите открытый датасет Iris (из sklearn.datasets) в переменную DataFrame и сохраните его в файл data.csv (pd.DataFrame.to_csv).

Для задания 2 в том же скрипте выполните предобработку: удалите лишние столбцы, при необходимости преобразуйте категориальные признаки в числовые, выполните масштабирование признаков (StandardScaler). Затем разберите данные на обучающую и тестовую выборки (train_test_split) и сохраните индексы разбиения в файлы train_idx.npy и test_idx.npy (numpy.save).

В задании 3 обучите классификатор (например, LogisticRegression) на обучающих данных, получите предсказания для тестовой выборки, вычислите метрики точности и F1-score. Сохраните обученную модель в файл model.joblib (joblib.dump). Сохраните метрики в файл metrics.txt (текстовый файл).

Также постройте матрицу ошибок (confusion matrix) с помощью matplotlib, отобразите её в виде тепловой карты и сохраните график в файл confusion_matrix.png (plt.savefig). Экспортируйте предсказания вместе с реальными метками в CSV файл predictions.csv.

В задании 4 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание выполненных заданий, скриншоты кода (из train_model.py), скриншот графика confusion_matrix.png, таблица метрик из metrics.txt, выводы. Вложите в отчёт все созданные файлы (data.csv, train_idx.npy, test_idx.npy, model.joblib, predictions.csv, confusion_matrix.png, metrics.txt).

Соберите все файлы проекта (директорию с кодом, данные и отчёт) в архив ZIP. В архиве должна быть папка src с файлом train_model.py, папка data с CSV и NPY-файлами, папка results с моделью, графиками и метриками, а также файл report.odt (или report.pdf).

Отчёт оформляется в ODT или PDF, содержит титульный лист, цель, последовательное описание четырёх заданий, скриншоты кода, скриншот графика confusion_matrix.png, таблицу метрик, выводы и список использованных файлов.

Сдача: упаковать проект в архив ZIP и отправить по e-mail преподавателю не позднее конца занятия.

Тема практической работы №8. Тестирование модели искусственного интеллекта на реальных данных информационной системы, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно тестировать готовую модель искусственного интеллекта на реальных данных информационной системы и документировать результаты.

Задание(я):

1. Подготовить набор реальных данных (CSV) для тестирования модели.

2. Сформировать тестовые сценарии с использованием pytest.

3. Запустить тесты, собрать метрики (accuracy, precision, recall) и сохранить их в CSV.

4. Визуализировать метрики на графиках и сохранить изображения.

5. Оформить отчёт и упаковать артефакты.

Методические указания по ходу выполнения работы:

В задании 1 загрузите файл CSV в pandas DataFrame и сохраните его как data.csv в папке data.

В задании 2 создайте файл test_model.py, где опишите функции-тесты, проверяющие предсказания модели на подготовленном наборе.

В задании 3 выполните pytest, вывод метрик запишите в файл metrics.csv в папке results.

В задании 4 с помощью matplotlib постройте графики метрик, сохраните их как accuracy.png, precision.png, recall.png в папке results.

В задании 5 подготовьте отчёт в формате ODT или PDF: титульный лист, описание выполненных заданий, скриншоты кода, таблицу metrics.csv и графики. Сохраните отчёт как report.odt.

Формат сдачи: архив ZIP, содержащий папки data, src (исходный код), results (CSV и PNG), и файл report.odt.

Отчёт ODT или PDF: титульный лист, краткое описание целей, пошаговое описание выполнения заданий, скриншоты кода, таблица метрик из metrics.csv, графики метрик, выводы.

Сдать архив ZIP с указанными файлами по email преподавателя до конца занятия.

Тема практической работы №9. Анализ данных в информационной системе с использованием искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного анализа данных, обучения и оценки простой модели искусственного интеллекта в Python, включая подготовку данных, визуализацию и сохранение артефактов.

Задание(я):

1. Подготовка данных: загрузить открытый датасет Iris, выполнить предобработку (удалить пропуски, привести типы). Сохранить чистый набор в CSV (data_clean.csv).

2. Исследовательский анализ: построить гистограммы распределения признаков и матрицу корреляций, сохранить графики в PNG (histograms.png, correlation.png).

3. Обучение модели: разделить данные на обучающую и тестовую выборки, обучить классификатор LogisticRegression, сохранить обученную модель с помощью joblib (model.joblib).

4. Оценка и визуализация результатов: вычислить метрики точности, построить ROC-кривую, сохранить метрики в CSV (metrics.csv) и график ROC в PNG (roc.png).

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для загрузки и предобработки данных, сохраните результат через DataFrame.to_csv.

В задании 2 используйте matplotlib и seaborn для построения графиков, каждый график сохраняйте функцией plt.savefig.

В задании 3 примените scikit-learn: функции train_test_split, LogisticRegression, joblib.dump для сохранения модели.

В задании 4 рассчитайте метрики accuracy_score, classification_report, roc_auc_score, постройте ROC-кривую с помощью matplotlib, сохраните результаты в указанные файлы.

Все скрипты разместите в отдельных .py файлах (prepare_data.py, explore.py, train_model.py, evaluate.py) в корневой папке проекта.

Формат сдачи: архив ZIP, содержащий папку с кодом, данными (исходный и чистый CSV), сохранённые графики, модель и отчёт.

Отчёт в формате ODT или PDF. Структура: титульный лист, цель работы, описание выполнения каждого задания, скриншоты выводов и сохранённых графиков, таблица метрик, выводы о качестве модели.

Сдача: отправить архив ZIP по электронной почте преподавателя не позднее конца занятия.

Тема практической работы №10. Создание отчета по производительности информационной системы с интеграцией искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного измерения и анализа производительности информационной системы, интегрированной с готовой моделью искусственного интеллекта, а также оформления итогового отчёта.

Задание(я):

1. Подготовка рабочей среды (установить Python, необходимые пакеты, загрузить готовую модель и пример датасета).

2. Запуск готовой модели на тестовых данных и фиксирование времени выполнения запросов.

3. Сбор системных метрик (загрузка CPU, использование памяти, время отклика) во время работы модели.

4. Визуализация собранных метрик: построить графики, сохранить их в файлы PNG.

5. Оформление итогового отчёта: описание выполненных шагов, скриншоты, интерпретация полученных графиков и выводы.

Методические указания по ходу выполнения работы:

В задании 1 создайте отдельную папку проекта, в ней файл requirements.txt с перечнем пакетов (например, scikit-learn, pandas, matplotlib, psutil) и выполните их установку через pip.

В задании 2 запустите скрипт, который загружает готовую модель (например, предобученный классификатор) и последовательно обрабатывает несколько примеров из выбранного датасета (Iris, MNIST и т.п.). Время выполнения каждого запроса фиксируйте и сохраняйте в CSV-файл.

В задании 3 одновременно с запуском модели собирайте системные показатели с помощью утилиты psutil: среднюю загрузку процессора, пиковое и текущее использование оперативной памяти. Результаты также сохраняйте в CSV-файл.

В задании 4 загрузите полученные CSV-файлы, постройте графики зависимости времени отклика от нагрузки CPU и памяти, а также гистограмму распределения времени отклика. Сохраните каждый график функцией plt.savefig('имя.png').

В задании 5 подготовьте отчёт в формате ODT или PDF: титульный лист, краткое описание целей и задач, последовательность выполнения, скриншоты терминала и графиков, таблицы с метриками, выводы о производительности системы. Все файлы (скрипты .py, CSV-данные, PNG-графики) упакуйте в один ZIP-архив.

Формат сдачи: архив ZIP, содержащий код, данные, графики и готовый отчёт, отправить преподавателю по указанному email до конца занятия.

Отчёт ODT или PDF: титульный лист, описание целей и задач, пошаговое изложение выполнения, скриншоты консоли, таблицы метрик, графики PNG с подписями, раздел «Выводы».

Сдача: упаковать все материалы в ZIP-архив и отправить по email преподавателю до окончания 4-часового занятия.

Тема практической работы №11. Интеграция моделей искусственного интеллекта в интерфейс информационной системы, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного интегрирования готовой модели машинного обучения в пользовательский интерфейс веб-приложения и оформлять результаты работы.

Задание(я):

1. Подготовка рабочей среды: установка Python, создание виртуального окружения, установка необходимых пакетов (Flask, scikit-learn, joblib, pandas, matplotlib).

2. Загрузка и проверка готовой модели: загрузить предобученный файл модели (например, logistic_regression_iris.pkl), выполнить предсказание на нескольких тестовых строках, сохранить результаты в CSV-файл.

3. Реализация веб-интерфейса: создать Flask-приложение с одной страницей, содержащей форму ввода признаков, обработать запрос, вывести предсказание модели, добавить вывод графика (например, распределения вероятностей) и сохранить его в файл PNG.

4. Тестирование и документирование: запустить приложение, проверить корректность работы формы, сделать скриншоты окна браузера с вводом и результатом, сохранить скриншоты, собрать весь код, модели и полученные файлы в один каталог.

5. Формирование отчёта и упаковка: подготовить отчёт в формате ODT или PDF согласно шаблону, включить титульный лист, описание выполненных заданий, скриншоты, ссылки на сохранённые файлы (график PNG, CSV с предсказаниями, сохранённый файл модели, исходный код). Упаковать всё в ZIP-архив.

Методические указания по ходу выполнения работы:

Создайте отдельную папку проекта и внутри неё поддиректории: data (для CSV), models (для .pkl), static (для PNG-графиков), templates (для HTML-форм).

Для задания 2 используйте модуль joblib для загрузки модели и pandas для формирования таблицы предсказаний, затем сохраните её функцией to_csv('data/predictions.csv').

В задании 3 создайте файл app.py, импортируйте Flask, загрузите модель из models, определите маршрут '/' с методом GET для отображения формы и маршрут '/predict' с методом POST для обработки ввода, сформируйте предсказание и отобразите его на странице. Сгенерированный график сохраняйте командой plt.savefig('static/probabilities.png').

Все графики, полученные в ходе выполнения, сохраняйте в формате PNG в папку static и указывайте их путь в отчёте.

Все исходные скрипты (app.py, load_model.py и т.п.) сохраняйте с расширением .py в корневой каталог проекта.

Отчёт оформляйте согласно следующей структуре: титульный лист; цель работы; описание каждого задания; скриншоты интерфейса (вставьте изображения из папки static); таблица предсказаний (скриншот CSV-файла); выводы о работе приложения.

После завершения соберите в один ZIP-архив следующие элементы: каталог проекта с кодом, моделью и данными; отчёт в ODT/PDF; скриншоты и графики.

Формат отчёта: ODT или PDF. Структура отчёта – титульный лист, цель работы, описание каждого задания, скриншоты веб-страницы с вводом и результатом, таблица предсказаний (скриншот CSV), график вероятностей (PNG), выводы и заключение.

Сдача: упаковать все материалы в ZIP-архив и отправить по электронной почте преподавателю до окончания занятия.

Тема практической работы №12. Автоматизация процессов в информационной системе с использованием искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного построения, сохранения и автоматизации простого модели машинного обучения для решения задачи классификации в информационной системе.

Задание(я):

1. Загрузка открытого датасета Iris, предобработка данных и разделение на обучающую и тестовую выборки. Сохранить подготовленные наборы в файлы train.csv и test.csv.

2. Обучить классификатор (например, LogisticRegression) на обучающей выборке, оценить точность на тестовой выборке, построить график зависимости точности от параметра regularization и сохранить график как accuracy.png.

3. Сохранить обученную модель в файл model.joblib с помощью joblib.

4. Создать скрипт-автоматизацию (predict.py), который загружает модель, принимает на вход новые данные из CSV-файла (input.csv), выполняет предсказание и сохраняет результаты в output.csv.

5. Подготовить отчёт, включающий описание каждого шага, скриншоты выполнения скриптов, таблицы с метриками и сохранённый график.

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для чтения датасета и разделения данных; результаты сохраните в указанные CSV-файлы.

В задании 2 обучите модель, вычислите метрику точности, постройте график зависимости точности от значения гиперпараметра regularization (используйте matplotlib) и сохраните его как accuracy.png.

В задании 3 сохраните полученную модель с помощью joblib.dump в файл model.joblib в корневой папке проекта.

В задании 4 реализуйте скрипт predict.py, который импортирует joblib, загружает model.joblib, читает входные данные из input.csv, делает предсказание и записывает результаты в output.csv.

В задании 5 подготовьте отчёт в формате ODT или PDF: титульный лист, цель работы, пошаговое описание заданий, скриншоты консольного вывода, таблицу с метриками, график accuracy.png и выводы.

Формат сдачи: архив ZIP, содержащий папку с исходным кодом (.py файлы), все CSV-файлы, сохранённый график, файл модели model.joblib и готовый отчёт.

Отчёт должен включать: титульный лист (название работы, ФИО, группа, дата); цель работы; описание каждого задания; скриншоты выполнения скриптов (консольный вывод); таблицу с метрикой точности; график accuracy.png; выводы и заключение. Оформление – стандартный стиль ODT/PDF, шрифт Times New Roman 12, межстрочный интервал 1,5.

Сдача: упаковать все материалы в архив ZIP и отправить по email преподавателю до окончания занятия.

Тема практической работы №13. Анализ бизнес-процессов для внедрения искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно проводить анализ бизнес-процессов и формировать обоснованные предложения по внедрению решений искусственного интеллекта

Задание(я):

1. Выбор простого бизнес-процесса (например, обработка клиентского заказа). Описание процесса в виде текста. Сохранить в файл process.txt.

2. Моделирование выбранного процесса с помощью блок-схемы в LibreOffice Draw. Сохранить схему в файл process.png.

3. Идентификация проблемных точек и возможностей применения ИИ. Оформить таблицу проблем и предложений в LibreOffice Calc и экспортировать в CSV (issues.csv).

4. Формирование выводов и рекомендаций по внедрению ИИ. Составить отчет в LibreOffice Writer, включив описание процесса, схему, таблицу и выводы. Сохранить как report.odt.

Методические указания по ходу выполнения работы:

В задании 1 опишите последовательность действий бизнес-процесса в виде простого текста, указывая ключевые этапы и участников.

В задании 2 откройте LibreOffice Draw, создайте блок-схему, отражающую этапы процесса, соедините их стрелками, подпишите каждый блок, затем сохраните изображение в формате PNG под именем process.png.

В задании 3 откройте LibreOffice Calc, создайте таблицу с двумя столбцами: «Проблема» и «Возможное ИИ-решение». Заполните таблицу выявленными проблемами и предложенными ИИ-инструментами, затем экспортируйте её в CSV-файл issues.csv.

В задании 4 подготовьте отчет, включив титульный лист, описание выбранного процесса, вставьте схему process.png, добавьте таблицу из issues.csv (можно скопировать в документ), сформулируйте выводы и рекомендации. Сохраните документ в формате ODT под именем report.odt.

Формат сдачи: создайте ZIP-архив, включив файлы process.txt, process.png, issues.csv и report.odt.

Отчёт должен быть в формате ODT или PDF и включать: титульный лист (название работы, ФИО, группа), описание процесса, скриншот (вставку) схемы process.png, таблицу проблем и предложений (из issues.csv) и раздел «Выводы и рекомендации».

Сдача: упаковать перечисленные файлы в архив ZIP и отправить по email преподавателю до конца занятия.

Тема практической работы №14. Моделирование бизнес-процесса с применением искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять полный цикл моделирования бизнес-процесса с применением искусственного интеллекта: от подготовки данных до обучения, оценки и сохранения модели.

Задание(я):

1. Подготовка рабочего окружения (установить Python, необходимые библиотеки, создать проектную папку).

2. Сбор и подготовка данных: загрузить открытый датасет "Iris" (или любой CSV с продажами), выполнить очистку и разбить на обучающую и тестовую выборки.

3. Обучение модели: построить простую регрессионную/классификационную модель с использованием scikit-learn.

4. Оценка модели и визуализация результатов: вычислить метрики, построить графики зависимости предсказаний от истинных значений, сохранить графики.

5. Сохранение артефактов и формирование отчёта: сохранить модель, данные, графики, исходный код и оформить отчёт.

Методические указания по ходу выполнения работы:

Для задания 1 создайте папку проекта, в ней подпапки data, models, plots, src. Установите библиотеки командой pip install scikit-learn pandas matplotlib joblib.

В задании 2 загрузите датасет в папку data, выполните предобработку (удаление пропусков, кодирование категориальных признаков). Сохраните полученный чистый набор в CSV (data/cleaned.csv). Разделите данные на train и test, сохраните разбиения в отдельные CSV-файлы (data/train.csv, data/test.csv).

В задании 3 в подпапке src создайте скрипт train_model.py, который читает подготовленные CSV-файлы, обучает выбранный алгоритм, выводит основные параметры модели. Сохраните обученную модель в файл models/model.joblib с помощью joblib.dump.

В задании 4 в скрипте evaluate.py загрузите модель и тестовый набор, вычислите метрики (accuracy, precision, recall или MSE). Постройте график сравнения предсказанных и истинных значений, сохраните его в папку plots (plots/metrics.png) командой plt.savefig('plots/metrics.png').

В задании 5 подготовьте отчёт в формате ODT или PDF: титульный лист, краткое описание каждого задания, скриншоты кода (из src), таблицы с метриками, включите сохранённые графики. Упакуйте в архив ZIP папки src, data, models, plots и готовый отчёт.

Формат сдачи: один ZIP-архив, содержащий все перечисленные папки и отчёт, отправить по email преподавателю до конца занятия.

Формат отчёта: ODT или PDF. Структура: титульный лист (название работы, ФИО, группа), вводная часть (цель, описание бизнес-процесса), разделы по каждому заданию (описание выполненных действий, скриншоты кода, таблицы с результатами), графики (вставки изображений), выводы и заключения.

Сдача: архив ZIP с папками src, data, models, plots и готовым отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №15. Оптимизация существующего бизнес-процесса с использованием искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного применения готовой модели машинного обучения для оптимизации простого бизнес-процесса (прогноз спроса) и умения сохранять артефакты работы.

Задание(я):

1. Подготовка данных (CSV). Сохранить файл data.csv.

2. Обучение линейной регрессионной модели. Сохранить модель в файл model.pkl.

3. Оценка качества модели и визуализация результатов. Сохранить график prediction.png и таблицу предсказаний predictions.csv.

4. Формирование отчёта и упаковка проекта. Сохранить отчёт в ODT/PDF и собрать архив.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки командой: pip install pandas scikit-learn matplotlib joblib.

Создайте рабочую папку проекта и в ней файл data.csv с двумя столбцами: "день" (число) и "продажи" (число). Заполните его простыми тестовыми данными (например, 30 строк).

В Python-скрипте load_data.py загрузите data.csv, разделите данные на признаки и целевую переменную, затем разбейте их на обучающую и тестовую выборки (например, 80/20).

В скрипте train_model.py обучите модель LinearRegression из scikit-learn на обучающей части, после чего сохраните её в файл model.pkl при помощи joblib.dump.

В скрипте evaluate.py загрузите модель из model.pkl, выполните предсказание на тестовой выборке, вычислите метрику MAE (Mean Absolute Error) и постройте график сравнения реальных и предсказанных значений. Сохраните график в prediction.png и таблицу предсказаний (день, реальное, предсказанное) в predictions.csv.

Подготовьте отчёт, включив: титульный лист, цель работы, краткое описание каждого задания, скриншоты кода (по желанию), таблицу предсказаний, график prediction.png, выводы о качестве модели и её применимости для оптимизации процесса.

Сохраните отчёт в формате ODT или PDF, проверьте наличие всех файлов (data.csv, load_data.py, train_model.py, evaluate.py, model.pkl, prediction.png, predictions.csv, отчёт).

Упакуйте всю папку проекта в архив ZIP.

Отчёт: ODT или PDF, содержит титульный лист, цель, описание заданий, скриншоты/фрагменты кода, график prediction.png, таблицу predictions.csv и выводы.

Сдача: отправить архив ZIP с полным проектом (код, данные, модели, графики, отчёт) по email преподавателя до конца занятия.

Тема практической работы №16. Тестирование искусственного интеллекта для автоматизации бизнес-операций, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного тестирования готовой модели искусственного интеллекта с использованием pytest

Задание(я):

1. Подготовка среды: установить необходимые библиотеки, загрузить готовую модель и тестовый набор данных. Результаты сохранить в файлы requirements.txt и data.csv.

2. Создать тестовый модуль: написать набор тестов, проверяющих корректность предсказаний модели на известных примерах. Сохранить код в файл test_model.py.

3. Запустить тесты и собрать отчёт: выполнить pytest с генерацией XML-отчёта и вывести сводку в консоль. Сохранить отчёт в файл test_report.xml.

4. Проанализировать результаты: построить график распределения ошибок (например, количество неверных предсказаний по классам) и сохранить его в файл errors.png.

5. Оформить итоговый отчёт: включить описание выполненных шагов, скриншоты вывода тестов и графика, выводы о качестве модели.

Методические указания по ходу выполнения работы:

Для задания 1 установите библиотеки pytest, scikit-learn и pandas командой pip install pytest scikit-learn pandas; создайте файл requirements.txt с перечислением установленных пакетов; загрузите готовую модель (например, файл model.joblib) и сохраните её в рабочую папку; подготовьте тестовый набор данных в формате CSV (data.csv).

В задании 2 создайте новый файл test_model.py; импортируйте модель, загрузите данные из data.csv; сформулируйте минимум три тестовых

функции, проверяющих предсказания модели на известных входных значениях и ожидаемых результатах; используйте ассерты для сравнения.

В задании 3 выполните команду pytest --junitxml=test_report.xml; убедитесь, что в консоли отображается количество пройденных и проваленных тестов; файл test_report.xml будет содержать детализированный отчёт.

В задании 4 при помощи pandas подсчитайте количество ошибок по каждому классу; постройте столбчатую диаграмму с помощью matplotlib; сохраните график командой plt.savefig('errors.png').

В задании 5 оформите документ отчёта в формате ODT или PDF: титульный лист, краткое описание целей, пошаговый перечень выполненных заданий, скриншоты вывода pytest, вставьте график errors.png, сделайте выводы о надёжности модели. Все файлы (requirements.txt, data.csv, model.joblib, test_model.py, test_report.xml, errors.png, отчёт) упакуйте в один ZIP-архив.

Отчёт — ODT или PDF, содержит титульный лист, описание целей, список выполненных заданий, скриншоты вывода pytest, график errors.png и раздел выводов.

Сдать: архив ZIP с кодом, данными, моделью, тестовым отчётом, графиком и готовым отчётом по электронной почте преподавателя не позднее окончания занятия.

Тема практической работы №17. Применение искусственного интеллекта для прогнозирования и аналитики в бизнесе, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять полный цикл построения, оценки и сохранения простой модели машинного обучения для прогнозирования бизнес-показателей.

Задание(я):

1. Подготовить рабочее окружение, установить необходимые библиотеки и создать структуру проекта.

2. Загрузить открытый датасет (например, датасет Iris, переименованный в бизнес-контекст), выполнить предварительный анализ и сохранить подготовленные данные в CSV.

3. Обучить простую модель (линейная регрессия или дерево решений) на подготовленных данных, оценить её точность, построить график зависимости метрики от параметра и сохранить модель и график.

4. Оформить отчёт, собрать все артефакты (скрипт .py, CSV, модель .joblib, график .png) в ZIP-архив и подготовить к сдаче.

Методические указания по ходу выполнения работы:

Установите библиотеки: pip install pandas scikit-learn matplotlib joblib.

Создайте папку проекта, внутри подпапки data, models, figures и scripts.

В скрипте scripts/prepare_data.py загрузите выбранный датасет, выполните базовую очистку и преобразование, разделите на обучающую и тестовую части, сохраните подготовленный набор в data/prepared.csv.

В скрипте scripts/train_model.py загрузите подготовленные данные, обучите выбранный алгоритм, вычислите метрику точности (например, R2 или accuracy), постройте график зависимости метрики от гиперпараметра, сохраните модель в models/model.joblib и график в figures/metric_plot.png.

В скрипте scripts/main.py объедините предыдущие шаги, чтобы запускать процесс последовательно.

Сохраните все результаты: CSV-файл, модель, график и скрипты.

Подготовьте отчёт в формате ODT или PDF: титульный лист, краткое описание каждого задания, скриншоты вывода консоли, вставьте сохранённый график, сделайте выводы о качестве модели.

Соберите всё в один ZIP-архив (например, project.zip) согласно структуре проекта.

Формат сдачи: архив ZIP, содержащий код, данные, модель, график и отчёт.

Отчёт ODT или PDF: титульный лист; цель работы; описание каждого задания; скриншоты консольного вывода; вставленный график metric_plot.png; выводы о точности модели и возможных улучшениях.

Сдача: отправить архив ZIP по электронной почте преподавателю до конца занятия.

Тема практической работы №18. Разработка автоматизированных отчетов с использованием искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять процесс создания автоматизированного отчета, используя готовую модель искусственного интеллекта для анализа текста и визуализации результатов.

Задание(я):

1. Подготовить набор данных: создать CSV-файл с несколькими текстовыми записями (например, отзывы клиентов). Сохранить как data.csv.

2. Установить необходимые библиотеки (pandas, matplotlib, scikit-learn, joblib) и загрузить готовую предобученную модель (например, модель классификации тональности, сохранённую в файле model.joblib).

3. Применить модель к текстовым данным, получить предсказания, построить график распределения предсказанных классов и сохранить его как distribution.png.

4. Сформировать автоматический отчёт: включить таблицу с исходными данными и предсказаниями, добавить график, сгенерировать текстовое резюме (например, средний процент положительных/отрицательных отзывов). Сохранить отчёт в формате PDF (report.pdf).

Методические указания по ходу выполнения работы:

Для задания 1 создайте в LibreOffice Calc таблицу с колонками 'id' и 'text', заполните минимум 10 строк, экспортируйте в CSV (data.csv).

Для задания 2 выполните pip install pandas matplotlib scikit-learn joblib. Поместите файл model.joblib в рабочую папку и загрузите его в Python-скрипте без написания кода в методических указаниях.

Для задания 3 обработайте каждую строку CSV-файла моделью, получите метки классов, сохраните предсказания в новый CSV (predictions.csv). Постройте гистограмму распределения меток, сохраните как distribution.png (plt.savefig('distribution.png')).

Для задания 4 создайте Python-скрипт, который собирает таблицу с оригинальными текстами и предсказанными метками, вставляет график distribution.png, рассчитывает процент положительных/отрицательных отзывов и сохраняет всё в PDF (используйте библиотеку reportlab или аналог). Сохраните PDF как report.pdf.

Формат сдачи: архив ZIP, содержащий файлы data.csv, predictions.csv, distribution.png, report.pdf, все .py скрипты и файл отчёта (ODT или PDF) с титульным листом, описанием выполненных заданий, скриншотами результатов и выводами.

Отчёт: ODT или PDF, титульный лист, краткое описание цели и задач, пошаговое описание выполнения, скриншоты таблиц, графика и итогового PDF-отчёта, выводы о работе модели.

Сдать: упаковать всё перечисленное в ZIP-архив и отправить по email преподавателю не позднее окончания двухчасового занятия.

Тема практической работы №19. Создание сценария искусственного интеллекта для управления бизнес-процессами, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного использования готовой модели ИИ для автоматизации простого бизнес-процесса

Задание(я):

1. Установить необходимые Python-пакеты (scikit-learn, pandas, matplotlib, joblib).

2. Скачать и разместить в рабочей папке готовый файл модели (model.joblib) и пример входных данных (requests.csv).

3. Написать скрипт (script.py), который загружает модель, читает requests.csv, применяет модель к каждому запросу и сохраняет результаты в predictions.csv.

4. В том же скрипте построить график метрик (например, матрица ошибок) и сохранить его как metrics.png.

5. Подготовить отчёт, включив описание выполненных шагов, скриншоты работы скрипта, таблицу predictions.csv и график metrics.png.

Методические указания по ходу выполнения работы:

Для установки пакетов используйте команду pip install в терминале.

Разместите файлы model.joblib и requests.csv в одной папке с вашим script.py.

В script.py импортируйте необходимые библиотеки, загрузите модель функцией joblib.load, загрузите данные через pandas.read_csv, выполните предсказание методом модели, сохраните результаты методом pandas.to_csv (имя файла predictions.csv).

Для построения графика используйте matplotlib: создайте фигуру, отобразите метрику и сохраните её командой plt.savefig('metrics.png').

Отчёт оформите в формате ODT или PDF: титульный лист, цель работы, описание каждого задания, скриншоты выполнения (терминал, результаты CSV, график), выводы.

Все файлы (script.py, model.joblib, requests.csv, predictions.csv, metrics.png, отчёт) упакуйте в один архив ZIP.

Отчёт: ODT или PDF, титульный лист, цель, пошаговое описание, скриншоты, таблица predictions.csv, график metrics.png, выводы.

Сдача: архив ZIP с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №20. Интеграция искусственного интеллекта в систему управления проектами, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного создания, обучения и интеграции простой модели машинного обучения в приложение управления проектами

Задание(я):

1. Подготовка учебного набора данных о задачах проекта (CSV).

2. Обучение регрессионной модели предсказания длительности задачи и сохранение модели (joblib).

3. Интеграция обученной модели в скрипт-утилиту управления проектами, генерация предсказаний, построение графика сравнения фактических и предсказанных длительностей, экспорт результатов в CSV и PNG.

Методические указания по ходу выполнения работы:

В задании 1 создайте файл tasks.csv с колонками: id, тип задачи, приоритет, оценка сложности (число), фактическая длительность (часы). Заполните минимум 15 строк примерами типовых задач проекта.

Сохраните файл tasks.csv в папку data/ проекта.

В задании 2 загрузите данные из CSV, разделите их на признаки и целевую переменную, обучите простую линейную регрессию (scikit-learn). Сохраните обученную модель в файл model.pkl (используйте joblib.dump). Поместите файл модели в папку models/.

В задании 3 напишите скрипт project_ai.py, который загружает model.pkl, принимает на вход новые записи задач (можно добавить их в отдельный CSV), выводит предсказанную длительность, сохраняет предсказания в файл predictions.csv (папка results/).

Постройте график фактической vs предсказанной длительности с помощью matplotlib, сохраните его как duration_comparison.png в папку results/. Убедитесь, что график сохраняется через plt.savefig().

Все исходные файлы (.py, .csv, .pkl, .png) упакуйте в один архив ZIP. В архив также включите файл отчёта.

Формат сдачи: архив ZIP, содержащий папки data/, models/, results/ и файл report.odt (или report.pdf).

Отчёт должен быть оформлен в ODT или PDF, содержать титульный лист, разделы: 1) Цель работы, 2) Описание выполнения каждого задания, 3) Скриншоты кода и терминала, 4) Таблицу с результатами предсказаний, 5) График duration_comparison.png, 6) Выводы о качестве модели.

Сдача: отправить готовый ZIP-архив по электронной почте преподавателю не позднее конца учебного занятия.

Тема практической работы №21. Автоматизация задач на основе искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять полный цикл разработки и автоматизации простого решения на основе искусственного интеллекта: от загрузки и предварительного анализа данных до обучения модели, её оценки, сохранения и применения в автоматическом скрипте.

Задание(я):

1. Загрузить открытый датасет Iris, выполнить базовый разведочный анализ и сохранить полученные статистические таблицы и графики в файлы CSV и PNG.

2. Обучить простой классификатор (например, LogisticRegression) на подготовленных данных, оценить его качество с помощью метрик точности и матрицы ошибок, сохранить обученную модель в файл.

3. Реализовать автоматический скрипт, который загружает сохранённую модель, принимает на вход новые признаки, выводит предсказание, сохраняет результат в CSV и генерирует график распределения предсказаний.

4. Подготовить отчёт и упаковать все материалы в архив для сдачи.

Методические указания по ходу выполнения работы:

Для выполнения заданий установите необходимые библиотеки командой pip install scikit-learn pandas matplotlib joblib.

Задание 1: создайте Python-скрипт (data_analysis.py), в котором загрузите датасет Iris через pandas, вычислите основные статистики (среднее, минимум, максимум) и сохраните их в файл stats.csv. Постройте гистограммы признаков, сохраните графики как hist_feature1.png, hist_feature2.png и т.д. При необходимости используйте matplotlib, сохраняйте графики функцией plt.savefig('имя.png').

Задание 2: создайте скрипт (train_model.py). Разделите данные на обучающую и тестовую выборки, обучите выбранный классификатор, вычислите точность и постройте матрицу ошибок. Сохраните обученную модель в файл model.joblib с помощью joblib.dump. Выведите метрики в консоль и сохраните их в файл metrics.txt.

Задание 3: создайте скрипт (predict.py). В нём загрузите модель из model.joblib, реализуйте функцию, принимающую массив новых признаков, выводящую предсказанный класс и сохраняющую результат в predictions.csv. Сформируйте график распределения предсказанных классов и сохраните его как predictions.png.

Задание 4: подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание каждого задания, скриншоты вывода консоли, таблицы CSV (в виде вложений или изображений), графики PNG, выводы и список использованных библиотек. Все исходные .py файлы, CSV, PNG, model.joblib и отчёт упакуйте в один ZIP-архив.

Формат сдачи: архив ZIP с названием <фамилия_имя_практика21>.zip, отправить по email преподавателю до окончания занятия.

Отчёт должен содержать титульный лист (название практики, ФИО студента, группа, дата), цель работы, пошаговое описание выполненных заданий, скриншоты консольного вывода, таблицы и графики (вставленные

как изображения), раздел с выводами и рекомендациями, список использованных библиотек и команд установки.

Сдача: подготовленный ZIP-архив отправить по email преподавателю до конца занятия.

Тема практической работы №22. Анализ результатов работы искусственного интеллекта в бизнесе, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного анализа результатов готовой модели ИИ в бизнес-задаче и представления выводов в виде отчёта.

Задание(я):

1. Подготовить рабочую папку, загрузить открытый датасет "Customer churn" (CSV) и готовый предобученный файл модели (joblib). Сохранить данные в папке data.

2. С помощью Python (pandas) загрузить данные, выполнить предобработку (удалить пропуски, преобразовать категориальные признаки в числовые). Сохранить обработанный набор в файл processed.csv.

3. Применить готовую модель к обработанным данным, получить предсказания, вычислить метрики точности (accuracy, precision, recall, F1). Сохранить метрики в файл metrics.txt.

4. Построить графики: ROC-кривая и матрица ошибок (confusion matrix). Сохранить графики в файлы roc.png и confusion.png. Включить их в отчёт.

Методические указания по ходу выполнения работы:

Создайте структуру проекта: корневая папка → data, results, src, report.

В папке src разместите скрипт analysis.py, в котором последовательно реализованы пункты 2-4.

Для загрузки модели используйте pip install joblib; для построения графиков – pip install matplotlib, seaborn; для расчёта метрик – pip install scikit-learn; для работы с данными – pip install pandas.

После выполнения скрипта убедитесь, что в папке results находятся файлы processed.csv, metrics.txt, roc.png, confusion.png.

Формат сдачи: архив ZIP, содержащий папки src, data, results и отчёт в формате ODT или PDF.

Отчёт должен включать титульный лист, описание выполненных заданий, скриншоты выводов консоли, сохранённые графики и интерпретацию метрик.

Отчёт ODT или PDF: титульный лист, цель работы, список задач, описания шагов, скриншоты вывода, вставленные графики (roc.png, confusion.png), таблица метрик из metrics.txt, выводы и рекомендации.

Сдать архив ZIP с кодом, данными, результатами и отчётом по email преподавателя не позднее окончания 2-часового занятия.

Тема практической работы №23. Построение отчета о внедрении искусственного интеллекта в бизнес-процесс, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного формирования полного отчёта о внедрении готовой модели искусственного интеллекта в выбранный бизнес-процесс, включая сбор требований, подготовку данных, оценку модели и оформление результатов.

Задание(я):

1. Выбор бизнес-процесса и формулировка задачи ИИ (пример: прогноз продаж, классификация заявок). Сохранить описание в файл description.txt.

2. Поиск и загрузка открытого датасета, соответствующего задаче (например, Iris, MNIST). Сохранить набор данных в файл data.csv.

3. Обучение простой модели с использованием scikit-learn (например, LogisticRegression для классификации). Сохранить обученную модель в файл model.joblib и метрики в файл metrics.txt.

4. Подготовка отчёта: описание задачи, данные, процесс подготовки, результаты модели, графики метрик. Сохранить отчёт в формате ODT/PDF (report.odt).

Методические указания по ходу выполнения работы:

Для задания 1 составьте короткое бизнес-описание (цель, ожидаемый эффект, критерии успеха) и запишите его в description.txt.

Для задания 2 используйте pip install scikit-learn и pandas, загрузите выбранный датасет, проведите базовый анализ (размер, типы столбцов) и экспортируйте его в CSV-файл data.csv.

Для задания 3 подготовьте данные (разделите на обучающую и тестовую выборки, выполните масштабирование при необходимости), обучите модель, вычислите точность и другие метрики, сохраните модель через joblib.dump('model.joblib') и запишите метрики в metrics.txt. Постройте график зависимости метрики от параметра (например, ROC-кривая) и сохраните его функцией plt.savefig('roc.png').

Для задания 4 оформите отчёт в LibreOffice: титульный лист, разделы по каждому пункту, вставьте скриншоты кода, таблицы данных и график roc.png. Сохраните документ как report.odt (или report.pdf).

Все файлы (description.txt, data.csv, model.joblib, metrics.txt, roc.png, report.odt) упакуйте в один ZIP-архив.

Отчёт ODT или PDF: титульный лист, цель проекта, описание бизнес-процесса, используемый датасет, этапы подготовки данных, описание модели, таблица метрик, график ROC (скриншот), выводы и рекомендации по внедрению.

Сдача: ZIP-архив с перечисленными файлами отправить по email преподавателю до конца занятия.

Тема практической работы №24. Модернизация бизнес-процессов на основе аналитики искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного проведения полного цикла аналитики данных с применением готовой модели машинного обучения для улучшения бизнес-процесса

Задание(я):

1. Загрузка и первичный анализ открытого датасета «Retail Sales» (CSV). Оценить структуру, типы данных, наличие пропусков. Сохранить результаты описательной статистики в файл stats.csv.

2. Предобработка данных: очистка от пропусков, кодирование категориальных признаков, масштабирование числовых признаков. Сохранить обработанный датасет в файл cleaned.csv.

3. Обучение простой регрессионной модели (LinearRegression) для прогнозирования объёма продаж. Разделить данные на обучающую и тестовую выборки, обучить модель, сохранить её в файл model.joblib.

4. Оценка качества модели: построить график фактических vs предсказанных значений, рассчитать метрики MAE и R^2, сохранить график как plot.png и метрики в файл metrics.txt. Подготовить краткий вывод о возможности применения модели для модернизации бизнес-процесса.

Методические указания по ходу выполнения работы:

Для выполнения всех заданий используйте Python 3.x, среды разработки LibreOffice для отчёта и Git для контроля версий.

Установите необходимые библиотеки командой: pip install pandas scikit-learn matplotlib joblib.

Все скрипты сохраняйте в отдельные файлы с расширением .py, названия файлов соответствуют номерам заданий (task1.py, task2.py и т.д.).

Результаты (CSV-файлы, PNG-график, файл модели, текстовый файл метрик) сохраняйте в подпапку results внутри рабочего каталога.

Код, данные и результаты упакуйте в один архив ZIP. В корне архива разместите отчёт в формате ODT или PDF, названный report.odt (или report.pdf).

В отчёте обязательно укажите титульный лист, описание каждого задания, скриншоты вывода скриптов (консоль, таблицы, графики), а также выводы по результатам модели.

Отчёт должен включать: титульный лист (название практики, ФИО, группа, дата), краткое описание цели, пошаговое описание выполнения заданий, таблицы/скриншоты результатов (stats.csv, cleaned.csv, metrics.txt), график plot.png, выводы о применимости модели. Формат – ODT или PDF.

Сдача: архив ZIP, содержащий код, данные, результаты и отчёт, отправить по электронной почте преподавателю не позже конца 4-часового занятия.

Тема практической работы №25. Реализация алгоритма искусственного интеллекта для анализа данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно выполнять полный цикл разработки простого алгоритма искусственного интеллекта для анализа данных: загрузка набора, обучение модели, оценка качества, визуализация результатов и сохранение артефактов.

Задание(я):

1. Установить необходимые библиотеки (pip install scikit-learn matplotlib pandas joblib). Загрузить открытый датасет Iris и сохранить его в файл data.csv.

2. Написать скрипт train_model.py, в котором выполнить предобработку данных, обучить классификатор DecisionTree, оценить точность на тестовой выборке и сохранить обученную модель в файл model.joblib.

3. В том же скрипте построить график важности признаков, сохранить его как feature_importance.png, а также экспортировать таблицу с предсказаниями в predictions.csv.

4. Сформировать отчёт, включив титульный лист, описание выполненных шагов, скриншоты кода, таблицы и график, а затем упаковать всё в архив.

Методические указания по ходу выполнения работы:

Для задания 1 используйте pandas для чтения датасета и сохранения в CSV; файл разместите в папке data/.

Для задания 2 создайте отдельный каталог src/ и поместите туда скрипт train_model.py; модель сохраняйте функцией joblib.dump в каталог models/.

Для задания 3 график сохраняйте функцией plt.savefig('feature_importance.png') в каталог results/; таблицу предсказаний сохраняйте через pandas.to_csv в results/.

Отчёт оформляйте в LibreOffice Writer, сохраняйте в формате ODT или PDF; включите скриншоты файлов и графика.

Финальный архив ZIP должен содержать папки data/, src/, models/, results/ и файл отчёт.odt/pdf.

Отчёт: титульный лист, цель работы, пошаговое описание заданий, скриншоты кода (src/train_model.py), таблица predictions.csv, график feature_importance.png, выводы о качестве модели.

Сдача: упаковать всё в архив ZIP и отправить по электронному адресу преподавателя до конца занятия.

Тема практической работы №26. Обучение модели искусственного интеллекта для обработки больших данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного обучения и оценки простой модели машинного обучения на большом наборе данных, включая подготовку данных, сохранение модели и визуализацию результатов.

Задание(я):

1. Установить необходимые библиотеки, загрузить открытый датасет (например, датасет Iris или synthetically generated CSV с 100 000 строк). Сохранить исходный CSV в папку data/.

2. Выполнить предварительную обработку: очистка от пропусков, кодирование категориальных признаков, масштабирование числовых столбцов. Сохранить обработанный набор в файл processed.csv в папку data/.

3. Разделить данные на обучающую и тестовую выборки, обучить выбранный алгоритм (например, линейную регрессию или случайный лес). Сохранить обученную модель в файл model.joblib в папку models/.

4. Оценить качество модели на тестовой выборке, вычислить метрики (RMSE, R^2). Построить график зависимости метрики от размера обучающего набора, сохранить его как plot.png в папку results/. Сохранить таблицу метрик в файл metrics.csv в папку results/.

5. Подготовить отчёт, включив титульный лист, описание каждого задания, скриншоты кода и результатов, выводы. Упаковать весь проект (папки data, models, results, скрипты .py и отчёт) в архив ZIP.

Методические указания по ходу выполнения работы:

Перед началом работы установите пакеты: pip install pandas scikit-learn matplotlib joblib.

Все скрипты сохраняйте в отдельные файлы с расширением .py, например, load_data.py, preprocess.py, train.py, evaluate.py.

Результаты (CSV, PNG, joblib) сохраняйте в указанные папки и фиксируйте пути в отчёте.

В отчёте разместите скриншоты содержимого файлов, графика plot.png и таблицы metrics.csv.

Формат сдачи: архив ZIP, содержащий весь исходный код, данные, модели, графики и отчёт в формате ODT или PDF.

Отчёт ODT или PDF: титульный лист, цель работы, список выполненных заданий с описанием, скриншоты кода, таблица metrics.csv, график plot.png, выводы и рекомендации.

Сдача: подготовленный ZIP-архив отправить по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №27. Применение метода кластеризации для анализа данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного применения метода кластеризации к открытым данным, построения визуализаций результатов и сохранения артефактов (графики, модели, CSV).

Задание(я):

1. Загрузка и предварительная подготовка набора данных Iris. Сохранить очищенный набор в файл data.csv.

2. Применение алгоритма K-means (k=3) к подготовленным данным. Сохранить полученную модель в файл model.joblib и таблицу с предсказанными кластерами в file clusters.csv.

3. Визуализация кластеров на двумерном графике (первый и второй признаков). Сохранить график в файл clusters.png.

4. Формирование отчёта о выполненной работе, включающего описание выполненных шагов, скриншоты кода и полученных артефактов, а также выводы.

Методические указания по ходу выполнения работы:

Подготовьте рабочую папку проекта, в ней создайте подкаталоги: data, model, figures, report.

В задании 1 используйте библиотеку pandas для чтения набора Iris, удалите любые строки с пропусками и сохраните полученный DataFrame в data/data.csv при помощи метода to_csv.

В задании 2 установите необходимые библиотеки (scikit-learn, joblib) через pip. С помощью класса KMeans обучите модель на данных из data/data.csv, задав количество кластеров k=3. Сохраните обученную модель в model/model.joblib при помощи joblib.dump. Добавьте столбец с номерами кластеров к исходному DataFrame и экспортируйте его в data/clusters.csv.

В задании 3 постройте двумерный график рассеяния (scatter) с осями – первые два признака набора. Точки раскрасьте в соответствии с номерами кластеров из columns. Сохраните график в figures/clusters.png используя plt.savefig.

В задании 4 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, последовательность выполненных заданий, скриншоты кода (из файлов .py), изображения графика, таблицы с результатами, выводы. Сохраните отчёт в корень проекта как report.odt (или report.pdf).

Для сдачи упакуйте всю рабочую папку в архив ZIP. В архиве должны присутствовать: папки data, model, figures, report, а также файл report.odt/pdf.

Формат сдачи: отправьте получившийся ZIP-архив по электронному адресу преподавателя до конца занятия.

Отчёт оформляется в ODT или PDF. Содержание: титульный лист (название практики, ФИО, группа), цель работы, описание каждого задания, скриншоты исходного кода (файлы .py), скриншоты полученного графика (clusters.png), таблицы с результатами (clusters.csv), выводы и личные наблюдения. Все изображения вставляются в текст отчёта, а также перечисляются пути к файлам.

Сдача: собрать все файлы проекта в ZIP-архив и отправить по указанному email преподавателя не позднее окончания двухчасовой практики.

Тема практической работы №28. Применение регрессионных методов для предсказаний, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить самостоятельный цикл работы с регрессионными моделями: подготовка данных, обучение, оценка качества, визуализация результатов и сохранение модели.

Задание(я):

1. Установить окружение и загрузить датасет California Housing; сохранить исходные данные в файл data.csv.

2. Выполнить предварительный анализ: изучить структуру данных, построить гистограммы и корреляционную матрицу; сохранить графики как hist.png и corr.png.

3. Обучить простую линейную регрессию на подготовленных данных; сохранить обученную модель в файл linear_model.pkl и метрики (MAE, RMSE) в файл metrics_linear.csv.

4. Обучить модель случайного леса регрессии; сохранить модель в файл rf_model.pkl и метрики в файл metrics_rf.csv.

5. Сравнить результаты обеих моделей, построить график сравнения ошибок (error_comparison.png); оформить выводы и сохранить их в отчёт.

Методические указания по ходу выполнения работы:

Для всех заданий используйте Python 3, библиотеки pandas, matplotlib, scikit-learn, joblib; установить их командой: pip install pandas matplotlib scikit-learn joblib.

Задание 1: создайте виртуальное окружение, запустите Python-скрипт, который загружает датасет California Housing через sklearn.datasets.fetch_california_housing, преобразует его в DataFrame и сохраняет в data.csv в папке work/.

Задание 2: в скрипте проведите описательный анализ (info(), describe()), постройте гистограммы для каждого признака и тепловую карту корреляций; сохраните изображения в файлы hist.png и corr.png в папке results/.

Задание 3: разделите данные на обучающую и тестовую выборки (например, 80/20), масштабируйте признаки, обучите LinearRegression, предскажите значения для тестовой части, вычислите MAE и RMSE; результаты запишите в metrics_linear.csv, модель сериализуйте с помощью joblib.dump в файл linear_model.pkl.

Задание 4: аналогично обучите RandomForestRegressor (укажите число деревьев, например 100), сохраните модель в rf_model.pkl и метрики в metrics_rf.csv.

Задание 5: загрузите обе модели, вычислите ошибки предсказаний на тестовой выборке, постройте столбчатый график сравнения MAE и RMSE для двух моделей, сохраните как error_comparison.png; подготовьте текстовый вывод с рекомендациями по выбору модели.

Все скрипты сохраняйте в отдельные файлы с расширением .py в папке src/.

Формат сдачи: один ZIP-архив, содержащий папки src/, work/, results/ и файл отчёт (ODT или PDF).

Отчёт ODT или PDF: титульный лист (название практики, ФИО, группа), краткое описание каждого задания, скриншоты кода и выводов, вставленные изображения (hist.png, corr.png, error_comparison.png), таблицы метрик (metrics_*.csv) и раздел выводов/рекомендаций.

Сдача: упаковать всё в ZIP-архив и отправить по электронной почте преподавателю не позднее окончания учебного занятия.

Тема практической работы №29. Валидация модели искусственного интеллекта для анализа данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного проведения валидации готовой модели искусственного интеллекта на примере простого классификатора

Задание(я):

1. Загрузка и подготовка датасета Iris; разделить данные на обучающую и тестовую выборки (70/30). Сохранить полученные CSV-файлы (train.csv, test.csv).

2. Загрузка предобученной модели (например, LogisticRegression, сохранённой в файле model.joblib) и применение её к тестовой выборке. Сохранить предсказания в CSV-файл predictions.csv.

3. Вычисление метрик качества (accuracy, precision, recall, f1-score). Сохранить таблицу метрик в CSV-файл metrics.csv и построить график зависимости точности от порога вероятности (если применимо). Сохранить график как accuracy_curve.png.

4. Сохранение всех полученных артефактов (CSV-файлы, график, исходный код .py) в отдельную папку проекта.

5. Подготовка отчёта: титульный лист, описание выполненных шагов, скриншоты кода и результатов, выводы. Сохранить отчёт в формате ODT или PDF.

Методические указания по ходу выполнения работы:

Для всех шагов используйте язык Python и стандартные библиотеки: pandas, scikit-learn, matplotlib, joblib.

Установите необходимые пакеты командой: pip install pandas scikit-learn matplotlib joblib.

При загрузке датасета используйте встроенный набор Iris из библиотеки scikit-learn.

Разделение данных выполните функцией train_test_split с параметром test_size=0.3 и random_state для воспроизводимости.

Сохраните полученные наборы данных в CSV-файлы при помощи метода to_csv (index=False).

Загрузите готовую модель из файла model.joblib функцией joblib.load.

Для получения предсказаний используйте метод predict и сохраните результаты в predictions.csv.

Вычислите метрики при помощи функций classification_report и accuracy_score; сформируйте таблицу и сохраните её в metrics.csv.

Постройте график зависимости точности от порога (если модель выдаёт вероятности) с помощью matplotlib; сохраните изображение командой plt.savefig('accuracy_curve.png').

Все скрипты разместите в отдельных .py файлах (например, data_preparation.py, evaluate_model.py).

Создайте папку "submission" и поместите туда: train.csv, test.csv, predictions.csv, metrics.csv, accuracy_curve.png, все .py файлы и готовый отчёт.

Упакуйте содержимое папки "submission" в архив ZIP.

Формат сдачи: архив ZIP с кодом, данными, графиками и отчётом.

Отчёт должен включать: титульный лист (название практики, ФИО студента, группа, дата), краткое описание задачи, пошаговое описание выполнения (с указанием использованных команд), скриншоты кода и результатов (таблицы CSV, график), таблицу метрик, выводы о качестве модели. Отчёт сохраняется в формате ODT или PDF.

Сдача: отправить подготовленный ZIP-архив по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №30. Оптимизация алгоритмов искусственного интеллекта для улучшения точности принимаемых решений, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного проведения полного цикла разработки и оптимизации простой модели машинного обучения для повышения точности принимаемых решений.

Задание(я):

1. Подготовка данных и базовое обучение модели.
2. Оценка качества базовой модели.
3. Применение методов оптимизации.
4. Сравнительный анализ и визуализация результатов.
5. Сохранение артефактов и подготовка отчёта.

Методические указания по ходу выполнения работы:

В задании 1 загрузите открытый датасет Iris, разделите его на обучающую и тестовую части, выполните предварительную обработку (например, стандартизацию признаков) и обучите простую классификационную модель (логистическую регрессию). Сохраните исходный код в файл baseline_model.py.

В задании 2 оцените точность базовой модели на тестовой выборке, вычислите метрики Accuracy, Precision, Recall и F1-score. Сохраните полученные значения в файл baseline_metrics.csv.

В задании 3 проведите оптимизацию модели: попробуйте изменить гиперпараметры (например, регуляризацию для логистической регрессии) и/или заменить алгоритм на более подходящий (например, дерево решений). Для выбора лучших параметров используйте перебор (grid search) на обучающей части. Сохраните оптимизированный код в файл

optimized_model.py и модель в файл optimized_model.joblib (через joblib.dump).

В задании 4 сравните метрики базовой и оптимизированной моделей, построив графики зависимости точности от выбранных параметров. Сохраните графики в файлы accuracy_comparison.png и f1_comparison.png (plt.savefig). Включите в отчёт скриншоты этих графиков.

В задании 5 упакуйте все файлы: .py-скрипты, .csv-отчёты, .png-графики, .joblib-модель в один архив ZIP. Подготовьте отчёт в формате ODT или PDF, включив титульный лист, описание каждого задания, таблицы метрик, скриншоты графиков и выводы о влиянии оптимизации на точность.

Отчёт ODT или PDF: титульный лист (название работы, ФИО, группа, дата); краткое описание целей; последовательное описание выполненных заданий; таблицы метрик (baseline_metrics.csv, optimized_metrics.csv); скриншоты графиков (accuracy_comparison.png, f1_comparison.png); выводы о результате оптимизации; список приложений (исходный код, сохранённые модели).

Сдача: архив ZIP, содержащий все .py-файлы, .csv-отчёты, .png-графики, .joblib-модель и готовый отчёт, отправить по email преподавателю до конца занятия.

Тема практической работы №31. Применение методов классификации для анализа данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно применять методы классификации к реальному набору данных, выполнять их обучение, оценку качества и визуализацию результатов.

Задание(я):

1. Подготовка данных (30 минут). Считать открытый датасет Iris, разделить на обучающую и тестовую части, сохранить подготовленные CSV-файлы.

2. Обучение модели (45 минут). Обучить классификатор LogisticRegression из scikit-learn, сохранить обученную модель в файл model.joblib.

3. Оценка и визуализация (45 минут). Рассчитать метрики точности, построить график зависимости точности от параметра C, сохранить график в файл accuracy.png и добавить выводы в отчет.

Методические указания по ходу выполнения работы:

Для всех заданий используйте Python 3, установив необходимые пакеты командой pip install pandas scikit-learn matplotlib joblib.

В задании 1 создайте скрипт preprocessing.py, который загружает датасет Iris, делит его на train и test (примерно 70/30) и сохраняет два файла train.csv и test.csv в папку data.

В задании 2 создайте скрипт train_model.py, который читает train.csv, обучает LogisticRegression, сохраняет модель в файл model.joblib в папку models и выводит в консоль значение точности на обучающем наборе.

В задании 3 создайте скрипт evaluate.py, который загружает test.csv и модель model.joblib, вычисляет точность на тестовом наборе, строит график точности в зависимости от параметра C (используйте несколько значений C), сохраняет график как accuracy.png в папку results и заносит метрики в файл metrics.csv.

Все скрипты должны быть сохранены с расширением .py, результаты – в указанных папках. По окончании работы подготовьте отчет, включив скриншоты вывода каждого скрипта, таблицу metrics.csv и изображение accuracy.png.

Формат сдачи: архив ZIP, содержащий папки data, models, results, все .py-файлы и готовый отчёт ODT или PDF.

Отчёт ODT или PDF: титульный лист, цель работы, описание каждого задания, скриншоты вывода скриптов, таблица metrics.csv, изображение accuracy.png, выводы и рекомендации.

Сдача: архив ZIP отправить по e-mail преподавателю до конца занятия.

Тема практической работы №32. Сравнение различных алгоритмов искусственного интеллекта на одном наборе данных, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно сравнивать несколько алгоритмов искусственного интеллекта на одном наборе данных, оценивать их качество и представлять результаты в виде графиков и сохранённых моделей.

Задание(я):

1. Установить необходимые библиотеки (scikit-learn, pandas, matplotlib, joblib) и подготовить рабочую папку. Сохранить исходный набор данных Iris в файл data.csv.

2. Реализовать три простых классификатора (логистическая регрессия, k-ближайших соседей, дерево решений), обучить их на подготовленном наборе и сохранить каждую модель в файл model_<alg>.joblib.

3. Оценить качество моделей с помощью метрик точности и матрицы ошибок, вывести результаты в консоль и записать в файл metrics.txt.

4. Построить график сравнения точности моделей, сохранить его как accuracy.png, добавить скриншот в отчёт.

Методические указания по ходу выполнения работы:

Для задания 1 создайте каталог project32, внутри подпапки data и models. С помощью pandas загрузите датасет Iris из библиотеки scikit-learn и экспортируйте его в CSV (data/iris.csv).

Для задания 2 в файле model_train.py реализуйте три классификатора, используя API scikit-learn. Обучите каждый на полной выборке и сохраните модели функцией joblib.dump в папку models (model_logreg.joblib, model_knn.joblib, model_tree.joblib).

Для задания 3 в том же скрипте вычислите точность (accuracy_score) и матрицу ошибок (confusion_matrix) для каждой модели. Запишите результаты

в текстовый файл results/metrics.txt, каждый блок начинайте с названия алгоритма.

Для задания 4 в отдельном скрипте plot_accuracy.py постройте столбчатую диаграмму сравнения точностей, подпишите оси и легенду. Сохраните рисунок в файл results/accuracy.png с помощью plt.savefig.

Все исходные .py файлы разместите в корневой папке проекта, результаты – в подпапке results. Оформите отчёт в ODT или PDF, включив титульный лист, краткое описание каждого задания, скриншоты вывода консоли и графика, а также выводы о сравнении алгоритмов.

Формат сдачи: один ZIP-архив, содержащий весь каталог проекта (код, данные, модели, результаты, отчёт).

Отчёт состоит из титульного листа, раздела «Описание заданий», «Результаты выполнения» (таблицы/скриншоты вывода, график accuracy.png), «Выводы». Все изображения вставляются в документ, текст оформлен в стиле академического отчёта.

Сдача: упаковать проект в архив ZIP и отправить по email преподавателю не позднее окончания занятия.

Тема практической работы №33. Автоматизация принятия решений с помощью искусственного интеллекта, объем часов 2

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно выполнять автоматизацию принятия решений с помощью простой модели машинного обучения, включая подготовку данных, обучение модели, оценку качества и предсказание новых образцов.

Задание(я):

1. Подготовить окружение и установить необходимые библиотеки (pip install scikit-learn pandas matplotlib joblib). Сохранить список пакетов в файл requirements.txt.

2. Загрузить открытый датасет Iris, выполнить предобработку, разделить на признаки и целевую переменную, сформировать обучающую и тестовую выборки. Сохранить предобработанные данные в CSV файл preprocessed.csv.

3. Обучить простую модель DecisionTreeClassifier на тренировочных данных, сохранить обученную модель в файл model.joblib. Сохранить модель в папку models.

4. Оценить качество модели на тестовой выборке, построить график зависимости точности от глубины дерева, сохранить график в файл accuracy.png и таблицу метрик в CSV.

5. Реализовать скрипт predict.py, который загружает модель и делает предсказания для новых образцов, сохраняет результаты в predictions.csv.

Методические указания по ходу выполнения работы:

Создайте виртуальное окружение, активируйте его и выполните указанные pip install. Сохраните список установленных пакетов в requirements.txt.

С помощью pandas загрузите датасет Iris, проведите базовую очистку, разделите данные на признаки и метки, затем с помощью train_test_split сформируйте обучающую и тестовую выборки. Сохраните полученные наборы в формате CSV.

Обучите модель DecisionTreeClassifier, используя scikit-learn, и сохраните её через joblib.dump в файл models/model.joblib.

Вычислите метрики (accuracy, precision, recall) на тестовой выборке, постройте график зависимости точности от параметра max_depth с помощью matplotlib и сохраните его функцией plt.savefig('accuracy.png'). Экспортируйте таблицу метрик в CSV файл metrics.csv.

Напишите скрипт predict.py, который принимает CSV файл с новыми данными, загружает модель из models/model.joblib, выводит предсказания в консоль и сохраняет их в файл predictions.csv.

Формат сдачи: архив ZIP, содержащий папку src с .py скриптами, папку data с исходными и промежуточными CSV, папку models с моделью, папку reports с отчетом и графиками.

Отчет в ODT или PDF: титульный лист, цель работы, описание каждого задания, скриншоты консольного вывода и графиков, таблица метрик, выводы и заключения.

Сдать архив ZIP по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №34. Внедрение модели искусственного интеллекта в систему поддержки принятия решений, объем часов 4

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Научиться самостоятельно внедрять готовую модель искусственного интеллекта в простую систему поддержки принятия решений

Задание(я):

1. Подготовить рабочее окружение, установить необходимые библиотеки и загрузить открытый датасет Iris.

2. Скачать предобученную модель (например, LogisticRegression, сохранённую в файл model.joblib) и разместить её в проекте.

3. Реализовать скрипт decision_support.py, который принимает на вход параметры цветка, получает предсказание от модели и выводит рекомендацию.

4. Оценить качество модели на тестовой части датасета, построить график зависимости точности от параметра, сохранить график в файл accuracy.png.

5. Оформить отчёт, включить скриншоты работы скрипта, графика и выводов, упаковать всё в ZIP-архив.

Методические указания по ходу выполнения работы:

В задании 1 создайте виртуальное окружение, установите пакеты: pip install scikit-learn pandas matplotlib joblib. Сохраните датасет в файл iris.csv.

В задании 2 разместите файл model.joblib в той же папке, где будет находиться скрипт, и убедитесь, что он доступен для загрузки через joblib.load.

В задании 3 в файле decision_support.py реализуйте чтение входных параметров (например, через argparse или ввод с клавиатуры), загрузку модели, получение предсказания и вывод рекомендации в консоль. Сохраните файл в корень проекта.

В задании 4 разделите данные на обучающую и тестовую части, вычислите метрики (accuracy, confusion matrix), постройте график зависимости точности от порога (если применимо) и сохраните его командой plt.savefig('accuracy.png').

В задании 5 подготовьте отчёт в формате ODT или PDF: титульный лист, цель работы, описание каждого задания, скриншоты консольного вывода и графика, выводы. Сохраните отчёт как report.odt (или report.pdf).

Для сдачи упакуйте в один ZIP-архив файлы: decision_support.py, model.joblib, iris.csv, accuracy.png, report.odt (или report.pdf).

Отчёт: ODT или PDF, титульный лист, цель, описание заданий, скриншоты консольного вывода, график accuracy.png, выводы и заключения.

Сдача: архив ZIP с кодом, данными, графиками и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №35. Тестирование алгоритмов искусственного интеллекта на реальных данных, объем часов 4

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного тестирования готовых моделей искусственного интеллекта на реальных датасетах и интерпретации полученных метрик.

Задание(я):

1. Выбрать открытый датасет (например, Iris) и подготовить CSV-файл с данными.

2. Установить необходимые Python-библиотеки (scikit-learn, pandas, matplotlib, joblib).

3. Загрузить предобученную модель (например, LogisticRegression) из scikit-learn и применить её к выбранному датасету.

4. Оценить качество модели с помощью метрик (accuracy, precision, recall, f1). Сохранить метрики в CSV-файл.

5. Построить графики зависимостей (например, ROC-кривая, матрица ошибок) и сохранить их в PNG-файлы.

6. Сохранить обученную модель в файл (joblib.dump).

7. Оформить отчёт, включив описание процесса, скриншоты кода, таблицы метрик и графики.

8. Упаковать весь материал (исходный .py-файл, CSV-файлы, PNG-графики, модель и отчёт) в ZIP-архив.

Методические указания по ходу выполнения работы:

Для задания 1 создайте папку data и поместите туда CSV-файл с выбранным датасетом.

Для задания 2 используйте команду pip install scikit-learn pandas matplotlib joblib.

Для задания 3 импортируйте модель из scikit-learn, загрузите данные из CSV и выполните предсказание.

Для задания 4 вычислите метрики, сформируйте таблицу и сохраните её командой pandas.DataFrame.to_csv.

Для задания 5 постройте необходимые графики, сохраните их функцией plt.savefig('имя.png') и включите в отчёт.

Для задания 6 сохраните модель с помощью joblib.dump в файл models/model.joblib.

Для задания 7 подготовьте отчёт в формате ODT или PDF: титульный лист, цель, описание шагов, таблицы метрик, скриншоты кода, PNG-графики, выводы.

Для задания 8 создайте ZIP-архив, включающий папки code, data, models, results и файл отчёта.

Отчёт оформляется в ODT или PDF, содержит титульный лист, цель работы, пошаговое описание выполнения заданий, таблицу метрик (CSV), скриншоты кода, PNG-графики, выводы и рекомендации по улучшению модели.

Сдача: упаковать все материалы в один ZIP-архив и отправить по e-mail преподавателю до конца занятия.

Тема практической работы №36. Анализ точности и эффективности решений, принятых с использованием искусственного интеллекта, объем часов 4

У3. Настраивать процесс обучения, выбирать подходящие датасеты и корректировать параметры обучения для калибровки.

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

Цель практической работы:

Освоить навыки самостоятельного анализа точности и эффективности моделей искусственного интеллекта на примере простых классификаторов.

Задание(я):

1. Загрузка и предобработка открытого датасета Iris. Сохранить подготовленные данные в CSV (data_prepared.csv).

2. Обучить два базовых классификатора (Logistic Regression и Decision Tree) на подготовленных данных. Сохранить обученные модели в файлы (logreg.pkl, dtree.pkl).

3. Оценить модели: построить графики зависимости точности от параметров, сохранить графики (accuracy_logreg.png, accuracy_dtree.png). Сохранить таблицу метрик в CSV (metrics.csv).

4. Сравнить полученные результаты, проанализировать эффективность моделей по метрикам и времени обучения. Сохранить текстовый файл с выводами (analysis.txt).

5. Подготовить отчёт, включив титульный лист, описание выполнения каждого задания, скриншоты графиков и выводов.

Методические указания по ходу выполнения работы:

В задании 1 используйте библиотеку pandas для чтения датасета Iris и выполните базовую очистку (удаление пропусков, кодирование категориальных признаков). Экспортируйте полученный набор в CSV.

В задании 2 импортируйте необходимые модели из scikit-learn, разделите данные на обучающую и тестовую выборки (примерно 80/20), обучите модели и сохраните их с помощью joblib.

В задании 3 вычислите метрики точности, полноты и F1-score для каждой модели на тестовой выборке. Постройте графики метрик с помощью matplotlib и сохраните их в файлы PNG. Сохраните таблицу всех метрик в CSV.

В задании 4 сопоставьте полученные метрики, оцените время обучения (можно измерить через time). Опишите, какая модель более эффективна в контексте задачи, и сохраните выводы в текстовый файл.

В задании 5 оформите отчёт в ODT или PDF: титульный лист, краткое описание каждого задания, вставьте скриншоты графиков, таблицу метрик и файл analysis.txt. Сохраните отчёт как report.odt (или report.pdf).

Формат сдачи: архив ZIP, содержащий папку src с .py файлами всех скриптов, папку data с CSV-файлами, папку models с сохранёнными моделями, папку figures с PNG-графиками, файл analysis.txt и готовый отчёт.

Отчёт в ODT или PDF: титульный лист, перечень выполненных заданий, скриншоты графиков (встроенные в документ), таблица метрик, выводы из analysis.txt, заключение.

Сдача: упаковать всё в архив ZIP и отправить по электронной почте преподавателю до конца занятия.

Тема практической работы №37. Анализ кейсов этических вопросов при использовании искусственного интеллекта, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно оценивать этические аспекты применения готовых моделей искусственного интеллекта и формировать обоснованные выводы по кейсам.

Задание(я):

1. Выбрать и изучить два типовых кейса применения ИИ (например, система распознавания лиц в общественных местах и автоматизированный отбор резюме).

2. Составить в таблице CSV перечень потенциальных этических вопросов для каждого кейса (приватность, дискриминация, прозрачность, ответственность).

3. Оценить значимость каждого вопроса по шкале 1-5, добавить комментарии. Сохранить таблицу как ethics_analysis.csv.

4. Сформировать выводы: сравнить уровень этических рисков в двух кейсах, предложить рекомендации по их снижению. Оформить выводы в документе ODT/PDF.

5. Подготовить короткую презентацию (5-6 слайдов) с ключевыми результатами и разместить файлы в архиве.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 используйте открытые источники (статьи, официальные описания) и сделайте короткие заметки в собственном блокноте.

В задании 2 создайте таблицу с колонками: «Кейс», «Этический вопрос», «Оценка важности», «Комментарий». Сохраните её как ethics_analysis.csv в рабочую папку.

В задании 3 добавьте столбец «Оценка важности» и заполните его числовыми значениями; комментарии оформляйте в отдельном столбце.

В задании 4 оформите документ отчёта: титульный лист, описание кейсов, таблица с результатами, выводы и рекомендации. Сохраните как report.odt (или report.pdf).

В задании 5 используйте LibreOffice Impress: создайте слайды с заголовками «Кейс 1», «Кейс 2», «Сравнительный анализ», «Рекомендации», «Выводы». Сохраните как presentation.odp (или .pdf).

Все созданные файлы (CSV, ODT/PDF, ODP/PDF) поместите в один каталог и упакуйте в архив ZIP.

Формат сдачи: архив ZIP, содержащий код (если использовался скрипт для создания CSV), данные и отчёт.

Отчёт ODT или PDF: титульный лист (название работы, ФИО, группа), краткое описание выбранных кейсов, таблица ethics_analysis.csv (вставка как изображение), раздел «Анализ этических вопросов», выводы и рекомендации, список использованных источников. Включить скриншоты таблицы и ключевых слайдов.

Сдача: упакованный ZIP-архив отправить по e-mail преподавателю не позднее окончания двухчасового занятия.

Тема практической работы №38. Исследование правовых аспектов использования искусственного интеллекта в бизнесе, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного исследования правовых аспектов применения искусственного интеллекта в бизнес-среде и оформления полученных выводов в структурированный отчёт.

Задание(я):

1. Сформировать перечень ключевых нормативных актов и рекомендаций, регулирующих использование ИИ в бизнесе.

2. Свести полученные сведения в таблицу с указанием названия документа, даты принятия, основной сферы применения и ключевых требований.

3. На основе таблицы подготовить аналитический обзор: описать основные риски, требования к прозрачности, защите данных и ответственности.

4. Оформить результаты в отчёт: титульный лист, описание целей, методика, таблица, аналитический обзор, выводы.

Методические указания по ходу выполнения работы:

В задании 1 используйте официальные источники (законодательные сайты, публичные документы). Список документов сохраните в текстовом файле sources.txt.

В задании 2 создайте таблицу в LibreOffice Calc, заполните столбцы: «Документ», «Дата», «Сфера применения», «Ключевые требования». Сохраните её как regulations.csv.

В задании 3 на основе таблицы напишите аналитический обзор в LibreOffice Writer, опираясь на выделенные в таблице требования. Сохраните файл как analysis.odt.

В задании 4 объедините все материалы (sources.txt, regulations.csv, analysis.odt) в один документ report.odt, включив титульный лист, оглавление и разделы, перечисленные в структуре отчёта.

Формат сдачи: архив ZIP, содержащий все перечисленные файлы (report.odt, regulations.csv, sources.txt).

Отчёт ODT или PDF: титульный лист (название работы, ФИО, группа), цель и задачи, методика исследования, таблица нормативных актов, аналитический обзор, выводы, список использованных источников. Включить скриншоты таблицы и ключевых пунктов документов.

Сдача: упаковать все файлы в архив ZIP и отправить по email преподавателю до окончания занятия.

Тема практической работы №39. Анализ рисков использования искусственного интеллекта в информационных системах, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно оценивать потенциальные риски применения искусственного интеллекта в информационных системах, используя базовый анализ данных

Задание(я):

1. Ознакомиться с вводным материалом по категориям рисков ИИ (этические, правовые, технические, социальные). Результат: список рисков в виде текста, сохранить в файл risks.txt.

2. Скачать подготовленный CSV-файл «ai_incidents.csv», содержащий примеры инцидентов с указанием категории риска и даты. Результат: файл данных оставьте без изменений.

3. С помощью Python (pandas) загрузить CSV-файл, посчитать количество инцидентов по каждой категории риска и сохранить полученную таблицу в файл «risk_counts.csv».

4. Построить столбчатую диаграмму распределения количества инцидентов по категориям риска, сохранить график в файл «risk_distribution.png».

5. Сформировать выводы о наиболее часто встречающихся рисках и предложить меры снижения. Результат: текстовый документ «conclusions.txt».

6. Подготовить отчёт (ODT или PDF) со структурой: титульный лист, цель работы, описание выполненных заданий, скриншоты кода и результатов (таблица, график), выводы и список использованных источников.

Методические указания по ходу выполнения работы:

Для работы используйте Python 3, установив необходимые библиотеки командой: pip install pandas matplotlib.

Все скрипты сохраняйте в отдельные .py файлы (например, load_data.py, analysis.py, plot.py) в одной папке.

Графики сохраняйте функцией plt.savefig('risk_distribution.png') без отображения окна.

Таблицу с результатами сохраняйте через метод to_csv('risk_counts.csv', index=False).

Текстовые результаты (список рисков, выводы) сохраняйте в .txt файлах.

Все файлы (скрипты, csv-данные, график, текстовые документы) упакуйте в один архив ZIP.

Формат сдачи: архив ZIP, содержащий код, данные, результаты и отчёт в ODT или PDF.

Отчёт должен включать скриншоты кода (по одному скриншоту на каждый скрипт) и скриншоты полученных файлов (таблица, график).

Отчёт ODT или PDF: титульный лист; цель практической работы; описание выполненных заданий; скриншоты кода; скриншот таблицы risk_counts.csv; скриншот графика risk_distribution.png; выводы; список использованных источников.

Сдача: подготовить архив ZIP с перечисленными файлами и отправить по электронной почте преподавателю до конца занятия.

Тема практической работы №40. Определение зон ответственности при использовании искусственного интеллекта, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного определения и документирования зон ответственности при внедрении систем искусственного интеллекта

Задание(я):

1. Ознакомиться с нормативными источниками (законы о защите данных, рекомендации по этике ИИ). Результат: список использованных источников сохранить в файле sources.txt.

2. Сформировать перечень типовых ролей, участвующих в жизненном цикле ИИ-системы (разработчик, поставщик, пользователь, владелец данных, регулятор). Результат: список ролей записать в файл roles.txt.

3. Составить таблицу «Зоны ответственности»: для каждой роли указать основные обязанности (разработка модели, сбор данных, обеспечение

прозрачности, контроль качества, соблюдение прав). Результат: таблицу оформить в LibreOffice Calc и экспортировать в CSV (responsibilities.csv).

4. Проанализировать полученную таблицу: выделить потенциальные риски и предложить меры снижения. Результат: текстовый документ analysis.txt.

5. Подготовить отчёт, включающий титульный лист, описание выполненных шагов, скриншоты таблицы из Calc, выводы и список источников. Результат: файл report.odt (или report.pdf).

Методические указания по ходу выполнения работы:

Для выполнения задания 1 используйте официальные ресурсы (например, сайт Федеральной службы по надзору в сфере связи, информационных технологий и массовых коммуникаций).

В задании 2 перечислите роли в отдельном текстовом файле, каждый пункт – отдельная строка.

В задании 3 создайте таблицу в LibreOffice Calc, столбцы – «Роль», «Обязанности», «Риски», «Меры снижения». После заполнения сохраните как CSV (responsibilities.csv).

В задании 4 сформулируйте выводы в текстовом файле, опираясь на заполненную таблицу.

В задании 5 оформите отчёт согласно структуре: титульный лист, цель работы, ход выполнения (по пунктам), скриншоты таблицы, выводы, список использованных источников. Скриншоты сделайте с помощью стандартного средства захвата экрана и вставьте в документ.

Все файлы (sources.txt, roles.txt, responsibilities.csv, analysis.txt, report.odt/pdf) упакуйте в один ZIP-архив.

Формат сдачи: ZIP-архив, содержащий весь исходный код (если использовались скрипты), данные и отчёт.

Отчёт в формате ODT или PDF: титульный лист, цель, описание каждого задания, скриншоты таблицы из Calc, выводы, список источников. Обязательно указать имена всех файлов, включённых в архив.

Сдать: ZIP-архив с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №41. Разработка рекомендаций по обеспечению безопасности искусственного интеллекта в информационных системах, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно проводить анализ угроз искусственного интеллекта и формировать практические рекомендации по их обеспечению в информационных системах

Задание(я):

1. Ознакомление с введением в безопасность ИИ. Результат: короткие заметки в текстовом файле notes.txt.

2. Составление списка основных угроз ИИ. Результат: таблица threats.csv с колонками «Угроза», «Описание», «Влияние».

3. Формулирование рекомендаций по снижению каждой угрозы. Результат: файл recommendations.txt.

4. Оформление рекомендаций в документе. Сохранить как report.odt.

5. Упаковка всех материалов в архив.

Методические указания по ходу выполнения работы:

В задании 1 изучите предоставленные учебные материалы (презентацию и статью) и сделайте краткие записи о ключевых понятиях безопасности ИИ в файле notes.txt.

В задании 2 на основе изученного материала составьте таблицу с минимум пятью типичными угрозами ИИ, используя LibreOffice Calc. Сохраните таблицу как threats.csv.

В задании 3 для каждой угрозы из таблицы сформулируйте конкретные рекомендации по её снижению. Запишите рекомендации в файл recommendations.txt, соблюдая нумерацию по угрозам.

В задании 4 создайте документ report.odt в LibreOffice Writer: титульный лист, раздел «Введение», «Список угроз», «Рекомендации»,

«Выводы». Вставьте в документ скриншоты таблицы threats.csv и фрагменты рекомендаций.

В задании 5 создайте папку project, поместите туда notes.txt, threats.csv, recommendations.txt, report.odt и любые скриншоты. Сожмите папку в архив project.zip.

Все файлы должны быть сохранены в указанной структуре и включены в архив для сдачи.

Отчет в формате ODT или PDF: титульный лист, описание выполненных заданий, скриншоты таблицы угроз и рекомендаций, выводы о важности безопасности ИИ.

Сдать архив ZIP (project.zip) по электронной почте преподавателю до окончания занятия.

Тема практической работы №42. Оценка правовых аспектов внедрения искусственного интеллекта в информационных системах, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного анализа правовых требований к использованию готовых моделей искусственного интеллекта и их документирования в информационных системах.

Задание(я):

1. Изучить основные правовые акты, регламентирующие применение ИИ (GDPR, закон РФ «О персональных данных», рекомендации по этике ИИ). Результат: список нормативных документов в текстовом файле (requirements.txt).

2. Сформировать таблицу соответствия требований к ИИ (прозрачность, объяснимость, безопасность, защита персональных данных) с критериями оценки. Результат: таблица сохранена в CSV (legal_requirements.csv).

3. Оценить готовую модель машинного обучения (например, предобученный классификатор из scikit-learn) на соответствие выбранным

требованиям. Результат: документ (model_assessment.txt) с выводами и скриншотом окна консоли (assessment.png).

4. Подготовить итоговый отчёт, включающий описание выполненных шагов, таблицу требований, результаты оценки модели и выводы. Результат: файл отчёта в формате ODT или PDF (report.odt).

Методические указания по ходу выполнения работы:

Для выполнения задания 1 откройте официальные сайты EU и РФ, найдите основные положения, касающиеся использования ИИ, и выпишите их в отдельный текстовый файл.

В задании 2 создайте таблицу с колонками: «Требование», «Краткое описание», «Критерий оценки», «Статус соответствия». Заполните её на основе материалов из задания 1 и сохраните в формате CSV.

В задании 3 загрузите готовую модель из библиотеки scikit-learn (например, DecisionTreeClassifier, обученную на датасете Iris). Проанализируйте, насколько модель удовлетворяет каждому требованию из таблицы: наличие возможности объяснения предсказаний, отсутствие использования личных данных, наличие документации и т.п. Сформулируйте выводы в текстовом файле и сделайте скриншот окна, где отображаются результаты анализа.

В задании 4 оформите отчёт согласно следующей структуре: титульный лист, цель практической работы, описание выполненных заданий, таблица требований (вставьте её как изображение), результаты оценки модели, выводы и рекомендации. Вставьте скриншоты из задания 3 в соответствующие разделы.

Все получившиеся файлы (requirements.txt, legal_requirements.csv, model_assessment.txt, assessment.png, report.odt) упакуйте в один архив ZIP.

Отчёт оформляется в формате ODT или PDF. Структура отчёта: 1) Титульный лист (название практики, ФИО студента, группа, дата); 2) Цель практической работы; 3) Описание выполненных заданий; 4) Таблица требований (в виде изображения); 5) Результаты оценки модели (текст + скриншот); 6) Выводы и рекомендации. В отчёте должны быть вставлены скриншоты (assessment.png) и таблица (из CSV).

Сдача: упаковать все файлы в архив ZIP и отправить его по электронной почте преподавателю не позднее окончания 2-х академических часов.

Тема практической работы №43. Проведение анализа конфиденциальности данных при использовании искусственного интеллекта, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно выполнять анализ конфиденциальности данных при использовании готовых моделей искусственного интеллекта

Задание(я):

1. Выбрать готовую модель (например, классификатор scikit-learn) и открытый датасет (Iris). Сохранить код в файл analysis.py.

2. Оценить, содержит ли набор данных персональную информацию (проверка наличия чувствительных признаков). Сохранить результаты проверки в CSV.

3. Применить простой инструмент аудита конфиденциальности (например, библиотеку privacy-metrics, установить через pip). Сгенерировать метрики (доля уникальных записей, риск раскрытия) и сохранить их в CSV.

4. Построить графики метрик (бар-диаграммы, гистограммы) и сохранить их как PNG. Подготовить короткий отчёт с описанием проведённых шагов, скриншотами кода и выводов.

Методические указания по ходу выполнения работы:

Для задания 1 создайте папку project43, в ней файл analysis.py, в котором загрузите выбранный датасет и модель.

Для задания 2 добавьте в analysis.py код, который выводит список признаков и проверяет их на наличие персональных данных; результаты запишите в file "privacy_check.csv" в той же папке.

Для задания 3 установите требуемую библиотеку командой pip install privacy-metrics, выполните расчёт метрик и сохраните их в файл "metrics.csv".

Для задания 4 используйте matplotlib: постройте графики из данных "metrics.csv" и сохраните их как "metrics.png". В отчёте включите скриншоты кода, таблицы CSV и графика.

Все файлы (analysis.py, privacy_check.csv, metrics.csv, metrics.png) упакуйте в один ZIP-архив.

Формат сдачи: архив ZIP, содержащий код, данные и отчёт в ODT или PDF.

Отчёт ODT или PDF: титульный лист, краткое описание каждой задачи, скриншоты кода, таблицы CSV, график metrics.png, выводы о конфиденциальности.

Сдать ZIP-архив с указанными файлами по email преподавателя до конца занятия.

Тема практической работы №44. Тестирование системы искусственного интеллекта на соблюдение правовых норм, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного тестирования готовой модели искусственного интеллекта на соответствие правовым нормам с использованием Python и pytest.

Задание(я):

1. Подготовка рабочей среды и загрузка готовой модели.

2. Создание набора тестов, проверяющих вывод модели на нарушение правовых ограничений (например, генерация запрещённого контента).

3. Запуск тестов, сбор и сохранение результатов в виде отчёта и графика (если требуется визуализация).

4. Оформление отчёта о проделанной работе.

Методические указания по ходу выполнения работы:

В задании 1 установите необходимые пакеты командой pip install pytest joblib (для загрузки модели) и загрузите модель в файл model.pkl, разместив её в каталоге project.

В задании 2 создайте файл tests.py, в котором определите функции-тесты, проверяющие, что модель не генерирует ответы, содержащие запрещённые темы (например, «насилие», «пропаганда наркотиков»). Для каждого теста используйте набор проверочных запросов, оформите их в виде списка строк. Сохраняйте файл tests.py в корневой каталог проекта.

В задании 3 запустите pytest, указав вывод результатов в файл results.txt (например, pytest > results.txt). Сохраните полученный файл в каталог project/reports. При необходимости постройте простой график количества прошедших/не прошедших тестов и сохраните его как report.png (plt.savefig('report.png')).

В задании 4 подготовьте отчёт в формате ODT или PDF. В отчёте должны быть: титульный лист, цель работы, описание каждого задания, скриншот вывода pytest (из results.txt), скриншот графика (report.png), выводы о соответствии модели правовым нормам. Сохраните отчёт как report.odt (или report.pdf) в корневой каталог проекта.

Формат сдачи: архив ZIP, содержащий каталог project с файлами model.pkl, tests.py, results.txt, report.png, report.odt (или pdf) и при необходимости файл requirements.txt с перечнем используемых пакетов.

Отчёт: ODT или PDF, титульный лист, цель, описание заданий, скриншоты (вывод pytest, графика), таблица результатов, выводы и рекомендации.

Сдача: упаковать все файлы проекта в архив ZIP и отправить по электронной почте преподавателю до окончания занятия.

Тема практической работы №45. Разработка отчета по соблюдению законодательства при внедрении искусственного интеллекта, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно готовить отчёт о соблюдении законодательства при внедрении готовой модели искусственного интеллекта.

Задание(я):

1. Сбор нормативных требований (законы, стандарты) в виде списка.

2. Выбор готовой модели ИИ (например, GPT-3, BERT, Stable Diffusion) и проверка её характеристик на соответствие требованиям (данные, конфиденциальность, объяснимость).

3. Формирование таблицы-чек-листа соответствия и построение графика уровня соответствия.

4. Оформление итогового отчёта, включающего описание модели, результаты чек-листа, график, выводы и рекомендации.

Методические указания по ходу выполнения работы:

Для задания 1 создайте документ (LibreOffice Writer) со списком применимых законов и нормативных актов, сохраните как requirements.txt.

Для задания 2 соберите сведения о выбранной модели из её официальной документации и запишите их в файл model_info.csv.

Для задания 3 сформируйте чек-лист в виде таблицы (CSV), где по каждому требованию укажите статус (соответствует/не соответствует). Постройте столбчатый график уровня соответствия и сохраните его как compliance.png (plt.savefig).

Для задания 4 подготовьте отчёт в формате ODT или PDF: титульный лист, цель работы, описание модели, таблица чек-листа, график, выводы и рекомендации. Вставьте скриншоты таблицы и графика.

Все созданные файлы (requirements.txt, model_info.csv, checklist.csv, compliance.png, отчёт.odt/pdf) упакуйте в один ZIP-архив.

Отчёт: ODT или PDF, титульный лист, цель, описание модели, таблица чек-листа (скриншот), график compliance.png, выводы, рекомендации.

Сдача: ZIP-архив с кодом, данными и отчётом отправить преподавателю по email до конца занятия.

Тема практической работы №46. Применение искусственного интеллекта для мониторинга соблюдения правовых норм, объем часов 2

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно применять готовую модель ИИ для автоматической оценки текстов на предмет соблюдения правовых норм и оформлять результаты в виде отчетных файлов.

Задание(я):

1. Подготовить набор тестовых текстов (10-15 предложений) в CSV-файле.

2. Загрузить готовую модель классификации и выполнить предсказание для всех строк.

3. Оценить точность предсказаний, построить график зависимости точности от количества проверяемых текстов.

4. Сохранить предсказания в CSV, график в PNG, модель в файл, оформить код в .py файл.

Методические указания по ходу выполнения работы:

В задании 1 создайте файл data.csv с колонками id и text, разместите его в рабочей папке.

В задании 2 выполните pip install scikit-learn joblib, загрузите модель из файла model.joblib, примените её к колонке text, результаты запишите в колонку prediction.

В задании 3 вычислите метрику accuracy, постройте столбчатый график, сохраните его как accuracy.png, укажите путь к файлу в отчёте.

В задании 4 сохраните предсказания в predictions.csv, разместите код в файле script.py, упакуйте все файлы в ZIP-архив.

Формат сдачи: архив ZIP, содержащий script.py, data.csv, predictions.csv, accuracy.png, model.joblib и отчёт ODT/PDF.

Отчёт ODT или PDF: титульный лист, цель, описание выполненных задач, скриншоты кода, таблица предсказаний, график accuracy.png, выводы.

Отправить архив ZIP преподавателю по email до конца занятия.

Тема практической работы №47. Моделирование системы защиты данных с применением искусственного интеллекта, объем часов 4

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного построения простой модели ИИ для обнаружения аномалий в данных и её интеграции в систему защиты данных.

Задание(я):

1. Подготовка рабочего окружения и установка необходимых библиотек. Результат: файл requirements.txt.

2. Формирование и предобработка обучающего набора данных (генерация synthetic-данных). Результат: data.csv.

3. Обучение модели обнаружения аномалий (IsolationForest). Результат: model.joblib.

4. Оценка качества модели, построение графиков ROC-кривой и распределения аномальных/нормальных точек. Результат: roc_curve.png, anomaly_plot.png.

5. Сохранение всех скриптов, модели и графиков, подготовка отчёта. Результат: набор .py файлов, архив с результатами.

Методические указания по ходу выполнения работы:

В задании 1 создайте файл requirements.txt со списком библиотек: scikit-learn, pandas, matplotlib, joblib. Установите их через pip install -r requirements.txt.

В задании 2 с помощью pandas и numpy сгенерируйте synthetic-датасет с двумя классами (нормальные и аномальные наблюдения) и сохраните его в файл data.csv.

В задании 3 загрузите data.csv, разделите его на обучающую и тестовую части, обучите модель IsolationForest, затем сохраните обученную модель в файл model.joblib с помощью joblib.dump.

В задании 4 примените модель к тестовым данным, вычислите метрики (precision, recall, f1) и постройте ROC-кривую и график распределения

аномалий. Сохраните графики в файлы roc_curve.png и anomaly_plot.png с помощью plt.savefig.

В задании 5 соберите все .py скрипты, файлы data.csv, model.joblib и графики в один каталог, упакуйте его в архив ZIP. Подготовьте отчёт в формате ODT или PDF, включив титульный лист, описание выполненных заданий, скриншоты кода и графиков, а также выводы о работе модели.

Формат сдачи: один ZIP-архив, содержащий каталог с кодом (.py), данными (CSV), моделью (joblib), графиками (PNG) и отчётом (ODT или PDF).

Отчёт должен включать: титульный лист (название работы, ФИО, группа), краткое описание цели и задач, пошаговое описание выполнения каждой задачи, скриншоты кода (из .py файлов) и полученных графиков, таблицу с метриками модели, выводы о качестве модели и её применимости в системе защиты данных.

Сдать: архив ZIP с кодом, данными, моделью, графиками и отчётом отправить по email преподавателя до окончания 4-х академических часов занятий.

Тема практической работы №48. Оценка возможных последствий при ошибках в работе искусственного интеллекта, объем часов 4

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного анализа последствий ошибок в работе моделей искусственного интеллекта, включая подготовку данных, обучение простой модели, моделирование типичных ошибок и оценку их влияния.

Задание(я):

1. Подготовка окружения, загрузка и сохранение датасета Iris в CSV.

2. Обучение базовой модели классификации (LogisticRegression) на чистых данных, сохранение модели.

3. Моделирование типичных ошибок (искажение меток, утечка данных), переобучение модели, построение и сохранение графиков метрик (confusion matrix, accuracy).

4. Анализ полученных результатов, формулирование выводов о потенциальных последствиях ошибок, подготовка отчёта.

Методические указания по ходу выполнения работы:

В задании 1 установите необходимые библиотеки (scikit-learn, pandas, matplotlib, joblib) с помощью pip, загрузите датасет Iris из sklearn, экспортируйте его в CSV файл (iris.csv) и разместите файл в папке data.

В задании 2 загрузите подготовленный CSV, разделите данные на обучающую и тестовую выборки, обучите модель LogisticRegression, оцените её точность на тестовом наборе, сохраните обученную модель в файл (model.pkl) с помощью joblib, результаты метрик запишите в файл metrics.txt.

В задании 3 создайте две версии искажённого набора: а) измените 10% случайных меток классов, б) добавьте в обучающий набор часть тестовых данных (утечка). Обучите модели на каждом искажённом наборе, постройте матрицы ошибок (confusion matrix) и графики зависимости точности от уровня искажения, сохраните графики в PNG файлы (confusion_original.png, confusion_noisy.png, confusion_leak.png). Сохраните новые модели в отдельные файлы (model_noisy.pkl, model_leak.pkl).

В задании 4 сравните полученные метрики и графики, опишите, как типичные ошибки могут влиять на надёжность ИИ-систем, сформулируйте рекомендации по предотвращению таких ошибок. В отчёте включите титульный лист, описание каждого задания, скриншоты графиков, таблицы метрик и выводы.

Формат сдачи: архив ZIP, содержащий папки data (CSV), models (pkl), graphs (PNG), файл metrics.txt и отчёт в формате ODT или PDF.

Отчёт оформляется в ODT или PDF, содержит титульный лист, перечень выполненных заданий, описание методики, скриншоты сохранённых графиков, таблицы метрик, анализ последствий ошибок и выводы.

Сдача: упаковать все файлы в архив ZIP и отправить по электронной почте преподавателю не позднее конца учебного дня.

МДК 03.03 Разработка промптов для искусственного интеллекта

Тема практической работы №1. Создание простого промпта для текстовой модели искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно формировать простой промт, отправлять его текстовой модели ИИ и сохранять полученные ответы для дальнейшего анализа.

Задание(я):

1. Установить Python-пакет openai (pip install openai). Сохранить скрипт в файл prompt_test.py.

2. Сформировать набор запросов (промтов) в виде списка строк и записать их в файл prompts.txt.

3. В скрипте реализовать отправку каждого промта к модели, получить ответы и сохранить их в CSV-файл responses.csv.

4. Сохранить скриншот вывода консоли (пример ответа модели) в файл screenshot.png и добавить его в отчёт.

5. Подготовить короткий отчёт с описанием выполненных шагов, таблицей из CSV и скриншотом.

Методические указания по ходу выполнения работы:

В задании 1 откройте терминал, выполните pip install openai и создайте файл prompt_test.py, где будет размещён основной код.

В задании 2 создайте текстовый файл prompts.txt, запишите в него 3-5 простых запросов (например, «Расскажи шутку», «Опиши погоду», «Дай совет по учебе»).

В задании 3 в файле prompt_test.py реализуйте чтение запросов из prompts.txt, последовательную отправку их модели через API openai,

получение текстовых ответов и запись пар «промт – ответ» в CSV-файл responses.csv (столбцы: prompt, response).

В задании 4 после выполнения скрипта откройте консоль, отобразите несколько полученных ответов и сделайте скриншот, сохранив его как screenshot.png.

В задании 5 составьте отчёт в формате ODT или PDF: титульный лист, перечень выполненных заданий, таблица-выдержка из responses.csv, скриншот консольного вывода, выводы о работе промта.

Формат сдачи: архив ZIP, содержащий файлы prompt_test.py, prompts.txt, responses.csv, screenshot.png и готовый отчёт.

Отчёт ODT или PDF: титульный лист (название работы, ФИО, группа), список задач с кратким описанием, таблица-выдержка из responses.csv, скриншот console_output.png, раздел «Выводы» (о качестве ответов и возможных улучшениях).

Сдача: упаковать все перечисленные файлы в архив ZIP и отправить по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №2. Тестирование промпта на генерацию текста с использованием модели искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно тестировать промпт для генерации текста с использованием готовой модели искусственного интеллекта

Задание(я):

1. Подготовка среды: установить Python, необходимые библиотеки и создать рабочую папку.

2. Загрузка готовой модели (например, GPT-2) и настройка скрипта для генерации текста.

3. Формулирование и сохранение нескольких вариантов промптов в текстовый файл.

4. Генерация текста для каждого промпта, сохранение результатов в отдельные файлы и запись метрик (длина, время генерации) в CSV.

5. Анализ полученных текстов: построение графика зависимости длины текста от варианта промпта, сохранение графика в PNG.

6. Оформление отчёта с описанием выполнения, скриншотами результатов и выводами.

Методические указания по ходу выполнения работы:

В задании 1 создайте папку "pract2" и внутри неё подпапки "data", "models", "results". Установите библиотеки через pip install transformers torch pandas matplotlib.

В задании 2 загрузите модель GPT-2 через библиотеку transformers, сохраните объект модели в подпапку "models". Подготовьте скрипт "generate.py" в корне папки.

В задании 3 составьте минимум три разных промпта, запишите их в файл "prompts.txt" в папку "data". Каждый промпт разместите на отдельной строке.

В задании 4 запустите скрипт, который читает промпты, генерирует текст, сохраняет каждый результат в отдельный файл "output_i.txt" в папку "results" и записывает метрики (номер промпта, количество токенов, время генерации) в файл "metrics.csv" в той же папке.

В задании 5 используйте pandas и matplotlib для построения столбчатой диаграммы, где по оси X – номер промпта, а по оси Y – количество сгенерированных токенов. Сохраните график как "length_chart.png" в папку "results".

В задании 6 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание выполнения каждого задания, скриншоты содержимого файлов "output_i.txt", таблица из "metrics.csv", изображение "length_chart.png", выводы о влиянии разных промптов на генерируемый текст.

Формат сдачи: упаковать весь каталог "pract2" (включая код, данные, результаты и отчёт) в ZIP-архив.

Формат отчёта: ODT или PDF с титульным листом, разделами «Цель», «Ход работы», «Результаты (скриншоты и таблицы)», «Графики», «Выводы». В разделе «Результаты» разместить скриншоты файлов output_i.txt и таблицу metrics.csv. В разделе «Графики» вставить изображение length_chart.png.

Сдача: отправить ZIP-архив с каталогом pract2 по электронной почте преподавателю не позже окончания двух академических часов.

Тема практической работы №3. Оптимизация созданного промпта для улучшения результатов работы модели искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно оптимизировать промт для улучшения качества ответов модели искусственного интеллекта.

Задание(я):

1. Сформировать базовый промт и получить ответы модели. Сохранить промт и ответы в файл prompts_baseline.csv.

2. Проанализировать полученные ответы, выявить недостатки (неполные, неоднозначные, неверные). Оформить результаты анализа в файл analysis.txt.

3. Применить техники улучшения промта (добавление контекста, уточнение задачи, примеры запросов) и получить новые ответы. Сохранить улучшенный промт и ответы в файл prompts_optimized.csv.

4. Сравнить базовые и оптимизированные ответы: построить таблицу сравнения и график изменения метрики релевантности (например, оценка по 5-балльной шкале). Сохранить график в файл comparison.png.

5. Подготовить отчёт с описанием всех шагов, скриншотами вывода, таблицей сравнения и графиком. Сохранить отчёт в формате ODT или PDF.

Методические указания по ходу выполнения работы:

В задании 1 используйте любой публичный LLM-сервис (например, модель OpenAI через командную строку или локальную модель в библиотеке transformers). Запишите использованный промт и полученные ответы в CSV-файл, где каждая строка содержит промт и соответствующий ответ.

В задании 2 проанализируйте ответы: оцените полноту, точность и ясность. Запишите выводы в текстовый файл, указав конкретные примеры проблемных ответов.

В задании 3 примените минимум три техники улучшения промта (уточнение цели, добавление примеров, разбивка вопроса на части). Сохраните каждый новый промт и полученный ответ в отдельный CSV-файл, аналогично заданию 1.

В задании 4 оцените ответы по единой шкале (например, 1-5). Сформируйте таблицу с колонками: тип промта (baseline/optimized), оценка. Постройте столбчатый график изменения средней оценки, сохраните его функцией plt.savefig('comparison.png').

В задании 5 подготовьте отчёт, включив: титульный лист, цель работы, описание каждого задания, скриншоты консольного вывода, таблицу сравнения, график, выводы о влиянии оптимизации промта. Сохраните документ в ODT или PDF.

Все исходные скрипты (Python-файлы), CSV-файлы, графики и отчёт упакуйте в один ZIP-архив для сдачи.

Формат отчёта: ODT или PDF. Структура: титульный лист; цель практической работы; описание выполненных заданий; скриншоты вывода консоли; таблица сравнения базовых и оптимизированных ответов; график comparison.png; выводы и рекомендации.

Сдача: упаковать весь проект (скрипты .py, CSV-файлы, график PNG, отчёт ODT/PDF) в архив ZIP и отправить по email преподавателю до конца учебного занятия.

Тема практической работы №4. Работа с параметрами промптов для достижения заданных целей при использовании искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного подбора и настройки параметров промтов (temperature, max_tokens, top_p) для получения требуемого результата от больших языковых моделей

Задание(я):

1. Подготовка окружения и базовый запрос к LLM. Результат (текст ответа) сохранить в файл answer_base.txt.

2. Создание трёх вариантов промта с разными настройками параметров (temperature, max_tokens, top_p) для генерации короткого рассказа в заданном стиле. Каждый ответ сохранить в отдельный файл (answer_1.txt, answer_2.txt, answer_3.txt).

3. Анализ полученных текстов: вычислить длину (количество символов) и количество вхождений ключевых слов стиля. Сформировать таблицу metrics.csv с колонками: variant, length, keyword_count. Построить график зависимости длины от значения temperature и сохранить как length_vs_temp.png.

Методические указания по ходу выполнения работы:

Установите необходимые библиотеки командой: pip install openai pandas matplotlib joblib

Создайте Python-скрипт (script.py), в котором будет выполнен запрос к модели, сохранены ответы и проведён анализ.

Для каждого варианта промта явно укажите используемые параметры (temperature, max_tokens, top_p) в коде.

Для подсчёта количества ключевых слов используйте простой поиск по строке.

График построить с помощью matplotlib и сохранить командой plt.savefig('length_vs_temp.png').

Сохраните все полученные файлы (txt, csv, png, script.py) в одну папку.

Подготовьте отчёт, включив в него: титульный лист, описание выполненных заданий, скриншоты кода, содержимое txt-файлов, таблицу metrics.csv и график length_vs_temp.png, выводы о влиянии параметров.

Формат сдачи: архив ZIP, содержащий папку с кодом, данными и отчётом.

Отчёт в формате ODT или PDF: титульный лист, список задач, описание каждого шага, скриншоты кода, содержимое файлов answer_*.txt, таблица metrics.csv, график length_vs_temp.png, выводы и заключения.

Сдать архив ZIP по электронной почте преподавателю не позднее конца учебного занятия.

Тема практической работы №5. Сравнение работы двух различных промптов при решении одной задачи с использованием искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного формирования и сравнения нескольких промтов для решения одной задачи в сервисе генеративного ИИ, а также анализа полученных результатов.

Задание(я):

1. Выбрать задачу (например, генерация короткого описания продукта)

2. Сформулировать два разных промта, ориентированных на разные стилистические требования.

3. С помощью онлайн-сервиса LLM (например, OpenAI ChatGPT) выполнить запросы с каждым промтом, получив два текста-ответа.

4. Сохранить полученные ответы в отдельные файлы формата TXT (answer_1.txt, answer_2.txt).

5. Оценить ответы по нескольким метрикам (соответствие задаче, читаемость, оригинальность) и занести оценки в таблицу CSV (metrics.csv).

6. Визуализировать сравнение метрик в виде столбчатой диаграммы и сохранить её как PNG (comparison.png).

7. Оформить отчет, включив описание задачи, промты, полученные ответы, таблицу метрик, график сравнения и выводы.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 определите конкретную задачу, которую будет решать ИИ; запишите её в документ.

В задании 2 сформулируйте два промта, различающихся по формулировке, тону или требуемому уровню детализации.

В задании 3 откройте выбранный сервис генеративного ИИ, последовательно введите каждый промт и зафиксируйте полученный ответ.

В задании 4 создайте в рабочей папке файлы answer_1.txt и answer_2.txt, скопировав в них соответствующие ответы.

В задании 5 оцените каждый ответ по трём критериям (соответствие задаче, читаемость, оригинальность) по шкале от 1 до 5, занесите результаты в таблицу metrics.csv (колонки: prompt, relevance, readability, originality).

В задании 6 постройте столбчатую диаграмму, где по оси X – названия критериев, а по оси Y – средние оценки двух промтов; сохраните график в файл comparison.png (используйте matplotlib, команда plt.savefig('comparison.png')).

В задании 7 подготовьте отчёт в формате ODT или PDF: титульный лист, описание задачи, тексты промтов, ответы (скриншоты или вставка из txt), таблицу метрик (вставьте из CSV), график comparison.png и раздел выводов с рекомендациями по выбору промтов.

Все исходные файлы (txt, csv, png, скрипт Python, если использовался) упакуйте в один ZIP-архив.

Формат сдачи: ZIP-архив, содержащий код, данные и отчёт, отправить по email преподавателю до конца занятия.

Отчёт оформляется в ODT или PDF, включает титульный лист (название работы, ФИО, группа, дата), описание задачи, два промта, ответы ИИ (скриншоты или вставка текста), таблицу метрик (в виде изображения или таблицы), график comparison.png, выводы и рекомендации. В конце отчёта укажите список вложенных файлов.

Сдача: подготовленный ZIP-архив отправить на электронную почту преподавателя не позднее конца второго академического часа.

Тема практической работы №6. Тестирование промптов с использованием вариаций структуры запросов к модели искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного тестирования промтов: формировать варианты запросов, получать ответы модели ИИ, сохранять результаты и проводить их количественный анализ.

Задание(я):

1. Подготовка среды разработки (установка Python, библиотеки, настройка API-ключа).

2. Формирование базового набора промтов (минимум 5 разных тем).

3. Создание вариантов каждого промта (изменение формулировки, добавление уточнений, изменение тона).

4. Выполнение запросов к модели ИИ для всех вариантов и запись ответов.

5. Экспорт полученных ответов в CSV-файл, включающий колонку «Вариант», «Ответ», «Время отклика».

6. Построение графика зависимости длины запроса от времени отклика и сохранение его в PNG.

7. Сохранение всех скриптов, CSV-файла и графика в отдельные папки, подготовка отчёта.

Методические указания по ходу выполнения работы:

Для задания 1 установите Python 3.10+, выполните pip install openai pandas matplotlib, сохраните ключ API в отдельном файле config.txt.

Для задания 2 создайте текстовый файл base_prompts.txt, каждую строку заполните отдельным базовым запросом.

Для задания 3 в отдельном файле prompt_variants.py сформируйте список вариантов, используя простые правила изменения текста (добавление «Пожалуйста», изменение вопроса в утверждение и т.п.).

Для задания 4 напишите скрипт run_prompts.py, который читает варианты, отправляет их модели через openai.Completion, измеряет время отклика и сохраняет ответы в переменные.

Для задания 5 создайте файл results.csv, запишите в него столбцы: VariantID, Prompt, Response, ResponseLength, LatencySeconds.

Для задания 6 в скрипте plot_latency.py постройте график (ось X – длина Prompt, ось Y – LatencySeconds) и сохраните его командой plt.savefig('latency_plot.png').

Для задания 7 организуйте структуру папок: /code – все .py-файлы, /data – CSV, /images – PNG, /report – готовый отчёт.

Формат сдачи: архив ZIP, содержащий папки code, data, images и файл отчёта в ODT или PDF.

Отчёт оформляется в ODT или PDF, включает титульный лист, описание целей и задач, пошаговое описание выполнения, скриншоты кода и вывода, таблицу результатов (CSV) в виде вставки, график latency_plot.png и выводы по анализу.

Сдача: упаковать все материалы в архив ZIP и отправить по email преподавателю до конца занятия.

Тема практической работы №7. Анализ и исправление ошибок в промпте для модели искусственного интеллекта, объем часов 2

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного анализа и исправления ошибок в промтах для моделей искусственного интеллекта

Задание(я):

1. Подготовить набор исходных промтов (CSV) и выполнить их через выбранную LLM-модель.

2. Проанализировать полученные ответы, выявить типичные ошибки (неясность, неоднозначность, отсутствие контекста).

3. Сформировать исправленные варианты промтов, протестировать их тем же способом и сравнить результаты.

4. Оформить результаты в отчёт, включив скриншоты ответов, таблицы сравнения и выводы.

Методические указания по ходу выполнения работы:

Для задания 1 создайте файл prompts_original.csv, разместив в нем 5-10 простых запросов к модели (по одной строке).

Напишите короткую Python-программу, использующую библиотеку openai (pip install openai), которая читает CSV, отправляет каждый запрос модели и сохраняет ответы в файл responses_raw.txt, а также сохраняет пары «промт-ответ» в файл results_raw.csv.

Для задания 2 откройте results_raw.csv, просмотрите ответы и в отдельном документе (TXT) запишите наблюдения о проблемах каждого ответа (неясность, отсутствие нужной информации, лишние детали).

Для задания 3 в файле prompts_corrected.csv разместите исправленные версии промтов, учитывая замечания из пункта 2. Запустите ту же

программу, изменив путь к CSV, и сохраните новые ответы в responses_corrected.txt и results_corrected.csv.

Сравните два набора ответов: в отдельной таблице (CSV) укажите исходный промт, исправленный промт, короткую оценку качества ответа (например, «недостаточно», «хорошо»).

Сохраните все файлы (CSV, TXT, PNG-скриншоты ответов в терминале) в одну папку.

Формат сдачи: архив ZIP, содержащий папку с кодом (.py), всеми CSV/TXT-файлами и отчёт в ODT или PDF.

Отчёт ODT/PDF: титульный лист, цель работы, описание каждого задания, таблица сравнения оригинальных и исправленных промтов с оценками, скриншоты ответов модели, выводы и личные рекомендации.

Сдать архив ZIP с кодом, данными и отчётом по email преподавателя не позднее конца занятия.

Тема практической работы №8. Изучение влияния длины промпта на результат работы модели искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно исследовать влияние длины промта на качество вывода модели искусственного интеллекта, используя Python и открытые модели.

Задание(я):

1. Установить необходимые Python-пакеты (pip install transformers, torch).

2. Скачать и загрузить предобученную языковую модель (например, GPT-2) в скрипт.

3. Сформировать набор промтов различной длины (короткие, средние, длинные) и сохранить их в файл prompts.txt.

4. Сгенерировать ответы модели для каждого промта, зафиксировать количество токенов в ответе и сохранить результаты (промт, длина промта, ответ, длина ответа) в CSV-файл results.csv.

5. Вычислить простую метрику качества (например, оценить связность ответа вручную или подсчитать среднюю длину ответа) и добавить её в CSV.

6. Построить график зависимости длины промта от длины ответа (и/или от выбранной метрики) и сохранить его как plot.png.

7. Сохранить объект модели в файл (joblib.dump или torch.save) как model.bin.

8. Подготовить отчёт и упаковать все файлы в архив.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 откройте терминал и выполните команды установки пакетов, затем проверьте их наличие командой pip list.

В задании 2 создайте Python-скрипт (load_model.py), импортируйте нужные классы из библиотеки transformers и загрузите модель в переменную.

В задании 3 подготовьте текстовый файл prompts.txt, в каждой строке разместив отдельный промт разной длины (пример: 5 слов, 15 слов, 30 слов).

В задании 4 создайте скрипт generate.py, который читает prompts.txt, последовательно передаёт каждый промт модели, получает ответ и записывает в results.csv столбцы: prompt, prompt_length, response, response_length, quality_metric.

В задании 5 добавьте в generate.py расчёт выбранной метрики (например, количество предложений в ответе) и запишите её в CSV.

В задании 6 используйте библиотеку matplotlib: загрузите results.csv, постройте график (ось X – длина промта, ось Y – длина ответа или метрика) и сохраните изображение командой plt.savefig('plot.png').

В задании 7 сохраните загруженную модель в файл model.bin с помощью соответствующей функции сохранения (torch.save или joblib.dump).

В задании 8 подготовьте отчёт в формате ODT или PDF, включив титульный лист, описание каждого задания, скриншоты кода, таблицу из results.csv и изображение plot.png. Упакуйте в ZIP-архив папку с: load_model.py, generate.py, prompts.txt, results.csv, plot.png, model.bin и отчёт.

Формат сдачи: ZIP-архив, отправить по электронной почте преподавателю до конца занятия.

Отчёт ODT или PDF: титульный лист, цель работы, список выполненных заданий, описание методики, таблица results.csv (вставка как изображение), скриншоты ключевых фрагментов кода, график plot.png, выводы о влиянии длины промта.

Сдача: архив ZIP, содержащий все .py скрипты, исходные данные, полученные файлы (CSV, PNG, модель) и отчёт, отправить по email преподавателю до конца занятия.

Тема практической работы №9. Создание сложного промпта для мультизадачной модели искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного создания и отладки сложного промта для мультизадачной модели искусственного интеллекта, включая формулирование целей, построение структуры промпта и анализ полученных ответов.

Задание(я):

1. Проанализировать требования к мультизадачной модели и сформулировать цели промта. Сохранить текстовый документ с описанием целей (prompt_requirements.txt).

2. Составить комплексный промт, объединяющий несколько задач (например, генерацию текста, классификацию и поиск фактов). Сохранить готовый промт в файл (complex_prompt.txt).

3. Запустить промт через выбранный API LLM (например, OpenAI), сохранить ответы в CSV-файл (responses.csv), подсчитать количество токенов в каждом ответе и построить график токенов vs. номер запроса (tokens_plot.png).

Методические указания по ходу выполнения работы:

В задании 1 изучите описание модели и перечислите, какие типы задач она может выполнять. Запишите цели промта в отдельный текстовый файл и укажите, какие входные данные и ожидаемые результаты нужны для каждой задачи.

В задании 2 постройте промт, используя три секции: вводные инструкции, примеры запросов-ответов и конкретные задачи. Убедитесь, что промт чётко разделён на части и легко читаем. Сохраните итоговый промт в файл complex_prompt.txt.

В задании 3 создайте простой Python-скрипт, который отправляет промт к API модели, получает ответы и записывает их в CSV-файл. Для каждого ответа подсчитайте количество токенов (можно воспользоваться функцией подсчёта токенов из библиотеки tiktoken). С помощью matplotlib постройте график количества токенов от номера запроса и сохраните его как tokens_plot.png. Все результаты (CSV, PNG, скрипт .py) разместите в отдельную папку.

Формат сдачи: один ZIP-архив, содержащий папку с кодом (.py), всеми сохранёнными файлами (txt, csv, png) и отчётом.

Отчёт в формате ODT или PDF, включающий: титульный лист (название модуля, название работы, ФИО студента, дата); раздел «Цели и задачи», где описаны результаты задания 1; раздел «Промт», где приведён текст из complex_prompt.txt и комментарии к структуре; раздел «Результаты выполнения», где представлены фрагменты CSV-файла, скриншоты вывода скрипта, график tokens_plot.png и краткий анализ полученных ответов; выводы и оценка выполненной работы.

Сдача: упаковать все материалы в ZIP-архив и отправить по электронной почте преподавателю не позже конца второго академического часа (через 2 часа после начала практической работы).

Тема практической работы №10. Работа с промптами для решения аналитических задач с использованием искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно формировать и использовать промпты для получения аналитических ответов от готовой модели искусственного интеллекта

Задание(я):

1. Подготовка рабочего окружения (установить Python, библиотеку openai, создать проект). Результат: файл requirements.txt с указанием pip install openai.

2. Составление набора промтов для аналитической задачи (например, анализ продаж по датасету Iris). Результат: файл prompts.txt, где каждый промт записан в отдельной строке.

3. Выполнение промтов через консольный клиент, сохранение полученных ответов. Результат: файл responses.csv с колонками «prompt» и «response».

4. Визуализация ключевых метрик полученных ответов (например, количество упомянутых классов). Результат: график metrics.png, сохранённый через plt.savefig.

5. Подготовка отчёта о проделанной работе. Результат: документ report.odt (или PDF) с описанием выполненных шагов, скриншотами консольного вывода и графика.

Методические указания по ходу выполнения работы:

В задании 1 создайте виртуальное окружение, установите требуемые пакеты и зафиксируйте их в requirements.txt.

В задании 2 сформулируйте минимум пять промтов, направленных на аналитический запрос к модели, и запишите их в prompts.txt.

В задании 3 последовательно выполните каждый промт, получив ответ от модели, и запишите пары «промт-ответ» в responses.csv.

В задании 4 постройте столбчатый график количества упоминаний разных категорий в ответах и сохраните его как metrics.png.

В задании 5 соберите отчёт: титульный лист, описание каждого задания, скриншоты вывода консоли, график metrics.png и выводы.

Формат сдачи: архив ZIP, содержащий папку с кодом (.py файлы), requirements.txt, prompts.txt, responses.csv, metrics.png и готовый отчёт (ODT или PDF).

Формат отчёта: ODT или PDF, включающий титульный лист, краткое описание практики, список выполненных заданий, скриншоты консольных выводов, график metrics.png и раздел выводов.

Сдача: упаковать все материалы в архив ZIP и отправить по электронной почте преподавателю не позже окончания двухчасового занятия.

Тема практической работы №11. Создание промпта для описания сложных задач (например, для анализа данных) с использованием искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно формировать эффективные промпты для описания сложных задач анализа данных с использованием готовых моделей искусственного интеллекта.

Задание(я):

1. Изучить базовые принципы построения промтов (структура, уточнение задачи, указание формата вывода). Оценить примерные требования к промту.

2. Сформулировать собственный промт для задачи анализа данных (например, «Проведи описательный анализ набора данных Iris», указать

желаемый формат вывода, ограничения). Сохранить текст промта в файл prompt.txt.

3. Проверить промт с помощью готовой модели ИИ (через веб-интерфейс, например, ChatGPT). Скопировать полученный ответ и сохранить в файл response.txt, сделав скриншот результата и сохранить как response.png. При необходимости улучшить промт и повторить проверку.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 изучите материалы курса о построении промтов, обратите внимание на элементы уточнения задачи и формата вывода.

В задании 2 откройте текстовый редактор LibreOffice Writer или любой простой редактор, создайте файл prompt.txt и запишите в него сформулированный промт. Сохраните файл в отдельную папку проекта.

В задании 3 откройте браузер, перейдите к онлайн-сервису готовой модели ИИ, введите содержимое файла prompt.txt в поле ввода и отправьте запрос. Скопируйте полученный ответ в файл response.txt, сделайте скриншот окна с ответом и сохраните как response.png в той же папке. При необходимости отредактируйте промт в prompt.txt и повторите процесс до получения удовлетворительного результата.

После выполнения всех пунктов упакуйте папку проекта (включая файлы prompt.txt, response.txt, response.png и любые вспомогательные файлы) в архив ZIP.

Подготовьте отчёт в формате ODT или PDF, включив титульный лист, описание каждого задания, скриншоты (response.png) и выводы о качестве полученного ответа.

Отчёт: ODT или PDF, титульный лист, разделы «Описание задачи», «Промт», «Результат ИИ», «Анализ и выводы», включающие скриншоты response.png и таблицу сравнения оригинального и улучшенного промтов.

Сдача: архив ZIP с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №12. Создание промпта для генерации творческого контента с использованием искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно разрабатывать, тестировать и документировать промпты для генерации творческого текста с помощью готовой модели искусственного интеллекта.

Задание(я):

1. Подготовить рабочее окружение: установить Python, создать виртуальное окружение и установить необходимые пакеты (pip install openai). Сохранить список установленных пакетов в файл requirements.txt.

2. Сформировать базовый промт для генерации короткого рассказа. Сохранить текст промта в файл prompt.txt.

3. Вызвать модель ИИ через консольный скрипт, передав промт, и получить сгенерированный текст. Сохранить результат в файл output.txt.

4. Проанализировать полученный результат: внести корректировки в промт (добавить ограничения, стиль, тему) и повторить запрос к модели. Сохранить улучшенный промт в файл prompt_refined.txt и новый результат в файл output_refined.txt.

5. Подготовить визуализацию сравнения: построить простой график количества слов в оригинальном и улучшенном тексте и сохранить его как word_counts.png.

6. Оформить отчёт, включив титульный лист, описание целей, пошаговый процесс, скриншоты консольных выводов, таблицу сравнения и график. Сохранить отчёт в формате ODT или PDF.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 создайте папку проекта, откройте терминал, создайте виртуальное окружение командой python -m venv venv, активируйте

его и установите пакет openai. Сохраните список пакетов, выполнив pip freeze > requirements.txt, в корневой каталог проекта.

В задании 2 откройте любой текстовый редактор, напишите промт, который задаёт тему, жанр и ограничения (например, "Напиши короткий фантастический рассказ в стиле Харуки Мураками, не более 200 слов.") и сохраните его как prompt.txt в папке проекта.

Для задания 3 создайте простой Python-скрипт (например, generate.py), который читает файл prompt.txt, отправляет его модели через API и выводит полученный текст в консоль. Перенаправьте вывод в файл output.txt, используя оператор > в командной строке.

В задании 4 откройте файл prompt.txt, добавьте уточнения (например, "используй образный язык и концовку-twist"), сохраните как prompt_refined.txt и повторите запуск скрипта, перенаправив результат в output_refined.txt.

Для задания 5 подсчитайте количество слов в каждом из файлов output.txt и output_refined.txt (можно воспользоваться утилитой wc или написать простой скрипт). С помощью библиотеки matplotlib постройте столбчатый график, отобразив количество слов для обеих версий, и сохраните его как word_counts.png (plt.savefig('word_counts.png')).

В задании 6 подготовьте отчёт: титульный лист (название работы, ФИО, группа), цель, список выполненных задач, скриншоты вывода консоли (сохраните как изображения и вставьте в документ), таблицу со сравнение количества слов и ключевых характеристик, график word_counts.png и выводы о том, как изменения в промте влияют на результат.

Отчёт оформляется в ODT или PDF, содержит титульный лист, цель практической работы, пошаговое описание выполненных заданий, скриншоты консольных выводов, таблицу сравнения (кол-во слов, основные особенности), график word_counts.png и раздел "Выводы".

Сдача: упаковать весь проект (все .py файлы, .txt, .png, requirements.txt и готовый отчёт) в архив ZIP и отправить по электронной почте преподавателю до конца учебного занятия.

Тема практической работы №13. Настройка промптов для работы с различными типами искусственного интеллекта (текст, изображения, голос), объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного создания и настройки промптов для работы с различными типами готовых моделей искусственного интеллекта (текст, изображения, голос).

Задание(я):

1. Настройка промта для текстовой модели. Сохранить промт в файл prompt_text.txt, получить ответ модели и сохранить его в файл answer_text.txt.

2. Настройка промта для модели генерации изображений. Сохранить описание изображения в файл prompt_image.txt, сгенерировать изображение и сохранить как result_image.png.

3. Настройка промта для модели синтеза речи. Сохранить текст для озвучивания в файл prompt_voice.txt, сгенерировать аудио-файл result_voice.wav.

Методические указания по ходу выполнения работы:

Подготовка окружения: установить Python, создать отдельную папку проекта, выполнить pip install openai, pip install diffusers[torch], pip install torch, pip install soundfile (или аналогичный пакет для TTS).

Для задания 1: открыть текстовый редактор, написать запрос-промт, сохранить в prompt_text.txt; запустить скрипт, который отправит промт в выбранную LLM-модель, получить ответ и записать его в answer_text.txt; убедиться, что файл сохранён в папке проекта.

Для задания 2: в prompt_image.txt записать короткое описание желаемого изображения; запустить скрипт, использующий модель генерации изображений, сохранить полученный файл как result_image.png в той же папке; проверить открытие изображения в просмотрщике.

Для задания 3: в prompt_voice.txt разместить текст для озвучивания; запустить скрипт, вызывающий модель синтеза речи, сохранить аудио-файл как result_voice.wav; прослушать файл через любой медиаплеер.

Все полученные файлы (prompt_*.txt, answer_text.txt, result_image.png, result_voice.wav) включить в архив.

Формат сдачи: архив ZIP, содержащий папку с кодом (.py файлы), всеми текстовыми и медиаресурсами, а также отчёт в формате ODT или PDF.

Отчёт ODT или PDF: титульный лист (название работы, ФИО, группа), краткое описание целей, пошаговое описание выполнения каждого задания, скриншоты вывода (текстовый ответ, изображение, аудио-плеер с воспроизведённым файлом), список сохранённых файлов и выводы.

Сдать архив ZIP (код, данные, отчёт) по электронной почте преподавателю не позднее окончания 2-х академических часов.

Тема практической работы №14. Анализ работы промптов с использованием контекста и без использования контекста в моделях искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно выполнять анализ влияния контекста на ответы готовой модели искусственного интеллекта, сравнивать результаты и оформлять выводы в виде отчёта.

Задание(я):

1. Подготовка рабочего окружения (установить Python-пакеты, создать папку проекта).

2. Формирование набора тестовых промтов (10 коротких вопросов).

3. Получение ответов модели без добавления контекста и сохранение их в файл answers_no_context.csv.

4. Получение ответов модели с добавлением простого контекста (например, вводный абзац) и сохранение их в файл answers_with_context.csv.

5. Вычисление сравниваемых метрик (длина ответа в токенах, количество уникальных слов) и запись результатов в файл comparison.csv.

6. Построение графика сравнения метрик (средняя длина и уникальность) и сохранение его как metrics.png.

7. Оформление отчёта с описанием выполненных шагов, таблицами, скриншотами и графиком.

Методические указания по ходу выполнения работы:

В задании 1 создайте отдельную папку проекта, откройте терминал и выполните команды: pip install transformers torch pandas matplotlib joblib. Сохраните список установленных пакетов в файл requirements.txt.

В задании 2 подготовьте текстовый файл prompts.txt, где каждая строка – отдельный промт. Пример содержимого: "Что такое искусственный интеллект?".

В задании 3 напишите скрипт (например, analyze.py), который загружает небольшую предобученную модель (distilgpt2) через библиотеку transformers, последовательно подаёт каждый промт из prompts.txt без контекста, сохраняет полученные ответы в CSV-файл с двумя столбцами: prompt и answer_no_context.

В задании 4 модифицируйте скрипт, добавив к каждому промту фиксированный вводный текст (контекст), например: "Для начинающего программиста: ". Сохраните ответы в отдельный CSV-файл с колонкой answer_with_context.

В задании 5 используя pandas, загрузите оба CSV-файла, вычислите для каждого ответа длину в токенах (можно использовать метод split()) и количество уникальных слов, добавьте эти значения в таблицу, затем рассчитайте средние показатели по каждому набору и запишите всё в comparison.csv.

В задании 6 постройте столбчатый график, где по оси X будут два столбца – «Без контекста» и «С контекстом», а по оси Y – средняя длина ответа и среднее количество уникальных слов (два графика на одном рисунке). Сохраните рисунок как metrics.png.

В задании 7 подготовьте отчёт в формате ODT или PDF. Отчёт должен содержать: титульный лист, цель работы, описание каждого задания, скриншоты кода и терминала, таблицы из CSV-файлов, график metrics.png и выводы о влиянии контекста.

Формат сдачи: в один ZIP-архив включите папку проекта с файлами analyze.py, prompts.txt, requirements.txt, answers_no_context.csv, answers_with_context.csv, comparison.csv, metrics.png и готовый отчёт.

Отчёт: ODT или PDF, титульный лист, цель, описание шагов, скриншоты выполнения, таблицы (CSV-данные), график metrics.png, выводы и заключение.

Сдача: ZIP-архив с кодом, данными и отчётом отправить по электронной почте преподавателю не позднее окончания двухчасового занятия.

Тема практической работы №15. Разработка промпта для автоматизации процессов с помощью искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного создания и тестирования промптов для готовых моделей искусственного интеллекта, научиться документировать процесс и оценивать полученные результаты.

Задание(я):

1. Подготовка среды. Установить необходимые Python-библиотеки, создать рабочую папку проекта.

2. Формулирование задачи. Описать реальную бизнес- или учебную задачу, решаемую с помощью ИИ, и определить требуемый тип ответа модели.

3. Составление промта. На основе описанной задачи написать текстовый запрос (промт), учитывая рекомендации по структуре и уточнению требований.

4. Тестирование промта. Запустить запрос к готовой модели (например, GPT-3/4) через Python-скрипт, получить ответы, при необходимости скорректировать промт.

5. Сбор и сохранение результатов. Сохранить финальный промт в файл *.txt, ответы модели – в файл *.csv, построить простой график количества токенов/времени ответа и сохранить его как *.png.

6. Оформление отчёта. Подготовить документ ODT/PDF с описанием всех шагов, скриншотами вывода и выводами.

Методические указания по ходу выполнения работы:

Для выполнения работы используйте Python 3.x и LibreOffice.

Установите необходимые библиотеки командой: pip install openai pandas matplotlib joblib.

Создайте папку проекта, внутри которой разместите файлы: prompt.txt, response.csv, tokens_plot.png, script.py.

В файле prompt.txt сохраните окончательный текст промта.

В файле response.csv запишите столбцы: "Запрос", "Ответ", "Количество_токенов", "Время_выполнения".

График количества токенов от номера запроса сохраните командой plt.savefig('tokens_plot.png').

Сохраните обучаемый объект (если используется) через joblib.dump, хотя в данном случае работает готовая модель.

Подготовьте отчёт: титульный лист, цель работы, описание каждого задания, скриншоты вывода из консоли и графика, выводы и возможные улучшения.

Все материалы (папка проекта + отчёт) упакуйте в архив ZIP.

Отчёт оформляется в ODT или PDF, включает титульный лист, цель, пошаговое описание заданий, скриншоты консольного вывода и графика, таблицу с результатами, выводы и рекомендации.

Сдача: архив ZIP с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №16. Оптимизация промпта на основе обратной связи, полученной от модели искусственного интеллекта, объем часов 4

У1. Анализировать задачи для выбора подходящих готовых моделей ИИ, учитывать их ограничения и возможности.

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного улучшения промпта для модели LLM на основе полученной от неё обратной связи

Задание(я):

1. Установить и настроить Python-окружение, установить необходимые библиотеки (pip install openai, pandas, matplotlib, joblib). Сохранить список установленных пакетов в файл requirements.txt. (30 минут)

2. Сформировать начальный промт, отправить его модели, получить ответ и сохранить ответ в файл response_1.txt, а также сохранить использованный промт в prompt_1.txt. (45 минут)

3. Проанализировать полученный ответ: выделить недостатки, сформулировать обратную связь и на её основе изменить промт. Сохранить новый промт в prompt_2.txt и ответ модели в response_2.txt. (60 минут)

4. Оценить улучшения: сравнить два ответа, подсчитать простую метрику (например, количество ключевых слов), построить график сравнения и сохранить как comparison.png; сохранить метрику в results.csv; сохранить финальную модель (если использовалась локальная) через joblib.dump в файл model.pkl. (45 минут)

Методические указания по ходу выполнения работы:

В задании 1 создайте виртуальное окружение, активируйте его и выполните установку перечисленных пакетов; список пакетов запишите в requirements.txt.

В задании 2 используйте API ключ модели LLM, отправьте текст из prompt_1.txt, полученный ответ запишите в response_1.txt; не забудьте фиксировать время запроса.

В задании 3 прочитайте response_1.txt, выпишите в отдельный документ недостатки (неполные ответы, неточности), сформулируйте уточняющие вопросы и дополнения, создайте новый промт и сохраните его в prompt_2.txt; повторите запрос и запишите ответ в response_2.txt.

В задании 4 проведите количественный анализ: подсчитайте количество целевых ключевых слов в каждом ответе, запишите результаты в results.csv; постройте столбчатый график сравнения и сохраните как comparison.png; если вы использовали локальную модель, сохраните её в model.pkl.

Формат сдачи: один ZIP-архив, содержащий папку с файлом requirements.txt, всеми .txt, .csv, .png, .pkl файлами, а также папкой src с .py скриптами, использованными в работе.

Отчёт в формате ODT или PDF. Структура: титульный лист, цель работы, описание каждого задания с указанием использованных файлов, скриншоты консольных выводов и графика comparison.png, таблица results.csv, выводы о том, насколько изменился ответ модели после оптимизации промта.

Сдача: упаковать всё в ZIP-архив и отправить по электронной почте преподавателю не позднее окончания 4-часового занятия.

Тема практической работы №17. Создание промпта для обработки текстовых данных с использованием модели искусственного интеллекта, объем часов 2

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц**.**

Цель практической работы:

Научиться самостоятельно создавать промпты для обработки текстовых данных и использовать готовую модель искусственного интеллекта для получения результатов.

Задание(я):

1. Подготовка рабочего окружения: установить Python, pip, создать виртуальное окружение, установить необходимые библиотеки (pip install transformers torch). Сохранить список установленных пакетов в файл requirements.txt.

2. Формирование и сохранение промта: написать текстовый файл prompt.txt, содержащий описанный запрос к модели (например, «Сократи следующий текст до 100 слов, сохранив смысл»).

3. Написание скрипта для обращения к модели: создать файл run_prompt.py, который загружает предобученную модель (например, distilbert-base-uncased), читает prompt.txt, подает его на вход модели и сохраняет полученный ответ в файл response.txt.

4. Анализ полученного ответа: открыть response.txt, оценить корректность и полноту результата, оформить выводы в документ analysis.txt.

5. Подготовка отчёта и упаковка результатов: создать файл report.odt (или PDF) с титульным листом, описанием выполненных заданий, скриншотами терминала и содержимым файлов prompt.txt, response.txt, analysis.txt. Сохранить все файлы в папку project и упаковать её в архив project.zip.

Методические указания по ходу выполнения работы:

Для задания 1 используйте командную строку: создайте виртуальное окружение, активируйте его, выполните pip install нужных пакетов и сохраните список пакетов командой pip freeze > requirements.txt.

Для задания 2 создайте простой текстовый файл prompt.txt в корне проекта и запишите в него ваш запрос.

Для задания 3 напишите скрипт run_prompt.py, который импортирует нужные модули, загружает модель, читает prompt.txt, получает ответ и записывает его в response.txt. Сохраните скрипт в корень проекта.

Для задания 4 откройте response.txt, сравните полученный текст с ожидаемым, запишите свои выводы в файл analysis.txt.

Для задания 5 оформите отчёт в ODT или PDF: титульный лист, цель работы, список выполненных заданий, скриншоты вывода терминала (например, вывод pip list, запуск скрипта), содержимое файлов prompt.txt, response.txt, analysis.txt, выводы. Все файлы (requirements.txt, prompt.txt,

run_prompt.py, response.txt, analysis.txt, report.odt/pdf) поместите в одну папку и упакуйте её в zip-архив.

Формат сдачи: один zip-архив, названный <фамилия>_<имя>_PW17.zip, содержащий код, данные и отчёт.

Отчёт ODT или PDF: титульный лист (название ПМ, номер ПР, ФИО, группа), цель работы, описание каждого задания, скриншоты терминала (pip install, запуск скрипта), содержимое файлов prompt.txt, response.txt, analysis.txt, выводы и заключение.

Сдача: отправить готовый zip-архив преподавателю по указанному каналу до конца занятия.

Тема практической работы №18. Оптимизация промптов для работы с большими объемами текстовых данных, объем часов 2

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно оптимизировать промты для работы с большими объёмами текстовых данных, оценивая эффективность по количеству токенов и времени выполнения.

Задание(я):

1. Подготовить набор текстов (10-20 файлов .txt, общий объём ~5 МБ). Сохранить в папке data/.

2. Написать простой скрипт (prompt_baseline.py), который последовательно отправляет каждый файл в LLM с базовым промтом и сохраняет ответы, количество использованных токенов и время выполнения в CSV (baseline.csv).

3. Модифицировать скрипт (prompt_optimized.py), улучшив промт (добавление инструкций, ограничение длины, использование шаблона). Сохранить новые ответы и метрики в CSV (optimized.csv).

4. Построить график сравнения количества токенов и времени выполнения для базового и оптимизированного промтов. Сохранить как comparison.png.

5. Сохранить все скрипты (.py), данные (.txt, .csv) и график в один архив (project.zip).

Методические указания по ходу выполнения работы:

В задании 1 создайте папку data/ и разместите в ней текстовые файлы; назовите их последовательно (doc1.txt, doc2.txt …).

В задании 2 используйте библиотеку openai (pip install openai) или аналогичную для доступа к LLM; реализуйте цикл чтения файлов, отправки запроса и записи результатов в CSV (название файла, токены, время).

В задании 3 измените промт, добавив ясные инструкции (например, «Сократи текст до 3-х предложений»), ограничьте максимальное количество токенов в запросе; результаты сохраняйте в отдельный CSV (optimized.csv).

В задании 4 загрузите оба CSV, постройте два графика (токены vs. файл, время vs. файл) на одном рисунке, подпишите оси и легенду; сохраните график в файл comparison.png (plt.savefig('comparison.png')).

В задании 5 упакуйте все файлы (data/, *.py, *.csv, *.png) в архив project.zip.

Формат сдачи: архив ZIP, содержащий код, данные и отчёт.

Отчёт в ODT или PDF: титульный лист (название работы, ФИО, группа), описание целей и задач, пошаговое описание выполненных действий, таблицы с метриками (из CSV), скриншот графика comparison.png, выводы о влиянии оптимизации промта. Объём отчёта – 2-3 страницы.

Сдача: отправить архив ZIP (project.zip) по электронной почте преподавателю не позднее конца занятия (2 академических часа).

Тема практической работы №19. Создание промпта для анализа тональности текста с использованием искусственного интеллекта, объем часов 2

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно формировать промпт для анализа тональности текста и получать результаты от готовой модели искусственного интеллекта, а также визуализировать полученные метки.

Задание(я):

1. Установить необходимые библиотеки (pip install openai pandas matplotlib). Сохранить список пакетов в файл requirements.txt.

2. Создать набор тестовых предложений (примерно 10-15 строк) и сохранить их в CSV-файл data.csv.

3. Сформировать текст промта, описывающий задачу анализа тональности и содержащий пример запроса-ответа. Сохранить промт в файл prompt.txt.

4. С помощью Python-скрипта отправить запрос к готовой модели (например, GPT) с использованием сохранённого промта и данных из data.csv, получить ответы и сохранить их в CSV-файл results.csv.

5. Проанализировать полученные метки тональности, построить гистограмму распределения (положительные/отрицательные/нейтральные) и сохранить график в файл sentiment.png.

6. Оформить отчёт, включив скриншоты вывода скрипта, таблицу results.csv и график sentiment.png.

Методические указания по ходу выполнения работы:

Для задачи 1 выполните команду установки библиотек в терминале и проверьте их наличие.

Для задачи 2 в LibreOffice Calc или любой таблице создайте колонку "text" и заполните её предложениями, затем экспортируйте в CSV.

Для задачи 3 в текстовом редакторе опишите цель анализа, формат входных данных и пример ожидаемого ответа модели; сохраните файл как prompt.txt в папке проекта.

Для задачи 4 напишите Python-скрипт, который читает data.csv, читает prompt.txt, формирует запрос к модели, получает ответ и записывает результат в results.csv. Скрипт сохраните как sentiment_analysis.py.

Для задачи 5 загрузите результаты из results.csv, подсчитайте количество каждой тональности, постройте гистограмму с помощью matplotlib и сохраните её как sentiment.png (используйте plt.savefig).

Для задачи 6 подготовьте отчёт в ODT или PDF: титульный лист, цель, описание выполненных задач, скриншоты терминала/вывода, таблицу результатов, график, выводы и список использованных файлов.

Все файлы (requirements.txt, data.csv, prompt.txt, sentiment_analysis.py, results.csv, sentiment.png, отчёт) упакуйте в один ZIP-архив.

Формат сдачи: ZIP-архив, отправить по электронной почте преподавателю до конца занятия.

Отчёт должен содержать: титульный лист (название работы, ФИО, группа), цель работы, пошаговое описание выполнения задач, скриншоты вывода скрипта (консоль), таблицу результатов (из results.csv), график распределения тональности (sentiment.png), выводы и список приложенных файлов. Оформление – стандартный стиль ODT/PDF, нумерация страниц.

Сдача: архив ZIP, содержащий все исходные файлы и отчёт, отправить по email преподавателю не позднее конца второго академического часа.

Тема практической работы №20. Разработка промпта для генерации технической документации с применением искусственного интеллекта, объем часов 2

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного создания и тестирования промпта для генерации технической документации с использованием готовой модели искусственного интеллекта

Задание(я):

1. Подготовка среды: установить Python, необходимые пакеты (pip install openai pandas matplotlib joblib) и создать рабочую директорию. Сохранить список установленных пакетов в файл requirements.txt.

2. Сбор примеров технической документации: создать CSV-файл examples.csv, в котором в одной колонке разместить несколько коротких описаний технических задач, а в другой – желаемый формат вывода (например, разделы "Введение", "Требования", "Описание" и т.д.).

3. Формулирование начального промта: написать текстовый файл prompt.txt, содержащий базовый запрос к модели (например, "Сгенерируй техническую документацию для задачи: {запрос}"), где {запрос} будет подставляться из CSV. Сохранить файл в рабочей директории.

4. Написание скрипта генерации: создать Python-скрипт generate.py, который читает examples.csv, подставляет запросы в промт, отправляет их модели (через API), получает ответы и сохраняет полученные документы в отдельные текстовые файлы (doc_1.txt, doc_2.txt …). Сохранить скрипт в рабочей директории.

5. Анализ качества: построить график зависимости длины сгенерированного текста от номера примера, сохранить как length_plot.png, а также сформировать таблицу metrics.csv с колонками «номер», «длина», «оценка качества» (оценка – простая оценка по количеству разделов).

6. Оптимизация промта: изменить текст в prompt.txt (добавить уточнения, например, указать стиль и форматирование), повторить запуск скрипта, сравнить новые метрики с предыдущими и сохранить сравнение в файл comparison.csv.

7. Оформление результатов: собрать все файлы (requirements.txt, examples.csv, prompt.txt, generate.py, doc_*.txt, length_plot.png, metrics.csv, comparison.csv) в один архив project.zip.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 создайте отдельную папку проекта и в ней выполните команды установки пакетов; результат запишите в requirements.txt.

В задании 2 используйте любой табличный редактор (LibreOffice Calc) для создания CSV-файла с примерами запросов и ожидаемыми разделами.

В задании 3 откройте текстовый редактор и запишите базовый промт, сохранив его как prompt.txt.

В задании 4 напишите скрипт, который последовательно обрабатывает строки CSV, формирует запрос к модели, сохраняет ответ в отдельный файл; скрипт сохраняется под именем generate.py.

В задании 5 вычислите длину каждого сгенерированного текста, запишите результаты в metrics.csv и постройте график length_plot.png, используя matplotlib; график сохраняйте через plt.savefig().

В задании 6 отредактируйте prompt.txt, запустите скрипт снова, сравните новые метрики с предыдущими и запишите результаты в comparison.csv.

В задании 7 упакуйте все перечисленные файлы в архив ZIP, назовите его project.zip.

Формат сдачи: один ZIP-архив, содержащий исходный код, данные, графики и отчёт.

Отчёт в формате ODT или PDF. Структура: титульный лист (название практики, ФИО, группа), краткое описание цели, пошаговое описание выполненных заданий, скриншоты кода и терминала, таблицы metrics.csv и comparison.csv, график length_plot.png, выводы о влиянии изменений в промте.

Сдать архив ZIP (project.zip) по электронной почте преподавателю не позже конца текущего учебного занятия.

Тема практической работы №21. Создание промпта для обработки изображений с использованием моделей искусственного интеллекта, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно создавать и тестировать промпт для генерации изображений с помощью готовой модели искусственного интеллекта.

Задание(я):

1. Установить Python-окружение и необходимые библиотеки (pip install diffusers transformers torch).

2. Выбрать открытый набор моделей (например, Stable Diffusion) и загрузить её в скрипт.

3. Сформировать текстовый промт, описывающий требуемое изображение, и сохранить его в файл prompt.txt.

4. Запустить скрипт, передав промт модели, получить сгенерированное изображение и сохранить его как result.png.

5. Сохранить скрипт в файл generate.py, а также экспортировать метаданные (промт, параметры) в CSV (results.csv).

6. Подготовить отчёт, включив титульный лист, описание выполненных шагов, скриншоты консоли и полученного изображения, выводы.

Методические указания по ходу выполнения работы:

Для задания 1 создайте виртуальное окружение, установите перечисленные пакеты через pip.

Для задания 2 в скрипте укажите имя модели и путь к её загрузке; используйте стандартный API библиотеки diffusers.

Для задания 3 в текстовом файле prompt.txt запишите один-единственный строковый запрос, описывающий желаемое изображение (например, "красочный закат над морем в стиле импрессионизма").

Для задания 4 запустите файл generate.py, передав путь к prompt.txt; полученный результат сохраните в файл result.png с помощью функции сохранения изображения.

Для задания 5 вместе с изображением сохраните в results.csv столбцы: prompt, seed, шаги, размер изображения, имя модели.

Для задания 6 оформите отчёт в ODT или PDF: титульный лист, список задач, описание каждого шага, скриншоты вывода терминала и изображения result.png, выводы о качестве промта.

Все файлы (generate.py, prompt.txt, result.png, results.csv, отчёт) упакуйте в один ZIP-архив.

Отчёт в ODT или PDF: титульный лист, перечень выполненных задач, описание каждого шага, скриншоты консольного вывода и изображения result.png, таблица из results.csv, выводы и рекомендации.

Сдача: ZIP-архив с кодом, данными и отчётом отправить по email преподавателю до окончания занятия.

Тема практической работы №22. Работа с промптами для генерации изображений по текстовому описанию, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно формировать промпты и использовать готовую модель искусственного интеллекта для генерации изображений по текстовому описанию, а также сохранять и документировать полученные результаты.

Задание(я):

1. Установить Python-окружение и необходимые библиотеки (pip install torch torchvision diffusers transformers). Результат: файл requirements.txt с перечнем установленных пакетов.

2. Скачивание и подготовка готовой модели (например, Stable Diffusion) через командную строку. Результат: файл model_info.txt с указанием использованной модели и версии.

3. Сформировать набор из 5-6 текстовых промтов (разные стили, тематики). Результат: файл prompts.txt, где каждый промт записан на новой строке.

4. Выполнить генерацию изображений для каждого промта, сохранить полученные файлы в формате PNG. Результат: папка images/ с 5-6 PNG-файлами; каждый файл назвать согласно номеру промта (e.g., prompt_01.png).

5. Сохранить метаданные (промт, параметры генерации, путь к файлу) в таблицу CSV. Результат: файл generation_log.csv.

6. Построить простой гистограммный график распределения средних яркостей сгенерированных изображений и сохранить его в PNG. Результат: файл brightness_histogram.png.

7. Подготовить отчет, включив титульный лист, описание выполненных шагов, скриншоты терминала и сгенерированных изображений, таблицу CSV и график. Результат: файл report.odt (или report.pdf).

Методические указания по ходу выполнения работы:

Перед началом работы создайте отдельную папку проекта и внутри неё подпапки: images/, data/, docs/.

Для установки библиотек используйте команду pip install с указанием всех названий пакетов; зафиксируйте их версии в файле requirements.txt.

Скачивание модели выполните через официальный CLI-инструмент diffusers, указав название модели; информацию о модели запишите в model_info.txt.

Промты запишите построчно в файл prompts.txt, следуя рекомендациям по разнообразию (разные объекты, стили, эпохи).

Для генерации изображений запустите скрипт, который читает prompts.txt, передаёт каждый промт модели и сохраняет результат в папку images/; названия файлов должны соответствовать номерам промтов.

Метаданные (промт, seed, шаги, путь к файлу) экспортируйте в CSV-формат, используя стандартный модуль csv; файл назовите generation_log.csv и разместите в папке data/.

Для расчёта средней яркости каждого изображения откройте файл, преобразуйте его в массив numpy и вычислите среднее значение; соберите данные и постройте гистограмму с помощью matplotlib; сохраните график как brightness_histogram.png в папке docs/.

Отчёт оформите согласно шаблону: титульный лист, цель, список выполненных заданий, скриншоты терминала (команды и вывод), изображения из папки images/, таблицу CSV (в виде вложения) и график. Сохраните в ODT или PDF.

После завершения упакуйте весь проект (все папки, файлы кода, requirements.txt, отчёт) в архив ZIP.

Формат отчёта: ODT или PDF. Структура: титульный лист, цель работы, описание каждого задания, скриншоты выполнения (терминал, генерация), галерея полученных изображений, таблица generation_log.csv (в виде вложения), график brightness_histogram.png, выводы и заключения.

Сдача: архив ZIP с полным содержимым проекта отправить по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №23. Настройка промпта для улучшения качества сгенерированных изображений, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно настраивать промпт для улучшения качества сгенерированных изображений с использованием готовой модели ИИ

Задание(я):

1. Установить необходимые библиотеки (pip install pillow diffusers transformers) и создать рабочую папку. Сохранить скрипт в файл prompt_setup.py.

2. С помощью готовой модели (например, Stable Diffusion) сгенерировать набор изображений по базовому промту. Сохранить изображения в папку images/base.

3. Проанализировать полученные изображения и сформулировать гипотезы по улучшению промта (добавление стилей, уточнение деталей).

4. Модифицировать промт согласно гипотезам, сгенерировать новые изображения, сохранить их в папку images/optimized.

5. Сравнить результаты: построить график количественной оценки (например, средний рейтинг качества) и сохранить в файл comparison.png.

6. Оформить отчет, включив описание шагов, скриншоты изображений, график сравнения и выводы.

Методические указания по ходу выполнения работы:

Для задания 1 выполните pip install указанных пакетов, создайте папки images/base и images/optimized, сохраните основной скрипт в prompt_setup.py.

В задании 2 запустите скрипт с базовым промтом, результаты сохраняйте в images/base, каждое изображение именуйте последовательно (img_01.png и т.д.).

Задание 3 – визуально оцените изображения, запишите наблюдения в текстовый файл notes.txt.

В задании 4 измените строку с промтом в скрипте согласно записям из notes.txt, запустите генерацию и сохраните новые файлы в images/optimized.

В задании 5 используйте библиотеку matplotlib для построения столбчатой диаграммы сравнения оценок, сохраните график командой plt.savefig('comparison.png').

В задании 6 подготовьте отчет в формате ODT или PDF: титульный лист, описание каждого задания, скриншоты примеров изображений, график comparison.png, выводы. Сохраните отчет как report.odt.

Все файлы (prompt_setup.py, notes.txt, images/*, comparison.png, report.odt) упакуйте в один ZIP-архив.

Отчет в ODT или PDF: титульный лист, короткое описание целей, пошаговое описание выполненных заданий, скриншоты примеров изображений (по 2-3 изображения из каждой папки), график comparison.png, таблица с оценками, выводы.

Сдать: ZIP-архив с кодом, данными и отчётом отправить по email преподавателю до конца занятия.

Тема практической работы №24. Оптимизация промптов для различных типов мультимедиа (изображения, видео) с использованием искусственного интеллекта, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного подбора и оптимизации промптов для генерации изображений и видеоконтента с помощью готовых моделей ИИ.

Задание(я):

1. Подготовка окружения. Установить необходимые Python-пакеты (pip install transformers diffusers torch opencv-python pandas matplotlib joblib). Сохранить список установленных пакетов в файл requirements.txt.

2. Выбор базовых промтов. Сформировать 3-5 простых текстовых запросов для генерации изображений и 2-3 запросов для генерации коротких видеоклипов. Записать их в файл prompts.csv.

3. Генерация контента. С помощью выбранных готовых моделей (например, Stable Diffusion для изображений и ModelScope VideoDiffusion для видео) сгенерировать медиаресурсы по каждому промту. Сохранить изображения в папку images/, видеоклипы – в папку videos/.

4. Оценка качества. Для изображений вычислить метрику сходства (например, CLIP-score) и сохранить результаты в CSV-файл metrics_images.csv. Для видео – оценить субъективный балл (например, “уровень детализации”) и сохранить в metrics_videos.csv.

5. Оптимизация промтов. На основе полученных метрик изменить формулировку промтов (добавить уточнения, изменить стиль) и повторить генерацию и оценку для улучшения результатов. Сохранить новые промты в файл prompts_optimized.csv, а новые метрики – в metrics_optimized.csv.

6. Визуализация результатов. Построить графики зависимости метрик от версии промта (исходный vs. оптимизированный) для изображений и видео, сохранить графики как images_score.png и videos_score.png.

7. Сохранение модели. Если использовалась дообучаемая часть модели, сохранить её состояние с помощью joblib.dump в файл model_optimized.joblib.

8. Подготовка отчёта. Оформить отчёт в ODT или PDF согласно шаблону (титульный лист, описание каждого задания, скриншоты сгенерированных медиа, таблицы метрик, графики, выводы).

Методические указания по ходу выполнения работы:

Создайте отдельные каталоги: data/, images/, videos/, results/.

Все CSV-файлы сохраняйте в папку data/.

Графики сохраняйте функцией plt.savefig('results/имя.png').

Модели сохраняйте командой joblib.dump(модель, 'results/model_optimized.joblib').

Код каждого шага разместите в отдельном .py файле (e.g., 01_setup.py, 02_generate.py, …) и сохраните в корневой каталог.

Для оценки CLIP-score установите библиотеку transformers и используйте предобученную модель CLIP; результаты запишите в CSV.

Отчёт упакуйте вместе с каталогом results/ и всеми .py файлами в один ZIP-архив.

Формат сдачи: архив ZIP с кодом, данными, графиками и отчётом.

Отчёт в ODT или PDF: титульный лист (название работы, ФИО, группа), краткое описание цели, пошаговое описание выполненных заданий, таблицы метрик (из CSV), скриншоты сгенерированных изображений и видеоклипов, графики (images_score.png, videos_score.png), выводы о влиянии оптимизации промтов.

Сдать архив ZIP через электронную почту преподавателя не позднее конца учебного дня, указав в теме письма «Практика 24 – Оптимизация промтов».

Тема практической работы №25. Разработка промпта для голосовых ассистентов на основе искусственного интеллекта, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно разрабатывать, тестировать и оценивать промпты для голосовых ассистентов на основе искусственного интеллекта.

Задание(я):

1. Изучить принципы построения промтов для голосовых ассистентов (структура, стилистика, ограничения). Результат: краткие конспекты сохранить в файл "notes.txt".

2. Сформулировать три варианта промта для заданного сценария (например, запрос погоды). Сохранить варианты в файл "prompts.txt".

3. Подготовить простую тестовую программу на Python, использующую библиотеку openai (pip install openai) для отправки промтов к модели и получения ответа. Сохранить код в файле "test_prompt.py".

4. Запустить тесты для каждого варианта промта, собрать ответы и оценить их по критериям (корректность, полнота, естественность). Сохранить результаты в файл "evaluation.csv" и построить график сравнения метрик (plt.savefig('metrics.png')).

5. Оформить выводы, добавить скриншоты консольного вывода и графика в отчет.

Методические указания по ходу выполнения работы:

Для задания 1 используйте учебные материалы курса и интернет-ресурсы, но не копируйте готовый текст; оформите собственные заметки в простом текстовом файле.

В задании 2 каждый промт должен быть записан в отдельной строке файла "prompts.txt"; укажите номер варианта.

В задании 3 создайте скрипт, который считывает промты из "prompts.txt", отправляет их модели и сохраняет ответы в CSV-файл; код сохраняйте в "test_prompt.py".

В задании 4 при оценке используйте балльную шкалу 1-5 по каждому критерию; результаты запишите в "evaluation.csv"; построенный график сохраните как "metrics.png".

В задании 5 подготовьте отчёт в формате ODT или PDF: титульный лист, описание выполненных заданий, таблицу оценок, график "metrics.png" и скриншоты вывода программы.

Формат сдачи: один ZIP-архив, содержащий папку с файлами "notes.txt", "prompts.txt", "test_prompt.py", "evaluation.csv", "metrics.png" и готовый отчёт.

Отчёт: ODT или PDF, титульный лист (название работы, ФИО, группа), раздел «Описание заданий», таблица оценок из "evaluation.csv", график "metrics.png", скриншоты консольного вывода, выводы и рекомендации.

Сдача: упаковать все указанные файлы в ZIP-архив и отправить по электронной почте преподавателю не позднее конца занятия.

Тема практической работы №26. Создание промпта для управления умными устройствами с помощью голосовых команд, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц**.**

Цель практической работы:

Освоить навыки самостоятельного создания и тестирования промтов для управления умными устройствами с помощью голосовых команд

Задание(я):

1. Исследовать требования к промту (какие устройства, какие команды, форматы ответов).

2. Сформулировать текст промта, учитывая ясность и ограничения LLM. Сохранить в файл prompt.txt.

3. Реализовать простую консольную программу (script.py), которая принимает текстовый ввод имитирующий голосовую команду, отправляет её вместе с промтом в модель и выводит ответ. Сохранить журнал запросов-ответов в файл log.csv.

4. Оценить полученные ответы, при необходимости скорректировать промт и повторить тестирование. Сохранить финальный вариант промта и журнал.

Методические указания по ходу выполнения работы:

В задании 1 изучите типичные сценарии управления умными устройствами (включить свет, регулировать температуру и т.п.) и запишите их в таблице.

В задании 2 напишите промт, включив в него инструкцию модели отвечать только конкретными командами устройства; сохраните текст в файл prompt.txt в рабочей папке.

В задании 3 создайте файл script.py; в нём реализуйте ввод строки от пользователя, формирование запроса к LLM (используйте pip install openai), вывод ответа на экран и запись пары запрос-ответ в log.csv.

В задании 4 проанализируйте ответы из log.csv, сравните их с ожидаемыми действиями, при необходимости отредактируйте prompt.txt и повторите шаг 3; окончательные файлы сохраняйте в той же папке.

Все результаты (prompt.txt, script.py, log.csv) упакуйте в один ZIP-архив.

Формат сдачи: архив ZIP с кодом, данными и отчётом.

Отчёт в ODT или PDF: титульный лист, описание каждого задания, скриншоты работы консоли, содержимое prompt.txt, таблица с результатами из log.csv, выводы и возможные улучшения.

Сдача: отправить ZIP-архив по email преподавателю до конца занятия.

Тема практической работы №27. Оптимизация промпта для улучшения распознавания речи моделью искусственного интеллекта, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Освоить навыки самостоятельного подбора и оптимизации промптов (подсказок) для улучшения качества распознавания речи готовой моделью искусственного интеллекта

Задание(я):

1. Подготовка рабочей среды (установить Python, pip, необходимые библиотеки: torch, transformers, datasets, jiwer). Время выполнения – 30 минут. Результат: файл requirements.txt с перечнем установленных пакетов.

2. Загрузка предобученной модели распознавания речи (например, Whisper) и проверка её работы на одном аудиофайле из открытого датасета (например, небольшая часть Common Voice). Результат: файл audio_input.wav и файл transcript_initial.txt с полученным текстом.

3. Формирование базового промта (по умолчанию) и запись полученных метрик (WER) в CSV-файл. Результат: файл metrics_initial.csv.

4. Разработка и применение изменённого промта (добавление указания языка, контекста, ограничений длины и т.п.), повторное распознавание того же аудио, сравнение метрик с базовым вариантом, построение графика

зависимости WER от вариантов промта. Результат: файл metrics_optimized.csv, график wer_comparison.png.

5. Сохранение финальной версии модели (если применялась дообучка), экспорт всех полученных артефактов (CSV, PNG, .py скрипты) в единую папку и упаковка её в ZIP-архив. Результат: архив project_pr27.zip.

6. Подготовка отчёта о выполненной работе (титульный лист, описание целей, пошаговый процесс, скриншоты терминала, таблицы метрик, график, выводы). Результат: файл report.odt (или report.pdf).

Методические указания по ходу выполнения работы:

Для выполнения задания 1 создайте файл requirements.txt, указав в нём названия и версии всех установленных пакетов (pip freeze > requirements.txt).

В задании 2 сохраните исходный аудиофайл в папку data/ и запишите полученный транскрипт в файл transcript_initial.txt в папку results/.

В задании 3 рассчитайте метрику Word Error Rate (WER) с помощью библиотеки jiwer, запишите значение в CSV-файл metrics_initial.csv (колонки: prompt_type, wer).

В задании 4 измените текст промта, повторите распознавание, заново вычислите WER и добавьте запись в CSV-файл metrics_optimized.csv. Постройте график сравнения (WER по типу промта) и сохраните его как wer_comparison.png (plt.savefig('wer_comparison.png')).

В задании 5 сохраните все скрипты (.py), CSV-файлы, PNG-график и файл requirements.txt в одну директорию, затем упакуйте её в архив ZIP под названием project_pr27.zip.

В задании 6 оформите отчёт согласно шаблону: титульный лист, цель работы, описание каждого задания, скриншоты вывода консоли, таблицы метрик, график, выводы и список использованных библиотек. Сохраните отчёт в формате ODT или PDF.

Формат сдачи: один ZIP-архив, содержащий папку с кодом, данными, графиками, CSV-файлами и готовый отчёт.

Формат отчёта: ODT или PDF. Структура: титульный лист; цель практической работы; последовательное описание выполненных заданий; скриншоты терминала/IDE; таблицы метрик в виде CSV-файлов (вставить

как изображения); график wer_comparison.png; выводы о влиянии промта на качество распознавания; список использованных библиотек и их версии.

Сдача: упаковать все материалы в один ZIP-архив (project_pr27.zip) и отправить по email преподавателю до конца занятия.

Тема практической работы №28. Разработка промпта для автоматической транскрибации голоса в текст с использованием искусственного интеллекта, объем часов 4

У6. Формировать запросы для получения данных из моделей ИИ, представлять результаты в виде графиков и таблиц.

Цель практической работы:

Научиться самостоятельно разрабатывать промпт и интегрировать готовую модель распознавания речи для получения текста из аудио-файла.

Задание(я):

1. Подготовка окружения: установить Python и необходимые библиотеки (pip install openai-whisper, pip install matplotlib, pip install pandas).

2. Получение тестового аудио: скачать открытый аудиофайл (например, «sample.wav») и разместить его в рабочей папке.

3. Формулировка промта: написать текстовое описание задачи для модели (например, «Транскрибировать речь в файле sample.wav, вывести результат без пунктуации»).

4. Реализация скрипта: написать Python-программу, которая загружает модель Whisper, передаёт в неё промт и аудио, получает транскрипт.

5. Сохранение и визуализация результатов: сохранить полученный текст в файл transcript.txt, измерить длительность аудио и количество символов, построить график «длительность (сек) – количество символов», сохранить график как plot.png.

6. Оформление отчёта и упаковка: подготовить отчёт, включить скриншоты кода и графика, собрать все файлы (скрипт .py, transcript.txt, plot.png, исходный аудио) в ZIP-архив.

Методические указания по ходу выполнения работы:

Для задания 1 используйте командную строку: pip install openai-whisper, pip install matplotlib, pip install pandas. Убедитесь, что Python версии 3.8+ установлен.

Для задания 2 поместите аудиофайл в подпапку "data" проекта. Назовите файл "sample.wav".

Для задания 3 промт необходимо записать в виде строки переменной prompt в скрипте; он будет передан модели вместе с путём к аудио.

Для задания 4 скрипт должен: загрузить модель Whisper, вызвать её с параметром prompt и путём к файлу, получить результат и вывести его в консоль.

Для задания 5: сохранить полученный текст в файл "transcript.txt" (использовать стандартный метод записи). Рассчитать длительность аудио (сек) и количество символов в тексте, построить график с помощью matplotlib, сохранить его командой plt.savefig('plot.png').

Для задания 6: в отчёте оформить титульный лист, краткое описание каждого задания, скриншоты кода и графика, выводы о работе модели. Сохранить отчёт в формате ODT или PDF. Упаковать в архив ZIP все файлы проекта (скрипт .py, data/sample.wav, transcript.txt, plot.png, отчёт).

Отчёт: ODT или PDF, титульный лист (название практики, ФИО, группа, дата), раздел «Описание выполнения задач» (по каждому пункту), раздел «Результаты» (скриншоты кода, график plot.png, фрагмент transcript.txt), раздел «Выводы».

Сдача: архив ZIP с кодом, данными и отчётом отправить по email преподавателя до конца занятия.

Тема практической работы №29. Тестирование эффективности промптов на реальных данных при работе с моделями искусственного интеллекта, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного тестирования эффективности промтов (запросов) при работе с готовыми моделями искусственного интеллекта на реальных данных

Задание(я):

1. Подготовка окружения и данных. Установить необходимые библиотеки, загрузить открытый датасет (например, датасет Iris) и сохранить его в файл data.csv.

2. Формирование набора промтов. Составить 5-6 разных формулировок запросов к модели, записать их в файл prompts.txt.

3. Получение ответов модели. С помощью выбранного API (например, OpenAI) выполнить запросы из prompts.txt к модели, сохранить ответы в файл responses.csv.

4. Оценка качества ответов. Для каждого ответа рассчитать метрику точности (например, сравнение с известными метками из датасета) и сохранить результаты в файл metrics.csv.

5. Визуализация результатов. Построить график зависимости метрики от номера промта, сохранить как plot.png.

6. Оформление отчёта. Подготовить документ ODT/PDF с титульным листом, описанием выполненных шагов, таблицами данных, скриншотами кода и графика, выводами.

Методические указания по ходу выполнения работы:

Установить Python и выполнить pip install openai pandas matplotlib joblib.

Создать каталог проекта, внутри него папки data, results, report.

Сохранить исходный датасет в data/data.csv.

Записать промты построчно в data/prompts.txt.

Скрипт для обращения к модели должен читать промты из файла, отправлять их через API и сохранять ответы в results/responses.csv.

Для оценки сравнить полученные ответы с целевыми метками из data/data.csv, вычислить точность и записать в results/metrics.csv.

График построить из results/metrics.csv, сохранить в results/plot.png.

Отчёт разместить в папке report, включив в него скриншоты файлов, таблиц и графика.

Для сдачи упаковать весь каталог проекта в архив ZIP.

Отчёт в формате ODT или PDF: титульный лист, цель работы, описание каждого задания, таблицы данных (data.csv, responses.csv, metrics.csv), скриншоты кода, график plot.png, выводы о влиянии разных промтов на качество модели.

Сдать архив ZIP с полным проектом (код, данные, результаты, отчёт) по e-mail преподавателя не позднее конца занятия.

Тема практической работы №30. Создание отчета по результатам работы промптов с моделями искусственного интеллекта, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного формирования, анализа и документирования результатов работы с готовыми моделями искусственного интеллекта при помощи промптов.

Задание(я):

1. Установить необходимые библиотеки Python (transformers, torch, pandas, matplotlib) и создать скрипт *.py для генерации ответов модели на набор подготовленных промтов. Результаты сохранить в файл responses.csv.

2. Провести базовый анализ полученных ответов: вычислить длину текста (кол-во символов) для каждого ответа, построить гистограмму распределения длин. График сохранить в файл length_distribution.png.

3. Сформировать отчёт ODT или PDF со структурой: титульный лист, цель работы, описание использованной модели и промтов, таблица-фрагмент из responses.csv, гистограмма length_distribution.png, выводы и рекомендации. Сохранить файл report.odt (или report.pdf).

Методические указания по ходу выполнения работы:

Для выполнения задания 1 установите пакеты командой pip install transformers torch pandas matplotlib в командной строке. В скрипте реализуйте загрузку предобученной модели из библиотеки transformers, чтение списка промтов из собственного текстового файла, получение ответов модели и запись пар «промт – ответ» в CSV-файл responses.csv. Сохраните скрипт под именем generate_responses.py.

Для задания 2 создайте отдельный скрипт analysis.py, в котором загрузите responses.csv с помощью pandas, вычислите длину каждого ответа, постройте гистограмму с помощью matplotlib и сохраните её в length_distribution.png с помощью plt.savefig('length_distribution.png').

В задании 3 подготовьте отчёт в LibreOffice Writer (или аналогичном редакторе). Титульный лист оформите согласно требованиям вуза (название практики, ФИО студента, группа, дата). В разделе «Методика» опишите, какие промты использовались, какая модель была загружена и как получены ответы. Вставьте таблицу-фрагмент из responses.csv (первые 5-10 строк) и график length_distribution.png. В разделе «Выводы» сформулируйте наблюдения о характере полученных ответов.

Все файлы (generate_responses.py, analysis.py, responses.csv, length_distribution.png, report.odt/pdf) упакуйте в один архив ZIP для сдачи.

Формат сдачи: архив ZIP с кодом, данными и отчётом, отправить по e-mail преподавателю до конца занятия.

Отчёт ODT или PDF с титульным листом, цель работы, описание используемой модели и промтов, таблицей-фрагментом из responses.csv, графиком length_distribution.png, выводами и рекомендациями. В отчёте должны быть скриншоты окна терминала с выводом команд и результаты анализа.

Сдача: упаковать все указанные файлы в архив ZIP и отправить по e-mail преподавателю до конца занятия.

Тема практической работы №31. Оптимизация промпта на основе результатов работы модели искусственного интеллекта, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно оптимизировать промт для готовой модели искусственного интеллекта, анализируя полученные ответы и улучшая формулировку запроса.

Задание(я):

1. Подготовить рабочее окружение и установить необходимые библиотеки. Сохранить список установленных пакетов в файл requirements.txt.

2. Реализовать скрипт prompt_opt.py, который принимает промт, генерирует ответ модели и сохраняет текст ответа в файл, а также вычисляет простые метрики (длина, количество слов). Сохранить результаты в CSV файл results.csv.

3. Провести первую генерацию с базовым промтом, сохранить ответ и метрики. Затем изменить промт, повторить генерацию, добавить новые строки в results.csv.

4. Построить график зависимости выбранной метрики (например, длина ответа) от номера итерации, сохранить изображение как metric_plot.png и включить его в отчёт.

Методические указания по ходу выполнения работы:

Для выполнения задания 1 используйте pip install transformers torch pandas matplotlib joblib и запишите версии пакетов в requirements.txt.

В задании 2 создайте файл prompt_opt.py, реализуйте загрузку модели gpt2 через библиотеку transformers, реализуйте функцию генерации текста и расчёта метрик, сохраняйте ответы в отдельные txt файлы (answer_1.txt, answer_2.txt) и метрики в results.csv.

В задании 3 сначала задайте простой промт (например, "Расскажи о пользе здорового питания.") и запустите скрипт, затем сформулируйте улучшенный промт, добавив уточнение (например, "В виде списка перечисли пять преимуществ здорового питания.") и повторите запуск, результаты добавьте в results.csv.

В задании 4 используйте matplotlib для построения линейного графика, где по оси X – номер итерации (1,2), по оси Y – выбранная метрика. Сохраните график командой plt.savefig('metric_plot.png').

Формат сдачи: архив ZIP, содержащий файл отчёта (ODT или PDF), папку с кодом (prompt_opt.py, requirements.txt), папку с полученными текстовыми ответами, файл results.csv и изображение metric_plot.png.

Отчёт оформляется в ODT или PDF, содержит титульный лист, описание целей и задач, пошаговое описание выполненных действий, скриншоты вывода скрипта, таблицу results.csv, график metric_plot.png и выводы о влиянии изменений промта.

Сдача: упаковать всё перечисленное в ZIP-архив и отправить по электронной почте преподавателю не позже конца занятия.

Тема практической работы №32. Тестирование промпта с вариациями структуры запросов к модели искусственного интеллекта, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно тестировать промпты с различными вариантами структуры запросов к модели искусственного интеллекта и анализировать полученные ответы.

Задание(я):

1. Подготовка среды: установить необходимые Python-пакеты, создать рабочую папку.

2. Сформировать набор тестовых запросов: минимум 5 базовых промтов и 3-4 их варианта (изменить порядок, добавить уточняющие фразы). Сохранить в файл prompts.txt.

3. Реализовать скрипт (prompt_test.py), который поочерёдно отправляет запросы к модели, фиксирует время ответа и сохраняет полученный текст в CSV-файл results.csv.

4. Сохранить полученные ответы и метрики: CSV-файл, а также скрипт для построения графика зависимости длины ответа от времени (plot.py).

5. Построить график (длина текста vs. время) и сохранить как plot.png.

6. Оформить отчёт, включив скриншоты терминала, фрагменты CSV и графика.

Методические указания по ходу выполнения работы:

Для установки пакетов используйте pip install openai pandas matplotlib.

Создайте папку work32 и внутри файлы prompts.txt, prompt_test.py, plot.py.

В prompts.txt запишите каждый запрос в отдельной строке.

В скрипте prompt_test.py реализуйте цикл чтения строк из prompts.txt, отправку их к модели (используйте API-ключ), измерение времени выполнения (time.time()), запись в CSV: запрос, ответ, длина ответа, время.

Сохраните CSV-файл в той же папке (results.csv).

В plot.py загрузите results.csv, постройте точечный график (по оси X – время, по оси Y – длина ответа) и сохраните как plot.png (plt.savefig('plot.png')).

Все файлы (prompt_test.py, plot.py, prompts.txt, results.csv, plot.png) упакуйте в один ZIP-архив.

Отчёт оформите в ODT или PDF: титульный лист, цель, список заданий, описание процесса, скриншоты выполнения (терминал, содержимое CSV, графика), выводы.

В архиве должны находиться: отчёт, все .py-файлы, prompts.txt, results.csv, plot.png.

Отчёт: ODT или PDF, титульный лист, цель, перечень заданий, пошаговое описание, скриншоты (терминал, CSV-фрагмент, график), выводы и рекомендации.

Сдача: ZIP-архив с кодом, данными и отчётом отправить по e-mail преподавателю до конца занятия.

Тема практической работы №33. Сравнение эффективности промптов при решении различных задач с использованием искусственного интеллекта, объем часов 2

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно сравнивать эффективность разных промптов при решении типовых задач с использованием готовых моделей искусственного интеллекта

Задание(я):

1. Установить необходимые Python-библиотеки (openai, pandas, matplotlib, tqdm). Сохранить список пакетов в файл requirements.txt.

2. Сформировать набор из трёх типовых задач (например, генерация текста, суммирование, ответы на вопросы) и для каждой задачи подготовить три варианта промта (короткий, расширенный, с указанием стиля). Сохранить промты в файл prompts.json.

3. С помощью API готовой модели (например, gpt-3.5-turbo) выполнить запросы для всех комбинаций задача-промт, собрать ответы и сохранить их в CSV-файл responses.csv.

4. Оценить полученные ответы по двум простым метрикам: (a) субъективная оценка качества (1-5) и (b) время отклика (секунды). Добавить оценки в CSV-файл.

5. Построить столбчатый график, сравнивающий среднюю оценку качества и среднее время отклика для разных типов промтов. Сохранить график в файл comparison.png.

6. Оформить отчёт, включив описание задач, таблицу с результатами, скриншоты кода и графика, выводы о том, какой тип промта более эффективен.

Методические указания по ходу выполнения работы:

Для задания 1 выполните команду pip install -r requirements.txt и проверьте, что библиотеки импортируются без ошибок. Сохраните файл requirements.txt в корень проекта.

Для задания 2 создайте файл prompts.json со структурой: {"task_name": [{"prompt_id": "short", "text": "..."}, ...]}. Каждый промт должен быть отдельной строкой. Файл разместите в папке data.

Для задания 3 напишите скрипт run_prompts.py, который читает prompts.json, последовательно отправляет запросы к модели, измеряет время выполнения и записывает ответы и метрики в responses.csv. Файл CSV разместите в папке results.

Для задания 4 откройте responses.csv в pandas, добавьте два столбца: quality (ручная оценка) и latency (время). Сохраните обновлённый CSV-файл в той же папке results.

Для задания 5 создайте скрипт plot_comparison.py, который читает results/responses.csv, вычисляет средние значения по каждому типу промта и строит столбчатый график с двумя наборами столбцов (качество, время). Сохраните график как results/comparison.png.

Для задания 6 подготовьте отчёт в формате ODT или PDF. В отчёте должны быть:

• Титульный лист (название работы, ФИО, группа, дата).

• Описание целей и задач.

• Таблица с результатами из responses.csv (можно вставить скриншот).

• Скриншоты кода из файлов run_prompts.py и plot_comparison.py.

• График comparison.png.

• Выводы о влиянии формулировки промта на качество и скорость ответа.

Формат сдачи: архив ZIP, содержащий папки data, results, скрипты .py, файл requirements.txt, отчёт ODT/PDF.

Отчёт ODT или PDF: титульный лист, раздел «Цели и задачи», описание набора задач и промтов, таблица результатов (скриншот CSV), скриншоты кода, график comparison.png, выводы и рекомендации.

Сдача: упаковать все файлы в архив ZIP и отправить по email преподавателя не позднее окончания 2-х академических часов.

Тема практической работы №34. Работа с промптами для решения сложных аналитических задач с использованием искусственного интеллекта, объем часов 4

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Научиться самостоятельно создавать и использовать промпты для получения решений аналитических задач с помощью готовых моделей искусственного интеллекта

Задание(я):

1. Подготовка среды разработки и создание файла prompts.md (результат сохранить в файл).

2. Формулирование трёх промтов для анализа датасета Iris (результат сохранить в prompts.md).

3. Выполнение промтов через Python-скрипт, сохранение полученных ответов в отдельные файлы code1.py, code2.py, code3.py; запуск полученного кода, сохранение графиков (PNG), модели (joblib) и таблиц (CSV).

4. Оформление отчёта: включить описание задач, скриншоты выполнения, ссылки на сохранённые файлы и выводы (результат – ODT или PDF).

Методические указания по ходу выполнения работы:

В задании 1 установите Python, выполните pip install openai pandas matplotlib scikit-learn joblib; создайте рабочую папку и файл prompts.md; время выполнения – 30 минут.

В задании 2 напишите три промта: (a) описание структуры датасета Iris, (b) построение гистограмм и диаграмм, (c) обучение простой модели

классификации; сохраняйте каждый промт в отдельной строке файла prompts.md; время – 1 час.

В задании 3 создайте Python-скрипт, который читает prompts.md, поочерёдно отправляет каждый промт модели (используйте openai), сохраняет полученный код в файлы code1.py-code3.py; запустите каждый полученный скрипт, полученные графики сохраняйте через plt.savefig('имя.png'), модель – joblib.dump('model.pkl'), таблицы – df.to_csv('имя.csv'); время – 1,5 часа.

В задании 4 подготовьте отчёт: титульный лист, цель, описание каждого задания, скриншоты консольного вывода и графиков, ссылки на файлы code*.py, *.png, *.pkl, *.csv, выводы о работе с промтами; время – 1 час.

Формат сдачи: архив ZIP, содержащий папку с кодом, данными, графиками и отчётом.

Отчёт в ODT или PDF: титульный лист, цель, последовательность заданий, скриншоты результатов, таблицы и графики, выводы; все файлы упомянуты в отчёте.

Сдача: отправить архив ZIP по email преподавателю до конца занятия.

Тема практической работы №35. Изучение влияния параметров промпта на качество работы модели искусственного интеллекта, объем часов 4

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного исследования влияния параметров промпта (длина, температура, top_p) на качество ответов модели искусственного интеллекта

Задание(я):

1. Установить необходимые Python-библиотеки (pip install openai pandas matplotlib joblib). Сохранить список пакетов в файл requirements.txt. (30 минут)

2. Сформировать набор тестовых запросов (10-15 вопросов) и записать их в CSV-файл data/prompts.csv (колонки id, prompt).

3. Реализовать скрипт src/prompt_experiment.py, который последовательно отправляет каждый запрос к модели с разными значениями температуры (0.2, 0.5, 0.8) и top_p (0.8, 0.9, 1.0), сохраняет ответы и параметры в CSV-файл output/results.csv.

4. Добавить в results.csv расчёт простых метрик качества: длина ответа (символы) и количество совпадений с эталоном (если задан).

5. С помощью matplotlib построить графики зависимости длины ответа от температуры и от top_p, сохранить их как PNG-файлы output/temperature_vs_length.png и output/top_p_vs_length.png.

6. Сформировать отчёт (ODT или PDF) с титульным листом, описанием целей, последовательностью выполнения, скриншотами кода, таблицей results.csv, графиками и выводами о влиянии параметров.

Методические указания по ходу выполнения работы:

Для задания 1 откройте терминал, выполните команды установки, затем выполните pip freeze > requirements.txt и поместите файл в корень проекта.

Для задания 2 создайте CSV-файл с двумя колонками (id, prompt) и сохраните его в папку data; каждый запрос должен быть отдельной строкой.

Для задания 3 скрипт должен считывать prompts.csv, формировать запросы к модели, менять параметры (temperature, top_p), получать ответы и записывать их вместе с параметрами в output/results.csv; используйте библиотеку pandas для чтения/записи.

Для задания 4 добавьте в скрипт расчёт длины строки ответа (len(answer)) и простого сравнения с эталоном (если он присутствует в отдельном CSV-файле); результаты запишите в новые колонки metrics_length и metrics_match.

Для задания 5 постройте линейные графики (temperature vs metrics_length, top_p vs metrics_length) с помощью matplotlib; каждый график сохраните функцией plt.savefig('имя.png') в папку output.

Для задания 6 подготовьте документ согласно шаблону: титульный лист (название практики, ФИО, группа), цель, описание выполненных заданий, скриншоты кода из src, таблицу результатов (скриншот или вставка CSV), графики PNG, выводы о влиянии параметров.

Формат сдачи: архив ZIP, содержащий папки data, output, src, файл requirements.txt и готовый отчёт.

Отчёт в формате ODT или PDF. Структура: титульный лист, цель практики, список выполненных заданий, скриншоты кода, таблица результатов (из results.csv), графики (PNG), раздел выводов о влиянии параметров промта.

Упаковать все файлы проекта в архив ZIP и отправить по электронной почте преподавателю до конца учебного дня.

Тема практической работы №36. Улучшение точности промпта для специфических задач с использованием искусственного интеллекта, объем часов 4

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного улучшения точности запросов (промтов) к модели искусственного интеллекта для решения конкретной задачи, используя методику анализа, итеративного уточнения и оценки результатов.

Задание(я):

1. Подготовка рабочего окружения (установка Python, необходимых библиотек).

2. Формулировка базового промта для выбранной задачи (например, классификация отзывов по тональности).

3. Получение ответов модели по базовому промту и сохранение результатов в CSV.

4. Анализ полученных ответов: выявление типичных ошибок и неточностей.

5. Разработка улучшенного промта (добавление контекста, примеров, уточнений).

6. Получение ответов модели по улучшенному промту, сохранение новых результатов в CSV.

7. Сравнительный анализ точности базового и улучшенного промтов (расчёт метрик, построение графика).

8. Оформление отчёта и упаковка всех материалов в архив.

Методические указания по ходу выполнения работы:

Для выполнения работы создайте отдельную папку проекта и внутри неё подпапки: data (для CSV-файлов), results (для графиков), code (для .py файлов).

Установите необходимые пакеты командой: pip install openai pandas matplotlib scikit-learn.

В файле code/01_prepare_environment.py опишите импорт библиотек и загрузку API-ключа (ключ храните в отдельном файле .env, не включайте его в архив).

В файле code/02_baseline_prompt.py сформулируйте простой промт, отправьте запрос к модели и сохраните полученные ответы в data/baseline_results.csv.

В файле code/03_analyze_baseline.py проведите сравнение ответов с эталонными метками (эти метки можно задать вручную в CSV) и запишите обнаруженные типичные ошибки в файл results/baseline_analysis.txt.

В файле code/04_improved_prompt.py создайте улучшенный промт, учитывающий замеченные недостатки, и сохраните ответы модели в data/improved_results.csv.

В файле code/05_evaluation.py рассчитайте метрики точности (например, точность, полноту, F1) для базового и улучшенного промтов, постройте график сравнения и сохраните его как results/accuracy_comparison.png.

В файле code/06_package.py соберите все файлы проекта в один архив ZIP (включите папки data, results, code и отчёт).

Отчёт оформляйте в ODT или PDF, структура отчёта: титульный лист, цель работы, описание каждого задания, скриншоты запросов/ответов, таблицы результатов, график сравнения, выводы о влиянии улучшения промта.

Все графики сохраняйте функцией plt.savefig('results/имя_файла.png'), а модели (если они использовались) – через joblib.dump. Данные экспортируйте в CSV через pandas.DataFrame.to_csv.

Отчёт должен быть в формате ODT или PDF, включать титульный лист, цель практики, пошаговое описание выполнения заданий, скриншоты запросов и ответов модели, таблицы с результатами (CSV-файлы), график сравнения точности (PNG) и раздел с выводами о влиянии улучшенного промта.

Сдача: упаковать все материалы (папки data, results, code и отчёт) в один ZIP-архив и отправить по электронной почте преподавателю до конца занятия.

Тема практической работы №37. Разработка промпта для работы с чувствительными данными с использованием искусственного интеллекта, объем часов 4

У4. Осуществлять мониторинг качества обучения моделей, выявлять отклонения и проблемы в результатах работы.

У5. Подготавливать отчёты и документировать результаты работы с моделями ИИ, используя стандарты и требования к оформлению.

Цель практической работы:

Освоить навыки самостоятельного создания и проверки безопасного промпта для работы с чувствительными данными в системе искусственного интеллекта.

Задание(я):

1. Ознакомление с принципами защиты чувствительных данных при использовании LLM.

2. Анализ требований к безопасному вводу и выводу данных.

3. Формулирование собственного промта, учитывающего ограничения по передаче конфиденциальной информации.

4. Тестирование разработанного промта с набором примерных запросов, фиксирование ответов и их соответствия требованиям.

5. Оформление отчёта: описание проделанных шагов, скриншоты тестов, выводы и рекомендации.

Методические указания по ходу выполнения работы:

В задании 1 изучите общие рекомендации по работе с конфиденциальной информацией и особенности LLM, используя учебные материалы курса и открытые источники.

В задании 2 составьте список требований к промту (например, отсутствие прямого указания на личные данные, использование маскирования, запрос подтверждения от пользователя). Сохраните список в текстовый файл requirements.txt.

В задании 3 на основе требований сформулируйте промт, который будет использоваться для взаимодействия с моделью. Сохраните готовый промт в файл prompt.txt.

В задании 4 подготовьте набор из 5-7 примерных запросов, содержащих псевдочувствительные данные (например, имена, номера счетов). Выполните запросы к модели (используйте pip install openai и локальный ключ доступа). Запишите полученные ответы в таблицу CSV (columns: request, response, compliance). Сохраните файл results.csv. При этом каждый полученный ответ необходимо проверить на соответствие требованиям из пункта 2 и отметить статус (соответствует/нарушает).

В задании 5 подготовьте отчёт в формате ODT или PDF. Структура отчёта: титульный лист, цель работы, описание каждого задания, скриншоты результатов тестирования (вставьте изображения сохранённых экранов), таблица results.csv, выводы и рекомендации по дальнейшему использованию

промта. Все файлы (prompt.txt, requirements.txt, results.csv, скриншоты, отчёт) упакуйте в один ZIP-архив.

Формат сдачи: архив ZIP, содержащий исходный код (если использовался), данные и отчёт, отправить по email преподавателю до конца занятия.

Отчёт: ODT или PDF, титульный лист, цель, описание шагов, скриншоты (png), таблица результатов (CSV) в виде вложения, выводы и рекомендации.

Сдача: ZIP-архив с кодом, данными и отчётом, отправить по email преподавателю до конца занятия.

II. Общие рекомендации

По всем вопросам, связанным с изучением дисциплины (включая самостоятельную работу), консультироваться с преподавателем.

III. Контроль и оценка результатов

Оценка за выполнение практической работы выставляется по пятибалльной системе и учитывается как показатель текущей успеваемости студента.

Оценивание ответа по пятибалльной системе осуществляется следующим образом:

По пятибалльной системе:

Качественная оценка индивидуальных образовательных достижений		**Критерии оценки результата**
балл (оценка)	**вербальный аналог**	
5	**отлично**	Практическая работа выполнена в полном объёме в соответствии с методическими указаниями. • Установлены и корректно использованы все требуемые библиотеки. • Корректно загружены, обработаны и сохранены датасеты (CSV, NPY). • Реализованы все этапы: предобработка, обучение, оценка, визуализация. • Модели обучены без ошибок, сохранены в требуемых форматах (joblib). • Метрики рассчитаны верно и сохранены в файлы (CSV, TXT). • Все графики построены и сохранены (PNG).

		• Структура проекта соответствует требованиям (data, models, results, src и т.д.). • Отчёт оформлен по требованиям: титульный лист, описание шагов, скриншоты, таблицы метрик, графики, выводы. • Архив собран корректно, все файлы присутствуют.
4	**хорошо**	Работа выполнена полностью, но допущены несущественные недочёты: • отдельные метрики/графики оформлены не оптимально; • незначительные ошибки в подписях, названиях файлов или структуре проекта; • выводы недостаточно обоснованы. Все основные этапы (загрузка, обучение, оценка, визуализация, сохранение) выполнены, архив содержит все ключевые файлы.
3	**удовлетворительно**	Выполнены большинство заданий, но: • часть этапов отсутствует (например, нет графика или не сохранена модель); • имеются ошибки в расчёте метрик или визуализации; • структура проекта нарушена; • отчёт неполный (нет скриншотов, таблиц, выводов). Работа демонстрирует частичное понимание процесса обучения моделей ИИ.
2	**не удовлетворительно**	Работа выполнена частично или с серьёзными ошибками: • отсутствуют ключевые этапы (нет обучения, оценки или визуализации); • файлы не сохраняются или отсутствуют; • модель не обучена или не работает; • отчёт отсутствует либо не соответствует требованиям. Цели практической работы не достигнуты.