Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка образовательных материалов и проектов в сфере Data Science»

по дисциплине «Проектный практикум»

Заказчик: Ильинский А. Д. _____

Куратор: Ильинский А. Д.

Ассистент кафедры ИИТ _____

Студенты команды _____

Белова А. В. _____

Шамоев Д. Ю. _____

Касаткин Г. Д. _____

Каниди Г. К. _____

Киселев М. С. _____

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью было получение необходимых знаний с помощью лабораторных работ для успешной реализации комплексного анализа данных и построение моделей машинного обучения в двух практических работах.

Задачи:

- Выполнение лабораторных работ,
- исследовательский анализ данных (EDA): визуализация, обработка пропусков, выявление закономерностей, корреляционный анализ,
- создание и оценка новых признаков (Feature Engineering) и анализ их важности,
- применение и сравнение моделей машинного обучения разных классов: линейных, деревьев решений, ансамблей, нейросетей,
- проведение кросс-валидации и выбор финальной модели,
- публикация работы на GitHub и командное обсуждение результатов.

В современном мире объем данных растет экспоненциально, и вместе с этим возрастает потребность в специалистах, способных эффективно извлекать ценную информацию из этих данных. Задачи анализа данных и машинного обучения становятся неотъемлемой частью цифровой трансформации в различных отраслях - от медицины и транспорта до медиаиндустрии и финансов. Таким образом, выполнение проекта способствует не только формированию технических навыков, но и развитию аналитического мышления, навыков интерпретации результатов и умения принимать обоснованные решения на основе данных. Всё это делает данный проект важным и своевременным элементом образовательной программы по направлению Data Science и машинного обучения.

Разрабатываемые в рамках проекта программные решения и аналитические модели имеют широкую область применения и демонстрируют универсальность методов анализа данных и машинного обучения.

Образование и учебные проекты:

Построенные модели, аналитика и визуализации могут использоваться в рамках учебных курсов, курсовых и выпускных квалификационных работ. Проект иллюстрирует полный цикл обработки данных и применим как учебное пособие по дисциплине «Машинное обучение».

Аналитика пользовательского поведения:

Задача классификации на основе данных о пассажирах «Титаника» демонстрирует, как можно анализировать табличные данные для принятия решений - например, в здравоохранении, страховании, транспорте и других сферах, где важно предсказывать исход событий по признакам пользователей.

Музыкальные стриминговые сервисы и рекомендации:

Регрессионная задача по предсказанию популярности треков Spotify отражает реальные потребности стриминговых платформ: персонализация контента, выявление хитов, анализ вкусов аудитории. Разработанные подходы могут быть применены в системах рекомендаций и маркетинговой аналитике.

По завершении проекта были созданы завершенные EDA-исследования по двум данным датасетам.

Повышение уровня владения инструментами pandas, numpy, matplotlib, seaborn, scikit-learn, развитие навыков Feature Engineering.

1. Выполнение лабораторных работ

В ходе выполнения заданий, каждым из участников в рамках учебного трека были выполнены такие лабораторные работы:

- Основы Python. Библиотека Numpy,
- Pandas и корреляционный анализ,
- линейная модель и градиентный спуск. Деревья, KNN,
- ансамбли, полносвязные нейронные сети.

В процессе выполнения лабораторных работ был получен всесторонний практический опыт, необходимый для решения задач анализа данных и машинного обучения. В ходе работы с библиотеками Python, такими как NumPy и Pandas, был приобретен опыт загрузки данных из различных источников, выполнять их очистка, обработка и трансформация для дальнейшего анализа. Освоены базовые операции с массивами и таблицами, работа с фильтрацией, агрегацией и преобразованием данных. Это позволило эффективно анализировать наборы данных, выявлять и устранять пропуски, формировать новые признаки и готовить данные для моделей.

Особое внимание было уделено визуализации данных: освоено построение графиков распределений, диаграмм рассеяния и тепловых карт, что позволило наглядно выявлять зависимости между признаками и анализировать результаты экспериментов. Опыт корреляционного анализа дал понимание того, как взаимосвязаны переменные в наборе данных и как эти связи могут быть использованы при построении моделей.

Важной частью работы стало освоение принципов построения моделей машинного обучения с нуля. Был приобретен опыт реализаций линейных моделей — таких как линейная регрессия и логистическая регрессия — а также реализация алгоритма градиентного спуска для обучения моделей. Это способствовало глубокому пониманию того, как работают методы оптимизации и как выбор параметров обучения (скорости обучения,

количества итераций) влияет на качество модели. В рамках экспериментов с моделями был приобретен опыт подбора гиперпараметров, настройки функций потерь и проверки качества классификации и регрессии на реальных наборах данных.

Следующим этапом стало освоение более сложных методов: полносвязных нейронных сетей и ансамблевых моделей. Были применены библиотеки scikit-learn и PyTorch для создания более сложных моделей, исследована их архитектура и оптимизированы параметры обучения. Было изучено, как методы ансамблирования (бэггинг, случайный лес) и нейросетевые подходы помогают повысить точность предсказаний, а также какие преимущества и ограничения связаны с использованием этих методов. Были приобретены навыки анализа результатов работы моделей, выявления ошибок классификации, интерпретации важности признаков и улучшения качества предсказаний.

Кроме того, особое внимание уделялось грамотной фиксации хода работы в Jupyter Notebook: все шаги анализа, код моделей, результаты экспериментов и выводы были структурированы и задокументированы в соответствии с требованиями. Регулярные обсуждения возникающих вопросов с преподавателем способствовали углублению понимания теоретических основ машинного обучения и практической реализации методов.

В целом, выполнение лабораторных работ позволило каждому участнику команды развить ключевые навыки, необходимые для решения задач анализа данных: от первичной обработки и анализа данных до построения и оценки качества моделей машинного обучения. Этот опыт стал прочной основой для выполнения последующих практических работ и формирования целостного понимания полного цикла работы с данными.

2. Работа по задачам «Титаник» и «Spotify»

Для оценки применимости и качества создаваемых моделей и решений был проведен анализ существующих решений и подходов в аналогичных задачах:

На платформе Kaggle задача предсказания выживания на Титанике является классическим соревнованием с десятками тысяч решений. Были изучены топовые ноутбуки, где применяются методы отбора признаков, ансамбли, работа с редкими категориями, а также усиленное feature engineering с использованием данных о родственниках, палубах и билетах. Наш проект учитывает лучшие практики: создание новых признаков, подбор моделей, кросс-валидация.

Spotify Popularity Prediction - аналогичные кейсы в открытых источниках. Подобные регрессионные задачи решаются в индустрии для создания рекомендательных систем. Среди аналогов - проекты с использованием моделей XGBoost и LightGBM, MLP, использование PCA и кластеризации по жанрам. В нашей работе также применены ансамблевые методы, анализ важности признаков, работа с категориальными переменными.

Программный продукт представляет собой модульную систему Jupyter Notebook-проектов, реализующую полный pipeline обработки данных и построения моделей.

Основные компоненты:

- Модуль загрузки и подготовки данных,
- импорт библиотек (pandas, numpy, matplotlib, seaborn, sklearn),
- загрузка датасетов (.csv),
- обработка пропусков, фильтрация, кодирование категориальных признаков, масштабирование,
- EDA и визуализация,
- построение графиков распределения,
- диаграммы корреляции,

- кластеризация и выявление выбросов,
- создание новых признаков,
- оценка важности признаков (Feature Importance),
- отбор признаков на основе корреляции и SHAP,
- модельный блок,
- построение базовых моделей,
- обучение деревьев решений, случайного леса, бустинга,
- построение MLP-нейросети,
- модуль валидации и метрик,
- кросс-валидация (5-fold),
- вычисление метрик (Accuracy, ROC-AUC, MAE, RMSE, R^2),
- визуализация кривых ошибок, precision-recall, ROC,
- сохранение и публикация на GitHub.

Jupyter Notebook выбран как наиболее удобный инструмент для прототипирования, визуализации и анализа данных.

Модульность архитектуры позволяет переиспользовать компоненты между задачами (например, блоки загрузки и анализа признаков).

На этапе EDA были выявлены некорректные значения и выбросы, которые устранялись вручную. Ошибки при обучении моделей возникали при несоответствии размерности данных (например, после one-hot encoding) — устранены через отладку в Jupyter. В процессе нормализации и кросс-валидации были оптимизированы параметры scaler, cv, что улучшило стабильность моделей.

Вот примеры наших тепловых карт корреляции числовых признаков в задаче «Spotify» и «Титаник» (см. рисунки 1-2):

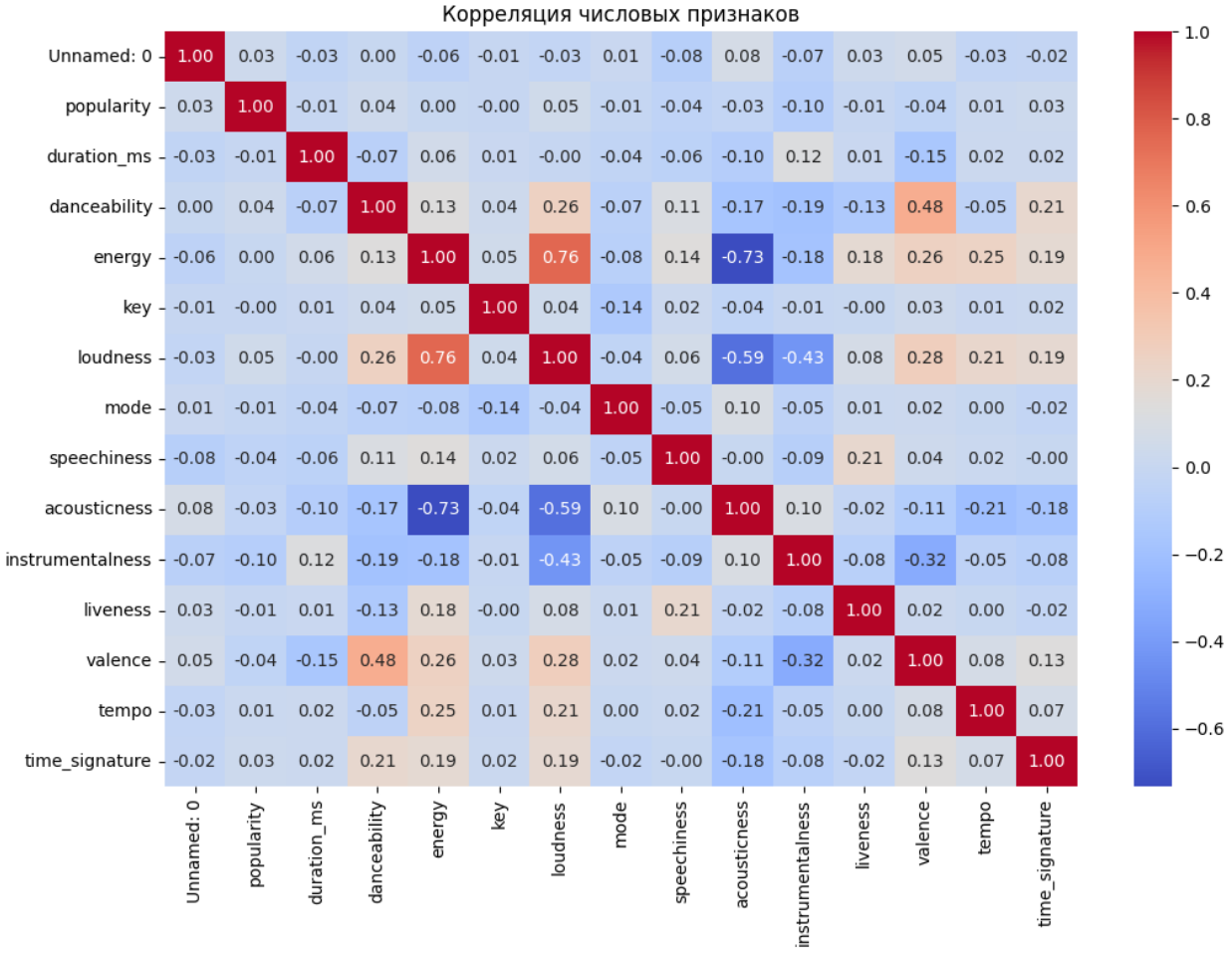

Рисунок 1 - корреляция числовых признаков

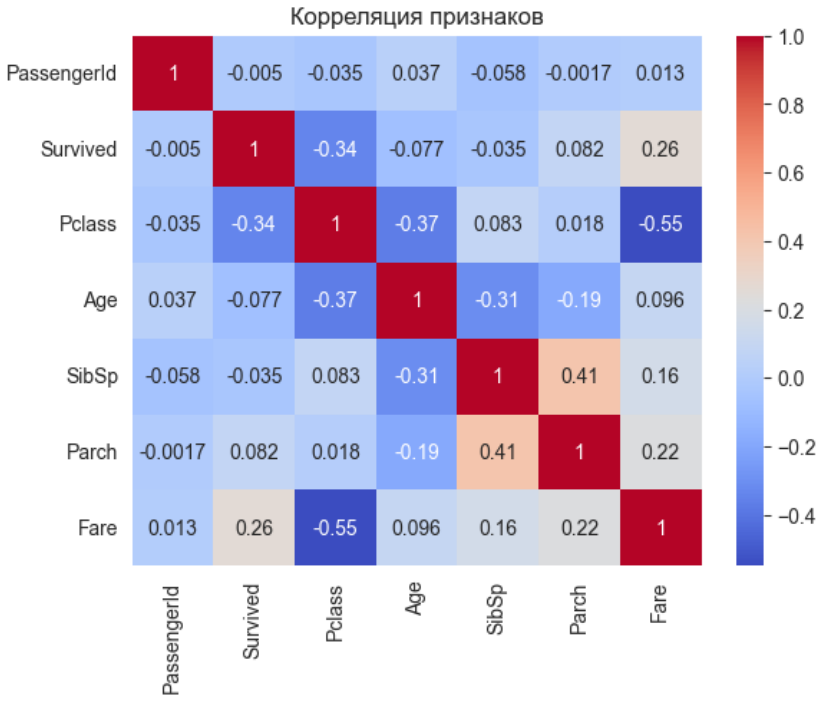

Рисунок 2 - корреляция признаков

3. Работа каждого участника

3.1 Белова А. В.

Занималась подготовкой данных и углубленным исследовательским анализом (EDA) для обоих кейсов проекта: Titanic и Spotify. В рамках Titanic были построены графики распределений по ключевым признакам, выполнен анализ пропусков, выявлены выбросы и аномалии, проведен анализ распределения категориальных признаков (пол, класс каюты, наличие родственников) и их влияние на целевую переменную (выживание). На основе визуализаций (гистограммы, boxplot, countplot) сделаны предварительные выводы о важности признаков, что позволило сократить количество исходных данных и сосредоточиться на ключевых характеристиках.

В проекте Spotify Белова А. В. провела оценку распределения целевой переменной (popularity), проверила наличие выбросов, проанализировала распределение категориальных признаков (жанр, explicit, artist name). Отдельное внимание было уделено анализу корреляционной матрицы, поиску мультиколлинеарных признаков, а также построению тепловых карт для визуализации связей между переменными. Белова А. В. подготовила блоки кода для визуализаций и статистического анализа, комментировала результаты и формировала выводы, что стало основой для дальнейшего этапа feature engineering и выбора моделей.

3.2 Каниди Г. К.

Отвечал за этап feature engineering и генерацию новых признаков для обеих задач. В Titanic были созданы признаки: размер семьи, наличие каюты по первой букве, длина имени, заголовок, а также бинарные индикаторы.

Для Spotify проведен анализ категориальных переменных, создан one-hot encoding для жанров, добавлен бинарный признак explicit, а также

признаки длины трека, плотности текста в названии. Особое внимание уделялось тестированию корреляции новых признаков с целевой переменной, анализу значимости с использованием feature importance (модели дерева решений) и SHAP. Провел отбор признаков на основе корреляционного анализа, а также реализовал код для оценки важности признаков и визуализации результата. Его работа позволила повысить интерпретируемость моделей и заложила основу для успешного обучения. В проекте Spotify применялся принцип «сильные признаки + минимизация шумов», что позволило сократить количество ненужных данных и улучшить производительность моделей.

3.3 Касаткин Г. Д.

Сосредоточился на разработке и тестировании моделей машинного обучения. В задаче Titanic были реализованы модели логистической регрессии, дерева решений и случайного леса. Особое внимание уделялось подбору гиперпараметров (максимальная глубина, минимальное количество выборок для разделения), настройке кросс-валидации (5-fold), а также вычислению метрик (Accuracy, ROC-AUC, Precision, Recall).

Касаткин Г. Д. подготовил код для оценки производительности моделей, визуализации кривых ROC, Precision-Recall и сравнительного анализа моделей. В задаче Spotify реализованы модели линейной регрессии, дерева решений, случайного леса и градиентного бустинга (XGBoost, LightGBM). Произведен подбор гиперпараметров с использованием GridSearchCV, проведена кросс-валидация, получены и проанализированы метрики (MAE, RMSE, R^2). Отдельное внимание уделялось обучению моделей на предварительно обработанных данных с разными наборами признаков. В процессе работы Касаткин Г. Д. обеспечивал стабильность кода, устранял ошибки, возникавшие из-за несоответствия размерностей данных после one-hot encoding, а также добавлял комментарии в ноутбуки для повышения

читаемости и удобства повторного использования.

3.4 Шамоев Д. Ю.

Занимался настройкой и обучением моделей глубокого обучения (MLP) для задач классификации (Titanic) и регрессии (Spotify). В рамках Titanic была реализована модель MLP с одним и несколькими скрытыми слоями, подобраны архитектура сети, функции активации (ReLU, Sigmoid), оптимизаторы (Adam, SGD), количество эпох, размер батча. Произведен анализ влияния различных архитектур на итоговую точность модели, построены графики зависимости функции потерь от эпох.

Для задачи Spotify создана MLP-модель с учетом регрессионной постановки задачи: оптимизирована структура сети, подобраны параметры обучения, протестированы разные функции потерь (MSE, MAE). Шамоев Д. Ю. отвечал за настройку early stopping, регулировку скорости обучения, проверку переобучения через визуализацию кривых обучения. Результаты экспериментов показали эффективность MLP на задаче регрессии, хотя ансамблевые методы (градиентный бустинг) показали более высокую стабильность. Также Шамоев Д. Ю. подготовил детальные выводы по обучению моделей, включая плюсы и минусы использования нейросетей на относительно малых датасетах.

3.5 Киселев М. С.

Занимался интеграцией результатов всех этапов проекта, а также оформлением итоговых ноутбуков, включая публикацию на GitHub. Подготовлены README-файлы, описания блоков кода, комментарии к визуализациям, ссылки на использованные источники.

Для задач Titanic и Spotify оформлены структурированные Jupyter Notebook, содержащие полное описание проведенного анализа, визуализаций,

построения моделей, кода для обучения, вычисленных метрик, графиков, а также итоговых выводов. В процессе работы Киселев М. С. проверял ноутбуки на наличие ошибок, устранял предупреждения, оптимизировал визуализацию (согласованная цветовая схема, заголовки, подписи осей). Для Spotify добавлены описания используемых библиотек, примеры запуска кода, инструкции для воспроизводимости экспериментов. Киселев М. С. координировал работу по созданию финального отчета, следил за соблюдением требований к оформлению, подготовил итоговую версию проекта для сдачи и презентации.

4. Эксперименты

4.1 Эксперименты по Titanic

Для задачи классификации выживания пассажиров Titanic было проведено сравнение нескольких моделей машинного обучения. В качестве метрики использовалась Accuracy, а для проверки устойчивости результатов применялась кросс-валидация с разбиением данных на 5 фолдов (StratifiedKFold, чтобы сохранять пропорции классов) (Таблица 1).

- Логистическая регрессия,
- дерево решений,
- случайный лес,
- градиентный бустинг (XGBoost),
- полносвязная нейронная сеть (MLP).

Таблица 1 - кросс-валидация для классических моделей

Модель	Средняя Accuracy	Стандартное отклонение
Logistic Regression	0.8238	0.0163
Decision Tree	0.7935	0.0257
Random Forest	0.8170	0.0196
XGBoost	0.8283	0.0266

Как видно из результатов, XGBoost показал наивысшее среднее качество (82.83%), чуть опередив логистическую регрессию. Случайный лес также показал достойный результат (81.70%). Дерево решений оказалось менее стабильным, его стандартное отклонение больше, что указывает на чувствительность модели к разным разбиениям данных.

Для нейросетевой модели (MLP) была построена простая архитектура из двух скрытых слоев (32 и 16 нейронов) с функцией активации ReLU и выходным слоем с сигмоидой. Обучение проводилось с использованием оптимизатора Adam и функцией потерь binary_crossentropy. В качестве метода оценки также использовалась 5-кратная кросс-валидация (KFold).

- Результаты MLP,
- средняя Accuracy: 0.8418,
- стандартное отклонение: 0.0267

Нейросеть показала чуть более высокое качество по сравнению с XGBoost, хотя различие не является статистически значимым. Это указывает на хороший потенциал использования MLP для подобных задач классификации, однако важно учитывать риски переобучения и потребность в настройке гиперпараметров.

Выводы по экспериментам Titanic: модели дерева решений и случайного леса демонстрируют хорошую интерпретируемость, но уступают по качеству XGBoost и MLP. XGBoost показывает стабильный результат без серьезной настройки гиперпараметров, что делает его универсальным решением. MLP имеет потенциал для дальнейшего улучшения качества, но требует более тщательной настройки и может быть чувствительным к размеру выборки. Логистическая регрессия выступает достойной базовой моделью с хорошим качеством и интерпретируемостью.

4.2 Эксперименты по Spotify

В задаче регрессии по предсказанию популярности треков (Spotify) также была проведена серия экспериментов с разными моделями. В качестве метрики использовалась среднеквадратичная ошибка (RMSE). Для проверки устойчивости результатов проводилась 5-кратная кросс-валидация на разных разбиениях данных.

- Используемые модели,
- линейная регрессия (Ridge),
- дерево решений,
- градиентный бустинг,
- MLP Regressor (нейросеть).

Средние значения RMSE по разным случайным разбиениям (с 5 разными random_state) (Таблица 2):

Таблица 2 - Средние значения RMSE по разным случайным разбиениям.

Модель	Среднее RMSE (приблизительное)
Ridge	~22.28
Decision Tree	~35.00–35.40
Gradient Boosting	~24.65
MLP Neural Net	сильно нестабильное (от 36.63 до 210.27)

Из полученных результатов видно, что Ridge-регрессия стабильно показывает наименьшее среднее RMSE (~22.28), что делает ее базовой и надежной моделью для данной задачи. Gradient Boosting показал близкие, но немного худшие результаты (RMSE ~24.65), однако продемонстрировал

высокую устойчивость.

Модель дерева решений оказалась менее точной (RMSE ~35), что связано с ограниченной способностью деревьев к обобщению и риском переобучения на шумных данных.

Нейросетевой регрессор (MLP) показал крайне нестабильные результаты: RMSE варьировался от 36 до более 200. Это указывает на высокую чувствительность MLP к настройкам, возможное переобучение и недостаточную пригодность для работы с подобными табличными данными без более глубокой настройки и обработки данных.

Таким образом, на тестовой выборке лучший результат показал градиентный бустинг (RMSE = 19.87), что указывает на его способность лучше обобщать данные в сравнении с линейной моделью.

Выводы по экспериментам Spotify: Ridge-регрессия — надежная и простая модель, подходит как базовый вариант. Градиентный бустинг показывает более высокое качество на тестовых данных и может быть рекомендован как финальная модель для задачи регрессии. Нейросеть (MLP) показала нестабильные результаты, требует дополнительной работы по улучшению архитектуры, подбору гиперпараметров и, возможно, увеличения объема данных. Дерево решений продемонстрировало ограниченные возможности для сложной регрессии, показывая более высокие ошибки.

ЗАКЛЮЧЕНИЕ

В ходе выполнения проектной работы были достигнуты все поставленные цели: участники команды освоили практические навыки анализа данных, построения и оценки моделей машинного обучения, разработки признаков и визуализации результатов. Работа над кейсами Titanic и Spotify позволила участникам получить опыт решения задач классификации и регрессии, что отражает широкий спектр реальных прикладных задач в области Data Science.

Каждый участник внёс значительный вклад в реализацию проекта: от предварительной обработки данных и визуализации до построения сложных моделей и их оптимизации. Были применены различные методы и инструменты анализа данных - от базовых библиотек Python до современных подходов, таких как ансамбли моделей и полносвязные нейронные сети. Подобный подход обеспечил глубокое понимание процессов, лежащих в основе машинного обучения, и укрепил навыки командной разработки.

Проект оформлен в виде модульной системы Jupyter Notebook, содержащей полный цикл обработки данных. Материалы проекта опубликованы на GitHub и могут быть использованы как основа для дальнейших исследований или в образовательных целях.

Таким образом, проект позволил не только закрепить теоретические знания, полученные в рамках курса, но и сформировать прочную базу практических навыков, необходимых для успешной карьеры в области анализа данных и машинного обучения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media.
2. Raschka, S., & Mirjalili, V. (2020). Python Machine Learning. Packt Publishing.
3. Kaggle. Titanic - Machine Learning from Disaster. URL: https://www.kaggle.com/competitions/titanic
4. Kaggle. Spotify Tracks Dataset. URL: https://www.kaggle.com/datasets/
5. Documentation - Scikit-learn. URL: https://scikit-learn.org/stable/documentation.html
6. Documentation - Pandas. URL: https://pandas.pydata.org/docs/
7. Documentation - Matplotlib and Seaborn. URL: https://matplotlib.org/, https://seaborn.pydata.org/
8. Molnar, C. (2022). Interpretable Machine Learning. URL: https://christophm.github.io/interpretable-ml-book/
9. Documentation - PyTorch. URL: https://pytorch.org/docs/
10. Журавлев Ю.И., Стрижов В.В. Методы машинного обучения. — М.: МЦНМО, 2021.

Приложение А

In [28]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

models = {
    "Ridge": Ridge(),
    "Decision Tree": DecisionTreeRegressor(),
    "Gradient Boosting": GradientBoostingRegressor(),
    "MLP Neural Net": MLPRegressor(max_iter=500)
}

for name, model in models.items():
    scores = cross_val_score(model, X, y, cv=5, scoring='neg_root_mean_squared_error')
    print(f"{name}: Среднее RMSE = {-scores.mean():.2f}")
```

```
Ridge: Среднее RMSE = 22.28
Decision Tree: Среднее RMSE = 34.73
Gradient Boosting: Среднее RMSE = 24.66
MLP Neural Net: Среднее RMSE = 210.27
```

вычисление лучшей модели

Приложение В

In [37]:

```
best_model = Ridge()
best_model.fit(X_train, y_train)

y_pred = best_model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f"Финальное RMSE на тесте: {rmse:.2f}")
```

```
Финальное RMSE на тесте: 22.04
```

вывод лучшей модели