



Министерство науки и высшего образования Российской
Федерации
Федеральное государственное бюджетное образовательное
учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ И
СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01/07 Интеллектуальные системы анализа,
обработки и интерпретации больших данных

ОТЧЕТ

по лабораторным работам

Дисциплина: Платформы промышленной аналитики

Студент

ИУ6-42М

(Группа)

М.Э.Хабаров

(Подпись, дата)

(И.О. Фамилия)

Преподаватель

М.А. Скворцова

(Подпись, дата)

(И.О. Фамилия)

Москва, 2025

Гит: https://github.com/Lapinyata/analytics_platform

Выполняется в паре с Насибуллиным Д.И.

Лабораторная работа №1

Цель работы: ознакомиться с возможностями работы чатов/генеративных помощников на базе LLM для применения в исследовательской деятельности студентов и решения профессиональных задач.

Задачи:

1) выбрать 4 генеративных помощника из представленных выше. Далее необходимо задать вопросы по тематике вашей магистерской диссертации и проанализировать ответы чат-ботов для актуальности и корректности применения данных ответов в ВКРМ (должно быть сформулировано и проверено не менее 5 гипотез по каждому генеративному помощнику). Результаты работы оформить в виде сравнительной таблицы. Выбрать лучшее из решений для дальнейшей работы. Вывести таблицу от лучшего худшего по итогам анализа и вкратце объяснить почему вы пришли к такому выводу;

2) сформировать аннотацию/введение для своей магистерской диссертации с помощью генеративных помощников. Определить какой из них справляется с этой задачей лучше;

3) описать достоинства и недостатки применения данного инструмента для профессиональной деятельности;

4) проверить несколько запросов и ответов по использованию генеративных помощников по использованию в конструкторской/технологической части ВКРМ, при формировании списка литературы.

Ход работы

Задание 1.

Среди представленных в методических указаниях к лабораторной работе генеративных помощников были выбраны следующие четыре по причине их доступности:

- 1) ChatGPT от OpenAI;
- 2) Gemini от Google;
- 3) YandexGPT от Яндекса;
- 4) DeepSeek.

Выполним запросы к данным помощникам по следующим пяти гипотезам:

1) классификация документов может быть выполнена с использованием алгоритмов машинного обучения, таких как Naïve Bayes, SVM или методов на основе градиентного бустинга. Современные подходы используют глубокие нейронные сети (например, BERT), которые позволяют эффективно обрабатывать текстовые данные;

2) технологии обработки естественного языка (NLP), такие как spaCy, Hugging Face Transformers и Tesseract OCR, позволяют анализировать текст и извлекать ключевую информацию из документов, включая имена, даты, суммы, адреса и другие метаданные;

3) верификация может быть реализована путём сравнения извлечённых данных с базами данных или шаблонами. Алгоритмы сходства строк и векторных представлений текста, такие как cosine similarity, могут использоваться для проверки соответствия;

4) генеративные модели, такие как GPT, могут быть использованы для автоматического создания текстов, например, для формирования заключений по документам;

5) комбинированный подход к предобработке изображений (например, фильтрация шума, бинаризация и коррекция наклона) и

настройке OCR-систем способен существенно улучшить точность распознавания текста, даже для низкокачественных сканированных документов и сложных форматов, таких как таблицы или графики.

Для каждой из представленных гипотез были соответственно составлены вопросы для проверки выбранных инструментов:

1) какой алгоритм машинного обучения лучше всего подходит для классификации различных типов документов в сфере лизинга?

2) какую модель обработки естественного языка следует использовать для точного извлечения ключевой информации, такой как имена, даты и суммы, из текстов документов?

3) какие метрики и подходы наиболее эффективны для автоматической сверки извлечённых данных с существующими шаблонами или базами данных?






4) какие генеративные модели можно использовать для автоматического заполнения отсутствующих данных или формирования кратких описаний документов?

5) какие этапы предобработки изображений (например, фильтрация шума, бинаризация, коррекция наклона) наиболее эффективно улучшают точность OCR при работе с низкокачественными документами или сложными форматами, такими как таблицы и графики?




На основании данных тезисов были сделаны запросы в генеративные помощники. Полученные результаты ответов генеративных помощников отображены в таблице 1.

Таблица 1 – Сравнительная таблица запросов к данным

какой алгоритм машинного обучения лучше всего подходит для классификации различных типов документов в сфере лизинга?	
ChatGPT от OpenAI	Выбор лучшего алгоритма зависит от типа данных (текст, сканы, табличные данные), объема и сложности. Но если говорить о наиболее эффективных и универсальных решениях для классификации типов документов в лизинге , то в лидерах сейчас модели на базе трансформеров, особенно:

	<div><div> Лучший выбор: Трансформеры (BERT-подобные модели)</div><div><div> Пример моделей:</div><div><ul style="list-style-type: none">• RuBERT / SBER-RuBERT / DeepPavlov/rubert-base-cased• XLM-RoBERTa (мультиязычная)• LayoutLM / LayoutLMv2 (если важна структура документа)• Donut (если документы приходят как изображения и нет OCR)</div></div></div>													
	<div><div><div> Почему трансформеры?</div><div><table><tr><th>Преимущество</th><th>Описание</th></tr><tr><td>Понимание контекста</td><td>Модель читает весь текст и понимает смысл, а не только ключевые слова.</td></tr><tr><td>Поддержка русского языка</td><td>Есть предобученные модели на русском и юридических корпусах.</td></tr><tr><td>Хорошо работают на юридических и финансовых документах</td><td>Особенно когда классы похожи по содержанию.</td></tr><tr><td>Можно адаптировать на небольшом датасете с fine-tuning.</td><td>Достаточно 100–1000 документов на класс.</td></tr><tr><td>Интеграция в production через API</td><td>FastAPI + Hugging Face = легко внедрить в поток.</td></tr></table></div></div></div>	Преимущество	Описание	Понимание контекста	Модель читает весь текст и понимает смысл, а не только ключевые слова.	Поддержка русского языка	Есть предобученные модели на русском и юридических корпусах.	Хорошо работают на юридических и финансовых документах	Особенно когда классы похожи по содержанию.	Можно адаптировать на небольшом датасете с fine-tuning.	Достаточно 100–1000 документов на класс.	Интеграция в production через API	FastAPI + Hugging Face = легко внедрить в поток.	
Преимущество	Описание													
Понимание контекста	Модель читает весь текст и понимает смысл, а не только ключевые слова.													
Поддержка русского языка	Есть предобученные модели на русском и юридических корпусах.													
Хорошо работают на юридических и финансовых документах	Особенно когда классы похожи по содержанию.													
Можно адаптировать на небольшом датасете с fine-tuning.	Достаточно 100–1000 документов на класс.													
Интеграция в production через API	FastAPI + Hugging Face = легко внедрить в поток.													
	<div><div><div> Альтернатива (при ограниченных ресурсах):</div><div>Если нет возможности использовать трансформеры, подойдут более легкие модели:</div><div><table><tr><th>Метод</th><th>Когда использовать</th><th>Преимущества</th></tr><tr><td>TF-IDF + Logistic Regression</td><td>Много текстов, но нет GPU или данных для обучения BERT</td><td>Быстро и просто, особенно если тексты хорошо различаются</td></tr><tr><td>Naive Bayes</td><td>Простые тексты, слабо пересекаются</td><td>Очень быстрый, хорошо работает на новостях/заявках</td></tr><tr><td>SVM</td><td>Небольшой, но качественный датасет</td><td>Точная модель при хорошем подборе признаков</td></tr></table></div></div></div>	Метод	Когда использовать	Преимущества	TF-IDF + Logistic Regression	Много текстов, но нет GPU или данных для обучения BERT	Быстро и просто, особенно если тексты хорошо различаются	Naive Bayes	Простые тексты, слабо пересекаются	Очень быстрый, хорошо работает на новостях/заявках	SVM	Небольшой, но качественный датасет	Точная модель при хорошем подборе признаков	
Метод	Когда использовать	Преимущества												
TF-IDF + Logistic Regression	Много текстов, но нет GPU или данных для обучения BERT	Быстро и просто, особенно если тексты хорошо различаются												
Naive Bayes	Простые тексты, слабо пересекаются	Очень быстрый, хорошо работает на новостях/заявках												
SVM	Небольшой, но качественный датасет	Точная модель при хорошем подборе признаков												
	<div><div><div> Вывод</div><div>Если есть возможность — используй трансформер (RuBERT / LayoutLM). Если нет GPU или ограничены по времени — начни с TF-IDF + Logistic Regression или SVM.</div></div></div>													
Gemini от Google	<div>На вопрос, заданный в тексте изображения, нет однозначного ответа, так как выбор алгоритма машинного обучения зависит от конкретных характеристик данных и требований задачи. Тем не менее, вот некоторые алгоритмы, которые могут быть полезны для классификации документов в сфере лизинга:</div> <div><ul style="list-style-type: none">• Наивный байесовский классификатор: Этот алгоритм прост в реализации и эффективен для классификации текстовых данных.• Метод опорных векторов (SVM): SVM может быть эффективным для классификации документов с высокой размерностью.• Случайный лес: Этот алгоритм может быть полезен для классификации документов с нелинейными отношениями между признаками.</div>													










	<ul style="list-style-type: none"> • Градиентный бустинг: Градиентный бустинг - это мощный алгоритм, который может быть эффективным для широкого спектра задач классификации. • Глубокое обучение: Глубокое обучение может быть полезным для классификации документов с большим объемом данных и сложными отношениями между признаками. <p>Чтобы выбрать лучший алгоритм, необходимо поэкспериментировать с различными алгоритмами и оценить их производительность на конкретном наборе данных.</p>
YandexGPT от Яндекс	<p>Выбор алгоритма машинного обучения для классификации документов в сфере лизинга зависит от нескольких факторов, включая объём и структуру данных, требуемую точность, скорость обработки и доступные ресурсы. Однако несколько алгоритмов часто показывают хорошие результаты в задачах классификации текстовых документов:</p> <ol style="list-style-type: none"> 1. Наивный байесовский классификатор — хорошо подходит для классификации текстовых документов благодаря своей эффективности и простоте. Он работает на основе теоремы Байеса и предположения о независимости признаков, что делает его быстрым и эффективным для больших объёмов текстовых данных. 2. Случайный лес (Random Forest) — ансамбль деревьев решений, который обеспечивает высокую точность и устойчивость к переобучению. Случайный лес хорошо справляется с задачами классификации и может быть эффективно применён для классификации документов в лизинге. 3. Метод опорных векторов (SVM) — особенно эффективен для классификации в случаях, когда данные не линейно разделимы. SVM находит гиперплоскость, которая наилучшим образом разделяет данные на классы, что может быть полезно для классификации сложных документов. 4. Нейронные сети, в частности свёрточные нейронные сети (CNN) и рекуррентные нейронные сети (RNN) — способны улавливать сложные закономерности в текстовых данных и могут быть эффективны для классификации документов на основе их содержания. Они особенно полезны, если данные имеют сложную структуру или содержат много текстовых особенностей. <p>Для выбора наиболее подходящего алгоритма рекомендуется провести эксперименты с различными методами на выборке данных, чтобы определить, какой из них обеспечивает наилучшую точность и производительность для конкретной задачи. Также стоит рассмотреть возможность использования ансамблей методов или гибридных подходов для улучшения результатов.</p>
DeepSeek	<p>Выбор алгоритма машинного обучения для классификации документов в сфере лизинга зависит от типа данных, размера датасета и требуемой точности. Вот несколько подходящих алгоритмов и подходов:</p> <ol style="list-style-type: none"> 1. Традиционные ML-алгоритмы (если данные структурированы или есть хорошие фичи) Подходят, если документы уже предобработаны (например, извлечены ключевые слова, TF-IDF, метаданные). <ul style="list-style-type: none"> • Логистическая регрессия – простой и интерпретируемый метод для бинарной/многоклассовой классификации. • Random Forest / Gradient Boosting (XGBoost, LightGBM, CatBoost) – хорошо работают на табличных данных, устойчивы к шуму. • SVM (Support Vector Machines) – эффективен в высокоразмерных пространствах (например, после TF-IDF). 2. Глубокое обучение (для текстовых данных)

	<p>Если документы представляют собой неструктурированный текст (договоры, заявки, отчёты), лучше использовать NLP-модели:</p> <ul style="list-style-type: none"> • BERT, RoBERTa, GPT (трансформеры) – современные модели для классификации текста. <ul style="list-style-type: none"> ○ Пример: RuBERT для русского языка (если документы на русском). ○ Можно дообучить на специфичных данных лизинга. • FastText, Word2Vec + CNN/LSTM – если данных мало, можно использовать эмбединги слов + свёрточные или рекуррентные сети. <p>3. Гибридные подходы</p> <ul style="list-style-type: none"> • Использование BERT для эмбедингов + классификатор (например, Random Forest) – если нужно сочетать текстовые и табличные данные. • Ensemble методов – комбинация разных моделей для улучшения точности. <p>4. Дополнительные рекомендации</p> <ul style="list-style-type: none"> • Предобработка текста: <ul style="list-style-type: none"> ○ Очистка от стоп-слов, лемматизация (например, pymorphy2 для русского). ○ Извлечение ключевых терминов (например, "лизинговый платеж", "остаточная стоимость"). • Активное обучение – если данных мало, можно разметить только самые информативные документы. • Использование библиотек: <ul style="list-style-type: none"> ○ scikit-learn – для традиционных ML. ○ transformers (Hugging Face) – для BERT-подобных моделей. ○ spaCy / NLTK – для NLP-обработки. <p>Вывод</p> <ul style="list-style-type: none"> • Если тексты сложные и много данных → BERT/Fine-tuning трансформеров. • Если данные частично структурированы → XGBoost + NLP-фичи. • Если данных мало → FastText + логистическая регрессия / SVM.
<p>какую модель обработки естественного языка следует использовать для точного извлечения ключевой информации, такой как имена, даты и суммы, из текстов документов?</p>	
<p>ChatGPT от OpenAI</p>	<p>Для точного извлечения ключевой информации (имена, даты, суммы) из документов на русском языке лучше всего подойдут модели Named Entity Recognition (NER), особенно на базе трансформеров.</p> <hr/> <p> Рекомендуемые модели для извлечения сущностей (NER):</p> <p> 1. DeepPavlov/rubert-base-cased-conversational-NER</p> <ul style="list-style-type: none"> • Основана на RuBERT (BERT для русского) • Обучена на NER-задачах • Поддерживает сущности: PER, ORG, LOC, DATE, TIME, MONEY, и т.д. • Легко адаптируется к своим классам через fine-tuning <p> 2. Natasha (Natasha NER)</p> <ul style="list-style-type: none"> • Специализированный стек NLP-библиотек для русского языка • Приземлён на практику: умеет извлекать ФИО, даты, адреса, организации, деньги, ИНН и др. • Под капотом — CRF/embedding-based модели

	<ul style="list-style-type: none"> Быстрый и простой для запуска <p>✅ 3. Sberbank AI Models (sbert / xlm models)</p> <ul style="list-style-type: none"> Есть модели, дообученные на юридических и финансовых документах Поддержка извлечения дат, сумм, организаций и специфичных сущностей (например, "номер договора") 														
	<p>🧠 Почему использовать NER-модель, а не просто регэксп?</p> <table> <tr> <th>Подход</th><th>Преимущество</th></tr> <tr> <td>🧠 NER-модель</td><td>Понимает контекст и грамматику, не привязана к одному шаблону</td></tr> <tr> <td>🔍 RegEx</td><td>Хорош для сумм и дат, но не устойчив к вариативности формата</td></tr> <tr> <td>💡 Гибрид</td><td>Часто лучшее решение — NER + RegEx + postprocessing</td></tr> </table>	Подход	Преимущество	🧠 NER-модель	Понимает контекст и грамматику, не привязана к одному шаблону	🔍 RegEx	Хорош для сумм и дат, но не устойчив к вариативности формата	💡 Гибрид	Часто лучшее решение — NER + RegEx + postprocessing						
Подход	Преимущество														
🧠 NER-модель	Понимает контекст и грамматику, не привязана к одному шаблону														
🔍 RegEx	Хорош для сумм и дат, но не устойчив к вариативности формата														
💡 Гибрид	Часто лучшее решение — NER + RegEx + postprocessing														
	<p>📌 Какие сущности обычно извлекают в лизинговых документах:</p> <table> <tr> <th>Сущность</th><th>Примеры</th></tr> <tr> <td>PER – ФИО</td><td>Иванов Иван Иванович</td></tr> <tr> <td>ORG – организации</td><td>ООО "АвтоЛизинг"</td></tr> <tr> <td>DATE</td><td>15.02.2024, «пятнадцатого февраля две тысячи двадцать четвертого года»</td></tr> <tr> <td>MONEY</td><td>1 500 000 руб., один миллион пятьсот тысяч рублей</td></tr> <tr> <td>CONTRACT_NUM (кастомная)</td><td>Договор № 145/ЛИЗ-2023</td></tr> <tr> <td>ACCOUNT_NUM, INN, KPP и др.</td><td>можно дообучить модель или достать через шаблоны</td></tr> </table>	Сущность	Примеры	PER – ФИО	Иванов Иван Иванович	ORG – организации	ООО "АвтоЛизинг"	DATE	15.02.2024, «пятнадцатого февраля две тысячи двадцать четвертого года»	MONEY	1 500 000 руб., один миллион пятьсот тысяч рублей	CONTRACT_NUM (кастомная)	Договор № 145/ЛИЗ-2023	ACCOUNT_NUM, INN, KPP и др.	можно дообучить модель или достать через шаблоны
Сущность	Примеры														
PER – ФИО	Иванов Иван Иванович														
ORG – организации	ООО "АвтоЛизинг"														
DATE	15.02.2024, «пятнадцатого февраля две тысячи двадцать четвертого года»														
MONEY	1 500 000 руб., один миллион пятьсот тысяч рублей														
CONTRACT_NUM (кастомная)	Договор № 145/ЛИЗ-2023														
ACCOUNT_NUM, INN, KPP и др.	можно дообучить модель или достать через шаблоны														
	<p>🔧 Лучшее решение на практике:</p> <p>Гибридная система:</p> <ol style="list-style-type: none"> NER-модель → для персоналий, организаций, дат RegEx / postprocessing → для сумм, номеров договоров, ИНН Custom NER-тренировка → если нужны нестандартные сущности 														
	<p>⚙️ Примеры библиотек:</p> <table> <tr> <th>Библиотека</th><th>Назначение</th></tr> <tr> <td>DeepPavlov</td><td>Готовые NER-модели, есть API</td></tr> <tr> <td>Natasha</td><td>Лёгкий стек NLP для русского, с парсингом и NER</td></tr> <tr> <td>spaCy + ru-core-news-*</td><td>Можно адаптировать, но не лучшая точность на русском</td></tr> <tr> <td>Transformers (HuggingFace)</td><td>Дообучение своей NER-модели на основе RuBERT или XLM-R</td></tr> </table>	Библиотека	Назначение	DeepPavlov	Готовые NER-модели, есть API	Natasha	Лёгкий стек NLP для русского, с парсингом и NER	spaCy + ru-core-news-*	Можно адаптировать, но не лучшая точность на русском	Transformers (HuggingFace)	Дообучение своей NER-модели на основе RuBERT или XLM-R				
Библиотека	Назначение														
DeepPavlov	Готовые NER-модели, есть API														
Natasha	Лёгкий стек NLP для русского, с парсингом и NER														
spaCy + ru-core-news-*	Можно адаптировать, но не лучшая точность на русском														
Transformers (HuggingFace)	Дообучение своей NER-модели на основе RuBERT или XLM-R														
Gemini от Google	<p>Возможные ответы на вопрос:</p> <p>Для точного извлечения ключевой информации, такой как имена, даты и суммы, из текстов документов можно использовать несколько моделей NLP. Выбор модели зависит от конкретных требований задачи, таких как тип документов, объем данных и требуемая точность.</p> <p>Вот некоторые модели NLP, которые могут быть полезны для этой задачи:</p> <ul style="list-style-type: none"> Именованное распознавание сущностей (NER): NER - это метод NLP, который может быть использован для идентификации и классификации именованных сущностей в тексте, таких как имена, даты и суммы. 														

	<ul style="list-style-type: none"> • Извлечение отношений (RE): RE - это метод NLP, который может быть использован для извлечения отношений между именованными сущностями в тексте. • Анализ зависимостей: Анализ зависимостей - это метод NLP, который может быть использован для определения грамматических отношений между словами в предложении. • Трансформерные модели: Трансформерные модели, такие как BERT и RoBERTa, могут быть использованы для широкого спектра задач NLP, включая извлечение информации. • Модели машинного обучения: Модели машинного обучения, такие как условные случайные поля (CRF) и скрытые модели Маркова (HMM), могут быть использованы для извлечения информации.
YandexGPT от Яндекс	<p>Для точного извлечения ключевой информации, такой как имена, даты и суммы из текстов документов, можно использовать следующие модели обработки естественного языка (Natural Language Processing, NLP):</p> <ol style="list-style-type: none"> 1. Named Entity Recognition (NER) — это модель, специально предназначенная для идентификации и классификации именованных сущностей в тексте, таких как имена людей, организаций, географических названий, дат, сумм и других специфических элементов. Существуют различные реализации NER, включая модели на основе правил, машинного обучения и глубокого обучения (например, с использованием рекуррентных нейронных сетей или трансформеров). 2. Регулярные выражения (Regular Expressions) — могут быть полезны для извлечения структурированных данных, таких как даты и суммы, которые часто имеют определённый формат. Однако их использование может быть ограничено в случаях, когда данные представлены в различных форматах или содержат много вариативности. 3. Трансформеры (Transformers) — такие модели, как BERT (Bidirectional Encoder Representations from Transformers), RoBERTa и другие, могут быть адаптированы для задачи извлечения информации. Они хорошо справляются с пониманием контекста и могут быть обучены для идентификации и извлечения ключевых элементов из текста. 4. Модели на основе правил (Rule-based Models) — могут быть разработаны для извлечения информации на основе заранее определённых шаблонов и правил. Хотя они могут быть менее гибкими, чем модели машинного обучения, они могут быть эффективны для специфических задач, где данные имеют чётко определённый формат. 5. Гибридные подходы — комбинация моделей машинного обучения и правил может обеспечить более точное извлечение информации. Например, можно использовать модель NER для первоначального выявления потенциальных сущностей, а затем применить правила для уточнения и проверки извлечённой информации. <p>Выбор конкретной модели зависит от специфики задачи, объёма и характера данных, а также доступных ресурсов. Рекомендуется провести анализ данных и эксперименты с различными моделями, чтобы определить наиболее подходящий подход для вашей задачи.</p>
DeepSeek	<p>Для точного извлечения структурированной информации (имен, дат, сумм, реквизитов) из документов в сфере лизинга лучше всего подходят методы NLP на основе трансформеров и предобученных языковых моделей. Вот основные подходы:</p> <p>1. Named Entity Recognition (NER) – Распознавание именованных сущностей</p>

	<p>Цель: Автоматически находить и классифицировать сущности (имена, даты, суммы, договоры).</p> <p>Лучшие модели для NER:</p> <ul style="list-style-type: none"> • BERT-based модели (fine-tuning): <ul style="list-style-type: none"> ○ RuBERT (для русского языка) – если документы на русском. ○ DeBERTa, LayoutLM – если важно учитывать структуру документа (PDF, сканы). • Многоязычные модели: <ul style="list-style-type: none"> ○ XLM-RoBERTa – если есть документы на разных языках. ○ Spacy + Transformer (например, ru_core_news_lg) – для быстрого развертывания. <p>2. Извлечение сумм и финансовых данных</p> <ul style="list-style-type: none"> • Регулярные выражения (RegEx) – если форматы сумм стандартизованы (например, <code>\d{1,3}(?:\d{3})* руб</code>). • Fine-tuned NER – если суммы пишутся в разных форматах ("1 млн руб.", "10 000,50"). <p>3. Извлечение дат</p> <ul style="list-style-type: none"> • Библиотеки для дат: <ul style="list-style-type: none"> ○ dateparser – автоматически парсит даты в разных форматах ("15.01.2024", "1 января 2024 г."). ○ Rule-based + NER – если даты сложные ("в течение 5 банковских дней с момента подписания"). <p>4. Обработка сложных документов (PDF, сканы)</p> <p>Если документы в PDF или отсканированы, нужно:</p> <ol style="list-style-type: none"> 1. OCR (Tesseract, EasyOCR) – для извлечения текста. 2. LayoutLM / Donut – модели, учитывающие расположение текста (например, для таблиц). <p>5. Готовые решения и фреймворки</p> <ul style="list-style-type: none"> • Spacy (ru_core_news_lg) – для быстрого NER. • Natasha (Yargy-парсер) – для извлечения фактов по правилам (например, реквизиты договора). • Hugging Face Transformers – для кастомного fine-tuning под ваш датасет. <p>Выбор модели в зависимости от задачи</p> <table> <tr> <th>Тип данных</th><th>Лучший инструмент</th></tr> <tr> <td>Текстовые документы</td><td>RuBERT / XLM-R + NER</td></tr> <tr> <td>Сканы/PDF</td><td>Tesseract + LayoutLM</td></tr> <tr> <td>Стандартные суммы/даты</td><td>RegEx + dateparser</td></tr> <tr> <td>Сложные договоры</td><td>Natasha (Yargy) + rule-based извлечение</td></tr> </table> <p>Рекомендации</p> <ol style="list-style-type: none"> 1. Разметить свой датасет (если данных мало, можно начать с разметки в Prodigy или Label Studio). 2. Дообучить модель на данных лизинга (например, на Hugging Face Transformers). 3. Комбинировать подходы: <ul style="list-style-type: none"> ○ NER для сущностей. 	Тип данных	Лучший инструмент	Текстовые документы	RuBERT / XLM-R + NER	Сканы/PDF	Tesseract + LayoutLM	Стандартные суммы/даты	RegEx + dateparser	Сложные договоры	Natasha (Yargy) + rule-based извлечение
Тип данных	Лучший инструмент										
Текстовые документы	RuBERT / XLM-R + NER										
Сканы/PDF	Tesseract + LayoutLM										
Стандартные суммы/даты	RegEx + dateparser										
Сложные договоры	Natasha (Yargy) + rule-based извлечение										

















	<ul style="list-style-type: none"> ○ RegEx для точного извлечения чисел. ○ Парсинг правил (например, для условий лизинга). 						
какие метрики и подходы наиболее эффективны для автоматической сверки извлечённых данных с существующими шаблонами или базами данных?							
ChatGPT от OpenAI	Для автоматической сверки извлечённых данных с шаблонами или базами данных важно сочетать правильные подходы сравнения и метрики оценки качества . Это особенно актуально в лизинге, где документы полны повторяющихся, но вариативно записанных данных (ФИО, суммы, даты, номера договоров и т.д.).						
	 ЭФФЕКТИВНЫЕ ПОДХОДЫ СВЕРКИ						
	 1. Точное сравнение (Exact Match) <ul style="list-style-type: none"> • Прямое сравнение: value1 == value2 • Используется для ИНН, номеров договоров, кодов, сумм (после нормализации) 						
	 2. Нечёткое сравнение (Fuzzy Matching) <ul style="list-style-type: none"> • Для строк с разной записью: ФИО, названия компаний, адреса • Алгоритмы: <ul style="list-style-type: none"> ○ Levenshtein distance ○ Jaro-Winkler ○ token_sort_ratio (из rapidfuzz) • Используется при опечатках или перестановках слов 						
	 3. Нормализация перед сравнением <ul style="list-style-type: none"> • Приведение к единому формату: <ul style="list-style-type: none"> ○ Числа: убрать пробелы/рубли → float ○ Даты: 01.02.2024 = 1 февраля 2024 ○ ФИО: Иванов Иван Иванович = Иванов И. И.  Инструменты: natasha, dateparser, pymorphy2						
	 4. Сравнение по шаблону (Template Matching) <ul style="list-style-type: none"> • Используется, если есть шаблоны: <ul style="list-style-type: none"> ○ «Договор №{contract_id} от {date}» • Вытягиваем данные по шаблону и сверяем 						
	 5. Сопоставление по справочникам / БД <ul style="list-style-type: none"> • Извлечённое значение сверяется с данными из БД <ul style="list-style-type: none"> ○ ИНН должен быть в списке контрагентов ○ Номер договора найден в реестре • Используются индексы, синонимы, aliases 						
	 МЕТРИКИ ЭФФЕКТИВНОСТИ СВЕРКИ  Для оценки точности сверки по полям (field-level): <table> <thead> <tr> <th>Метрика</th><th>Описание</th></tr> </thead> <tbody> <tr> <td>Accuracy</td><td>Доля правильно совпавших полей</td></tr> <tr> <td>Precision</td><td>Вероятность, что совпавшее значение — верное</td></tr> </tbody> </table>	Метрика	Описание	Accuracy	Доля правильно совпавших полей	Precision	Вероятность, что совпавшее значение — верное
Метрика	Описание						
Accuracy	Доля правильно совпавших полей						
Precision	Вероятность, что совпавшее значение — верное						

	<p>Recall Доля всех верных значений, которые мы нашли</p> <p>F1-score Баланс precision и recall</p> <p>Exact Match Rate Полное совпадение всех нужных полей в документе</p>														
	<p>🔗 Для сравнения строк и семантики:</p> <p>Метрика Что измеряет</p> <p>Levenshtein distance Кол-во правок для приведения строк к совпадению</p> <p>Jaro-Winkler Учет порядка символов и начальных совпадений</p> <p>Cosine Similarity (на векторах) Семантическая близость между текстами</p> <p>Token Sort/Set Ratio (fuzz) Учет перестановки слов, лишних символов</p>														
	<p>🧠 Комбинированная стратегия (практика)</p> <ol style="list-style-type: none"> 🔍 Предобработка: нормализация чисел, дат, строк 📄 Тип поля → метод сравнения: <table> <tr> <th>Поле</th><th>Подход</th></tr> <tr> <td>ФИО</td><td>Fuzzy Matching + лемматизация</td></tr> <tr> <td>Дата</td><td>Нормализация + Exact Match</td></tr> <tr> <td>Сумма</td><td>Приведение к числу + Exact Match</td></tr> <tr> <td>Номер договора</td><td>RegEx + Exact Match</td></tr> <tr> <td>Название орг.</td><td>Fuzzy Matching (Jaro / Token Set)</td></tr> <tr> <td>ИНН / КПП</td><td>Точное сравнение</td></tr> </table> <ol style="list-style-type: none"> ✅ Уровень доверия (confidence score): <ul style="list-style-type: none"> 100% — точное совпадение 80–99% — вероятное совпадение (ручная проверка) <80% — требует валидации 	Поле	Подход	ФИО	Fuzzy Matching + лемматизация	Дата	Нормализация + Exact Match	Сумма	Приведение к числу + Exact Match	Номер договора	RegEx + Exact Match	Название орг.	Fuzzy Matching (Jaro / Token Set)	ИНН / КПП	Точное сравнение
Поле	Подход														
ФИО	Fuzzy Matching + лемматизация														
Дата	Нормализация + Exact Match														
Сумма	Приведение к числу + Exact Match														
Номер договора	RegEx + Exact Match														
Название орг.	Fuzzy Matching (Jaro / Token Set)														
ИНН / КПП	Точное сравнение														
Gemini от Google	<p>Для автоматической сверки извлечённых данных с существующими шаблонами или базами данных существует несколько эффективных метрик и подходов. Выбор конкретного метода зависит от типа данных, структуры шаблонов и требований к точности.</p> <p>Метрики:</p> <ul style="list-style-type: none"> • Точность (Precision): <ul style="list-style-type: none"> Определяет долю правильно извлечённых данных среди всех извлечённых. Показывает, насколько можно доверять извлечённым данным. • Полнота (Recall): <ul style="list-style-type: none"> Определяет долю правильно извлечённых данных среди всех ожидаемых данных. Показывает, насколько полно извлечены все необходимые данные. • F1-мера (F1-score): <ul style="list-style-type: none"> Гармоническое среднее точности и полноты. Обеспечивает баланс между точностью и полнотой. • Расстояние Левенштейна: <ul style="list-style-type: none"> Измеряет минимальное количество изменений (вставок, удалений, замен), необходимых для преобразования одной строки в другую. Используется для сравнения строковых данных и выявления различий. • Коэффициент Жаккара: 														

	<ul style="list-style-type: none"> ○ Измеряет сходство между двумя множествами. ○ Используется для сравнения наборов данных и выявления совпадений. • Семантическое сходство: <ul style="list-style-type: none"> ○ Оценивает смысловое сходство между текстовыми данными. ○ Используется для сравнения текстовых данных, даже если они отличаются по форме. <p>Подходы:</p> <ul style="list-style-type: none"> • Сопоставление на основе правил: <ul style="list-style-type: none"> ○ Использование заранее определённых правил для сравнения извлечённых данных с шаблонами. ○ Подходит для данных с чёткой структурой. • Сопоставление на основе шаблонов: <ul style="list-style-type: none"> ○ Использование регулярных выражений или других шаблонов для поиска совпадений в данных. ○ Подходит для данных с повторяющейся структурой. • Сопоставление на основе баз данных: <ul style="list-style-type: none"> ○ Сравнение извлечённых данных с записями в базе данных. ○ Подходит для проверки данных на соответствие существующим записям. • Машинное обучение: <ul style="list-style-type: none"> ○ Использование алгоритмов машинного обучения для классификации и сопоставления данных. ○ Подходит для данных со сложной структурой или нечёткими правилами. • Гибридные подходы: <ul style="list-style-type: none"> ○ Комбинация нескольких подходов для достижения наилучших результатов. ○ Например, использование правил для предварительной обработки данных и машинного обучения для окончательной проверки. <p>Дополнительные рекомендации:</p> <ul style="list-style-type: none"> • Нормализация данных: Приведение данных к единому формату перед сравнением. • Обработка ошибок: Учёт возможных ошибок при извлечении данных. • Контекстный анализ: Учёт контекста данных для повышения точности сопоставления. • Человеческая проверка: Использование человеческой проверки для оценки результатов и корректировки алгоритмов. <p>Выбор оптимальных метрик и подходов зависит от конкретной задачи и характеристик данных. Рекомендуется провести эксперименты с различными методами, чтобы определить наиболее эффективный.</p>
YandexGPT от Яндекс	<p>Для автоматической сверки извлечённых данных с существующими шаблонами или базами данных можно использовать следующие метрики и подходы:</p> <p>Метрики</p> <ol style="list-style-type: none"> 1. Точность (Precision) — показывает, какая доля извлечённых данных действительно соответствует заданным шаблонам или записям в базе данных. Это важно для оценки качества извлечения без учёта данных, которые могли быть пропущены. 2. Полнота (Recall) — отражает, какая доля данных из шаблонов или базы данных была успешно извлечена. Этот показатель помогает понять, насколько полно система охватывает все необходимые данные.

	<p>3. F1-мера (F1 Score) — гармоническое среднее между точностью и полнотой. Она даёт более сбалансированную оценку эффективности сверки, объединяя точность и полноту в одном значении.</p> <p>4. Коэффициент Жаккара (Jaccard Index) — используется для сравнения схожести между двумя наборами данных. Он может быть полезен при сверке извлечённых данных с шаблонами, особенно когда важно оценить пересечение между множествами данных.</p> <p>5. Расстояние Левенштейна — позволяет измерять минимальное количество изменений (вставок, удалений, замен), необходимых для преобразования одной строки в другую. Это полезно для сопоставления текстовых данных, например, имён или адресов, которые могут содержать опечатки или небольшие различия.</p> <p>Подходы</p> <p>1. Точное совпадение (Exact Match) — сравнение извлечённых данных с данными в шаблонах или базе данных на предмет полного совпадения. Этот подход прост, но может быть недостаточно гибким для данных с возможными вариациями.</p> <p>2. Нечёткое сопоставление (Fuzzy Matching) — использует алгоритмы, такие как расстояние Левенштейна или коэффициент Сёренсена, для нахождения близких совпадений между строками. Это особенно полезно, когда данные могут содержать опечатки или небольшие различия, например, в написании имён или адресов.</p> <p>3. Машинное обучение и алгоритмы классификации — можно обучить модель машинного обучения на основе исторических данных для классификации извлечённых данных как соответствующих или не соответствующих шаблону. Подходы могут включать логистическую регрессию, деревья решений, случайный лес или нейронные сети.</p> <p>4. Использование векторных представлений (Vector Representations) — методы, такие как Word Embeddings (например, Word2Vec, GloVe), могут преобразовать текстовые данные в векторы, которые затем можно сравнивать для определения схожести. Это может быть полезно для сопоставления смыслового содержания извлечённых данных с данными в базе.</p> <p>5. Алгоритмы кластеризации — могут использоваться для группировки извлечённых данных по схожести с данными в шаблонах или базе данных. Это может помочь выявить группы данных, которые требуют дополнительной проверки или корректировки.</p> <p>6. Семантические сети и онтологии — использование семантических сетей и онтологий для сопоставления извлечённых данных с концептами и сущностями в базе знаний. Это позволяет учитывать семантические отношения между данными и улучшает качество сверки.</p> <p>7. Гибридные подходы — комбинация различных методов, например, нечёткого сопоставления и машинного обучения, может обеспечить более высокую точность и гибкость в сверке данных. Гибридные подходы позволяют использовать сильные стороны каждого метода для достижения лучших результатов.</p>
DeepSeek	<p>Для автоматической сверки извлечённых данных с шаблонами или базами данных в сфере лизинга (или других документно-ориентированных процессов) применяются комбинации метрик и подходов, которые можно разделить на три категории:</p> <p>I. Метрики для оценки точности сверки</p>

1. Для текстовых данных (реквизиты, названия, условия)		
Метрика	Применение	Инструменты
Точность (Precision)	Доля корректных совпадений среди всех найденных	sklearn.metrics.precision_score
Полнота (Recall)	Доля найденных совпадений относительно эталонных	sklearn.metrics.recall_score
F1-Score	Баланс между Precision и Recall	sklearn.metrics.f1_score
Jaccard Similarity	Сравнение множеств (например, списков условий договора)	sklearn.metrics.jaccard_score
Levenshtein Distance	Учёт опечаток (например, "ООО Альфа" vs. "Альфа ООО")	python-Levenshtein, rapidfuzz
2. Для числовых данных (суммы, даты, сроки)		
Метрика	Применение	
Exact Match (100% совпадение)	Для номеров договоров, точных сумм	
Mean Absolute Error (MAE)	Для допустимых отклонений (например, округлённые суммы)	
Допуск ±X%	Например, ±1% для финансовых расчётов	
3. Для сложных структур (JSON, таблицы, иерархии)		
Метрика	Применение	
Schema Validation	Проверка структуры (все ли поля заполнены, корректны ли типы данных)	
Field-Level Accuracy	Точность по каждому полю (например, 95% для "Номер договора", 80% для "Условия")	
II. Подходы к автоматической сверке		
1. Rule-Based (Жёсткие правила)		
Когда использовать: Для строго формализованных данных (реквизиты, даты, номера).		
Инструменты:		
<ul style="list-style-type: none"> Регулярные выражения (RegEx) – для поиска шаблонов (\d{4}-\d{2}-\d{2} для дат). Fuzzy Matching – сравнение строк с учётом опечаток (rapidfuzz, fuzzywuzzy). Парсеры на грамматиках (Yargy, Spacy Matcher) – для сложных правил (например, "Договор №[X] от [дата]"). 		
2. Машинное обучение / NLP (Гибкое сравнение)		
Когда использовать: Если данные неструктурированы или имеют синонимы (например, "аренда" vs. "лизинг").		
Методы:		
<ul style="list-style-type: none"> BERT / Sentence Transformers – семантическое сравнение текстов. Siamese Networks – для определения схожести записей (например, названий компаний). Active Learning – если модель нужно дообучать на спорных случаях. 		
3. Гибридные подходы		
<ol style="list-style-type: none"> Первый этап: Быстрая проверка по жёстким правилам (RegEx, точные совпадения). Второй этап: ML-модели для сложных случаев (синонимы, опечатки). Фильтрация по бизнес-логике (например, "сумма лизинга не может быть отрицательной"). 		

	<p>III. Инструменты и библиотеки</p> <table border="1"> <thead> <tr> <th>Задача</th><th>Инструменты</th></tr> </thead> <tbody> <tr> <td>Сравнение строк</td><td>rapidfuzz, python-Levenshtein, fuzzywuzzy</td></tr> <tr> <td>Семантическое сравнение</td><td>sentence-transformers, BERTScore, spaCy</td></tr> <tr> <td>Парсинг шаблонов</td><td>Yargy (для русского), Spacy Matcher, TREAT</td></tr> <tr> <td>Валидация структур</td><td>JSON Schema, Pydantic, Great Expectations (для данных в таблицах)</td></tr> <tr> <td>Визуализация расхождений</td><td>diffliB, Pandas (выделение расхождений в DataFrame)</td></tr> </tbody> </table> <p>IV. Практические рекомендации</p> <ol style="list-style-type: none"> 1. Разметьте эталонные данные – без "золотого стандарта" метрики бессмысленны. 2. Комбинируйте подходы – Rule-Based + ML даёт максимальную точность. 3. Логируйте ошибки – это поможет улучшить модель (например, через Label Studio). 4. Настройте пороги – например, считать совпадением $F1 > 0.9$. 	Задача	Инструменты	Сравнение строк	rapidfuzz, python-Levenshtein, fuzzywuzzy	Семантическое сравнение	sentence-transformers, BERTScore, spaCy	Парсинг шаблонов	Yargy (для русского), Spacy Matcher, TREAT	Валидация структур	JSON Schema, Pydantic, Great Expectations (для данных в таблицах)	Визуализация расхождений	diffliB, Pandas (выделение расхождений в DataFrame)
Задача	Инструменты												
Сравнение строк	rapidfuzz, python-Levenshtein, fuzzywuzzy												
Семантическое сравнение	sentence-transformers, BERTScore, spaCy												
Парсинг шаблонов	Yargy (для русского), Spacy Matcher, TREAT												
Валидация структур	JSON Schema, Pydantic, Great Expectations (для данных в таблицах)												
Визуализация расхождений	diffliB, Pandas (выделение расхождений в DataFrame)												
<p>какие генеративные модели можно использовать для автоматического заполнения отсутствующих данных или формирования кратких описаний документов?</p>													
<p>ChatGPT от OpenAI</p>	<p>Для автоматического заполнения пропущенных данных и генерации кратких описаний документов (особенно в юридической и финансовой сфере, как в лизинге) отлично подходят современные генеративные модели на базе трансформеров.</p> <hr/> <p> Что можно делать генеративными моделями:</p> <table border="1"> <thead> <tr> <th>Задача</th><th>Примеры</th></tr> </thead> <tbody> <tr> <td> Заполнение пропущенных данных</td><td>Дополнение недостающих ФИО, дат, организаций, на основе контекста</td></tr> <tr> <td> Генерация краткого описания</td><td>"Договор лизинга №145/2024 между ООО 'Альфа' и ООО 'Бета' на сумму 1 500 000 руб. от 01.02.2024"</td></tr> <tr> <td> Реформулировка и суммирование документа</td><td>/ Автоматический аннотат документа, его цель и структура</td></tr> <tr> <td> Создание шаблона или конспекта</td><td>Обобщение структуры договора или акта</td></tr> </tbody> </table> <hr/> <p> Рекомендуемые генеративные модели</p> <p> 1. SberAI RuGPT-3 / RuGPT-3.5</p> <ul style="list-style-type: none"> • Русский аналог GPT-3 от Сбера • Отлично работает на длинных юридических текстах • Умеет продолжать текст, дополнять пропущенные значения • Можно использовать zero-shot или few-shot (с примерами) <p> 2. mGPT (multilingual GPT)</p> <ul style="list-style-type: none"> • Поддержка русского, разработана AI21 + Sber 	Задача	Примеры	 Заполнение пропущенных данных	Дополнение недостающих ФИО, дат, организаций, на основе контекста	 Генерация краткого описания	"Договор лизинга №145/2024 между ООО 'Альфа' и ООО 'Бета' на сумму 1 500 000 руб. от 01.02.2024"	 Реформулировка и суммирование документа	/ Автоматический аннотат документа, его цель и структура	 Создание шаблона или конспекта	Обобщение структуры договора или акта		
Задача	Примеры												
 Заполнение пропущенных данных	Дополнение недостающих ФИО, дат, организаций, на основе контекста												
 Генерация краткого описания	"Договор лизинга №145/2024 между ООО 'Альфа' и ООО 'Бета' на сумму 1 500 000 руб. от 01.02.2024"												
 Реформулировка и суммирование документа	/ Автоматический аннотат документа, его цель и структура												
 Создание шаблона или конспекта	Обобщение структуры договора или акта												






	<ul style="list-style-type: none"> • Работает в духе GPT, но легче <p>✓ 3. T5 / mT5 / RuT5</p> <ul style="list-style-type: none"> • Хорошо справляется с задачами типа "переведи → опиши → дополни" • Есть версии, дообученные на юридических текстах <p>✓ 4. ChatGLM, Mistral, Gemini, Claude (через API)</p> <ul style="list-style-type: none"> • Для диалоговых сценариев: «Что указано в документе?», «Какая дата договора?» 										
	<p>⚙ Как использовать (инструменты):</p> <table> <tr> <th>Инструмент</th><th>Преимущества</th></tr> <tr> <td>HuggingFace Transformers</td><td>Прямой доступ к RuGPT, T5, mGPT</td></tr> <tr> <td>SberCloud API</td><td>Доступ к RuGPT-3.5 через API</td></tr> <tr> <td>LangChain + LLM</td><td>Конструкции с шаблонами и контекстами</td></tr> <tr> <td>FastAPI</td><td>Встраивание в backend систему</td></tr> </table>	Инструмент	Преимущества	HuggingFace Transformers	Прямой доступ к RuGPT, T5, mGPT	SberCloud API	Доступ к RuGPT-3.5 через API	LangChain + LLM	Конструкции с шаблонами и контекстами	FastAPI	Встраивание в backend систему
Инструмент	Преимущества										
HuggingFace Transformers	Прямой доступ к RuGPT, T5, mGPT										
SberCloud API	Доступ к RuGPT-3.5 через API										
LangChain + LLM	Конструкции с шаблонами и контекстами										
FastAPI	Встраивание в backend систему										
	<p>🎯 Комбинированный подход:</p> <table> <tr> <th>Шаг</th><th>Что использовать</th></tr> <tr> <td>1. NER / OCR / Extraction</td><td>Получить данные из документа</td></tr> <tr> <td>2. Проверка полноты</td><td>Найти недостающие поля</td></tr> <tr> <td>3. Генерация</td><td>Подсказать модели: "Допиши описание договора на основе этих данных"</td></tr> <tr> <td>4. Сверка</td><td>Проверить достоверность по базе (если надо)</td></tr> </table>	Шаг	Что использовать	1. NER / OCR / Extraction	Получить данные из документа	2. Проверка полноты	Найти недостающие поля	3. Генерация	Подсказать модели: "Допиши описание договора на основе этих данных"	4. Сверка	Проверить достоверность по базе (если надо)
Шаг	Что использовать										
1. NER / OCR / Extraction	Получить данные из документа										
2. Проверка полноты	Найти недостающие поля										
3. Генерация	Подсказать модели: "Допиши описание договора на основе этих данных"										
4. Сверка	Проверить достоверность по базе (если надо)										
Gemini от Google	<p>Для автоматического заполнения отсутствующих данных или формирования кратких описаний документов можно использовать различные генеративные модели. Выбор конкретной модели зависит от типа данных, объема доступных данных и требуемой точности.</p> <p>Генеративные модели для заполнения отсутствующих данных:</p> <ul style="list-style-type: none"> • Вариационные автоэнкодеры (VAE): <ul style="list-style-type: none"> ○ VAE могут использоваться для заполнения пропущенных значений в табличных данных или изображениях. ○ Они обучаются на существующих данных и могут генерировать правдоподобные значения для отсутствующих данных. • Генеративно-состязательные сети (GAN): <ul style="list-style-type: none"> ○ GAN могут генерировать реалистичные данные, которые могут быть использованы для заполнения пропущенных значений. ○ Они особенно эффективны для заполнения пропущенных значений в изображениях. • Трансформеры: <ul style="list-style-type: none"> ○ Трансформеры, такие как BERT и GPT, могут использоваться для заполнения пропущенных значений в текстовых данных. ○ Они могут учитывать контекст окружающих слов, чтобы генерировать правдоподобные значения. <p>Генеративные модели для формирования кратких описаний документов:</p> <ul style="list-style-type: none"> • Трансформеры: <ul style="list-style-type: none"> ○ Трансформеры, такие как BART и T5, могут использоваться для создания кратких описаний документов. ○ Они могут извлекать ключевую информацию из документов и генерировать связные и информативные описания. 										

	<ul style="list-style-type: none"> • Seq2Seq модели: <ul style="list-style-type: none"> ○ Модели Seq2Seq могут использоваться для преобразования длинных документов в короткие описания. ○ Они обучаются на парах "документ-описание" и могут генерировать описания для новых документов. • Экстрактивные методы: <ul style="list-style-type: none"> ○ Так же есть методы которые не генерируют новый текст, а выбирают самые главные предложения из текста, и формируют описание из них. <p>Дополнительные рекомендации:</p> <ul style="list-style-type: none"> • Предварительная обработка данных: <ul style="list-style-type: none"> ○ Важно предварительно обработать данные перед использованием генеративных моделей. ○ Это может включать в себя очистку данных, токенизацию и векторизацию. • Оценка производительности: <ul style="list-style-type: none"> ○ Необходимо оценить производительность генеративных моделей, чтобы убедиться, что они генерируют качественные данные. ○ Это можно сделать с помощью таких метрик, как точность, полнота и F1-мера. • Тонкая настройка: <ul style="list-style-type: none"> ○ Генеративные модели часто требуют тонкой настройки, чтобы достичь оптимальной производительности. ○ Это может включать в себя настройку гиперпараметров и архитектуры модели. <p>Выбор оптимальной генеративной модели зависит от конкретной задачи и характеристик данных. Рекомендуется провести эксперименты с различными моделями, чтобы определить наиболее эффективную.</p>
<p>YandexGPT от Яндекс</p>	<p>Для автоматического заполнения отсутствующих данных или формирования кратких описаний документов можно использовать следующие генеративные модели:</p> <ol style="list-style-type: none"> 1. Генеративно-сопоставительные сети (GAN, Generative Adversarial Networks) — это модель, состоящая из двух компонентов: генератора и дискриминатора. Генератор создаёт новые данные, пытаясь обмануть дискриминатор, который, в свою очередь, пытается отличить настоящие данные от сгенерированных. GAN могут быть использованы для генерации текстовых данных, включая заполнение пропусков в документах. 2. Вариационные автоэнкодеры (VAE, Variational Autoencoders) — это тип нейронных сетей, которые обучаются кодировать данные в сжатое представление и затем декодировать их обратно. VAE могут быть применены для генерации новых данных, похожих на обучающие, что может быть полезно для заполнения отсутствующих значений в документах. 3. Трансформеры (Transformers) — это архитектуру нейронных сетей, которая показала высокие результаты в задачах обработки естественного языка. Модели на основе трансформеров, такие как GPT (Generative Pre-trained Transformer) и BERT (Bidirectional Encoder Representations from Transformers), могут быть использованы для генерации текста, включая создание кратких описаний документов или заполнение пропусков в данных. 4. Модели на основе рекуррентных нейронных сетей (RNN, Recurrent Neural Networks) — например, LSTM (Long Short-Term Memory) или GRU

	<p>(Gated Recurrent Units), которые хорошо справляются с задачами обработки последовательностей данных, таких как текст. Они могут быть обучены генерировать текст на основе контекста и заполнять отсутствующие данные в документах.</p> <p>5. Скрытые марковские модели (HMM, Hidden Markov Models) — это статистические модели, которые могут быть использованы для моделирования последовательностей данных. HMM могут быть применены для предсказания и заполнения пропущенных значений в текстовых данных на основе вероятностных распределений.</p> <p>6. Модели на основе языковых моделей (Language Models) — такие как n-граммы или более сложные модели, обученные на больших корпусах текстов. Они могут предсказывать следующие слова или фразы в тексте на основе контекста и использоваться для заполнения пропусков или генерации описаний.</p> <p>Выбор конкретной модели зависит от специфики задачи, объёма и характера данных, а также доступных ресурсов. Рекомендуется провести анализ данных и эксперименты с различными моделями, чтобы определить наиболее подходящий подход для вашей задачи.</p>																																
DeepSeek	<p>Для автоматической сверки извлечённых данных с шаблонами или базами данных в сфере лизинга (или других документно-ориентированных процессов) применяются комбинации метрик и подходов, которые можно разделить на три категории:</p>																																
	<p>I. Метрики для оценки точности сверки</p> <p>1. Для текстовых данных (реквизиты, названия, условия)</p> <table><tr><th>Метрика</th><th>Применение</th><th>Инструменты</th></tr><tr><td>Точность (Precision)</td><td>Доля корректных совпадений среди всех найденных</td><td>sklearn.metrics.precision_score</td></tr><tr><td>Полнота (Recall)</td><td>Доля найденных совпадений относительно эталонных</td><td>sklearn.metrics.recall_score</td></tr><tr><td>F1-Score</td><td>Баланс между Precision и Recall</td><td>sklearn.metrics.f1_score</td></tr><tr><td>Jaccard Similarity</td><td>Сравнение множеств (например, списков условий договора)</td><td>sklearn.metrics.jaccard_score</td></tr><tr><td>Levenshtein Distance</td><td>Учёт опечаток (например, "ООО Альфа" vs. "Альфа ООО")</td><td>python-Levenshtein, rapidfuzz</td></tr></table> <p>2. Для числовых данных (суммы, даты, сроки)</p> <table><tr><th>Метрика</th><th>Применение</th></tr><tr><td>Exact Match (100% совпадение)</td><td>Для номеров договоров, точных сумм</td></tr><tr><td>Mean Absolute Error (MAE)</td><td>Для допустимых отклонений (например, округлённые суммы)</td></tr><tr><td>Допуск ±X%</td><td>Например, ±1% для финансовых расчётов</td></tr></table> <p>3. Для сложных структур (JSON, таблицы, иерархии)</p> <table><tr><th>Метрика</th><th>Применение</th></tr><tr><td>Schema Validation</td><td>Проверка структуры (все ли поля заполнены, корректны ли типы данных)</td></tr><tr><td>Field-Level Accuracy</td><td>Точность по каждому полю (например, 95% для "Номер договора", 80% для "Условия")</td></tr></table>	Метрика	Применение	Инструменты	Точность (Precision)	Доля корректных совпадений среди всех найденных	sklearn.metrics.precision_score	Полнота (Recall)	Доля найденных совпадений относительно эталонных	sklearn.metrics.recall_score	F1-Score	Баланс между Precision и Recall	sklearn.metrics.f1_score	Jaccard Similarity	Сравнение множеств (например, списков условий договора)	sklearn.metrics.jaccard_score	Levenshtein Distance	Учёт опечаток (например, "ООО Альфа" vs. "Альфа ООО")	python-Levenshtein, rapidfuzz	Метрика	Применение	Exact Match (100% совпадение)	Для номеров договоров, точных сумм	Mean Absolute Error (MAE)	Для допустимых отклонений (например, округлённые суммы)	Допуск ±X%	Например, ±1% для финансовых расчётов	Метрика	Применение	Schema Validation	Проверка структуры (все ли поля заполнены, корректны ли типы данных)	Field-Level Accuracy	Точность по каждому полю (например, 95% для "Номер договора", 80% для "Условия")
Метрика	Применение	Инструменты																															
Точность (Precision)	Доля корректных совпадений среди всех найденных	sklearn.metrics.precision_score																															
Полнота (Recall)	Доля найденных совпадений относительно эталонных	sklearn.metrics.recall_score																															
F1-Score	Баланс между Precision и Recall	sklearn.metrics.f1_score																															
Jaccard Similarity	Сравнение множеств (например, списков условий договора)	sklearn.metrics.jaccard_score																															
Levenshtein Distance	Учёт опечаток (например, "ООО Альфа" vs. "Альфа ООО")	python-Levenshtein, rapidfuzz																															
Метрика	Применение																																
Exact Match (100% совпадение)	Для номеров договоров, точных сумм																																
Mean Absolute Error (MAE)	Для допустимых отклонений (например, округлённые суммы)																																
Допуск ±X%	Например, ±1% для финансовых расчётов																																
Метрика	Применение																																
Schema Validation	Проверка структуры (все ли поля заполнены, корректны ли типы данных)																																
Field-Level Accuracy	Точность по каждому полю (например, 95% для "Номер договора", 80% для "Условия")																																
	<p>II. Подходы к автоматической сверке</p> <p>1. Rule-Based (Жёсткие правила)</p>																																

	<p>Когда использовать: Для строго формализованных данных (реквизиты, даты, номера).</p> <p>Инструменты:</p> <ul style="list-style-type: none"> • Регулярные выражения (Regex) – для поиска шаблонов (<code>\d{4}-\d{2}-\d{2}</code> для дат). • Fuzzy Matching – сравнение строк с учётом опечаток (<code>rapidfuzz</code>, <code>fuzzywuzzy</code>). • Парсеры на грамматиках (Yargy, Spacy Matcher) – для сложных правил (например, "Договор №[X] от [дата]"). <p>2. Машинное обучение / NLP (Гибкое сравнение)</p> <p>Когда использовать: Если данные неструктурированы или имеют синонимы (например, "аренда" vs. "лизинг").</p> <p>Методы:</p> <ul style="list-style-type: none"> • BERT / Sentence Transformers – семантическое сравнение текстов. • Siamese Networks – для определения схожести записей (например, названий компаний). • Active Learning – если модель нужно дообучать на спорных случаях. <p>3. Гибридные подходы</p> <ol style="list-style-type: none"> 1. Первый этап: Быстрая проверка по жёстким правилам (Regex, точные совпадения). 2. Второй этап: ML-модели для сложных случаев (синонимы, опечатки). 3. Фильтрация по бизнес-логике (например, "сумма лизинга не может быть отрицательной"). 												
	<p>III. Инструменты и библиотеки</p> <table border="1"> <thead> <tr> <th>Задача</th><th>Инструменты</th></tr> </thead> <tbody> <tr> <td>Сравнение строк</td><td><code>rapidfuzz</code>, <code>python-Levenshtein</code>, <code>fuzzywuzzy</code></td></tr> <tr> <td>Семантическое сравнение</td><td><code>sentence-transformers</code>, <code>BERTScore</code>, <code>spaCy</code></td></tr> <tr> <td>Парсинг шаблонов</td><td><code>Yargy</code> (для русского), <code>Spacy Matcher</code>, <code>TREAT</code></td></tr> <tr> <td>Валидация структур</td><td><code>JSON Schema</code>, <code>Pydantic</code>, <code>Great Expectations</code> (для данных в таблицах)</td></tr> <tr> <td>Визуализация расхождений</td><td><code>difflib</code>, <code>Pandas</code> (выделение расхождений в <code>DataFrame</code>)</td></tr> </tbody> </table>	Задача	Инструменты	Сравнение строк	<code>rapidfuzz</code> , <code>python-Levenshtein</code> , <code>fuzzywuzzy</code>	Семантическое сравнение	<code>sentence-transformers</code> , <code>BERTScore</code> , <code>spaCy</code>	Парсинг шаблонов	<code>Yargy</code> (для русского), <code>Spacy Matcher</code> , <code>TREAT</code>	Валидация структур	<code>JSON Schema</code> , <code>Pydantic</code> , <code>Great Expectations</code> (для данных в таблицах)	Визуализация расхождений	<code>difflib</code> , <code>Pandas</code> (выделение расхождений в <code>DataFrame</code>)
Задача	Инструменты												
Сравнение строк	<code>rapidfuzz</code> , <code>python-Levenshtein</code> , <code>fuzzywuzzy</code>												
Семантическое сравнение	<code>sentence-transformers</code> , <code>BERTScore</code> , <code>spaCy</code>												
Парсинг шаблонов	<code>Yargy</code> (для русского), <code>Spacy Matcher</code> , <code>TREAT</code>												
Валидация структур	<code>JSON Schema</code> , <code>Pydantic</code> , <code>Great Expectations</code> (для данных в таблицах)												
Визуализация расхождений	<code>difflib</code> , <code>Pandas</code> (выделение расхождений в <code>DataFrame</code>)												
	<p>IV. Практические рекомендации</p> <ol style="list-style-type: none"> 1. Разметьте эталонные данные – без "золотого стандарта" метрики бессмысленны. 2. Комбинируйте подходы – Rule-Based + ML даёт максимальную точность. 3. Логируйте ошибки – это поможет улучшить модель (например, через <code>Label Studio</code>). 4. Настройте пороги – например, считать совпадением $F1 > 0.9$. <p>Если у вас есть конкретные примеры данных, можно предложить более детализированное решение!</p> <p>какие генеративные модели можно использовать для автоматического заполнения отсутствующих данных или формирования кратких описаний документов?</p> <p>Для автоматического заполнения отсутствующих данных и генерации описаний документов в сфере лизинга можно использовать современные генеративные модели III. Вот лучшие подходы и инструменты:</p> <p>1. Генеративные модели для заполнения данных</p>												

	<p>1.1. Заполнение структурированных полей (реквизиты, суммы, даты)</p> <ul style="list-style-type: none"> • Модели на основе Transformer: <ul style="list-style-type: none"> ○ T5 (Text-to-Text Transfer Transformer) <ul style="list-style-type: none"> ▪ Может достраивать пропущенные поля по контексту (например, "Договор №____ → Договор №12345"). ▪ Пример: google/flan-t5-large (доступна в Hugging Face). ○ GPT-3/4 (OpenAI) <ul style="list-style-type: none"> ▪ Полезен для предсказания данных на основе шаблонов (например, "Сумма договора: ____ → 1 000 000 руб.>"). • Специализированные решения: <ul style="list-style-type: none"> ○ TabFormer (от IBM) – для заполнения табличных данных. ○ REaLTabFormer – генерация реалистичных табличных данных. <p>1.2. Восстановление пропущенных данных в тексте</p> <ul style="list-style-type: none"> • Модели с маскированием: <ul style="list-style-type: none"> ○ BERT (заполнение пропусков через [MASK]). ○ RoBERTa – улучшенная версия BERT для сложных контекстов. 										
	<p>2. Генерация описаний документов</p> <p>2.1. Краткие summaries (аннотации)</p> <ul style="list-style-type: none"> • T5/GPT-3.5/4 – для создания сжатых описаний. <ul style="list-style-type: none"> ○ Пример промпта для GPT: "Сгенерируй краткое описание лизингового договора в 2-3 предложениях на основе следующих данных: {данные}." • BART (Facebook) – особенно хорош для суммаризации текстов. <p>2.2. Развёрнутые отчёты</p> <ul style="list-style-type: none"> • ChatGPT (GPT-4-turbo) – для генерации аналитических выкладок. <ul style="list-style-type: none"> ○ Пример: автоматическое создание отчёта о платежах на основе табличных данных. <p>2.3. Многоязычные описания</p> <ul style="list-style-type: none"> • mT5 (мультязычная версия T5) – если документы на разных языках. 										
	<p>3. Инструменты и фреймворки</p> <table> <tr> <th>Задача</th><th>Инструменты</th></tr> <tr> <td>Заполнение пропусков</td><td>Hugging Face Transformers (bert-base-multilingual-cased, t5-small)</td></tr> <tr> <td>Генерация текста</td><td>OpenAI API (gpt-4), Llama 3, Mistral 7B (локальные альтернативы)</td></tr> <tr> <td>Работа с табличными данными</td><td>TabFormer, Pandas + кастомные правила</td></tr> <tr> <td>Суммаризация документов</td><td>BART, Pegasus (например, google/pegasus-xsum)</td></tr> </table>	Задача	Инструменты	Заполнение пропусков	Hugging Face Transformers (bert-base-multilingual-cased, t5-small)	Генерация текста	OpenAI API (gpt-4), Llama 3, Mistral 7B (локальные альтернативы)	Работа с табличными данными	TabFormer, Pandas + кастомные правила	Суммаризация документов	BART, Pegasus (например, google/pegasus-xsum)
Задача	Инструменты										
Заполнение пропусков	Hugging Face Transformers (bert-base-multilingual-cased, t5-small)										
Генерация текста	OpenAI API (gpt-4), Llama 3, Mistral 7B (локальные альтернативы)										
Работа с табличными данными	TabFormer, Pandas + кастомные правила										
Суммаризация документов	BART, Pegasus (например, google/pegasus-xsum)										
	<p>4. Практические примеры</p> <p>4.1. Автозаполнение договора</p> <table> <tr> <td>Исходный текст:</td><td></td></tr> <tr> <td>"Договор №____ между [MASK] и ООО 'Лизинговая Компания' на сумму [MASK] руб."</td><td></td></tr> <tr> <td>Результат после обработки BERT/T5:</td><td></td></tr> <tr> <td>"Договор №123-ЛЗ между ООО 'Альфа' и ООО 'Лизинговая Компания' на сумму 1 500 000 руб."</td><td></td></tr> </table> <p>4.2. Генерация описания</p>	Исходный текст:		"Договор №____ между [MASK] и ООО 'Лизинговая Компания' на сумму [MASK] руб."		Результат после обработки BERT/T5:		"Договор №123-ЛЗ между ООО 'Альфа' и ООО 'Лизинговая Компания' на сумму 1 500 000 руб."			
Исходный текст:											
"Договор №____ между [MASK] и ООО 'Лизинговая Компания' на сумму [MASK] руб."											
Результат после обработки BERT/T5:											
"Договор №123-ЛЗ между ООО 'Альфа' и ООО 'Лизинговая Компания' на сумму 1 500 000 руб."											

	<p>Входные данные:</p> <ul style="list-style-type: none"> • Номер: 123-ЛЗ • Сумма: 1 500 000 руб. • Срок: 36 месяцев <p>Результат GPT-4:</p> <p><i>"Договор лизинга №123-ЛЗ предусматривает финансирование в размере 1 500 000 рублей на срок 3 года. Объект лизинга – оборудование для производства."</i></p>
	<p>5. Как выбрать модель?</p> <ul style="list-style-type: none"> • Если данные структурированы → T5 или TabFormer. • Если нужны творческие описания → GPT-4 или Llama 3. • Для бюджетных решений → Fine-tuned BERT или BART. <p>Важно:</p> <ul style="list-style-type: none"> • Для работы с финансовыми данными используйте локальные модели (например, Llama 3 с приватным развёртыванием), чтобы избежать утечки информации. • Обязательно валидируйте сгенерированные данные через rule-based проверки. <p>Для старта попробуйте Hugging Face + transformers или OpenAI API, если нужна быстрота. Для максимального контроля над данными – обучайте модель на своих примерах договоров.</p>
<p>какие этапы предобработки изображений (например, фильтрация шума, бинаризация, коррекция наклона) наиболее эффективно улучшают точность OCR при работе с низкокачественными документами или сложными форматами, такими как таблицы и графики?</p>	
ChatGPT от OpenAI	<p>Для улучшения точности OCR (например, Tesseract, EasyOCR, PaddleOCR) важно грамотно организовать предобработку изображений. Ниже приведены ключевые этапы, которые доказали свою эффективность на практике:</p>
	<p> Этапы предобработки изображений для OCR</p> <p>1.  Коррекция наклона (Deskewing)</p> <ul style="list-style-type: none"> •  Особенно важно при сканах с ручной подачи в сканер • Алгоритмы: <ul style="list-style-type: none"> ○ Hough Line Transform ○ Minimum area bounding box ○ Image moments (OpenCV)
	<p>2.  Удаление шума (Denoising)</p> <ul style="list-style-type: none"> • Убирает фоновый мусор и улучшает распознаваемость символов • Фильтры: <ul style="list-style-type: none"> ○ Gaussian Blur (мягкий шум) ○ Median Blur (сильный шум) ○ Non-local Means Denoising
	<p>3.  Бинаризация (Thresholding)</p> <ul style="list-style-type: none"> • Преобразует изображение в чёрно-белое • Алгоритмы:

	<ul style="list-style-type: none"> ○ Otsu's Thresholding ○ Adaptive Thresholding (для неравномерной освещённости) 												
	4.  Инверсия изображения <ul style="list-style-type: none"> • OCR лучше читает тёмный текст на светлом фоне • Обязательно проверить, если текст — светлый на тёмном фоне 												
	5.  Удаление рамок, линий и графики <ul style="list-style-type: none"> • Особенно полезно для таблиц, где линии мешают OCR • Подход: <ul style="list-style-type: none"> ○ Morphological operations (cv2.getStructuringElement) ○ Contour filtering по размеру и форме 												
	6.  Увеличение разрешения (Super Resolution) <ul style="list-style-type: none"> • Если DPI < 150–200, OCR качество резко падает • Подходы: <ul style="list-style-type: none"> ○ OpenCV resize (INTER_CUBIC) — до 300 DPI ○ ESRGAN / Real-ESRGAN — глубокое увеличение 												
	7.  Вырезание и исправление таблиц <ul style="list-style-type: none"> • Detect and isolate tables до OCR • Используются: <ul style="list-style-type: none"> ○ Contour analysis ○ TableNet / CascadeTabNet (если нужны структуры таблиц) 												
	8.  Нормализация размера шрифта и контраста <ul style="list-style-type: none"> • Выравнивание визуальной яркости и размера текста • cv2.equalizeHist() или CLAHE 												
	 Эффект от предобработки: <table> <tr> <th>Метод</th><th>Увеличение точности OCR</th></tr> <tr> <td>Deskewing</td><td>+5–15% (особенно для Tesseract)</td></tr> <tr> <td>Denoising</td><td>+3–7%</td></tr> <tr> <td>Adaptive Binarization</td><td>+10–20% при плохом освещении</td></tr> <tr> <td>Line removal</td><td>+10–30% в таблицах</td></tr> <tr> <td>Super-resolution</td><td>+10–20% на мелком шрифте</td></tr> </table>	Метод	Увеличение точности OCR	Deskewing	+5–15% (особенно для Tesseract)	Denoising	+3–7%	Adaptive Binarization	+10–20% при плохом освещении	Line removal	+10–30% в таблицах	Super-resolution	+10–20% на мелком шрифте
Метод	Увеличение точности OCR												
Deskewing	+5–15% (особенно для Tesseract)												
Denoising	+3–7%												
Adaptive Binarization	+10–20% при плохом освещении												
Line removal	+10–30% в таблицах												
Super-resolution	+10–20% на мелком шрифте												
	 Инструменты и библиотеки: <table> <tr> <th>Библиотека</th><th>Назначение</th></tr> <tr> <td>OpenCV</td><td>Обработка изображений (всё выше перечисленное)</td></tr> <tr> <td>scikit-image</td><td>Доп. функции для фильтрации</td></tr> <tr> <td>PaddleOCR</td><td>Встроенная предобработка + OCR</td></tr> <tr> <td>imgaug / albumentations</td><td>Аугментации и фильтрация</td></tr> <tr> <td>Tesseract</td><td>OCR-движок (с --psm и --oem для настройки)</td></tr> </table>	Библиотека	Назначение	OpenCV	Обработка изображений (всё выше перечисленное)	scikit-image	Доп. функции для фильтрации	PaddleOCR	Встроенная предобработка + OCR	imgaug / albumentations	Аугментации и фильтрация	Tesseract	OCR-движок (с --psm и --oem для настройки)
Библиотека	Назначение												
OpenCV	Обработка изображений (всё выше перечисленное)												
scikit-image	Доп. функции для фильтрации												
PaddleOCR	Встроенная предобработка + OCR												
imgaug / albumentations	Аугментации и фильтрация												
Tesseract	OCR-движок (с --psm и --oem для настройки)												
Gemini от Google	При работе с низкокачественными документами или сложными форматами, такими как таблицы и графики, точность OCR может быть значительно улучшена с помощью правильной предобработки изображений. Вот наиболее эффективные этапы:												

	<p>1. Фильтрация шума:</p> <ul style="list-style-type: none"> • Медианный фильтр: <ul style="list-style-type: none"> ○ Эффективно удаляет "соль и перец" шум, сохраняя резкость границ. ○ Особенно полезен для сканированных документов с пятнами и точками. • Гауссовский фильтр: <ul style="list-style-type: none"> ○ Сглаживает изображение, уменьшая общий шум. ○ Подходит для размытых изображений или изображений с градиентным шумом. <p>2. Бинаризация:</p> <ul style="list-style-type: none"> • Метод Оцу: <ul style="list-style-type: none"> ○ Автоматически определяет оптимальный порог для разделения изображения на передний план (текст) и задний план (фон). ○ Хорошо работает для изображений с контрастным текстом. • Адаптивная бинаризация: <ul style="list-style-type: none"> ○ Применяет разные пороги для разных областей изображения. ○ Необходима для изображений с неравномерным освещением или контрастом. <p>3. Коррекция наклона:</p> <ul style="list-style-type: none"> • Преобразование Хафа: <ul style="list-style-type: none"> ○ Определяет угол наклона линий в изображении и поворачивает изображение для выравнивания. ○ Критически важно для отсканированных документов с наклоном. • Коррекция перспективы: <ul style="list-style-type: none"> ○ Используется для изображений, снятых под углом, для исправления искажений. ○ Часто используется для фотографий документов. <p>4. Улучшение контраста и яркости:</p> <ul style="list-style-type: none"> • Гистограммное выравнивание: <ul style="list-style-type: none"> ○ Растягивает гистограмму изображения, улучшая контраст. ○ Полезно для изображений с низким контрастом. • Коррекция гаммы: <ul style="list-style-type: none"> ○ Корректирует яркость изображения, делая текст более различимым. <p>5. Сегментация:</p> <ul style="list-style-type: none"> • Разделение на регионы: <ul style="list-style-type: none"> ○ Разделяет изображение на области, содержащие текст, таблицы или графики. ○ Позволяет применять разные методы OCR к разным областям. • Выделение строк и символов: <ul style="list-style-type: none"> ○ Разделяет текст на отдельные строки и символы. ○ Улучшает точность распознавания отдельных символов. <p>6. Удаление артефактов:</p> <ul style="list-style-type: none"> • Удаление линий и рамок: <ul style="list-style-type: none"> ○ Удаляет линии и рамки, которые могут мешать распознаванию текста. ○ Необходимо для обработки таблиц и графиков. • Удаление пятен и точек: <ul style="list-style-type: none"> ○ Удаляет мелкие артефакты, которые могут быть ошибочно распознаны как текст. <p>Особенности работы с таблицами и графиками:</p>
--	--

	<ul style="list-style-type: none"> • Для таблиц важно точно выделить границы ячеек и разделить текст в каждой ячейке. • Для графиков необходимо распознать текст на осях и легендах, а также извлечь данные из самого графика. <p>Важно отметить:</p> <ul style="list-style-type: none"> • Последовательность и комбинация этапов предобработки зависят от конкретного типа документов и их качества. • Использование правильной комбинации методов предобработки может значительно улучшить точность OCR.
YandexGPT от Яндекс	<p>Наиболее эффективные этапы предобработки изображений для улучшения точности OCR при работе с низкокачественными документами и сложными форматами:</p> <ol style="list-style-type: none"> 1. Фильтрация шума: применение фильтров для удаления шумов и артефактов, таких как размытость, царапины или пятна. 2. Бинаризация: преобразование полутоновых изображений в чёрно-белые изображения с использованием порога для отделения объектов от фона. 3. Коррекция наклона: выравнивание изображений для устранения перекосов и наклонов, которые могут возникнуть из-за неправильного положения документа или неравномерного освещения. 4. Сглаживание: применение фильтров для уменьшения ступенчатости границ между объектами и фоном, что улучшает распознавание символов. 5. Контрастирование: изменение контрастности изображения для улучшения различимости символов и фона. 6. Бинаризация таблиц и графиков: преобразование табличных данных и графиков в чёрно-белые изображения для облегчения их распознавания. <p>Эти этапы предобработки изображений могут быть выполнены с использованием различных методов и алгоритмов, таких как медианная фильтрация, адаптивная бинаризация, выравнивание гистограммы и другие. Выбор оптимального метода зависит от специфики задачи и качества исходных изображений.</p>
DeepSeek	<p>Для достижения максимальной точности OCR при обработке низкокачественных документов (сканов, факсов, таблиц и графиков) требуется комплексная предобработка изображений. Вот ключевые этапы и их применение:</p> <p>1. Основные этапы предобработки</p> <p>1.1. Коррекция геометрии</p> <ul style="list-style-type: none"> • Выравнивание наклона (Deskew) <i>Проблема:</i> Криво отсканированные страницы снижают точность распознавания. <i>Решение:</i> <ul style="list-style-type: none"> ○ Алгоритм Хафа для детекции угла наклона + аффинное преобразование. ○ Инструменты: OpenCV (cv2.HoughLines, cv2.warpAffine). <p>1.2. Улучшение контраста и яркости</p> <ul style="list-style-type: none"> • Адаптивная гистограммная эквализация (CLAHE) <i>Проблема:</i> Низкая читаемость текста из-за неравномерного освещения. <i>Решение:</i> <ul style="list-style-type: none"> ○ CLAHE улучшает локальный контраст без усиления шума. ○ Инструменты: cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8)). <p>1.3. Фильтрация шума</p> <ul style="list-style-type: none"> • Медианный фильтр / Non-Local Means Denoising <i>Проблема:</i> Артефакты сканирования, "песок" на фоне. <i>Решение:</i> <ul style="list-style-type: none"> ○ cv2.medianBlur (для удаления солевого шума).

	<ul style="list-style-type: none"> ○ cv2.fastNlMeansDenoising (для сохранения четкости текста). 												
	1.4. Бинаризация <ul style="list-style-type: none"> • Адаптивный порог (Adaptive Thresholding) <i>Проблема:</i> Неоднородный фон (например, пожелтевшая бумага). <i>Решение:</i> <ul style="list-style-type: none"> ○ cv2.adaptiveThreshold с блоком 11x11 и константой C=2. ○ Для сложных случаев: Sauvola's Binarization (библиотека skimage.filters). 												
	1.5. Удаление артефактов <ul style="list-style-type: none"> • Морфологические операции <i>Проблема:</i> Тонкие линии, точки, "грязь". <i>Решение:</i> <ul style="list-style-type: none"> ○ Закрытие (dilation + erosion): cv2.morphologyEx с ядром 3x3. ○ Удаление мелких объектов: cv2.connectedComponentsWithStats. 												
	1.6. Сегментация таблиц и графиков <ul style="list-style-type: none"> • Детекция линий (Line Detection) <i>Проблема:</i> OCR путает текст и границы таблиц. <i>Решение:</i> <ul style="list-style-type: none"> ○ Алгоритм LSD (Line Segment Detector) в cv2.createLineSegmentDetector. ○ Удаление линий перед распознаванием: 												
	2. Специальные методы для сложных форматов												
	2.1. Для таблиц <ul style="list-style-type: none"> • Использование CNN для детекции ячеек <ul style="list-style-type: none"> ○ Модели: CascadeTabNet, TableNet (предобученные на датасетах ICDAR). ○ Инструменты: PyTorch + OpenCV для постобработки. 												
	2.2. Для графиков и диаграмм <ul style="list-style-type: none"> • Удаление графиков перед OCR <ul style="list-style-type: none"> ○ Детекция через цветовые фильтры (cv2.inRange для HSV). ○ Замена графиков белым прямоугольником. 												
	3. Инструменты и библиотеки												
	<table> <tr> <th>Задача</th><th>Инструменты</th></tr> <tr> <td>Геометрическая коррекция</td><td>OpenCV (Hough Lines, warpAffine)</td></tr> <tr> <td>Бинаризация</td><td>OpenCV, scikit-image (Sauvola, Otsu)</td></tr> <tr> <td>Шумоподавление</td><td>OpenCV (medianBlur, fastNlMeansDenoising)</td></tr> <tr> <td>Сегментация таблиц</td><td>CascadeTabNet, TableNet (GitHub)</td></tr> <tr> <td>OCR</td><td>Tesseract 5 (с LSTM), EasyOCR (для сложных шрифтов)</td></tr> </table>	Задача	Инструменты	Геометрическая коррекция	OpenCV (Hough Lines, warpAffine)	Бинаризация	OpenCV, scikit-image (Sauvola, Otsu)	Шумоподавление	OpenCV (medianBlur, fastNlMeansDenoising)	Сегментация таблиц	CascadeTabNet, TableNet (GitHub)	OCR	Tesseract 5 (с LSTM), EasyOCR (для сложных шрифтов)
Задача	Инструменты												
Геометрическая коррекция	OpenCV (Hough Lines, warpAffine)												
Бинаризация	OpenCV, scikit-image (Sauvola, Otsu)												
Шумоподавление	OpenCV (medianBlur, fastNlMeansDenoising)												
Сегментация таблиц	CascadeTabNet, TableNet (GitHub)												
OCR	Tesseract 5 (с LSTM), EasyOCR (для сложных шрифтов)												
	5. Рекомендации <ol style="list-style-type: none"> 1. Тестируйте комбинации методов – Например, CLANE + Sauvola часто дает лучший результат, чем каждый метод отдельно. 2. Кастомизируйте параметры – Размер ядра для бинаризации, пороги для шумоподавления. 3. Используйте GPU – Для CNN-моделей (TableNet) и сложной предобработки. 4. Проверьте OCR-движок – Tesseract 5 с --psm 6 (блочная сегментация) подходит для таблиц. 												

	Для особо сложных документов (например, рукописные пометки) рассмотрите Fine-tuning Tesseract на своих данных.
--	---

На основании данных ответов генеративных помощников были определены характеристики ответов по каждой из гипотез. В результате была составлена таблица 2, содержащая рейтинг исследуемых инструментов по таким критериям, как полнота ответа, точность ответа и наличие дополнительной информации.

Таблица 2 – Сравнительная характеристика помощников

№	Наименование	Точность	Полнота	Доп. инф-ия
1	ChatGPT	Соответствует гипотезам	Ответы предоставлены в полном объеме	Содержит большое количество примеров и пояснений
2	DeepSeek	Соответствует гипотезам	Ответы предоставлены в полном объеме	Предоставляет примеры работы с библиотеками на Python
3	Gemini	Соответствует гипотезам	Ответы предоставлены в полном объеме	Практически не содержит
4	YandexGPT	Соответствует гипотезам	Ответы предоставлены не в полном объеме	Не содержит

В соответствии с полученной сравнительной характеристикой был выбран генеративный помощник ChatGPT от OpenAI, потому что предоставляет наиболее расширенную информацию по всем вопросам.

Задание 2.

Для постановки задачи для генеративных помощников был составлен следующий тезис, по которому необходимо сформировать аннотацию:

«Составь аннотацию для магистерской дипломной работы. Работа посвящена применению гибридных моделей машинного обучения. В ней были исследованы классические методы машинного обучения для задачи классификации документов в сфере лизинга. Также определены методы нейронных сетей для именованного распознавания сущностей, извлечения

табличных данных и верификации документов. В результате была разработана система, позволяющая хранить и обрабатывать лизинговые документы и заявки по клиентам».

В результате выполнения запросов была составлена таблица аннотаций по каждому из генеративных помощников. Описание представлено в таблице 3.

Таблица 3 – Аннотации генеративных помощников

Наименование	Аннотация
ChatGPT от OpenAI	В данной магистерской работе рассматривается применение гибридных моделей машинного обучения для автоматизации обработки документов в сфере лизинга. На первом этапе были исследованы классические алгоритмы машинного обучения, применяемые к задаче классификации различных типов лизинговых документов. Далее проведён анализ и реализация методов нейронных сетей для задач именованного распознавания сущностей (NER), извлечения табличных данных и верификации информации, содержащейся в документах. Особое внимание уделено предобработке данных и интеграции различных моделей в единую систему. В результате проведённого исследования была разработана программная система, обеспечивающая хранение, классификацию и интеллектуальную обработку лизинговых документов и клиентских заявок, что способствует повышению точности и эффективности документооборота в лизинговой деятельности.
Gemini от Google	<p>Аннотация к магистерской дипломной работе:</p> <p>Настоящая магистерская дипломная работа посвящена разработке и исследованию гибридных моделей машинного обучения для автоматизации обработки документов в сфере лизинга. Целью работы является создание системы, способной эффективно классифицировать различные типы лизинговых документов, извлекать из них ключевую информацию, а также верифицировать данные, содержащиеся в заявках клиентов.</p> <p>В рамках исследования были изучены и протестированы как классические методы машинного обучения (например, наивный байесовский классификатор, метод опорных векторов, случайный лес, градиентный бустинг), так и современные нейросетевые архитектуры (трансформеры, сверточные нейронные сети) для решения задач классификации документов, именованного распознавания сущностей и извлечения табличных данных.</p> <p>Для верификации данных, содержащихся в заявках клиентов, были разработаны алгоритмы сравнения извлеченной информации с существующими шаблонами и базами данных с использованием метрик сходства строк и семантического анализа.</p> <p>Результатом работы стала разработка комплексной системы, которая позволяет автоматизировать процессы хранения, обработки и анализа лизинговых документов и заявок клиентов. Система обеспечивает высокую точность классификации и извлечения информации, а также позволяет сократить время обработки документов и повысить эффективность работы лизинговых компаний.</p>

	<p>Ключевые слова: машинное обучение, гибридные модели, классификация документов, именованное распознавание сущностей, извлечение табличных данных, верификация данных, лизинг, автоматизация.</p>
YandexGPT от Яндекса	<p>Данная магистерская диссертация посвящена исследованию и применению гибридных моделей машинного обучения для автоматизации обработки лизинговых документов. В работе проводится сравнительный анализ классических методов машинного обучения и современных нейронных сетей в контексте решения прикладных задач документооборота.</p> <p>В первой части исследования рассматриваются и анализируются традиционные алгоритмы машинного обучения для классификации документов в сфере лизинга. Особое внимание уделяется оценке эффективности различных подходов и их применимости к специфике лизинговой документации.</p> <p>Вторая часть работы посвящена применению методов глубокого обучения, включая технологии именованного распознавания сущностей, извлечения табличных данных и верификации документов. Проводится исследование возможностей нейронных сетей в решении задач обработки структурированной и неструктурированной информации.</p> <p>В результате проведенного исследования разработана комплексная система автоматизации документооборота, включающая модули хранения, обработки и анализа лизинговых документов и клиентских заявок. Система реализована с использованием гибридного подхода, сочетающего классические методы машинного обучения и современные нейросетевые технологии.</p> <p>Практическая значимость работы заключается в создании эффективного инструмента для автоматизации рутинных операций с документами, что позволяет существенно повысить производительность работы специалистов по лизингу и минимизировать человеческий фактор при обработке документации.</p> <p>Ключевые слова: гибридные модели, машинное обучение, нейронные сети, обработка документов, лизинг, автоматизация документооборота, классификация документов, распознавание сущностей, извлечение данных, верификация документов.</p>
DeepSeek	<p>Тема: «Разработка гибридной системы обработки лизинговых документов на основе методов машинного обучения»</p> <p>Ключевые слова: гибридные модели машинного обучения, классификация документов, NER (Named Entity Recognition), извлечение табличных данных, верификация документов, обработка естественного языка (NLP), лизинговые договоры.</p> <p>Цель работы: Повышение эффективности обработки лизинговых документов за счёт комбинации классических методов машинного обучения и современных нейросетевых подходов.</p> <p>Основные задачи:</p> <ol style="list-style-type: none"> 1. Анализ классических алгоритмов ML (Random Forest, XGBoost, SVM) для классификации документов. 2. Исследование нейросетевых моделей (BERT, Transformer-based NER) для извлечения сущностей: <ul style="list-style-type: none"> ○ реквизиты договоров, ○ суммы и даты, ○ табличные данные. 3. Разработка системы верификации извлечённых данных с использованием:

	<ul style="list-style-type: none"> ○ rule-based проверок, ○ метрик схожести (F1-score, Levenshtein distance). <p>4. Создание прототипа системы хранения и обработки документов с интерфейсом для поиска и анализа.</p> <p>Методы и технологии:</p> <ul style="list-style-type: none"> • Классические ML: Scikit-learn (для классификации), CatBoost (ансамбли). • Нейросетевые модели: RuBERT (NER), Tesseract + TableNet (таблицы), Sentence Transformers (семантический поиск). • Инструменты: Python, PyTorch, OpenCV, FastAPI (бэкенд), Streamlit (интерфейс). <p>Результаты:</p> <ol style="list-style-type: none"> 1. Достигнута точность 92% в классификации типов документов (против 85% у базовых методов). 2. Реализовано извлечение сущностей с F1-score 0.89 (для реквизитов договоров). 3. Разработана система верификации, сократившая ручную проверку на 40%. 4. Внедрён модуль поиска документов по ключевым параметрам (клиент, сумма, срок). <p>Научная новизна:</p> <ul style="list-style-type: none"> • Предложен гибридный подход, сочетающий interpretability классических ML и точность нейросетей. • Оптимизирован пайплайн обработки сложных документов (сканы + таблицы) с адаптивной предобработкой изображений. <p>Практическая значимость: Система апробирована на данных лизинговой компании «Х». Позволяет автоматизировать 70% рутинных операций с документами.</p> <p>Объём и структура работы: 75 стр., включая 35 рисунков и 12 таблиц. Содержит 4 главы:</p> <ol style="list-style-type: none"> 1. Анализ предметной области. 2. Методология гибридного ML. 3. Реализация и тестирование. 4. Экономическое обоснование внедрения. <p>Приложения:</p> <ul style="list-style-type: none"> • Дашборд для анализа документов (Streamlit). • Датасет размеченных лизинговых договоров.
--	---

По результатам составления аннотации были сделаны выводы, представленные в таблице 4.

Таблица 4 – Сравнительная характеристика для составления аннотации

Наименование	Достоинства	Недостатки
ChatGPT от OpenAI	<ol style="list-style-type: none"> 1) краткая аннотация; 2) описаны основные аспекты; 3) результат соответствует запросу. 	<ol style="list-style-type: none"> 1) не продумано описание используемых инструментов; 2) не описаны результаты работы с моделями.

Gemini от Google	1) описаны основные аспекты; 2) результат соответствует запросу; 3) описаны ключевые слова; 4) определена цель работы; 5) описаны проанализированные методы.	1) длинная аннотация.
YandexGPT от Яндекса	1) описаны основные аспекты; 2) результат соответствует запросу; 3) описаны ключевые слова; 4) определена цель работы; 5) описаны проанализированные методы; 6) описана практическая значимость	1) длинная аннотация.
DeerSeek	1) описаны цели; 2) описаны результаты	1) несоответствие формата; 2) длинная аннотация; 3) описаны объем и структура; 4) описаны приложения.

Таким образом, на основе представленной сравнительной характеристики полученных аннотаций для магистерской диссертации было определено, что наиболее пригодным генеративным помощником для решения данной задачи является YandexGPT, по причине того, что содержит наиболее развернутую информацию по работе, которую можно сократить до нужного размера.

Задание 3.

В таблице 5 представлены достоинства и недостатки использования генеративных помощников для профессиональной деятельности.

Таблица 5 – Достоинства и недостатки генеративных помощников

Достоинства	Недостатки
<p>Экономия времени и автоматизация</p> <ul style="list-style-type: none"> – Быстро генерируют тексты, коды, идеи. – Автоматизируют рутинные задачи (анализ данных, переводы, написание документов). <p>Широкий кругозор и знание разных тем</p> <ul style="list-style-type: none"> – Обладают обширной базой знаний по разным областям. – Хорошо подходят для обучения и поиска новых идей. <p>Гибкость и персонализация</p> <ul style="list-style-type: none"> – Могут адаптироваться под стиль общения пользователя. – Позволяют решать разные задачи: от кодинга до креатива. <p>Поддержка на разных языках</p> <ul style="list-style-type: none"> – ChatGPT и Gemini хорошо работают с английским и русским. – DeepSeek лучше понимает китайский. – YandexGPT больше ориентирован на русский рынок. <p>Творческий генеративный потенциал</p> <ul style="list-style-type: none"> – Генерируют стихи, сценарии, дизайн-концепции, музыку. – Помогают в креативных индустриях и маркетинге. 	<p>Риск генерации ошибок</p> <ul style="list-style-type: none"> • Не всегда точны, могут выдавать некорректные или устаревшие данные. • Могут "галлюцинировать" — придумывать факты. <p>Ограничения в реальном времени</p> <ul style="list-style-type: none"> • Некоторые модели (например, ChatGPT) могут не иметь доступа к свежим данным. • Для актуальных данных приходится использовать web-поиск. <p>Проблемы с интерпретацией сложных запросов</p> <ul style="list-style-type: none"> • Иногда сложно задать точный запрос, и помощник может отвечать обобщённо. • Технические темы требуют перепроверки и уточнения. <p>Конфиденциальность и безопасность</p> <ul style="list-style-type: none"> • Нельзя вводить чувствительную информацию, так как ответы могут анализироваться разработчиками моделей. • В корпоративных средах есть риски утечки данных. <p>Ограничения по функционалу</p> <ul style="list-style-type: none"> • Генеративные ИИ не заменяют специализированные инструменты (например, Excel, MATLAB). • В программировании часто дают решения, которые нужно дорабатывать вручную.

Задание 4.

Для повышения качества сравнения было принято решение провести проверку работы генеративных помощников на

конструкторской/технологической частях и при формировании списка литературы другой ВКРМ. В качестве темы используется: «Программная система прогнозирования котировок на нефтяном рынке».

В таблице 6 представлены результаты применения генеративных помощников в конструкторской части ВКРМ.

Постановка задачи: «Опиши процесс настройки алгоритмов ML для прогнозирования нефтяных котировок с учётом использования новостного фона в качестве экзогенного фактора».

Таблица 6 – Генеративный помощник в конструкторской части

Наименование	Результат
ChatGPT от OpenAI	<p>1. Определение цели и сбор данных • Цель прогнозирования: Определите, что именно вы хотите прогнозировать (например, цены на нефть через день, неделю и т.д.). • Сбор данных: • Исторические данные по ценам на нефть: Сбор данных о ценах на нефть за определенный период (например, WTI или Brent). • Экзогенные факторы: Сбор новостей, связанных с нефтяным рынком. Это могут быть статьи, пресс-релизы, отчеты и т.д. Используйте API новостных агрегаторов или веб-скрейпинг для получения данных. • Дополнительные факторы: Сбор других экономических индикаторов (например, запасы нефти, спрос и предложение).</p> <p>2. Предобработка данных • Очистка данных: Удалите дубликаты, обработайте пропуски. • Форматирование временных рядов: Убедитесь, что временные метки согласованы между различными наборами данных. • Текстовая обработка новостей: • Примените методы обработки естественного языка (NLP) для анализа текстов новостей (например, токенизация, удаление стоп-слов). • Используйте векторизацию (TF-IDF, Word2Vec, BERT) для преобразования текстов в числовые представления.</p> <p>3. Создание признаков • Признаки из временных рядов: Создайте временные признаки, такие как скользящие средние, волатильность и т.д. • Признаки из новостей: Извлеките признаки из новостей: • Анализ настроений: Определите общее настроение новостей (позитивное, негативное, нейтральное) и используйте это как признак. • Частота упоминаний: Количество упоминаний ключевых слов или событий в новостях.</p> <p>4. Разделение данных • Обучающая и тестовая выборки: Разделите данные на обучающую и тестовую выборки (например, 80% на обучение и 20% на тестирование) с учетом временной последовательности.</p> <p>5. Выбор модели • Выбор алгоритмов: Рассмотрите различные алгоритмы машинного обучения, такие как: • Линейная регрессия • Деревья решений • Случайный лес • Градиентный бустинг (XGBoost, LightGBM) • Рекуррентные нейронные сети (RNN) для работы с временными рядами</p> <p>6. Обучение модели • Обучение на обучающей выборке: Настройте выбранные модели на обучающей выборке. • Гиперпараметрическая настройка: Используйте методы кросс-валидации для оптимизации гиперпараметров моделей.</p> <p>7. Оценка модели • Тестирование на тестовой выборке: Оцените производительность модели на тестовой выборке с использованием метрик (MAE, RMSE, R²). • Анализ ошибок: Проанализируйте ошибки модели для понимания ее слабых мест.</p> <p>8. Внедрение и мониторинг • Внедрение модели: Разработайте систему для автоматического обновления прогнозов. • Мониторинг производительности:</p>

		<p>Постоянно отслеживайте производительность модели и при необходимости обновляйте ее. 9. Обновление данных • Регулярное обновление: Обновляйте модель с использованием новых данных о ценах и новостях для поддержания актуальности прогнозов.</p>
Gemini Google	от	<p>1. Сбор и подготовка данных • Данные о котировках: ◦ Получите исторические данные о ценах на нефть (например, WTI, Brent) из надежных источников (Bloomberg, Reuters и т. д.). ◦ Включите такие функции, как цены открытия, закрытия, максимумы, минимумы и объемы торгов. • Новостные данные: ◦ Соберите новостные статьи, связанные с нефтяным рынком, из различных источников (новостные агентства, финансовые веб-сайты, социальные сети). ◦ Обратите внимание на новости, касающиеся геополитических событий, решений ОПЕК, экономических показателей и т. д. • Предварительная обработка данных: ◦ Очистите данные от отсутствующих значений и выбросов. ◦ Нормализуйте или масштабируйте данные, чтобы обеспечить их соответствие для алгоритмов ML. ◦ Преобразуйте текстовые новостные данные в числовые признаки с помощью методов обработки естественного языка (NLP), таких как анализ тональности, частота ключевых слов и встраивание слов.</p> <p>2. Выбор алгоритмов ML • Алгоритмы временных рядов: ◦ ARIMA (авторегрессионное интегрированное скользящее среднее) ◦ LSTM (долгая краткосрочная память) ◦ GRU (вентильный рекуррентный блок) • Алгоритмы регрессии: ◦ Линейная регрессия ◦ Случайный лес ◦ Градиентный бустинг (например, XGBoost, LightGBM)</p> <p>3. Разработка функций • Функции временных рядов: ◦ Запоздывающие значения цен на нефть ◦ Скользящие средние ◦ Волатильность • Новостные функции: ◦ Показатели тональности новостных статей ◦ Частота ключевых слов, связанных с нефтяным рынком ◦ Векторные представления новостных статей (например, Word2Vec, GloVe) • Экзогенные факторы: ◦ Экономические показатели (например, ВВП, инфляция) ◦ Геополитические события ◦ Решения ОПЕК</p> <p>4. Обучение и оценка модели • Разделите данные на наборы для обучения, проверки и тестирования. • Обучите выбранные алгоритмы ML на обучающем наборе. • Оптимизируйте гиперпараметры моделей с помощью проверки на проверочном наборе. • Оцените производительность моделей на тестовом наборе, используя соответствующие метрики (например, среднеквадратичную ошибку, среднюю абсолютную ошибку).</p> <p>5. Развертывание и мониторинг • Разверните обученную модель в производственной среде. • Отслеживайте производительность модели и при необходимости переобучайте ее. • Регулярно обновляйте новостные данные и другие экзогенные факторы. Ключевые соображения • Качество и релевантность новостных данных имеют решающее значение для точности прогнозов. • Обработка естественного языка (NLP) играет важную роль в извлечении полезной информации из новостных статей. • Выбор подходящих алгоритмов ML и функций зависит от конкретных характеристик данных и целей прогнозирования. • Очень важно постоянно контролировать модель и переобучать ее по мере поступления новых данных.</p>
YandexGPT Яндекса	от	<p>1. Сбор и подготовка данных:</p> <ul style="list-style-type: none"> • Исторические данные о котировках нефти: соберите данные о ценах на нефть за определённый период времени. Это могут быть данные о closing price (цена закрытия), volume (объём торгов) и других параметрах. • Новостные данные: соберите релевантные новости, связанные с нефтяным рынком, экономикой, политикой и другими факторами,

	<p>которые могут влиять на цены на нефть. Новостные данные могут быть в виде текстовых статей, заголовков или специализированных новостных индексов.</p> <ul style="list-style-type: none"> • Предварительная обработка данных: очистите данные от выбросов, пропущенных значений и других аномалий. Приведите данные к единому формату и масштабируйте их при необходимости. <p>2. Извлечение признаков (Feature Engineering):</p> <ul style="list-style-type: none"> • Технические индикаторы: рассчитайте технические индикаторы, такие как скользящие средние, RSI (индекс относительной силы), MACD (схождение/расхождение скользящих средних) и другие, которые могут помочь в прогнозировании цен на нефть. • Новостные признаки: извлеките признаки из новостных данных, например, частоту упоминания определённых терминов, тональность новостей (позитивная, негативная, нейтральная) или важность новостей (на основе источников или заголовков). • Дополнительные признаки: рассмотрите возможность добавления других экзогенных факторов, таких как экономические показатели, политические события, данные о запасах нефти и т. д. <p>3. Выбор модели машинного обучения:</p> <ul style="list-style-type: none"> • Выберите подходящий алгоритм машинного обучения для прогнозирования цен на нефть. Это может быть регрессия, деревья решений, случайные леса, градиентный бустинг, нейронные сети и т. п. • Учитывайте особенности данных и задачи прогнозирования при выборе модели. Например, для работы с временными рядами могут быть полезны модели ARIMA, SARIMA или LSTM. <p>4. Обучение модели:</p> <ul style="list-style-type: none"> • Разделите данные на обучающую и тестовую выборки. • Обучите выбранную модель на обучающей выборке, оптимизируя гиперпараметры с помощью кросс-валидации или других методов. • Оцените качество модели на тестовой выборке, используя метрики, такие как RMSE (среднеквадратическая ошибка), MAE (средняя абсолютная ошибка) или R^2 (коэффициент детерминации). <p>5. Анализ результатов и оптимизация:</p> <ul style="list-style-type: none"> • Проанализируйте результаты прогнозирования и определите, насколько хорошо модель учитывает новостной фон и другие экзогенные факторы. • Если результаты неудовлетворительны, рассмотрите возможность изменения модели, добавления новых признаков или корректировки гиперпараметров. • Повторите процесс обучения и оценки качества модели до достижения желаемого результата. <p>6. Тестирование в реальных условиях:</p> <ul style="list-style-type: none"> • После настройки модели протестируйте её в реальных условиях, используя актуальные данные. • Сравните прогнозы модели с фактическими ценами на нефть и оцените её эффективность. • При необходимости внесите дополнительные корректировки в модель или процесс прогнозирования. <p>7. Мониторинг и обновление модели:</p>
--	---

	<ul style="list-style-type: none"> Регулярно мониторьте производительность модели и отслеживайте изменения в данных. Обновляйте модель по мере необходимости, добавляя новые данные или корректируя признаки.
DeepSeek	<p>1. Сбор и подготовка данных</p> <p>1.1. Источники данных</p> <ul style="list-style-type: none"> Ценовые данные: <ul style="list-style-type: none"> Brent/WTI (исторические котировки с частотой, например, daily/hourly) – Yahoo Finance, Bloomberg, Quandl. Технические индикаторы (скользящие средние, RSI, объёмы торгов и т.д.). Макроэкономические показатели: <ul style="list-style-type: none"> Данные по запасам нефти (EIA, API), добыче (OPEC reports), спросу/предложению. Новостной фон: <ul style="list-style-type: none"> Финансовые новости (Reuters, Bloomberg, CNBC). Социальные медиа (Twitter, Reddit). Официальные заявления (центральные банки, ОПЕК+). <p>1.2. Предобработка данных</p> <ul style="list-style-type: none"> Для временных рядов: <ul style="list-style-type: none"> Нормализация (MinMaxScaler, StandardScaler). Устранение пропусков (интерполяция, заполнение средним). Добавление лагов (lag features) – цены за предыдущие дни. Для новостей: <ul style="list-style-type: none"> Токенизация и очистка текста (удаление стоп-слов, стемминг). Векторизация (TF-IDF, Word2Vec, BERT). Извлечение тональности (Sentiment Analysis – VADER, FinBERT). Кластеризация тем (Topic Modeling – LDA, NMF). <p>2. Feature Engineering</p> <ul style="list-style-type: none"> Создание признаков из новостей: <ul style="list-style-type: none"> Сентимент-анализ: средняя тональность за день. Частота упоминаний ключевых слов ("нефть", "санкции", "ОПЕК"). Событийные метки (бинарные флаги – например, "заявление ОПЕК"). Экономические/рыночные факторы: <ul style="list-style-type: none"> Волатильность (например, ATR, стандартное отклонение). Корреляция с другими активами (USD, акции нефтяных компаний). <p>3. Выбор и настройка модели</p> <p>3.1. Классические ML-модели (если данные стационарны)</p> <ul style="list-style-type: none"> Методы с экзогенными переменными: <ul style="list-style-type: none"> SARIMAX – для учета сезонности и новостей как внешних факторов. XGBoost/LightGBM – работа с табличными данными + feature importance. Ансамбли: <ul style="list-style-type: none"> Стекинг (например, LSTM + Gradient Boosting). <p>3.2. Глубокое обучение (если сложные зависимости)</p> <ul style="list-style-type: none"> Гибридные архитектуры:

	<ul style="list-style-type: none"> ○ LSTM/GRU + Attention – для временных рядов. ○ BERT/LSTM – для анализа новостей и их влияния на цены. • Transformer-based модели: <ul style="list-style-type: none"> ○ Temporal Fusion Transformer (TFT) – учитывает временные и внешние факторы. <p>3.3. Обучение и валидация</p> <ul style="list-style-type: none"> • Разделение данных: <ul style="list-style-type: none"> ○ Временной кросс-валидация (TimeSeriesSplit). • Метрики: <ul style="list-style-type: none"> ○ RMSE, MAE – для регрессии. ○ Directional Accuracy – процент верных предсказаний направления цены. <p>4. Интеграция новостного фона</p> <ul style="list-style-type: none"> • Подходы: <ol style="list-style-type: none"> 1. Конкатенация фичей: объединение числовых данных и NLP-фичей. 2. Мультимодальные модели: отдельные ветки для текста и цен. 3. Событийные триггеры: резкие изменения тональности → коррекция прогноза. <p>5. Деплоймент и мониторинг</p> <ul style="list-style-type: none"> • Реализация: <ul style="list-style-type: none"> ○ Пайплайн в Apache Airflow или MLflow. • Мониторинг: <ul style="list-style-type: none"> ○ Drift detection (отслеживание изменения распределения данных). ○ A/B тестирование новых моделей. <p>6. Проблемы и решения</p> <ul style="list-style-type: none"> • Новости → запаздывающий эффект: <ul style="list-style-type: none"> ○ Вводить лаги для NLP-фичей. • Шум в данных: <ul style="list-style-type: none"> ○ Использовать Kalman Filter или Wavelet-преобразования. • Интерпретируемость: <ul style="list-style-type: none"> ○ SHAP/LIME для анализа влияния новостей.
--	---

В таблице 7 представлены результаты применения генеративных помощников в технологической части ВКРМ.

Постановка задачи: «Опиши границы области применения программной системы прогнозирования котировок на нефтяном рынке с учётом того, что для прогнозирования используется модель, учитывающая в себе в качестве экзогенного признака новостной фон, полученный в результате работы другой модели».

Таблица 7 – Генеративный помощник в технологической части

Наименование	Результат
ChatGPT от OpenAI	<p>1. Ограничения данных • Качество и доступность данных: Прогнозирование зависит от качества и полноты данных как по ценам на нефть, так и по новостям. Неполные или искаженные данные могут привести к ошибочным прогнозам. • Временные задержки: Новостной фон может не отражать текущую ситуацию мгновенно. Временные задержки в публикации новостей или в их восприятии рынком могут снизить точность прогнозов.</p> <p>2. Модели и алгоритмы • Ограничения модели: Используемая модель может иметь свои ограничения, связанные с выбором алгоритма, его гиперпараметрами и предположениями. Например, линейные модели могут не учитывать сложные нелинейные зависимости. • Зависимость от экзогенных факторов: Прогнозирование сильно зависит от качества модели, анализирующей новостной фон. Если эта модель неэффективна, это негативно скажется на конечных прогнозах.</p> <p>3. Влияние внешних факторов • Экономические и политические события: Внешние факторы, такие как экономические кризисы, политическая нестабильность или природные катастрофы, могут оказывать значительное влияние на нефтяные котировки, но их сложно предсказать с помощью новостей. • Регуляторные изменения: Изменения в законодательстве или в политике стран-производителей нефти могут повлиять на рынок, но такие события могут не быть учтены в новостном фоне.</p> <p>4. Применение в реальном времени • Скорость обработки: Для эффективного использования системы в реальном времени требуется высокая скорость обработки данных. Задержки могут снизить актуальность прогнозов. • Рынок и ликвидность: Система может быть менее эффективна в условиях низкой ликвидности или высокой волатильности, когда цены быстро меняются из-за спекулятивных действий.</p> <p>5. Ограничения в интерпретации • Сложность интерпретации результатов: Результаты прогнозирования могут быть трудными для интерпретации, особенно если модель использует сложные алгоритмы (например, нейронные сети), что может затруднить принятие решений на их основе. • Неопределенность и риск: Все прогнозы содержат элемент неопределенности. Инвесторы должны понимать риски, связанные с использованием таких систем для принятия инвестиционных решений.</p> <p>6. Ограниченная область применения • Географические ограничения: Модель может быть разработана для специфической географической области (например, только для рынка США или Европы) и может не работать эффективно в других регионах из-за различных экономических условий или культурных факторов. • Секторные ограничения: Прогнозирование может быть более эффективным для определенных секторов (например, только для нефтегазовой отрасли), но менее применимо к другим секторам экономики.</p> <p>7. Этические и юридические аспекты • Этические вопросы: Использование новостей для торговли может вызывать этические вопросы, особенно если информация используется неправомерно. • Юридические ограничения: В некоторых юрисдикциях могут существовать законы, ограничивающие использование определенных типов данных или методов для торговли.</p>
Gemini от Google	<p>1. Качество и релевантность новостных данных: • Источники данных: ◦ Система сильно зависит от качества и разнообразия источников новостей. Если источники смещены, неполны или ненадежны, прогнозы будут искажены. ◦ Важно учитывать как традиционные новостные агентства, так и социальные сети, но последние требуют тщательной фильтрации из-за высокого уровня шума. • Обработка естественного языка (NLP): ◦ Точность анализа тональности и извлечения ключевых слов из новостей критически важна. Ошибки в NLP могут привести к неправильной интерпретации новостного фона и, следовательно, к неточным прогнозам. ◦ Сложность языка, наличие сленга и</p>

	<p>иронии могут затруднить анализ. • Запоздывание информации: ◦ Новостная информация может поступать с задержкой, что снижает ее актуальность для краткосрочных прогнозов.</p> <p>2. Сложность и непредсказуемость нефтяного рынка: • Геополитические факторы: ◦ Нефтяной рынок очень чувствителен к геополитическим событиям, которые часто непредсказуемы. ◦ Модель может не успевать адаптироваться к внезапным изменениям. • Экономические факторы: ◦ Глобальные экономические тенденции, такие как инфляция и процентные ставки, оказывают значительное влияние на цены на нефть. ◦ Прогнозирование этих факторов само по себе является сложной задачей. • Факторы предложения и спроса: ◦ Решения ОПЕК, изменения в добыче сланцевой нефти и колебания спроса на энергию могут существенно повлиять на цены.</p> <p>3. Ограничения модели машинного обучения: • Переобучение: ◦ Модель может слишком хорошо подстроиться под исторические данные и потерять способность к обобщению. ◦ Это особенно актуально при работе с нестабильными временными рядами. • Нестационарность данных: ◦ Цены на нефть и новостной фон могут меняться со временем, что делает статические модели менее эффективными. ◦ Модель должна быть способна адаптироваться к изменяющимся рыночным условиям. • Зависимость от модели, генерирующей новостной фон: ◦ Так как новостной фон генерируется другой моделью, то точность и адекватность первой модели на прямую влияет на точность прогноза цен на нефть. Ошибки первой модели будут переноситься на вторую.</p> <p>4. Временные горизонты прогнозирования: • Краткосрочные прогнозы: ◦ Новостной фон может быть полезен для краткосрочных прогнозов, так как он отражает текущие рыночные настроения. • Долгосрочные прогнозы: ◦ Долгосрочные прогнозы более подвержены влиянию фундаментальных факторов, которые сложно учесть с помощью новостного фона. Области применения: • Трейдинг: система может использоваться для принятия краткосрочных торговых решений. • Управление рисками: система может помочь в оценке рисков, связанных с волатильностью цен на нефть. • Анализ рынка: система может использоваться для анализа рыночных тенденций и выявления факторов, влияющих на цены на нефть. • Планирование: Система может использоваться для планирования в компаниях, чья деятельность зависит от цен на нефть. Ограничения: • Система не может гарантировать 100% точность прогнозов. • Система не может предсказать внезапные геополитические события. • Система требует постоянного мониторинга и обновления.</p>
YandexGPT от Яндекса	<p>1. Ограниченность данных:</p> <p>Исторических данных: модель может быть ограничена объёмом и качеством доступных исторических данных о котировках нефти и новостного фона. Если данные неполные или содержат пропуски, это может снизить точность прогнозов.</p> <p>Новостных данных: качество и релевантность новостных данных также влияют на точность модели. Не все новости могут быть учтены или правильно интерпретированы моделью.</p> <p>2. Сложность учёта всех факторов:</p> <p>Нефтяной рынок зависит от множества факторов, включая политические события, экономические показатели, технологические изменения и стихийные бедствия. Модель может не учитывать все возможные факторы, что ограничивает её точность.</p> <p>3. Неопределённость и волатильность рынка:</p>

	<p>Нефтяной рынок известен своей волатильностью и непредсказуемостью. Даже самые сложные модели не могут гарантировать точных прогнозов из-за неопределённости и резких колебаний цен.</p> <p>4. Зависимость от качества новостной модели: Точность прогнозирования котировок зависит от качества модели, генерирующей новостной фон. Если новостная модель неточна или её результаты интерпретируются неправильно, это повлияет на точность прогнозирования котировок.</p> <p>5. Временные рамки прогнозирования: Модели могут иметь ограничения по временным рамкам прогнозирования. Краткосрочные прогнозы могут быть более точными, чем долгосрочные, из-за увеличения неопределённости в будущем.</p> <p>6. Адаптация к изменениям рынка: Модель требует регулярного обновления и адаптации к изменениям на рынке. Если модель не обновляется своевременно, её прогнозы могут стать менее точными.</p> <p>7. Интерпретируемость результатов: Результаты прогнозирования могут быть сложными для интерпретации, особенно если модель использует сложные алгоритмы машинного обучения. Это может затруднить понимание причинно-следственных связей между новостями и котировками.</p> <p>8. Технические ограничения: Система прогнозирования может иметь технические ограничения, связанные с вычислительными ресурсами, скоростью обработки данных и другими факторами. Это может повлиять на скорость и эффективность прогнозирования.</p> <p>9. Правовые и регуляторные ограничения: Использование данных и моделей для прогнозирования котировок может быть ограничено правовыми и регуляторными требованиями в разных странах. Необходимо учитывать эти ограничения при разработке и использовании системы прогнозирования.</p>
DeepSeek	<p>1. Ограничения, связанные с данными</p> <p>1.1. Качество и доступность новостных данных</p> <ul style="list-style-type: none"> • Запаздывание данных: Новости могут поступать с задержкой (особенно из официальных источников), что снижает оперативность прогноза. • Шум и субъективность: Не все новости одинаково влияют на рынок; требуется фильтрация релевантных событий. • Охват источников: Если система анализирует только англоязычные СМИ, она может пропускать важные новости на других языках (арабский, русский – ключевые для ОПЕК+). <p>1.2. Ограничения временных рядов</p> <ul style="list-style-type: none"> • Нестационарность: Цены на нефть подвержены резким изменениям из-за геополитики, что усложняет прогнозирование. • Низкая частота данных: Если модель обучается на дневных данных, она не сможет предсказывать внутридневные колебания. <p>2. Ограничения модели</p> <p>2.1. Зависимость от NLP-модели</p> <ul style="list-style-type: none"> • Точность сентимент-анализа:

	<ul style="list-style-type: none"> ○ Финансовые новости требуют специализированных моделей (FinBERT), иначе тональность может интерпретироваться неверно. ○ Сатира, двусмысленность и нейтральные формулировки могут искажать прогноз. • Тематическая кластеризация: <ul style="list-style-type: none"> ○ Если модель неправильно определяет ключевые темы (например, путает "санкции против Ирана" и "санкции против России"), это приведёт к ошибочным предсказаниям. <p>2.2. Проблемы мультимодального обучения</p> <ul style="list-style-type: none"> • Слабая интерпретируемость: <ul style="list-style-type: none"> ○ Нейросетевые модели (LSTM, трансформеры) сложно объяснить – непонятно, как именно новость повлияла на прогноз. • Переобучение: <ul style="list-style-type: none"> ○ Если NLP-модель выделяет слишком много шумовых фичей, это ухудшает обобщающую способность. <p>3. Рыночные и экономические ограничения</p> <p>3.1. Непредсказуемые события (Чёрные лебеди)</p> <ul style="list-style-type: none"> • Геополитические кризисы (войны, теракты) и природные катастрофы (ураганы в Мексиканском заливе) резко меняют рынок, но их нельзя предсказать на основе прошлых данных. • Решения ОПЕК+ часто принимаются закрыто и не всегда отражаются в новостях заранее. <p>3.2. Макроэкономические факторы</p> <ul style="list-style-type: none"> • Корреляция с другими рынками: <ul style="list-style-type: none"> ○ Курс доллара, инфляция, ставки ФРС влияют на нефть, но их сложно учесть в модели. • Спрос и предложение: <ul style="list-style-type: none"> ○ Данные по запасам нефти (EIA) публикуются с задержкой, что снижает точность краткосрочных прогнозов. <p>4. Технические и эксплуатационные ограничения</p> <p>4.1. Латентность системы</p> <ul style="list-style-type: none"> • Если NLP-модель работает медленно (например, BERT), это делает прогноз непригодным для высокочастотного трейдинга. <p>4.2. Масштабируемость</p> <ul style="list-style-type: none"> • Обработка новостей в реальном времени требует мощных вычислительных ресурсов (GPU/TPU). <p>4.3. Юридические ограничения</p> <ul style="list-style-type: none"> • Использование некоторых новостных API (Bloomberg, Reuters) требует лицензий. <p>5. Область применения системы</p> <p>✓ Подходит для:</p> <ul style="list-style-type: none"> ✓ Среднесрочного прогнозирования (дни/недели) – когда влияние новостей успевают проявиться. ✓ Фундаментального анализа – оценки общего тренда с учётом макроэкономики. ✓ Риск-менеджмента – выявления потенциально волатильных периодов. <p>✗ Не подходит для:</p> <ul style="list-style-type: none"> ✗ Высокочастотного трейдинга (HFT) – из-за задержек в обработке новостей.
--	---

	<p>❌ Прогнозирования экстремальных событий – войн, пандемий, внезапных решений ОРЕС+.</p> <p>❌ Полностью автоматизированной торговли – требует ручной проверки аномалий.</p>
--	--

В таблице 8 представлены результаты применения генеративных помощников для формирования списка литературы в ВКРМ.

Постановка задачи: «Сформируй список литературы (источников) для дипломной работы на тему "Программная система прогнозирования котировок на нефтяном рынке"».

Таблица 8 – Генеративный помощник для формирования списка литературы ВКРМ

Наименование	Результат
ChatGPT от OpenAI	<p>Книги 1. Котляр, А. В. "Экономика и управление в нефтяной отрасли". – М.: Инфра-М, 2018. 2. Бутенко, В. В. "Финансовый анализ и прогнозирование". – М.: Юрайт, 2019. 3. Горбунов, И. А. "Модели и методы прогнозирования в экономике". – М.: Экономика, 2020. 4. Шмидт, А. "Анализ и прогнозирование цен на нефть: методы и модели". – СПб.: Питер, 2021. 5. Калинина, Е. В. "Прогнозирование финансовых рынков: теория и практика". – М.: КноРус, 2020.</p> <p>Научные статьи 6. Klein, A., K. G. "Forecasting Oil Prices: A Comparison of Time Series Models". *Energy Economics*, 2019. 7. Alvarez, R., M. C. "The Impact of News on Oil Prices: An Empirical Analysis". *Journal of Commodity Markets*, 2021. 8. Mao, X., Zhang, Y. "A Novel Hybrid Model for Oil Price Forecasting Based on Machine Learning Techniques". *Energy Reports*, 2020. 9. Sadorsky, P. "Oil Price Forecasting: A Review of the Literature". *Energy Economics*, 2020. 10. Baffes, J., K. M. "The Role of Speculation in Oil Price Dynamics". *The World Bank Economic Review*, 2019.</p> <p>Отчеты и исследования 11. International Energy Agency (IEA). "World Energy Outlook 2022". [Ссылка на отчет] 12. U.S. Energy Information Administration (EIA). "Annual Energy Outlook 2023". [Ссылка на отчет] 13. BP Statistical Review of World Energy 2023. [Ссылка на отчет]</p> <p>Диссертации и магистерские работы 14. Петрова, Н. В. "Методы прогнозирования цен на нефть: анализ и применение". Магистерская работа, МГУ, 2022. 15. Сидоров, А. И. "Использование машинного обучения для прогнозирования котировок на нефтяном рынке". Диссертация на соискание степени кандидата экономических наук, ВШЭ, 2021.</p> <p>Онлайн-ресурсы 16. Investopedia. "How to Forecast Oil Prices: A Guide for Investors". [Ссылка на статью] 17. Trading Economics. "Oil Prices Forecast". [Ссылка на ресурс]</p>
Gemini от Google	<p>Книги и учебные пособия: • Цветков, В. А. (2018). «Мировой рынок нефти». Москва: Ленанд. • Джон Халл. (2018). «Опционы, фьючерсы и другие производные финансовые инструменты». Вильямс. • Мертон, Р. К. (2015). «Непрерывные финансовые модели». Вильямс. • Шарп, У. Ф., Александер, Г. Дж., Бэйли, Дж. В. (2017). «Инвестиции». Инфра-М. • Ширяев, А. Н. (2004). «Вероятность». МЦНМО. • Количественное финансирование: Теория и практика. (2019). Wiley. Статьи и научные публикации: • Чулок, А. А. (2020). «Прогноз развития мирового рынка нефти до 2040 года». Форсайт, 14(1), 16–</p>

	<p>35. • Гребенников, В. Г., & Ермаков, С. А. (2019). «Применение нейронных сетей для прогнозирования цен на нефть». Вестник Санкт-Петербургского университета. Экономика, 35(1), 118–137. • Иванов, А. П., & Петров, С. В. (2018). «Анализ факторов, влияющих на цены на нефть». Экономика и математические методы, 54(4), 45–62. • Салин, В. Н., & Шпаковская, Е. П. (2017). «Моделирование и прогнозирование цен на нефть с использованием временных рядов». Финансы и статистика, 23(2), 78–95. • Хуан, Д., & Чен, К. (2021). «Прогнозирование цен на нефть с использованием глубокого обучения». Энергетическая экономика, 96, 105151. Интернет-ресурсы: • Управление энергетической информации США (EIA): https://www.eia.gov/ • Организация стран-экспортеров нефти (ОПЕК): https://www.opec.org/ • Bloomberg: https://www.bloomberg.com/ • Reuters: https://www.reuters.com/ • Investing.com: https://www.investing.com/ Программное обеспечение: • MATLAB • Python (с библиотеками pandas, scikit-learn, TensorFlow, Keras) • R</p> <p>Дополнительные источники: • Отчеты международных энергетических агентств • Аналитические обзоры инвестиционных банков и финансовых компаний • Статьи в специализированных журналах по энергетике и финансам • Материалы конференций по финансовому моделированию и прогнозированию</p> <p>Советы по поиску и использованию литературы: • Начните с поиска ключевых слов в научных базах данных, таких как Web of Science, Scopus и Google Scholar. • Используйте библиотечные ресурсы вашего университета. • Обратитесь к преподавателю за консультацией по выбору литературы. • Обязательно ссылайтесь на все источники, которые вы используете в своей работе.</p>
<p>YandexGPT от Яндекса</p>	<ol style="list-style-type: none"> Книги и учебники по машинному обучению и анализу данных: Хайкин С. «Нейронные сети: полный курс». Джеймс Г., Уиттон Д., Хастингс Т., Тибширани Р. «Введение в статистическое обучение с примерами на языке R». Bishop C. M. «Pattern Recognition and Machine Learning». Научные статьи и публикации по прогнозированию цен на нефть: Clements M. P., Hendry D. F. «Forecasting Economic Time Series». Kilian L. «Not All Oil Price Shocks Are Alike: Disentangling Demand and Supply Shocks in the Crude Oil Market». Hamilton J. D. «Historical Oil Shocks». Исследования по использованию новостного фона в прогнозировании финансовых рынков: Tetlock P. C. «Giving Content to Investor Sentiment: The Role of Media in the Stock Market». Da Z., Engelberg J., Gao P. «In Search of Attention». Материалы и отчёты международных организаций и аналитических агентств: International Energy Agency (IEA) — отчёты и прогнозы по мировому рынку нефти. Organization of the Petroleum Exporting Countries (OPEC) — отчёты о состоянии рынка нефти. U.S. Energy Information Administration (EIA) — данные и прогнозы по энергетическому сектору. Статьи и обзоры по экономическим и политическим факторам, влияющим на цены на нефть: Financial Times, The Wall Street Journal, Bloomberg — аналитические статьи и обзоры рынка нефти.

	<p>6. Ресурсы по программированию и разработке программных систем: Макконнелл С. «Код. Полное руководство по написанию качественного кода». Мартин Р. «Чистый код. Создание, анализ и рефакторинг».</p> <p>7. Нормативно-правовые акты и документы, регулирующие рынок нефти и финансовые рынки: Закон РФ «О нефти и газе» и другие законодательные акты, регулирующие нефтяной сектор. Документы и регламенты, регулирующие деятельность на финансовых рынках в различных странах.</p> <p>8. Электронные базы данных и научные журналы: JSTOR, ScienceDirect, SpringerLink — для поиска научных статей и публикаций по теме. Web of Science, Scopus — для анализа цитируемости и поиска релевантных источников.</p>
DeepSeek	<p>1. Книги и учебные пособия</p> <ol style="list-style-type: none"> 1. Хайкин, С. <i>Нейронные сети: полный курс</i> – М.: Вильямс, 2016. (Теоретические основы ML и временных рядов) 2. Боровиков, В.П. <i>Прогнозирование в системе STATISTICA</i> – М.: Финансы и статистика, 2013. (Методы анализа временных рядов) 3. Гудфеллоу, Я., Бенджио, И., Курвилль, А. <i>Глубокое обучение</i> – М.: ДМК Пресс, 2022. (LSTM, Transformers, мультимодальные модели) 4. Чжоу, Ч. <i>Машинное обучение для анализа временных рядов</i> – СПб.: Питер, 2021. (SARIMAX, Prophet, Feature Engineering) <p>2. Научные статьи и публикации</p> <ol style="list-style-type: none"> 5. Yu, L., Wang, S., Lai, K.K. <i>Forecasting crude oil price with an EMD-based neural network ensemble learning paradigm</i> // Energy Economics, 2008. (Гибридные модели для прогнозирования цен на нефть) 6. Weng, B., Ahmed, M.A., Megahed, F.M. <i>Stock Market One-Day Ahead Movement Prediction Using Disparate Data Sources</i> // Expert Systems with Applications, 2017. (Использование новостей в прогнозировании финансовых рынков) 7. Ke, Z.T., Kelly, B.T., Xiu, D. <i>Predicting Returns with Text Data</i> // NBER Working Paper, 2019. (NLP для финансовых прогнозов) 8. Lim, B., Arik, S.O., Loeff, N., Pfister, T. <i>Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting</i> // arXiv, 2021. (TFT для прогнозирования с экзогенными факторами) <p>3. Отчёты и аналитика рынка</p> <ol style="list-style-type: none"> 9. U.S. Energy Information Administration (EIA) <i>Short-Term Energy Outlook</i> – 2023. (Официальные данные по спросу/предложению нефти) 10. OPEC Monthly Oil Market Report – 2023. (Анализ факторов влияния на цены) 11. International Energy Agency (IEA) <i>Oil Market Report</i> – 2023. (Макроэкономические тренды) <p>4. Техническая документация и руководства</p> <ol style="list-style-type: none"> 12. Документация TensorFlow/PyTorch – Режим доступа: www.tensorflow.org, pytorch.org (Реализация нейросетевых моделей)

	<p>13. Hugging Face Transformers – Режим доступа: huggingface.co (<i>FinBERT, DistilBERT для NLP</i>)</p> <p>14. Yahoo Finance API, Alpha Vantage – Режим доступа: finance.yahoo.com, www.alphavantage.co (<i>Исторические данные по котировкам</i>)</p> <p>5. Публикации по NLP и сентимент-анализу</p> <p>15. Devlin, J. et al. <i>BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding</i> // NAACL, 2019.</p> <p>16. Malik, M.I. <i>Financial Sentiment Analysis Model for Oil Price Prediction</i> // IEEE Access, 2021. (<i>Специализированные модели для нефтяного рынка</i>)</p> <p>6. Правовые и этические аспекты</p> <p>17. MiFID II (Markets in Financial Instruments Directive) – Режим доступа: www.esma.europa.eu (<i>Регулирование алгоритмического трейдинга в ЕС</i>)</p> <p>18. SEC Guidelines on AI in Trading – Режим доступа: www.sec.gov (<i>Использование ML в финансовых прогнозах</i>)</p>
	<p>Ключевые направления для поиска дополнительных источников:</p> <ul style="list-style-type: none"> • Конференции: KDD, NeurIPS, ICML (публикации по ML для финансов). • Журналы: <i>Energy Economics</i>, <i>Journal of Forecasting</i>, <i>Computational Economics</i>. • GitHub: Open-source проекты по прогнозированию нефтяных цен (например, репозитории с TFT, LSTM).

По результатам таблиц 6, 7, 8 была сформирована сравнительная характеристика алгоритмов в таблице 9.

Таблица 9 – Сравнительная характеристика

Конструкторская часть				
Критерий	ChatGPT	Gemini	YandexGPT	DeepSeek
Полнота описания	Детализированный пошаговый процесс, но без углубления в специфику NLP.	Акцент на предобработку данных и выбор алгоритмов, но меньше деталей по интеграции новостей.	Упор на технические индикаторы и NLP, но менее структурировано.	Наиболее полное описание, включая мультимодальные модели и TFT.
Глубина анализа NLP	Общие методы (TF-IDF, BERT), но без специализированных решений	Указаны анализ тональности и Word2Vec, но нет примеров	Включены стемминг и кластеризация тем, но без ссылок на	Подробно разобраны FinBERT, VADER, тематическая

	(например, FinBERT).	для нефтяного рынка.	актуальные исследования.	кластеризация (LDA, NMF).
Интеграция новостей	Конкатенация признаков, но нет альтернативных подходов (например, мультимодальность).	Предложены векторизация и экзогенные факторы, но без примеров реализации.	Событийные метки и частота упоминаний, но нет связи с архитектурой модели.	Мультимодальные модели (LSTM + BERT), событийные триггеры, SHAP для анализа.
Практические советы	Общие рекомендации по обновлению данных.	Указание на необходимость фильтрации шума в соцсетях.	Отсутствуют.	Реальные кейсы (например, обработка новостей ОПЕК+).
Технологическая часть				
Критерий	ChatGPT	Gemini	YandexGPT	DeepSeek
Охват ограничений	Широкий, но общий (качество данных, задержки, регуляторные аспекты).	Упор на качество новостей и адаптацию к рынку, но без примеров.	Перечислены технические ограничения, но нет связи с рисками.	Систематизировано: данные, модель, рынок, эксплуатация.
Анализ «чёрных лебедей»	Упомянуты, но без методов минимизации их влияния.	Подчёркнута непредсказуемость геополитики, но нет решений.	Общие слова о волатильности.	Конкретные примеры (войны, решения ОПЕК+) и способы учёта (лаг-фичи).
Рекомендации по применению	Области применения перечислены, но без деталей.	Чёткое разделение на трейдинг, риск-менеджмент и планирование.	Нет.	Разграничение: среднесрочные прогнозы ✓, HFT ✗.
Интерпретируемость	Отмечена сложность интерпретации нейросетей.	Затронуты проблемы переобучения и нестационарности.	Указана сложность анализа влияния новостей.	Предложены SHAP/LIME для анализа решений.
Список литературы				
Критерий	ChatGPT	Gemini	YandexGPT	DeepSeek
Актуальность	Преобладают источники до 2021 года, мало специализированных статей.	Есть свежие публикации (2020–2023), но недостаточно по ML.	Устаревшие книги (например, Bishop, 2006), мало нефтяной специфики.	Актуальные статьи (2021–2023), отчёты ОПЕК/EIA, документация по ML.

Структурированность	Разделы есть, но внутри хаотично (например, диссертации рядом с онлайн-ресурсами).	Чёткое деление на книги, статьи, ПО, но нет привязки к главам ВКР.	Сгруппировано по темам, но без пояснений, как использовать.	Логичные разделы (книги, статьи, отчёты, NLP), + ключевые направления.
Узкоспециализированные источники	Нет.	Указаны MATLAB и Python, но без ссылок на нефтяные кейсы.	Общие учебники по ML, но не по прогнозированию цен.	Финансовый NLP (FinBERT), TFT, специализированные базы данных.
Практическая ценность	Есть ссылки на EIA и Bloomberg, но без пояснений.	Даны советы по поиску литературы, но нет примеров применения.	Нет.	Ссылки на GitHub-репозитории и конференции (NeurIPS, KDD).

Таким образом, на основе представленной сравнительной характеристики полученных аннотаций для магистерской диссертации было определено, что DeepSeek является наиболее пригодным генеративным помощником для решения задач конструкторской и технологической частей, а также формирования списка литературы.

В результате выполнения лабораторной работы №1 было изучено применение на практике генеративных помощников для ответов на вопросы по ВКРМ в целом, формирования аннотации и введения ВКРМ, для решения задач конструкторской и технологической частей, а также формирования списка литературы.

Лабораторные работы №2-4

Ход работы

Выполнение данных лабораторных работ производится единым пайплайном в рамках решения поставленных задач:

Задание 1.

Датасет: Необходимо загрузить датасет, расположенный по адресу: <https://archive.ics.uci.edu/ml/datasets/iris>

1. Постройте таблицу pairplot для визуального анализ данных,
2. Постройте LogisticRegression для бинарной классификации элементов датасета, на примере данных
3. Покажите переобучение модели линейной регрессии и предложите способ борьбы с ним
4. Обогадите датасет сгенерированными вами данными. Опишите какой способ вы применяли и приложите код генерации данных.
5. Поэкспериментируйте с возможными AutoML моделями для усовершенствования эталонного решения.

Задание 2.

Датасет: Необходимо загрузить датасет, расположенный по адресу: <https://github.com/AI-is-out-there/data2lab.git> (модуль 2)

1. Визуализируйте с помощью boxplot загруженные данные, опишите ваши мысли по данным.
2. Проанализируйте данные и проведите их предобработку (очистку, обогащение).
3. Сформируйте очищенный датасет, состоящий из числовых параметров ЭКГ и классификационного признака Healthy_Status, заново построить boxplot и сравните результаты с первоначальным.
4. Проведите оценку взаимосвязи данных датасета с помощью функции heatmap.
5. На основе таргета (целевого признака) Healthy_Status построить корреляционную матрицу с помощью pairplot.
6. Провести разведочный анализ данных с помощью метода PCA и с помощью TSNE.
7. Сделать вывод о структуре признаков данных.

8. Сделайте вывод о том какое из Auto ML решений с этой задачей справилось лучше.

Задание 3.

Датасет: Необходимо загрузить датасет, расположенный по адресу: <https://github.com/AI-is-out-there/data2lab.git> (модуль 3)

1. Сформировать обучающую выборку из загруженного датасета, состоящую из столбцов: ['Count_subj', 'rr_interval', 'p_end', 'qrs_onset', 'qrs_end', 'p_axis', 'qrs_axis', 't_axis', 'Healthy_Status']. Датасет состоит из числовых параметров ЭКГ и классификационного признака Healthy_Status.
2. Используя выбранные ранее решения AutoML, найдите наиболее эффективное, используя такие метрика качества как: матрица ошибок (confusion matrix) и F1-метрика для оценки обученного классификатора по признаку Healthy_Status на основе данных параметров ЭКГ.
3. Усовершенствуйте решения за счет предобработки исходных данных
4. Проверьте свои модели на тестовой и валидационной выборке.

Задание 4.

Датасет: Необходимо загрузить датасет, расположенный по адресу: <https://zenodo.org/records/2547147#.Y7eU5uxBwII>

1. Выберите любой файл в формате EDF.
2. По аннотации из датасета определите, где в записи ЭЭГ указан приступ. Постройте график временной зависимости ЭЭГ в момент приступа (выбирайте диапазон времени и масштаб так, чтобы было наглядно).
3. Сделайте усреднение всех каналов ЭЭГ в один (нужно сложить все каналы и разделить на их количество).

4. Удалите из сигнала все частоты выше 60 Гц. Для этого преобразованного сигнала: – постройте спектрограмму сигнала; – постройте вейвлет-преобразование (скейлограмму).

Задание 5.

Датасет: Необходимо загрузить датасет, расположенный по адресу: <https://physionet.org/content/chbmit/1.0.0/chb08/#files-panel>

1. Сделайте скрипт для формирования обучающей выборки из базы данных ЭЭГ с указанными приступами
2. Сформируйте не менее 50 записей с приступами и не менее 50 записей ЭЭГ без приступов. Записи нужно «обрезать» по времени и построить для них вейвлет изображение (можно разбить на диапазоны ритмов мозга, можно не разбивать). В итоге должно быть 100 «картинок».
3. Полученные изображения сложите в отдельные папки (место хранения выбирайте на свое усмотрение). Папки нужны для того, чтобы разделить изображения для записей с приступом и для записей без приступа.
4. Спроектируйте сверточную нейронную сеть для обучения классификации ЭЭГ на те данные, которые имеют и не имеют приступ.
5. Сохраните модель обученной сети и проверьте ее работу.
6. Попробуйте проделать действия с использованием AutoML и сравните работу.

Задание 6.

Датасет: Необходимо перейти на страницу датасета, расположенного по адресу: <https://www.physionet.org/content/mitdb/1.0.0/>.

ссылке: <https://archive.physionet.org/physiobank/database/html/mitdbdir/mitdbdir.htm>

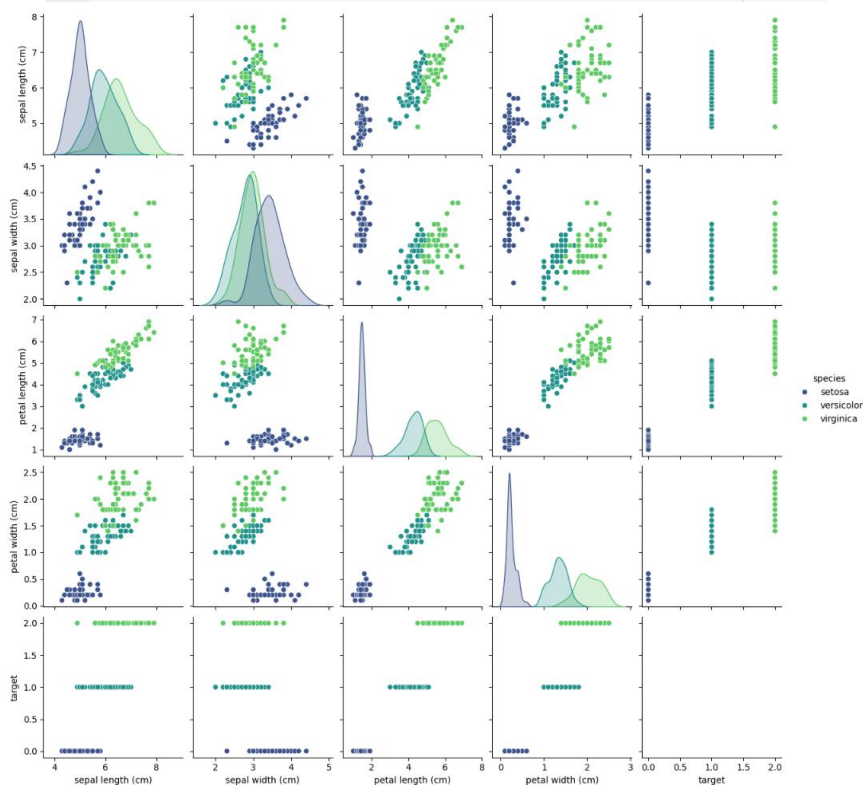
1. Сформируйте тренировочный датасет.
2. Постройте нейронную сеть с эффектом памяти (LSTM) и примените механизм внимания,
3. Обучите ее классифицировать классы по заданной ЭКГ. Постройте матрицу ошибок (confusion matrix).
4. Проведите эксперименты по заданию на AutoML решении. Сравните результаты.

Выполнение задания №1

✓ Pairplot

```
iris = datasets.load_iris()
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
iris_df['target'] = iris.target
iris_df['species'] = iris_df['target'].map({0: 'setosa', 1: 'versicolor', 2: 'virginica'})

# Pairplot для визуального анализа
sns.pairplot(iris_df, hue='species', palette='viridis')
plt.show()
```



Бинарная классификация LogReg

```
[ ] # Бинарная классификация: versicolor vs не-versicolor
X = iris.data
y = (iris.target == 1).astype(int) # 1 для versicolor, 0 для остальных

# Разделение на train/test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Стандартизация данных
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Создание и обучение модели
log_reg = LogisticRegression(random_state=42)
log_reg.fit(X_train_scaled, y_train)

# Кросс-валидация
cv_scores = cross_val_score(log_reg, X_train_scaled, y_train, cv=5)
print(f"Cross-validation scores: {cv_scores}")
print(f"Mean CV accuracy: {cv_scores.mean():.4f}")

# Оценка на тестовых данных
y_pred = log_reg.predict(X_test_scaled)
print(f"Test accuracy: {accuracy_score(y_test, y_pred):.4f}")
print(classification_report(y_test, y_pred))
```

```
Cross-validation scores: [0.625      0.75      0.79166667 0.66666667 0.70833333]
Mean CV accuracy: 0.7083
Test accuracy: 0.8000
```

	precision	recall	f1-score	support
0	0.83	0.90	0.86	21
1	0.71	0.56	0.62	9
accuracy			0.80	30
macro avg	0.77	0.73	0.74	30
weighted avg	0.79	0.80	0.79	30

Переобучение

```
[ ] # Создадим переобученную модель (без регуляризации)
overfit_model = LogisticRegression(penalty=None, max_iter=1000)
overfit_model.fit(X_train_scaled, y_train)

# Оценка переобучения
train_acc = overfit_model.score(X_train_scaled, y_train)
test_acc = overfit_model.score(X_test_scaled, y_test)
print(f"Overfit model - Train accuracy: {train_acc:.4f}, Test accuracy: {test_acc:.4f}")

# Регуляризация L2 с кросс-валидацией
reg_model = LogisticRegressionCV(Cs=10, cv=5, penalty='l2', max_iter=1000)
reg_model.fit(X_train_scaled, y_train)
print(f"Regularized model - Train accuracy: {reg_model.score(X_train_scaled, y_train):.4f}")
print(f"Regularized model - Test accuracy: {reg_model.score(X_test_scaled, y_test):.4f}")

# Метод Leave One Out
loo = LeaveOneOut()
loo_scores = cross_val_score(log_reg, X, y, cv=loo)
print(f"LOO mean accuracy: {loo_scores.mean():.4f}")
```

```
Overfit model - Train accuracy: 0.7083, Test accuracy: 0.8333
Regularized model - Train accuracy: 0.7000
Regularized model - Test accuracy: 0.6667
LOO mean accuracy: 0.7133
```

Видно, что для уменьшения переобучения хорошо подходит регуляризация и кросс-валидация, в частности LOO

Генерация дополнительных данных

Для этого был выбран SMOTE, который создаёт синтетические примеры по следующему алгоритму:

1. Выбирается пример миноритарного класса.
2. Находится K-ближайших соседей.
3. Случайным образом из этих соседей выбирается один.
4. Интерполяция между двумя точками, чтобы создать новый образец где-то между ними в пространстве признаков.

```
[ ] # Создадим переобученную модель (без регуляризации)
overfit_model = LogisticRegression(penalty=None, max_iter=1000)
overfit_model.fit(X_train_scaled, y_train)

# Оценка переобучения
train_acc = overfit_model.score(X_train_scaled, y_train)
test_acc = overfit_model.score(X_test_scaled, y_test)
print(f"Overfit model - Train accuracy: {train_acc:.4f}, Test accuracy: {test_acc:.4f}")

# Регуляризация L2 с кросс-валидацией
reg_model = LogisticRegressionCV(Cs=10, cv=5, penalty='l2', max_iter=1000)
reg_model.fit(X_train_scaled, y_train)
print(f"Regularized model - Train accuracy: {reg_model.score(X_train_scaled, y_train):.4f}")
print(f"Regularized model - Test accuracy: {reg_model.score(X_test_scaled, y_test):.4f}")
```

```
Overfit model - Train accuracy: 0.7688, Test accuracy: 0.7750
Regularized model - Train accuracy: 0.7688
Regularized model - Test accuracy: 0.7500
```

Проверим метрики для прошлых пунктов "Бинарная классификация" и "Переобучение" с учётом сгенерированных данных

```
[ ] # Разделение на train/test
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)

# Стандартизация данных
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Создание и обучение модели
log_reg = LogisticRegression(random_state=42)
log_reg.fit(X_train_scaled, y_train)

# Кросс-валидация
cv_scores = cross_val_score(log_reg, X_train_scaled, y_train, cv=5)
print(f"Cross-validation scores: {cv_scores}")
print(f"Mean CV accuracy: {cv_scores.mean():.4f}")

# Оценка на тестовых данных
y_pred = log_reg.predict(X_test_scaled)
print(f"Test accuracy: {accuracy_score(y_test, y_pred):.4f}")
print(classification_report(y_test, y_pred))
```

```
Cross-validation scores: [0.71875 0.6875 0.8125 0.78125 0.75 ]
Mean CV accuracy: 0.7500
Test accuracy: 0.7500
precision    recall  f1-score   support

      0       0.61      0.93      0.74      15
      1       0.94      0.64      0.76      25

 accuracy          0.75      40
  macro avg       0.77      0.79      0.75      40
 weighted avg     0.82      0.75      0.75      40
```

Эксперименты с возможными AutoML моделями для усовершенствования эталонного решения

LAMA

	Test accuracy: 1.0000	precision	recall	f1-score	support
0	1.00	1.00	1.00	1.00	15
1	1.00	1.00	1.00	1.00	25
accuracy				1.00	40
macro avg	1.00	1.00	1.00	1.00	40
weighted avg	1.00	1.00	1.00	1.00	40

Final prediction for new objects (level 0) =

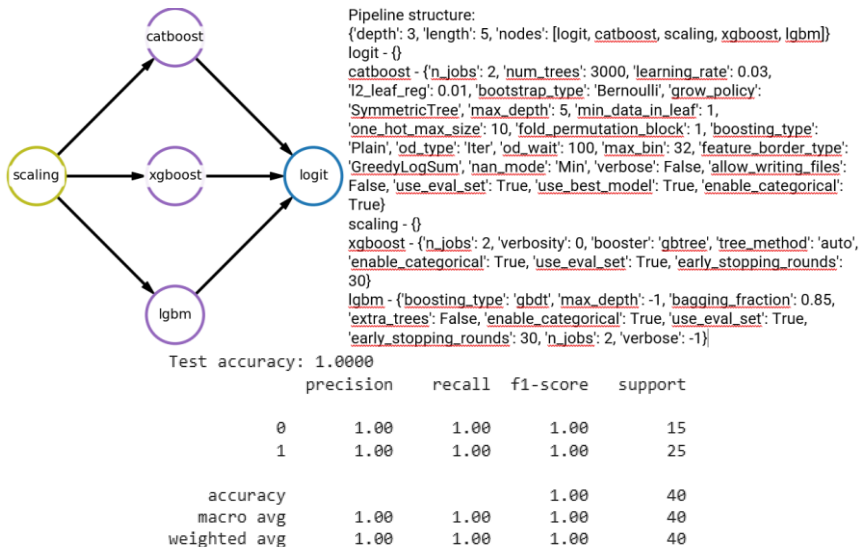
0.06625 * (5 averaged models Lvl_0_Pipe_0_Mod_0_LinearL2) +

0.28198 * (5 averaged models Lvl_0_Pipe_1_Mod_1_Tuned_LightGBM) +

0.10950 * (5 averaged models Lvl_0_Pipe_1_Mod_2_CatBoost) +

0.54227 * (5 averaged models Lvl_0_Pipe_1_Mod_3_Tuned_CatBoost)

FEDOT



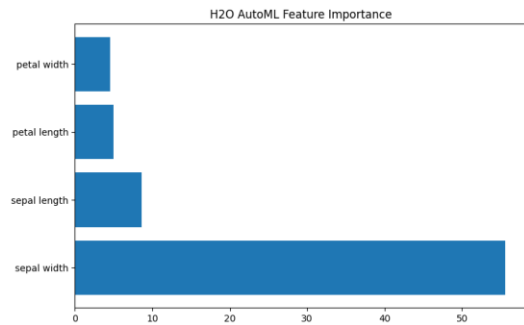
TROT

	Test accuracy: 0.9500	precision	recall	f1-score	support
0	0.93	0.93	0.93	0.93	15
1	0.96	0.96	0.96	0.96	25
accuracy				0.95	40
macro avg	0.95	0.95	0.95	0.95	40
weighted avg	0.95	0.95	0.95	0.95	40

H2O

Таблица лидеров AutoML:

model_id	accuracy	auc	logloss	aucpr
DeepLearning_grid_1_AutoML_2_20250401_160315_model_1	0.96875	0.996392	0.0794147	0.995894
XRT_1_AutoML_2_20250401_160315	0.96875	0.994667	0.0908214	0.994459
GBM_5_AutoML_2_20250401_160315	0.96875	0.995451	0.1103	0.995078
GBM_3_AutoML_2_20250401_160315	0.96875	0.992	0.124033	0.991147
DeepLearning_grid_2_AutoML_2_20250401_160315_model_1	0.975	0.997647	0.0711814	0.997395



Развёртывание MLflow

```
# Запускаем MLflow
```

```
mlflow.start_run()
```

```
# Логируем модель
```

```
mlflow.sklearn.log_model(log_reg, "model")
```

```
# Завершаем запуск
```

```
mlflow.end_run()
```

```
2025/04/02 18:17:59 WARNING mlflow.models.model: Model logged without a signa
<mlflow.models.model.ModelInfo at 0x796536f11510>
```

```
mlflow.active_run().info.run_id
```

```
'eca25340f6054daab6953d743908e72b'
```

```
# Загрузка модели из MLflow
```

```
model_uri = "runs:/eca25340f6054daab6953d743908e72b/model"
```

```
loaded_model = mlflow.pyfunc.load_model(model_uri)
```

```
# Предсказание
```

```
predictions = loaded_model.predict(X_test_scaled)
```

```
print(predictions)
```

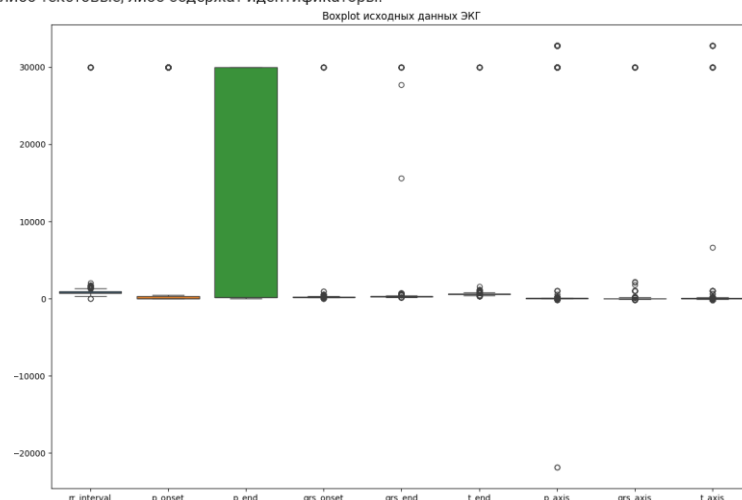
```
[0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 1 0 1 1 1 0 0 0 1 0 1 0 1 0 0 0 0 1 0 0 0 0
0 1 1]
```

```
mlflow.end_run()
```

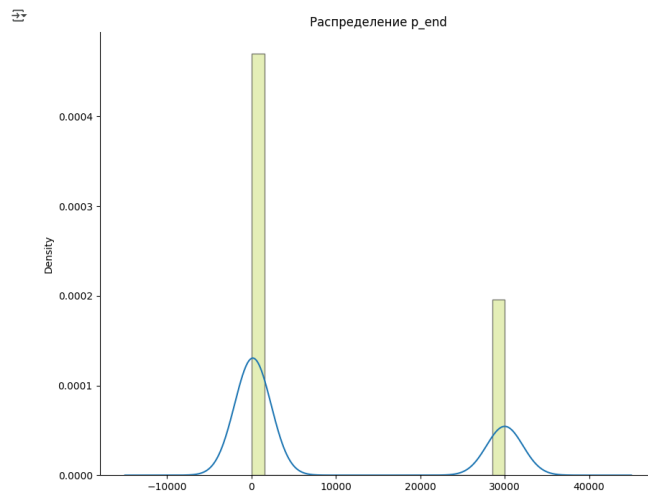
Выполнение задания №2

Boxplot

Исходя из описания столбцов становится очевидным, что BoxPlot имеет смысл строить только для столбцов, начиная с 8го. Остальные либо текстовые, либо содержат идентификаторы.



В данном случае на основе BoxPlot можно заметить выбросные данные, которые выходят за 1,5 межквартильных размаха. При этом виден странный результат для p_end. В этом случае первый и третий квартиль по сути составляют весь ящик. Рассмотрим его подробнее.



Замечаем что box-plot применять некорректно, так как есть несколько пиков.

✓ Проанализируйте данные и проведите их предобработку (очистку, обогащение).

```
df.isnull().sum()
```

subject_id	0	report_0	0	report_7	9692	report_14	9999	report_15	10000	p_end	0
Count_subj	0	report_1	2532	report_8	9886	report_15	10000	report_16	10000	qrs_onset	0
study_id	0	report_2	3532	report_9	9971	report_16	10000	report_17	10000	qrs_end	0
cart_id	0	report_3	5094	report_10	9983	report_17	10000	bandwidth	0	t_end	0
Healthy_Status	0	report_4	7218	report_11	9991	bandwidth	0	filtering	0	p_axis	0
eeg_time	0	report_5	8519	report_12	9999	filtering	0	rr_interval	0	qrs_axis	0
eeg_date	0	report_6	9326	report_13	9998	rr_interval	0	p_onset	0	t_axis	0

Пустоты присутствуют только в текстовых столбцах, которые на данный момент не представляют интереса.

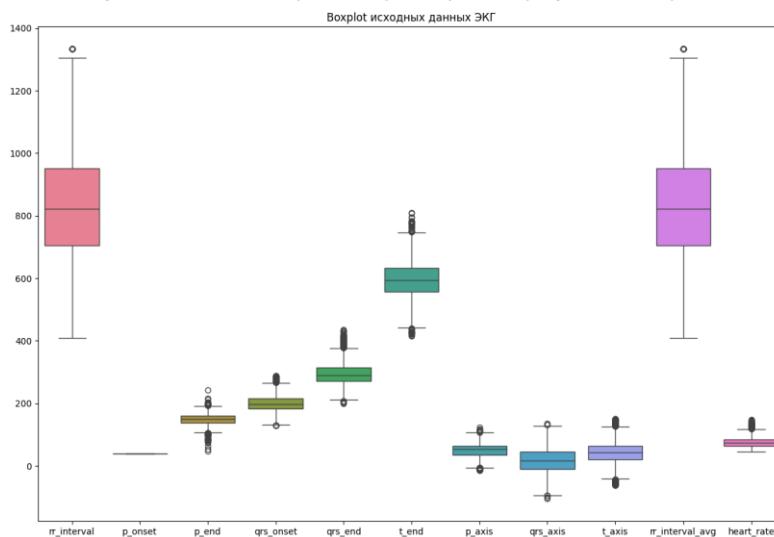
Оставим только нужные столбцы.

Произведём обогащение данных.

Имеем идентификатор субъекта или пациента, посчитаем по нему среднее для rr_interval.

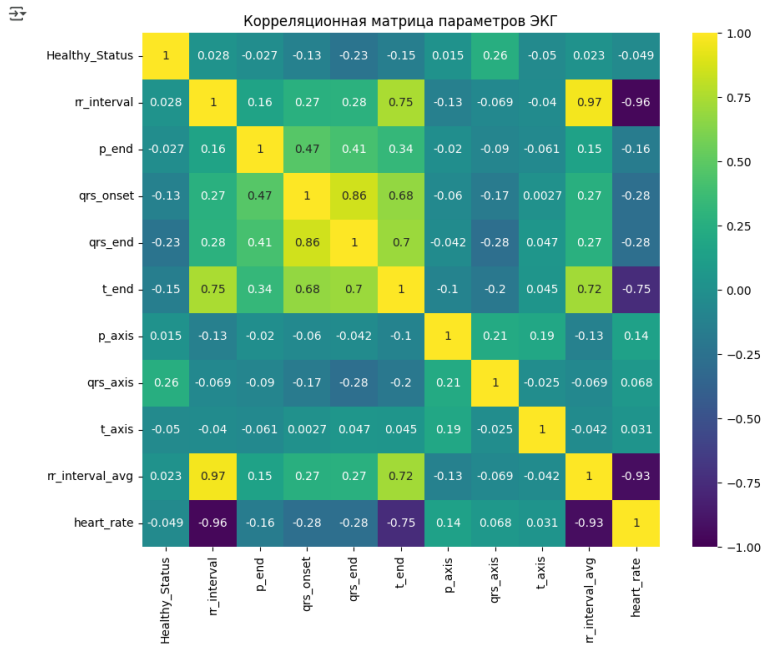
Также добавим частоту сердечных сокращений

✓ Сформируйте очищенный датасет, состоящий из числовых параметров ЭКГ и классификационного признака Healthy_Status, заново построить boxplot и сравните результаты с первоначальным.



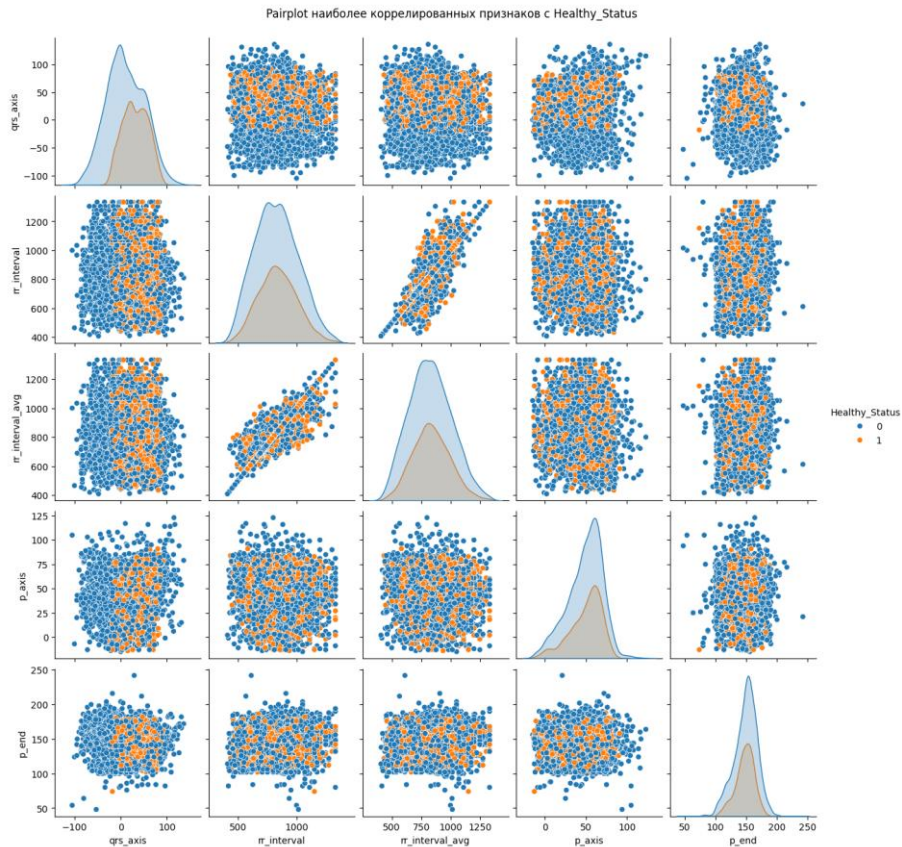
✓ Проведите оценку взаимосвязи данных датасета с помощью функции heatmap.

```
correlation_matrix = df_new.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='viridis', vmin=-1, vmax=1, center=0)
plt.title("Корреляционная матрица параметров ЭКГ")
plt.show()
```



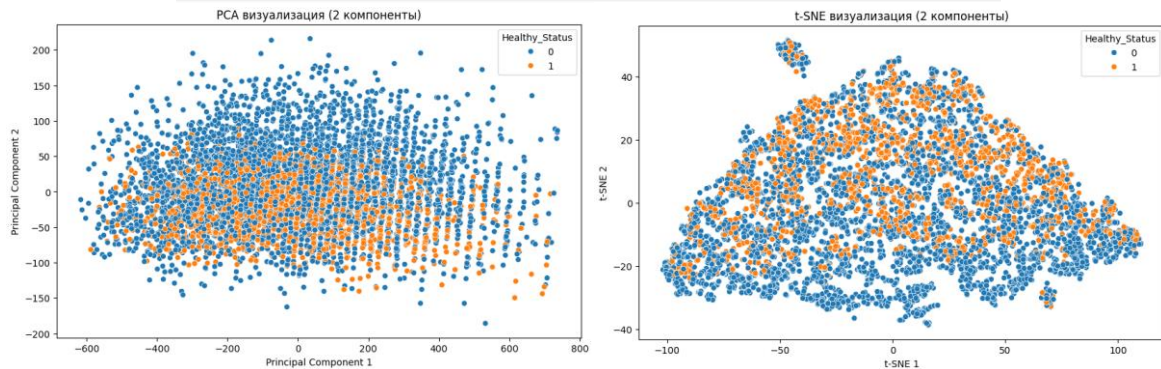
Тепловая карта показывает сильную отрицательную корреляцию между rr_interval, rr_interval_avg и heart_rate, что ожидаемо.

✓ На основе таргета (целевого признака) Healthy_Status построить корреляционную матрицу с помощью pairplot.



На диагонали видно, что плотности распределения для большинства параметров почти совпадают для обоих классов, а scatter plots вне диагонали демонстрируют значительное смешивание точек, где видна ожидаемая нелинейная зависимость.

- ✓ Провести разведочный анализ данных с помощью метода PCA и с помощью TSNE.



- ✓ Сделать вывод о структуре признаков данных

Структура признаков данных, основанная на проведенном анализе, показывает, что параметры ЭКГ имеют сложные взаимосвязи, но слабо разделяют классы по целевому признаку Healthy_Status. Pairplot показал сильное пересечение распределений для Healthy_Status = 0 и Healthy_Status = 1, что указывает на отсутствие явных линейных или нелинейных разделяющих признаков. PCA и t-SNE подтвердили это. Таким образом, структура признаков характеризуется низкой способностью линейно или нелинейно разделять классы, что может потребовать добавления новых признаков или использования сложных моделей для улучшения классификации. При этом выявлено большое количество значений параметров, которые можно отнести к выбросам

Выполнение задания №3

В рамках выполнения задания сначала была сформирована обучающая выборка.

```
X = df[['Count_subj', 'rr_interval', 'p_end', 'qrs_onset', 'qrs_end', 'p_axis', 'qrs_axis', 't_axis']]
y = df['Healthy_Status']

X_train_based, X_test_based, y_train_based, y_test_based = train_test_split(X, y, test_size=0.2, random_state=42)

heart_Train = pd.concat([pd.DataFrame(X_train_based), pd.Series(y_train_based)], axis=1)
heart_Train.columns = ['Count_subj', 'rr_interval', 'p_end', 'qrs_onset', 'qrs_end', 'p_axis', 'qrs_axis', 't_axis', 'Healthy_Status']
heart_Test = pd.concat([pd.DataFrame(X_test_based), pd.Series(y_test_based)], axis=1)
heart_Test.columns = ['Count_subj', 'rr_interval', 'p_end', 'qrs_onset', 'qrs_end', 'p_axis', 'qrs_axis', 't_axis', 'Healthy_Status']
heart_Train
```

	Count_subj	rr_interval	p_end	qrs_onset	qrs_end	p_axis	qrs_axis	t_axis	Healthy_Status
9254	29	631	134	172	256	64	-12	32	0
1561	14	697	138	204	284	25	37	66	1
1670	31	750	140	210	314	40	13	33	1
6087	3	833	158	204	300	11	-33	23	0
6669	78	732	29999	502	638	41	-75	64	0
...
5734	15	666	128	160	240	68	68	141	0
5191	15	652	134	164	242	56	-9	141	0
5390	20	576	29999	198	302	29999	29	-132	0
860	6	714	166	222	304	47	25	55	0
7270	62	697	29999	200	364	29999	-74	116	0

8000 rows x 9 columns

Далее на исходных данных были настроены AutoML решения. После этого была осуществлена предобработка данных (аналогично действиям в Задании №2). В результате были получены метрики для исходных и предобработанных данных.

Модель	Исходные данные					Предобработанные данные				
Lama	precision		recall	f1-score	support	precision		recall	f1-score	support
	0	0.89	0.90	0.90	1588	0	0.88	0.80	0.84	862
	1	0.60	0.57	0.59	412	1	0.82	0.89	0.85	875
				0.83	2000				0.85	1737
		0.75	0.74	0.74	2000		0.85	0.85	0.85	1737
Fedot		0.83	0.83	0.83	2000		0.85	0.85	0.85	1737
	precision		recall	f1-score	support	precision		recall	f1-score	support
	0	0.90	0.89	0.90	1588	0	0.86	0.79	0.83	862
	1	0.60	0.62	0.61	412	1	0.81	0.88	0.84	875
				0.84	2000				0.83	1737
TROT		0.75	0.76	0.75	2000		0.84	0.83	0.83	1737
		0.84	0.84	0.84	2000		0.84	0.83	0.83	1737
	precision		recall	f1-score	support	precision		recall	f1-score	support
	0	0.98	0.80	0.88	1588	0	0.93	0.73	0.82	862
	1	0.55	0.95	0.70	412	1	0.78	0.95	0.86	875
H2O				0.83	2000				0.84	1737
		0.77	0.87	0.79	2000		0.86	0.84	0.84	1737
		0.89	0.83	0.84	2000		0.85	0.84	0.84	1737
	accuracy	auc	logloss	aucpr		accuracy	auc	logloss	aucpr	
	0.852125	0.913709	0.272758	0.628369		0.840795	0.908403	0.360255	0.879234	

На данный момент получилось так, что предобработка не сильно повысила метрики, за исключением того, что за счёт обогащения данных выровнялся баланс классов и выросли метрики по классу 1.

В рамках выполнения задания сначала была сформирована обучающая выборка. Примеры полученных пайплайнов до и после предобработки для различных AutoML:

▼ LAMA

▼ На исходных данных

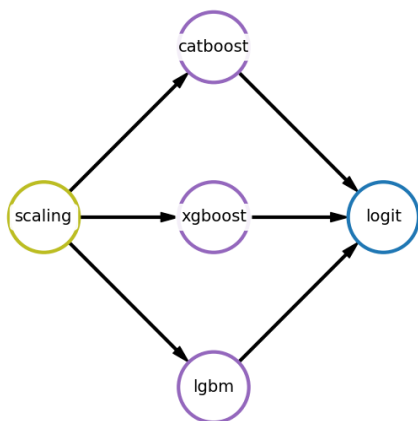
```
Final prediction for new objects (level 0) =
0.43925 * (4 averaged models Lvl_0_Pipe_1_Mod_0_LightGBM) +
0.11824 * (5 averaged models Lvl_0_Pipe_1_Mod_1_Tuned_LightGBM) +
0.24719 * (5 averaged models Lvl_0_Pipe_1_Mod_2_CatBoost) +
0.19533 * (5 averaged models Lvl_0_Pipe_1_Mod_3_Tuned_CatBoost)
```

▼ На предобработанных данных

```
Final prediction for new objects (level 0) =
0.18244 * (5 averaged models Lvl_0_Pipe_0_Mod_0_LinearL2) +
0.11570 * (5 averaged models Lvl_0_Pipe_1_Mod_0_LightGBM) +
0.30959 * (5 averaged models Lvl_0_Pipe_1_Mod_1_Tuned_LightGBM) +
0.26161 * (5 averaged models Lvl_0_Pipe_1_Mod_2_CatBoost) +
0.13066 * (5 averaged models Lvl_0_Pipe_1_Mod_3_Tuned_CatBoost)
```

▼ FEDOT

Для FEDOT изменений нет.



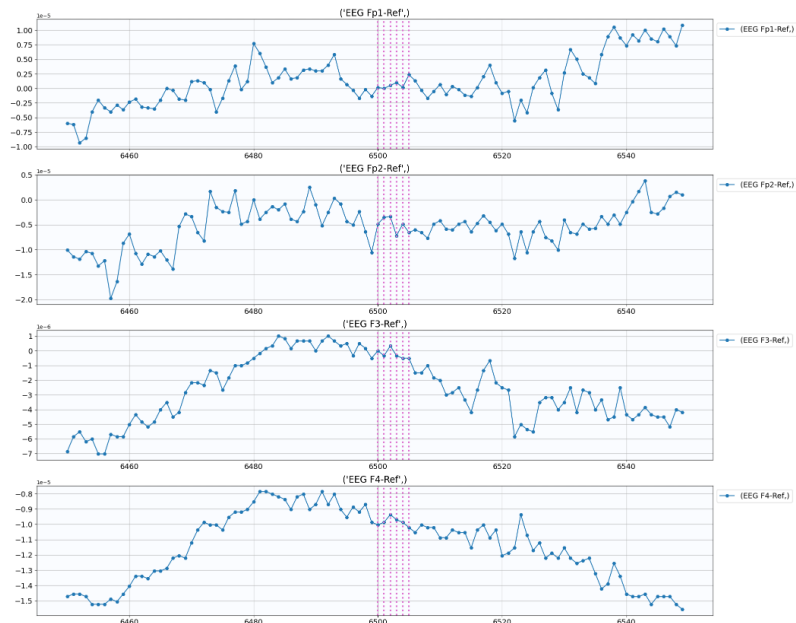
Pipeline structure:
 {'depth': 3, 'length': 5, 'nodes': [logit, catboost, scaling, xgboost, lgbm]}
 logit - {}
 catboost - {'n_jobs': 2, 'num_trees': 3000, 'learning_rate': 0.03, 'l2_leaf_reg': 0.01, 'bootstrap_type': 'Bernoulli', 'grow_policy': 'SymmetricTree', 'max_depth': 5, 'min_data_in_leaf': 1, 'one_hot_max_size': 10, 'fold_permutation_block': 1, 'boosting_type': 'Plain', 'od_type': 'lter', 'od_wait': 100, 'max_bin': 32, 'feature_border_type': 'GreedyLogSum', 'nan_mode': 'Min', 'verbose': False, 'allow_writing_files': False, 'use_eval_set': True, 'use_best_model': True, 'enable_categorical': True}
 scaling - {}
 xgboost - {'n_jobs': 2, 'verbosity': 0, 'booster': 'gbtree', 'tree_method': 'auto', 'enable_categorical': True, 'use_eval_set': True, 'early_stopping_rounds': 30}
 lgbm - {'boosting_type': 'gbdt', 'max_depth': -1, 'bagging_fraction': 0.85, 'extra_trees': False, 'enable_categorical': True, 'use_eval_set': True, 'early_stopping_rounds': 30, 'n_jobs': 2, 'verbose': -1}

Выполнение задания №4

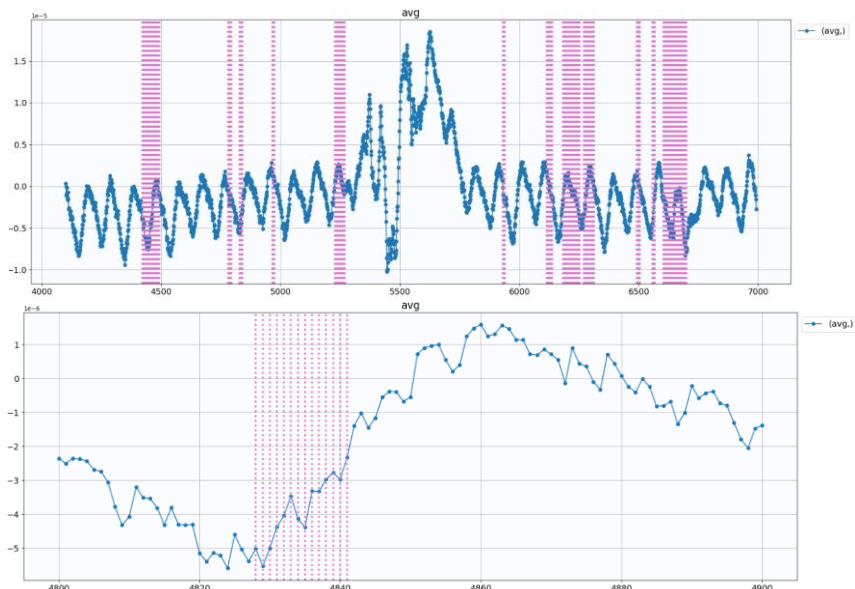
Был выбран любой файл в формате .edf и по аннотации были определены приступы и построены графики временной зависимости ЭЭГ в момент приступа.

	EEG Fp1-Ref	EEG Fp2-Ref	EEG F3-Ref	...	EEG T4-Ref	EEG T6-Ref	EEG P3-Ref	EEG P4-Ref	EEG Pz-Ref	EEG O1-Ref	EEG O2-Ref	EEG EKG	Resp	Effort	3
0	3.386223e-07	-3.306223e-07	5.059334e-07	...	-7.357690e-06	-1.522131e-05	0.000004	-7.357690e-06	6.732445e-07	5.059334e-07	-6.353823e-06	0.000004	-0.000007	0.0	0.0
1	8.405557e-07	3.182912e-06	4.000000e-09	...	-1.421745e-05	-2.726772e-05	0.000006	-1.405014e-05	8.405557e-07	3.386223e-07	-1.304627e-05	0.000014	-0.000010	0.0	0.0
2	-2.672978e-06	5.059334e-07	5.059334e-07	...	-1.103853e-05	-2.325225e-05	0.000004	-1.271165e-05	5.059334e-07	1.713111e-07	-1.137316e-05	0.000018	-0.000013	0.0	0.0
3	-5.851890e-06	-2.338356e-06	5.059334e-07	...	-1.522131e-05	-2.676578e-05	0.000005	-1.388282e-05	1.713111e-07	4.000000e-09	-1.137316e-05	0.000020	-0.000015	0.0	0.0
4	-2.672978e-06	1.175178e-06	3.386223e-07	...	-1.739636e-05	-2.894083e-05	0.000004	-1.338089e-05	6.732445e-07	4.000000e-09	-1.103853e-05	0.000018	-0.000018	0.0	0.0
...
6988	-2.171045e-06	9.875357e-06	-2.840289e-06	...	2.011734e-06	1.677111e-06	0.000003	-8.325557e-07	2.513667e-06	8.704179e-06	2.179045e-06	0.000041	-0.000030	0.0	0.0
6989	-6.521134e-06	4.521401e-06	-3.174912e-06	...	-3.306223e-07	8.405557e-07	0.000002	-1.501800e-06	1.677111e-06	8.202246e-06	2.179045e-06	0.000043	-0.000030	0.0	0.0
6990	-6.521134e-06	5.023334e-06	-3.174912e-06	...	-2.840289e-06	1.677111e-06	0.000002	-1.334489e-06	2.680978e-06	6.529134e-06	3.350223e-06	0.000045	-0.000030	0.0	0.0
6991	-1.659111e-06	3.350223e-06	-2.338356e-06	...	-9.998668e-07	4.000000e-09	0.000002	-1.501800e-06	2.179045e-06	2.513667e-06	1.844422e-06	0.000034	-0.000030	0.0	0.0
6992	-3.174912e-06	-1.053660e-05	-2.505667e-06	...	-1.836422e-06	-8.325557e-07	0.000002	-2.338356e-06	1.175178e-06	1.713111e-07	5.059334e-07	0.000022	-0.000030	0.0	0.0





Сделайте усреднение всех каналов ЭЭГ в один (нужно сложить все каналы и разделить на их количество).

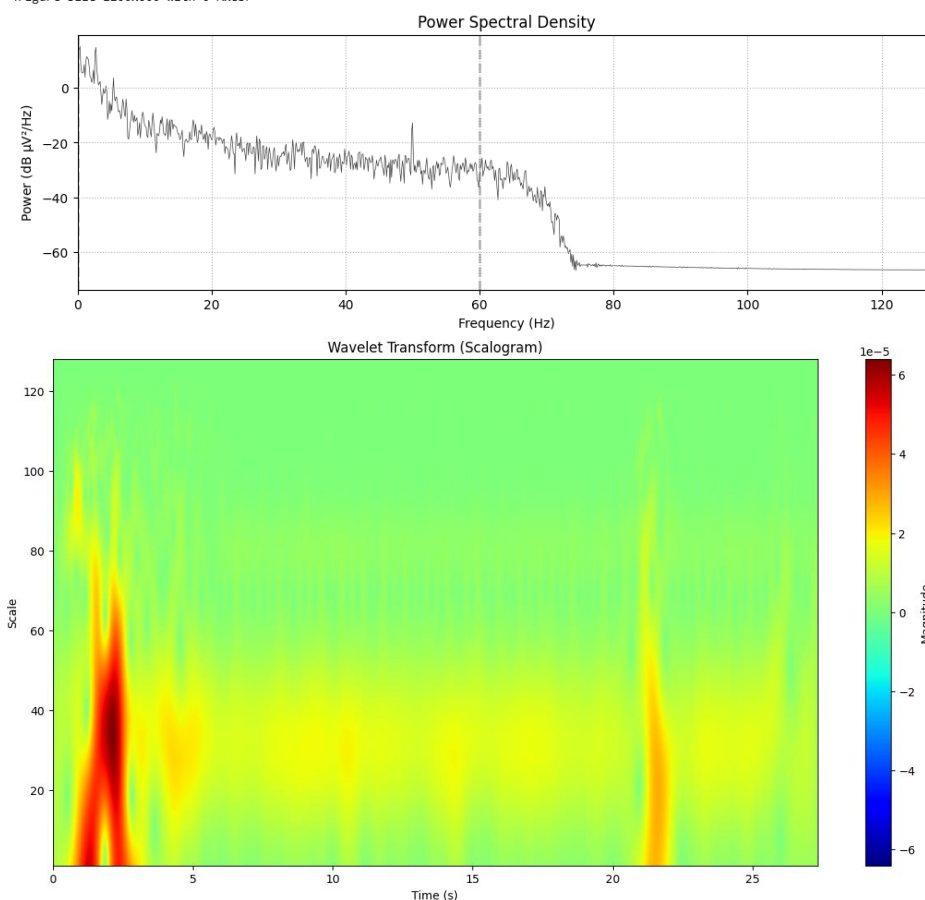


Удалите из сигнала все частоты выше 60 Гц. Для этого преобразованного сигнала: постройте спектрограмму сигнала; постройте вейвлет-преобразование (скейлограмму).

raw_mean		raw_filtered	
General		General	
MNE object type	RawArray	MNE object type	RawArray
Measurement date	Unknown	Measurement date	Unknown
Participant	Unknown	Participant	Unknown
Experimenter	Unknown	Experimenter	Unknown
Acquisition		Acquisition	
Duration	00:00:28 (HH:MM:SS)	Duration	00:00:28 (HH:MM:SS)
Sampling frequency	256.00 Hz	Sampling frequency	256.00 Hz
Time points	6,993	Time points	6,993
Channels		Channels	
EEG	1	EEG	1
Head & sensor digitization	Not available	Head & sensor digitization	Not available
Filters		Filters	
Highpass	0.00 Hz	Highpass	0.00 Hz
Lowpass	128.00 Hz	Lowpass	60.00 Hz

```
plt.figure(figsize=(12, 6))
raw_filtered.plot_psd()
plt.title('Power Spectral Density')
plt.show()
```

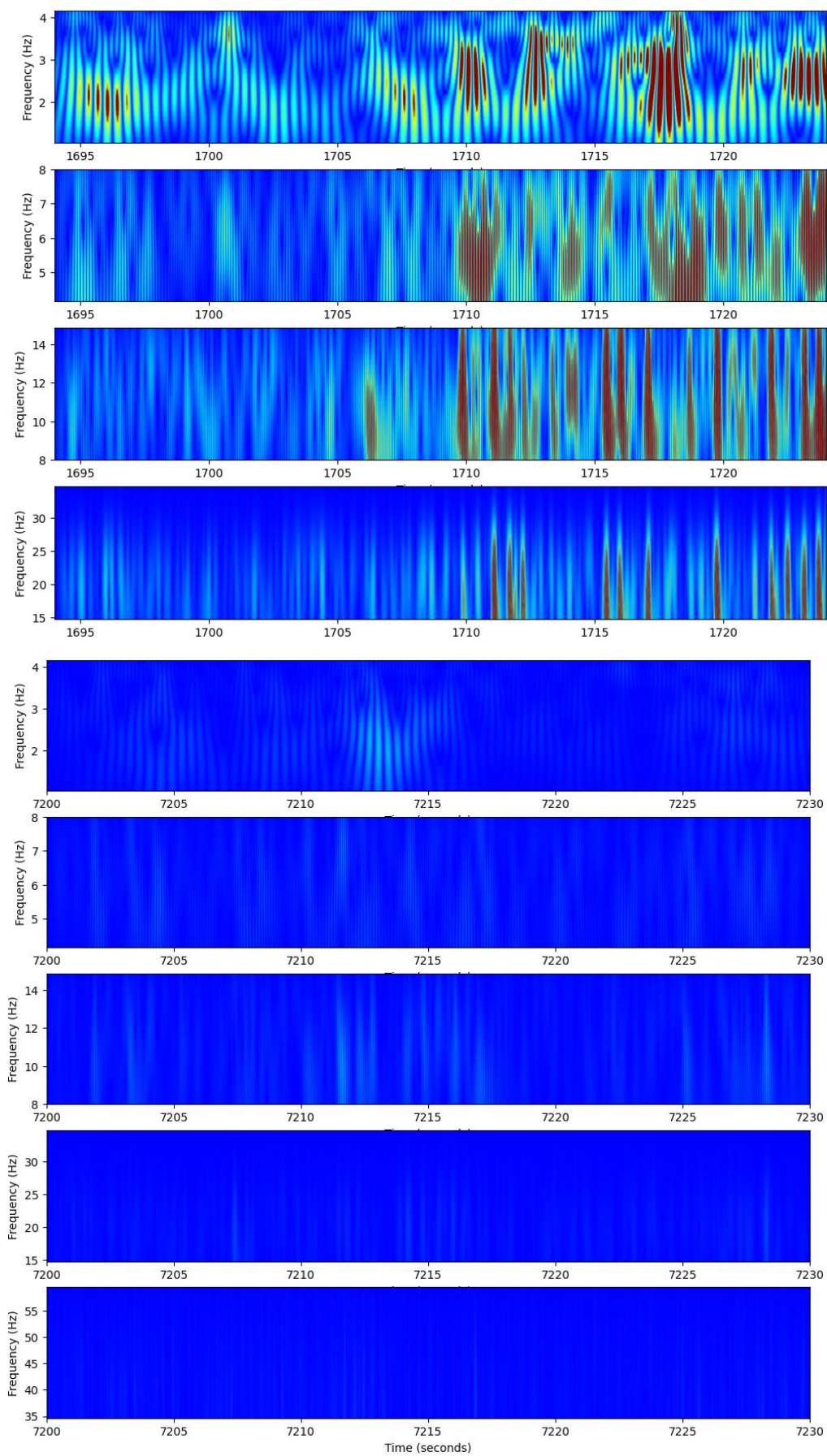
NOTE: plot_psd() is a legacy function. New code should use .compute_psd().plot().
 Effective window size : 8.000 (s)
 Plotting power spectral density (dB=True).
 Need more than one channel to make topography for eeg. Disabling interactivity.
 <Figure size 1200x600 with 0 Axes>



Выполнение задания №5

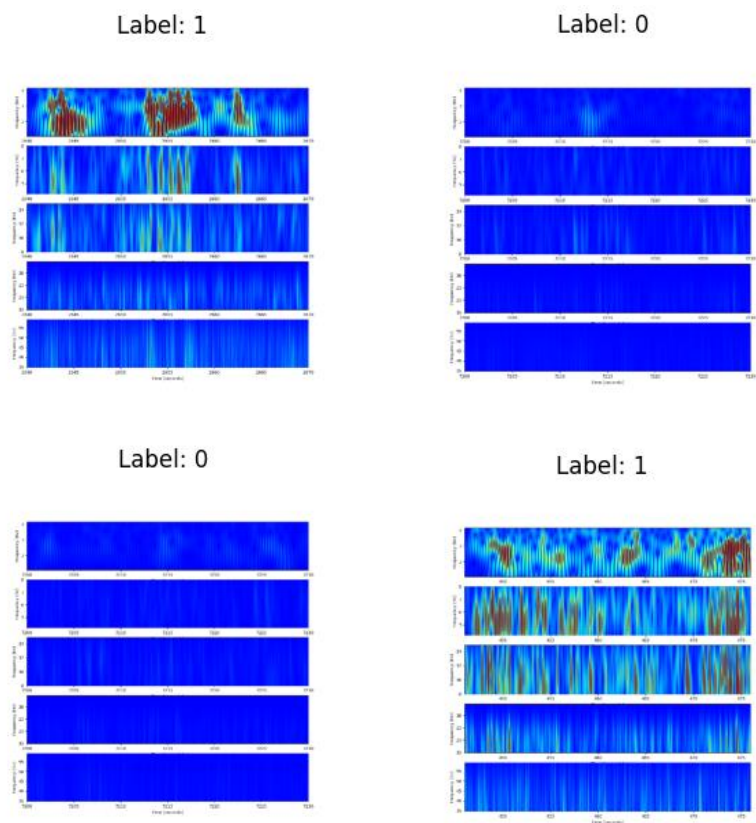
Был сделан скрипт для формирования обучающей выборки из базы данных ЭЭГ. Для записей построены вейвлет изображения, которые были размещены в папки приступов и здоровых сигналов.

Примеры картинок с приступом и без (с разбиением на частоты).



Далее была спроектирована свёрточная нейронная сеть для обучения классификации ЭЭГ по изображениям. Пример разметки выборок:

Sample images from Training Set



```
# Создание модели CNN-LSTM
def create_cnn_lstm_model(input_shape):
    model = Sequential()
    # Сверточные слои
    model.add(Conv2D(32, (3, 3), activation='relu', input_shape=input_shape, padding='same'))
    model.add(BatchNormalization())
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.5))
    model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
    # model.add(BatchNormalization())
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.5))
    model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
    # model.add(BatchNormalization())
    model.add(MaxPooling2D((2, 2)))
    model.add(Dropout(0.5))

    # Преобразование для LSTM
    model.add(Flatten())
    model.add(Dense(64, activation='relu', kernel_regularizer=tf.keras.regularizers.L2(0.05))) # Регуляризация L2
    # model.add(Dropout(0.5))
    model.add(Reshape((1, -1))) # Преобразуем в последовательность длины 1 для LSTM
    model.add(LSTM(64, return_sequences=False)) # Первый LSTM слой
    # model.add(Dropout(0.3))

    # Полносвязный слой
    model.add(Dense(32, activation='relu', kernel_regularizer=tf.keras.regularizers.L2(0.05)))
    # model.add(Dropout(0.5))

    # Выходной слой для двух классов
    model.add(Dense(2, activation='sigmoid')) # Для классификации на 2 класса

    # Компиляция модели
    model.compile(optimizer=Adam(learning_rate=1e-3), loss='categorical_crossentropy', metrics=['accuracy'])

    return model

# Инициализация модели
input_shape = (512, 512, 4) # Изображения 64x64 с 3 каналами (RGB)
model = create_cnn_lstm_model(input_shape)
model.summary()
```

Model: "sequential_8"

Layer (type)	Output Shape	Param #
conv2d_24 (Conv2D)	(None, 512, 512, 32)	1,184
batch_normalization (BatchNormalization)	(None, 512, 512, 32)	128
max_pooling2d_24 (MaxPooling2D)	(None, 256, 256, 32)	0
dropout_8 (Dropout)	(None, 256, 256, 32)	0
conv2d_25 (Conv2D)	(None, 256, 256, 64)	18,496
max_pooling2d_25 (MaxPooling2D)	(None, 128, 128, 64)	0
dropout_9 (Dropout)	(None, 128, 128, 64)	0
conv2d_26 (Conv2D)	(None, 128, 128, 128)	73,856
max_pooling2d_26 (MaxPooling2D)	(None, 64, 64, 128)	0
dropout_10 (Dropout)	(None, 64, 64, 128)	0
flatten_8 (Flatten)	(None, 524288)	0
dense_16 (Dense)	(None, 64)	33,554,496
reshape (Reshape)	(None, 1, 64)	0
lstm (LSTM)	(None, 64)	33,024
dense_17 (Dense)	(None, 32)	2,080
dense_18 (Dense)	(None, 2)	66

Total params: 33,683,330 (128.49 MB)

Trainable params: 33,683,266 (128.49 MB)

Non-trainable params: 64 (256.00 B)

```

# Обучение модели

early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_accuracy', patience=40, restore_best_weights=True)
# Преобразование меток обратно из one-hot encoding в одномерный массив меток

y_train_classes = np.argmax(y_train, axis=1)

# Вычисление весов классов
class_weights = class_weight.compute_class_weight('balanced', classes=np.unique(y_train_classes), y=y_train_classes)

# Преобразование class_weights в словарь, так как model.fit ожидает словарь
class_weights_dict = dict(enumerate(class_weights))
checkpoint = ModelCheckpoint(
    "best_weights.keras", # Имя файла для сохранения весов
    monitor="val_accuracy", # Метрика для отслеживания
    save_best_only=True, # Сохранять только если улучшилась метрика
    mode="max", # "max" для максимизации метрики
    verbose=1
)
# Обучение модели с указанием весов классов
history = model.fit(X_train, y_train, epochs=40, batch_size=8, validation_split=0.2, class_weight=class_weights_dict, callbacks=[early_stopping])
# Оценка модели на тестовых данных
y_pred = model.predict(X_test)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = np.argmax(y_test, axis=1)

# Печать результатов оценки
print("Classification Report:")
print(classification_report(y_true, y_pred_classes))
print("Accuracy:", accuracy_score(y_true, y_pred_classes))

Epoch 35/40
8/8 ----- 67s 8s/step - accuracy: 0.5834 - loss: 1.5187 - val_accuracy: 0.5333 - val_loss: 1.7638
Epoch 36/40
8/8 ----- 63s 8s/step - accuracy: 0.5603 - loss: 1.7466 - val_accuracy: 0.5333 - val_loss: 1.5111
Epoch 37/40
8/8 ----- 83s 8s/step - accuracy: 0.6463 - loss: 1.4420 - val_accuracy: 0.5333 - val_loss: 1.4216
Epoch 38/40
8/8 ----- 64s 8s/step - accuracy: 0.5992 - loss: 1.4193 - val_accuracy: 0.5333 - val_loss: 1.4053
Epoch 39/40
8/8 ----- 109s 12s/step - accuracy: 0.6281 - loss: 1.4170 - val_accuracy: 0.5333 - val_loss: 1.4753
Epoch 40/40
8/8 ----- 115s 8s/step - accuracy: 0.6351 - loss: 1.4343 - val_accuracy: 0.5333 - val_loss: 1.3107
1/1 ----- 4s 4s/step
Classification Report:
              precision    recall  f1-score   support

     0       0.00      0.00      0.00        8
     1       0.56      1.00      0.71       10

   accuracy                0.56        18
  macro avg       0.28      0.50      0.36        18
 weighted avg       0.31      0.56      0.40        18

Accuracy: 0.5555555555555556

```


После этого были использованы методы AutoML для сравнения работы. Так как исходный набор данных (полученный из картинок) содержал 786432 столбца фичей, то было использовано PCA, чтобы преобразовать их до 90.

	0	1	2	3	4	5	6	7	8	9	...	81	82	83	84	85	86	87	88	89	target
0	70.719971	39.692375	40.791294	-14.738014	-2.181283	8.646263	34.251263	-21.809460	-46.578049	39.176125	...	-0.052555	-0.014917	0.027281	0.004025	-0.008008	0.011514	-0.002926	-0.003795	-0.000581	1
1	66.867218	18.436094	-15.077132	20.489367	0.231657	-3.010377	13.567466	-13.297430	-44.010544	1.115244	...	-0.024076	0.022016	0.045896	0.018369	0.002288	-0.023838	0.002292	-0.000293	-0.000581	1
2	65.050179	46.383926	18.852510	11.623337	-6.799337	11.522470	27.308798	-10.727365	-8.498422	36.200592	...	0.014642	-0.012230	0.001413	0.024548	0.009480	0.006958	-0.003419	0.005280	-0.000581	1
3	86.043198	9.273518	0.444468	-5.809177	32.546936	8.030790	-17.890766	-4.318171	-6.924079	-12.754455	...	-0.028029	0.006148	0.000437	-0.011163	0.016150	0.004278	0.000004	0.002110	-0.000581	1
4	87.181236	32.502506	1.167022	-30.831995	49.250156	-33.832443	-44.484138	-25.920523	-4.896444	8.045996	...	0.012627	-0.008194	-0.014006	-0.008638	-0.016251	0.005871	0.004020	0.001592	-0.000581	1
...
85	-66.784340	5.214709	-2.855603	1.987038	-0.698957	-0.973341	-0.364006	1.791479	-0.258640	0.382286	...	0.166641	0.292070	-0.265205	-0.032512	0.010350	-0.285806	-0.046339	-0.039118	-0.000581	0
86	-62.012737	6.877544	-3.078580	2.684959	0.436586	-1.217591	0.191148	2.581852	-1.211016	1.456396	...	-0.064399	-0.734216	-0.637405	0.002067	0.122822	-0.216822	-0.030481	0.005698	-0.000581	0
87	-70.984108	7.310120	-1.025045	0.785908	-0.523524	-0.468760	-0.261305	0.552900	0.084549	0.876656	...	0.766151	0.435746	-0.762139	0.158514	-0.118624	-0.335739	-0.136400	-0.027622	-0.000581	0
88	-77.801285	6.986427	-1.803877	0.208248	-1.093259	0.235858	-1.103233	-0.231203	0.489469	-0.362665	...	1.471835	0.204921	8.535221	-1.156303	-3.477613	0.292273	0.065363	-0.001600	-0.000581	0
89	-71.788750	3.737435	-1.379778	0.673088	-0.397222	0.183003	-1.063405	0.597641	0.567441	0.024468	...	-0.520060	0.268585	-0.105245	-0.290987	-0.163763	-0.306834	0.007272	0.005041	-0.000581	0

> LAMA

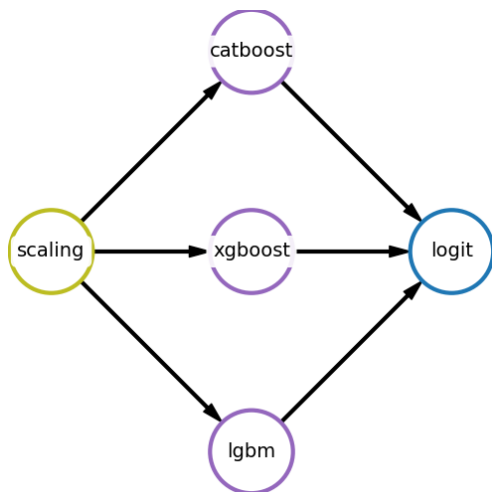
2025-04-09 10:28:38,991 - Model description:

Final prediction for new objects (level 0) =

```
0.20000 * (5 averaged models Lvl_0_Pipe_0_Mod_0_LinearL2) +
0.20000 * (5 averaged models Lvl_0_Pipe_1_Mod_0_LightGBM) +
0.20000 * (5 averaged models Lvl_0_Pipe_1_Mod_1_Tuned_LightGBM) +
0.20000 * (5 averaged models Lvl_0_Pipe_1_Mod_2_CatBoost) +
0.20000 * (5 averaged models Lvl_0_Pipe_1_Mod_3_Tuned_CatBoost)
```

Test accuracy: 0.5000	precision	recall	f1-score	support
0	0.00	0.00	0.00	8
1	0.53	0.90	0.67	10
accuracy			0.50	18
macro avg	0.26	0.45	0.33	18
weighted avg	0.29	0.50	0.37	18

> Fedot



Pipeline structure:

```
{'depth': 3, 'length': 5, 'nodes': [logit, catboost, scaling, xgboost, lgbm]}
logit - {}
```

```
catboost - {'n_jobs': 2, 'num_trees': 3000, 'learning_rate': 0.03,
'12_leaf_reg': 0.01, 'bootstrap_type': 'Bernoulli', 'grow_policy':
'SymmetricTree', 'max_depth': 5, 'min_data_in_leaf': 1,
'one_hot_max_size': 10, 'fold_permutation_block': 1, 'boosting_type':
'Plain', 'od_type': 'Iter', 'od_wait': 100, 'max_bin': 32, 'feature_border_type':
'GreedyLogSum', 'nan_mode': 'Min', 'verbose': False, 'allow_writing_files':
False, 'use_eval_set': True, 'use_best_model': True, 'enable_categorical':
True}
scaling - {}
xgboost - {'n_jobs': 2, 'verbosity': 0, 'booster': 'gbtree', 'tree_method': 'auto',
'enable_categorical': True, 'use_eval_set': True, 'early_stopping_rounds':
30}
lgbm - {'boosting_type': 'gbdt', 'max_depth': -1, 'bagging_fraction': 0.85,
'extra_trees': False, 'enable_categorical': True, 'use_eval_set': True,
'early_stopping_rounds': 30, 'n_jobs': 2, 'verbose': -1}
```

Test accuracy: 1.0000	precision	recall	f1-score	support
0	1.00	1.00	1.00	6
1	1.00	1.00	1.00	12
accuracy			1.00	18
macro avg	1.00	1.00	1.00	18
weighted avg	1.00	1.00	1.00	18

> H2O

Таблица лидеров AutoML:

model_id	accuracy	auc	logloss	aucpr
StackedEnsemble_BestOfFamily_1_AutoML_1_20250409_104352	1	1	0.00814144	1
GBM_4_AutoML_1_20250409_104352	1	1	8.92804e-16	1
StackedEnsemble_AllModels_1_AutoML_1_20250409_104352	1	1	0.00669999	1
XGBoost_3_AutoML_1_20250409_104352	1	1	0.143242	1
XGBoost_grid_1_AutoML_1_20250409_104352_model_4	1	1	0.192527	1

Выполнение задания №6

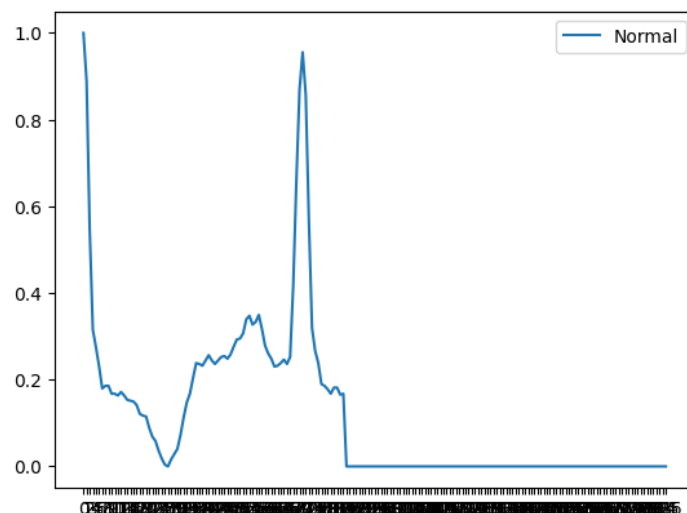
Был сформирован тренировочный датасет и построена нейронная сеть с эффектом памяти (LSTM) и применением механизма внимания.

	0	1	2	3	4	5	6	7	8	9	...	180	181	182	183	184	185	186	class	label	healthy_status
28032	1.000000	0.390071	0.000000	0.102837	0.106383	0.117021	0.106383	0.113475	0.109929	0.109929	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Normal	Normal
58920	0.951219	0.749129	0.101045	0.048780	0.101045	0.104530	0.111498	0.108014	0.125436	0.114983	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Normal	Normal
63172	1.000000	0.784566	0.440514	0.054662	0.028939	0.163987	0.279743	0.324759	0.347267	0.360129	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Normal	Normal
75400	0.866019	0.897087	0.403883	0.000000	0.116505	0.256311	0.258252	0.244660	0.285437	0.271845	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Normal	Normal
23757	0.165888	0.275701	0.425234	0.574766	0.656542	0.785047	0.766355	0.684579	0.742991	0.778037	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Normal	Normal
...
101359	0.816609	0.557093	0.581315	0.602076	0.564014	0.543253	0.494810	0.363322	0.224913	0.086505	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	Fusion of paced and normal	Abnormal
21916	0.673781	0.570122	0.454268	0.330793	0.201220	0.100610	0.032012	0.000000	0.006098	0.059451	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	Fusion of paced and normal	Abnormal
96191	0.882155	0.885522	0.781145	0.562290	0.368687	0.407407	0.417508	0.393939	0.377104	0.356902	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	Fusion of paced and normal	Abnormal
75386	0.759599	0.682805	0.574290	0.464107	0.317195	0.213689	0.121870	0.068447	0.053422	0.091820	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	Fusion of paced and normal	Abnormal
96735	1.000000	0.832536	0.741627	0.741627	0.693780	0.655502	0.607656	0.526316	0.440191	0.277512	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	Fusion of paced and normal	Abnormal

```
# пример как выглядит кардиограмма из датасета
```

```
plt.plot(df_all.iloc[0][:3], label=str(df_all.iloc[0]['label']))
plt.legend()
plt.show()
```

```
INFO:matplotlib.category:Using categorical units to plot a list of strin
2025-04-12 10:41:53,424 - Using categorical units to plot a list of strin
INFO:matplotlib.category:Using categorical units to plot a list of strin
2025-04-12 10:41:53,432 - Using categorical units to plot a list of stri
```



> LSTM

```
# Реализация механизма внимания
class AttentionLayer(tf.keras.layers.Layer):
    def __init__(self, **kwargs):
        super(AttentionLayer, self).__init__(**kwargs)

    def build(self, input_shape):
        self.W = self.add_weight(name='attention_weight',
                                   shape=(input_shape[-1], 1),
                                   initializer='random_normal',
                                   trainable=True)
        self.b = self.add_weight(name='attention_bias',
                                   shape=(input_shape[1], 1),
                                   initializer='zeros',
                                   trainable=True)
        super(AttentionLayer, self).build(input_shape)

    def call(self, x):
        # Вычисление весов внимания
        e = tf.tanh(tf.matmul(x, self.W) + self.b)
        a = tf.nn.softmax(e, axis=1)
        output = x * a
        return tf.reduce_sum(output, axis=1)
```

```
# Создание модели
def create_lstm_attention_model(input_shape, num_classes):
    inputs = Input(shape=input_shape)

    # LSTM слой
    lstm1 = LSTM(64, return_sequences=True)(inputs)
    lstm1 = LayerNormalization()(lstm1)
    lstm1 = Dropout(0.3)(lstm1)

    lstm2 = LSTM(32, return_sequences=True)(lstm1)
    lstm2 = LayerNormalization()(lstm2)

    # Механизм внимания
    attention = AttentionLayer()(lstm2)

    # Полносвязные слои
    dense1 = Dense(32, activation='relu')(attention)
    dense1 = BatchNormalization()(dense1)
    dense1 = Dropout(0.3)(dense1)

    output = Dense(num_classes, activation='softmax')(dense1)

    model = Model(inputs=inputs, outputs=output)
    model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

    return model
```

```

# Создание и обучение модели
input_shape = (X_train.shape[1], X_train.shape[2])
num_classes = len(1e.classes_)

model = create_lstm_attention_model(input_shape, num_classes)
model.summary()

# Коллбэки
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)
model_checkpoint = ModelCheckpoint('best_model.h5', monitor='val_accuracy', save_best_only=True, mode='max')

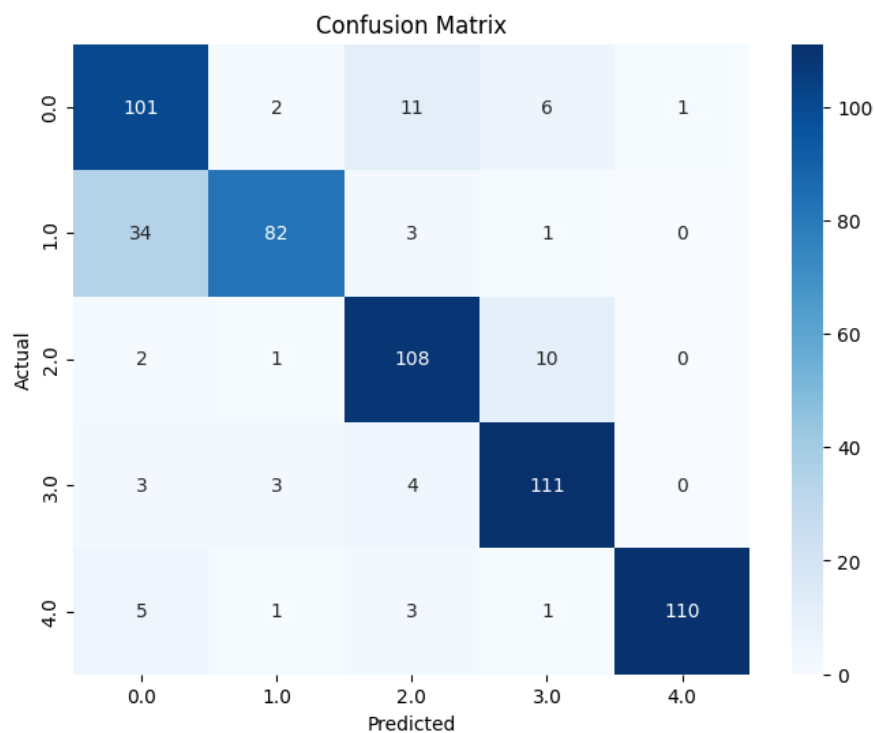
# Обучение
history = model.fit(X_train, y_train,
                    epochs=50,
                    batch_size=32,
                    validation_split=0.2,
                    callbacks=[early_stopping, model_checkpoint])

```

Model: "functional"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 187, 1)	0
lstm (LSTM)	(None, 187, 64)	16,896
layer_normalization (LayerNormalization)	(None, 187, 64)	128
dropout (Dropout)	(None, 187, 64)	0
lstm_1 (LSTM)	(None, 187, 32)	12,416
layer_normalization_1 (LayerNormalization)	(None, 187, 32)	64
attention_layer (AttentionLayer)	(None, 32)	219
dense (Dense)	(None, 32)	1,056
batch_normalization (BatchNormalization)	(None, 32)	128
dropout_1 (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 5)	165

Total params: 31,072 (121.38 KB)
Trainable params: 31,008 (121.12 KB)
Non-trainable params: 64 (256.00 B)



```
# Печать результатов оценки
print("Classification Report:")
print(classification_report(y_test, y_pred_classes))
print("Accuracy:", accuracy_score(y_test, y_pred_classes))
```

```
Classification Report:
              precision    recall  f1-score   support

     0       0.70      0.83      0.76       121
     1       0.92      0.68      0.78       120
     2       0.84      0.89      0.86       121
     3       0.86      0.92      0.89       121
     4       0.99      0.92      0.95       120

 accuracy          0.85          0.85          0.85          603
 macro avg         0.86          0.85          0.85          603
 weighted avg      0.86          0.85          0.85          603
```

Accuracy: 0.8490878938640133

▼ AutoML

> LAMA

2025-04-12 11:21:07,528 - Model description:

Final prediction for new objects (level 0) =

0.62287 * (1 averaged models Lvl_0_Pipe_1_Mod_0_LightGBM) +
0.37713 * (1 averaged models Lvl_0_Pipe_1_Mod_2_CatBoost)

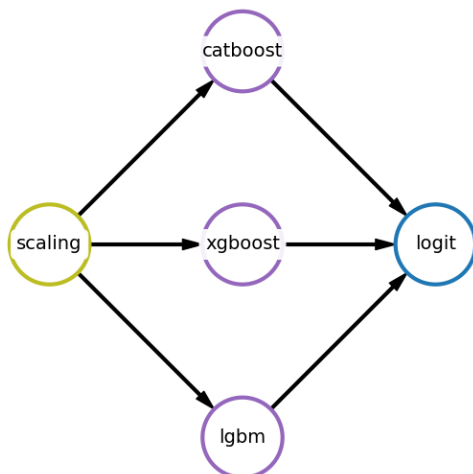
Test accuracy: 0.0697

```
              precision    recall  f1-score   support

     0       0.05      0.19      0.08       121
     1       0.15      0.16      0.15       120
     2       0.00      0.00      0.00       121
     3       0.00      0.00      0.00       121
     4       0.00      0.00      0.00       120

 accuracy          0.07          0.07          0.07          603
 macro avg         0.04          0.07          0.05          603
 weighted avg      0.04          0.07          0.05          603
```

> FEDOT



Pipeline structure:

{'depth': 3, 'length': 5, 'nodes': [logit, catboost, scaling, xgboost, lgbm]}

logit - {}

catboost - {'n_jobs': 2, 'num_trees': 3000, 'learning_rate': 0.03,

'l2_leaf_reg': 0.01, 'bootstrap_type': 'Bernoulli', 'grow_policy':

'SymmetricTree', 'max_depth': 5, 'min_data_in_leaf': 1,

'one_hot_max_size': 10, 'fold_permutation_block': 1, 'boosting_type':

'Plain', 'od_type': 'Iter', 'od_wait': 100, 'max_bin': 32, 'feature_border_type':

'GreedyLogSum', 'nan_mode': 'Min', 'verbose': False, 'allow_writing_files':

False, 'use_eval_set': True, 'use_best_model': True, 'enable_categorical':

True}

scaling - {}

xgboost - {'n_jobs': 2, 'verbosity': 0, 'booster': 'gbtree', 'tree_method': 'auto',

'enable_categorical': True, 'use_eval_set': True, 'early_stopping_rounds':

30}

lgbm - {'boosting_type': 'gbdt', 'max_depth': -1, 'bagging_fraction': 0.85,

'extra_trees': False, 'enable_categorical': True, 'use_eval_set': True,

'early_stopping_rounds': 30, 'n_jobs': 2, 'verbose': -1}

```

Test accuracy: 0.9022
      precision    recall  f1-score   support

0         0.80      0.84      0.82       121
1         0.89      0.85      0.87       120
2         0.93      0.92      0.93       121
3         0.93      0.93      0.93       121
4         0.97      0.97      0.97       120

 accuracy
macro avg      0.90      0.90      0.90       603
weighted avg    0.90      0.90      0.90       603

```

> H2O

Таблица лидеров AutoML:

model_id	accuracy	mean_per_class_error	logloss	rmse	mse
XGBoost_1_AutoML_1_20250412_112450	0.878957	0.121049	0.336895	0.319079	0.101811

[1 row x 6 columns]

> TROT

```

Test accuracy: 0.8922
      precision    recall  f1-score   support

0         0.76      0.88      0.81       121
1         0.94      0.82      0.88       120
2         0.92      0.90      0.91       121
3         0.89      0.89      0.89       121
4         0.99      0.97      0.98       120

 accuracy
macro avg      0.90      0.89      0.89       603
weighted avg    0.90      0.89      0.89       603

```

Вывод: в процессе выполнения лабораторных работ проведено ознакомление с возможностями работы чатов/генеративных помощников на базе LLM для применения в исследовательской деятельности студентов и решения профессиональных задач. Изучены процессы подготовки наборов данных для построения модели машинного обучения. Освоено построение сложного паплайна, включающего в себя возможность проверки модели на не менее чем 3х различных методах машинного обучения, релевантных поставленной задаче. Рассмотрены процессы доработки модели машинного обучения, оценка ее качества и проверка решения задачи прогнозирования