BAUM AI PREDICT

Руководство администратора

Версия 1.0.0

Москва

2023

СОДЕРЖАНИЕ:

1. Общие положения

- 1.1. Основные понятия и определения

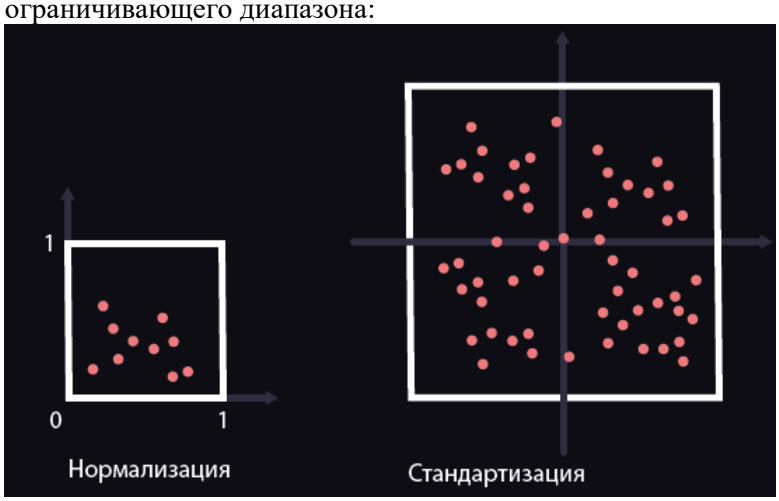
Таблица 1 – Основные термины, используемые в документе, и их определения

Термин	Определение
Apache Kafka	Брокер сообщений, реализующий паттерн Producer-Consumer. Данные из одного и того же топика могут считываться множеством консьюмер-групп одновременно.
Apache Spark	Фреймворк с открытым исходным кодом для реализации распределенной обработки неструктурированных и слабоструктурированных данных.
BPMN	С англ. Business Process Model and Notation. Нотация определяющая способ визуализации процессов в виде диаграмм с определенным набором блоков и взаимосвязей.
Временной ряд	Совокупность наблюдений, собранных за определенный временной интервал. Этот тип данных используется для поиска долгосрочного тренда, прогнозирования будущего и прочих видов анализа. Анализ временных рядов позволяет обнаруживать тенденции и закономерности в исследуемых процессах, строить прогнозы и предсказывать будущие изменения в бизнесе, на производстве, и в других областях.
Выборка	Случайное подмножество генеральной совокупности.
Датасет	С англ. *Data set*, набор данных. Коллекция из логических записей, хранящихся в виде *кортежа*. Набор данных можно сравнить с файлом, но в отличие от файла он является одновременно и каталогом, и файлом файловой системы, и не может содержать в себе другие наборы. Файловая система ориентирована на хранение записей, которые являются неделимыми единицами хранения. Множества записей объединяются в группы, которые и называются наборами данных. Записи в наборах данных используются приложениями, например, как входные данные. Так, записями набора данных могут быть как текстовые данные, так и изображения. К набору данных можно обратиться, указав точное место его хранения, или, если ранее для набора было зазервировано имя в файловой системе, по имени (второй вариант не реализован). Также *датасетом* называются данные, которые пользователь загрузил с локального компьютера, а Система при загрузке выполнила их предварительную обработку. Например, в систему загружается временной ряд в формате csv, и при загрузке он преобразуется в *структурированные данные* или по-другому датасет. Такой датасет пригоден для использования в моделях машинного обучения. В качестве *датасета* в настоящем документе также выступает набор изображений или файлов, которые используются для решения одной задачи.

Дерево решений	Инструмент прогнозного моделирования. Строится с помощью алгоритмического подхода, который разделяет набор данных на основе различных условий. Относится к классу обучения с учителем, используется для задач классификации и регрессии. Цель в том, чтобы создать модель, которая предсказывает значение целевой переменной, изучая правила принятия решений, выведенные из характеристик данных. Алгоритм: 1. Выбрать лучший атрибут, который разделяет наблюдения на группы. 2. Задать соответствующий вопрос. 3. Следовать по путям ответов. 4. Вернуться к шагу 1.
Дисперсия	Мера удаленности того или иного значения выборки от среднего значения. Рассчитывается по формуле: $\sigma^2 = \frac{\Sigma(x_i - \bar{x})^2}{N}$ σ^2 x_i $\bar{x}$ N Стандартное отклонение рассчитывается как квадратный корень из полученной цифры.
Категориальная переменная (качественные данные)	Это данные с ограниченным числом уникальных значений или *категорий* (например, пол, страна проживания, номер группы, категория товаров, и т.п.). Категориальные поля могут быть как текстовыми, так и числовыми, в которых категории закодированы *числовыми кодами* (например, 0=женский, а 1=мужской). Номинальные поля, порядковые поля и флаги являются категориальными полями. -*Набор* (номинальная переменная). Поле, значения которого представляют категории без естественного упорядочивания (например, подразделение компании, в котором работает сотрудник). -*Упорядоченный набор* (порядковая переменная). Поле, значения которого представляют категории с некоторым естественным для них упорядочением (например, оценки, представляющие степень удовлетворенности или уверенности, или баллы, оценивающие предпочтение). -*Флаг*. Поле или переменная с двумя отдельными значениями, например Да и Нет.
Классификация	Задача машинного обучения, которая ставит своей целью назначить метку класса наблюдениям из предметной области. Основные типы классификации: • бинарная классификация; • мультиклассовая классификация; • классификация по нескольким меткам; • несбалансированная классификация.
Кластеризация	Техника обучения без учителя, которая включает в себя группирование или кластеризацию точек данных. Чаще всего она

	используется для сегментации потребителей, выявления мошенничества и классификации документов. Кластеризация (или кластерный анализ) – это задача разбиения множества объектов на группы, называемые кластерами. Главное отличие от классификации состоит в том, что перечень групп четко не задан и определяется в процессе работы алгоритма.
Машинное обучение (ML – Machine learning)	Тренировка математической модели на исторических данных для того, чтобы прогнозировать какое-то событие или явление на новых данных. То есть попытка заставить алгоритмы программ совершать действия на основе предыдущего опыта, а не только на основе имеющихся данных. Для обучения нужны исторические данные (обучающая выборка) и значение целевой переменной (то, что прогнозируем), которое соответствует заданным историческим данным. Модель наблюдает и находит зависимости между данными и целевой переменной. Эти зависимости используются моделью для нового набора данных, чтобы прогнозировать целевую переменную, которая неизвестна. Машинное обучение включает в себя целый набор методов и алгоритмов, которые могут предсказать какой-то результат по входным данным. Алгоритмов машинного обучения большое множество: одни эффективны для решения одного типа задач, вторые – для другого. **Суть технологии машинного обучения** Говоря в общем, машинное обучение – это обучение компьютерной программы или алгоритма постепенному улучшению исполнения поставленной задачи. Машинное обучение обозначает множество математических, статистических и вычислительных методов для разработки алгоритмов, способных решить задачу не прямым способом, а на основе поиска закономерностей в разнообразных входных данных. Решение вычисляется не по четкой формуле, а по установленной зависимости результатов от конкретного набора признаков и их значений. Например, если каждый день в течение недели земля покрыта снегом и температура воздуха существенно ниже нуля, то вероятнее всего, наступила зима. Поэтому машинное обучение применяется для диагностики, прогнозирования, распознавания и принятия решений в различных прикладных сферах: от медицины до банковской деятельности.
Мониторинг	Процесс наблюдения и регистрации данных о каком-либо объекте на неразрывно примыкающих друг к другу интервалах времени, в течение которых значения данных существенно не изменяются .
Мониторинг состояния	Наблюдение за состоянием объекта для определения и предсказания момента перехода в предельное состояние. Результат мониторинга состояния объекта представляет собой совокупность диагнозов составляющих его субъектов, получаемых на неразрывно примыкающих друг к другу интервалах времени, в течение которых состояние объекта существенно не изменяется. Принципиальное отличие от

	мониторинга параметров является наличие интерпретатора измеренных параметров в терминах состояния – экспертной системы поддержки принятия решений о состоянии объекта и дальнейшем управлении.
Наблюдение (строка, запись, точка, сущность)	Ценные данные, собираемые во время исследования или эксперимента. Вместе с масштабом анализа определяет совокупность. *Эмпирические исследования* – практические эксперименты с результатами на основе реального опыта, а не теории или убеждений. Основополагающим принципом *Науки о данных* является приоритет наблюдения над предположением. Типы наблюдений: ● *Числовой*: целые (integer), вещественные (real number), числа с плавающей точкой (float). ● *Булевый* (boolean) – принимает значения 1/0 (да/нет). ● *Категориальный*. Например, жанры кино: комедия, ужасы, мелодрама. ● *Текстовый*. ● *Вектор*.
Нейронная сеть (или Искусственная нейронная сеть)	Представляет собой *математическую модель*, а также её программное или аппаратное воплощение, построенную по принципу организации и функционирования биологических нейронных сетей – сетей нервных клеток живого организма. Нейронные сети решают задачу: по точкам находят функцию. Происходит это путем *минимизации ошибки* – сводится к минимуму «расстояние» между значениями, предсказываемыми нейронной сетью, и значениями, которые наблюдаются. Под *архитектурой нейронной сети* понимается ее устройство – последовательность нейронов и связей между ними.
Нормализация	Техника преобразования значений признака, масштабирующая значения таким образом, что они расположены в диапазоне от 0 до 1.
Обучение с учителем	Контролируемое обучение – метод машинного обучения, при котором модель обучается на размеченных данных. Например, исследовав опухли, установив их размер, плотность и другие метрики, мы передаем эти данные модели с обязательной пометкой, какое наблюдение к какому строению (доброкачественному или злокачественному) относится. Алгоритмы контролируемого обучения подразделяются на следующие модели: классификация, регрессия.
Пайплайн	Последовательность стадий, внутри которых расположены задачи. Расположены они таким образом, что выход каждого элемента является входом следующего.
Признак	Объективная характеристика, характерная черта или свойство, которое может быть определено или измерено. В статистике независимые переменные X используются для предсказания зависимого признака Y
Регрессия (в математической статистике)	Математическое выражение, отражающее связь между зависимой переменной **y** и независимыми переменными **x**. Алгоритмы регрессии используются для контролируемого обучения моделей искусственного интеллекта. Модели обучают прогнозировать числовые значения целевых переменных.

Стандартизация	Техника преобразования значений признака, адаптирующая признаки с разными диапазонами значений к моделям машинного обучения, использующих дистанцию для прогнозирования. Это разновидность нормализации с использованием стандартизированной оценки преобразует значения так, что из каждого наблюдения каждого признака вычитается среднее значение и результат делится на стандартное отклонение этого признака: $x_{i,\text{станд.}} = \frac{x_i - \mu}{\sigma}$ $x_{i,\text{станд.}}$ x_i – исходный элемент, μ – среднее арифметическое, σ – стандартное отклонение. Преобразование необходимо, поскольку признаки датасета могут иметь большие различия между своими диапазонами, и для моделей машинного обучения это спровоцирует искаженное восприятие данных. Стандартизация, в отличие от нормализации, не имеет ограничивающего диапазона: 
Стандартизированная оценка	Метрика, характеризующая удаленность наблюдения от среднего значения совокупности данных.
Стандартное отклонение	Мера разброса в наборе числовых данных. Показывает, насколько далеко от среднего арифметического находятся точки данных. Чем меньше стандартное отклонение, тем более сгруппированы данные вокруг центра (среднего). Чем отклонение больше, тем больше разброс значений.
Тренировочные (обучающие) данные	Часть датасета, обучающая основа модели машинного обучения. Является одной из составляющих разделенного набора данных наряду с *тестовыми* и *валидационными* данными.
Валидационные (тестовые) данные	Часть датасета, основа для проверки работоспособности модели машинного обучения.
Числовая переменная	Numeric variable - переменная, выраженная различными видами чисел.

Целевая (зависимая) переменная	Target variable – признак датасета, который предстоит предсказывать модели машинного обучения. Зависимой ее называют, поскольку в ходе разведочного анализа данных выявляется корреляция между одной или несколькими переменными-предикторами и рассматриваемым целевым признаком.

2. Введение

- 2.1. Область применения

.ПО предиктивного анализа систем хранения данных (СХД) **BAUM AI PREDICT** представляет собой удобный инструмент для системного аналитика или администратора СХД для решения задачи прогнозирования загрузки СХД и превентивного ремонта составных частей СХД в режиме визуального программирования (моделирования).

ПО предназначено для решения задачи прогнозирования загрузки СХД и превентивного ремонта составных частей СХД для снижения рисков недоступности и отказа СХД.

2.2. Уровни подготовки пользователей

В Системе предполагается наличие ролей пользователей – *аналитик*, моделирующий с помощью конструктора искусственного интеллекта работу бизнес-процесса, и *оператор*, обладающий возможностью только просматривать *результаты запуска конструктора*. Платформа позволяет настраивать уровни доступа к разделам системы для разных пользователей, подробней об этом в разделе Администрирование.

3. Регистрация и авторизация пользователя в Системе

Для начала работы с Системой необходимо пройти процедуру **регистрации**, для этого:

1. Перейдите на страницу регистрации:

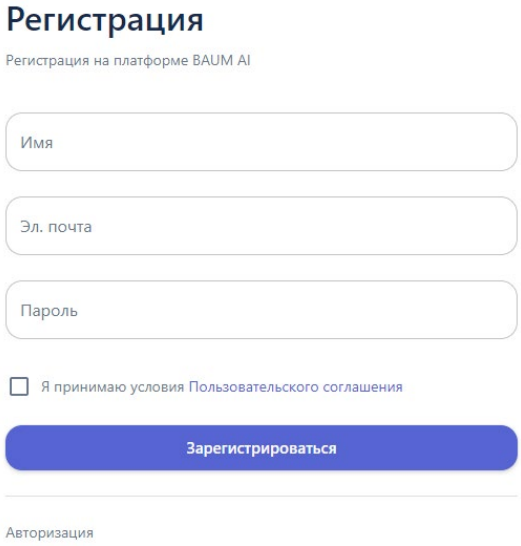


Рисунок 3.1 – Страница регистрации пользователя в Системе

По ссылке «Авторизация» доступен переход на страницу авторизации.

2. Заполните следующие поля:
   - Имя пользователя;
   - Адрес электронной почты;
   - Пароль для входа в Систему.
3. Ознакомьтесь с условиями пользовательского соглашения. Регистрация возможна только при условии их принятия.
4. Нажмите кнопку «Зарегистрироваться».

После этого шага выполняется регистрация пользователя в Системе – в БД **MongoDB** создается новая запись с уникальным идентификатором пользователя.

5. Вы можете сохранить связку логин и пароль для автоматического заполнения при последующей авторизации на текущем устройстве.

В случае, если регистрация была пройдена ранее, при входе в систему осуществляется процедура **авторизации.** Для этого:

1. Перейдите на страницу авторизации:

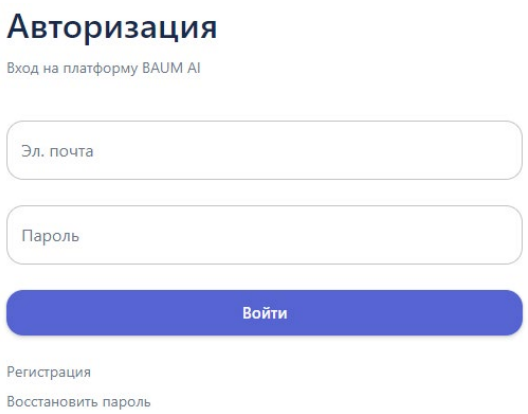


Рисунок 3.2 – Страница авторизации пользователя в Системе

По ссылке «Регистрация» доступен переход на страницу регистрации.

По ссылке «Восстановить пароль» доступен переход на страницу восстановления пароля.

2. Введите следующую информацию:
   - Адрес электронной почты;
   - Пароль для входа в Систему.
3. Нажмите кнопку «Войти».

4. Личный кабинет пользователя

Личный кабинет – это персональная страница на Платформе, доступ к которой есть только у одного пользователя (его владельца).

Как работать с личным кабинетом:

1. Перейдите на страницу личного кабинета одним из способов:
   1.1. На панели инструментов нажмите на кнопку с инициалами/аватаркой пользователя (кнопка БА) –> кнопку «Профиль» (кнопка):

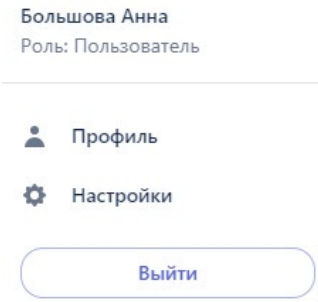


Рисунок 4.1 – Переход в личный кабинет

   1.2. В левом верхнем углу Главного окна в разделе с информацией о пользователе нажмите кнопку с инициалами/аватаркой пользователя:

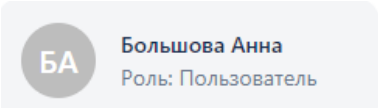


Рисунок 4.2 – Отображение данных пользователя

2. Откроется страница «Профиль» по умолчанию на вкладке «Главная»:

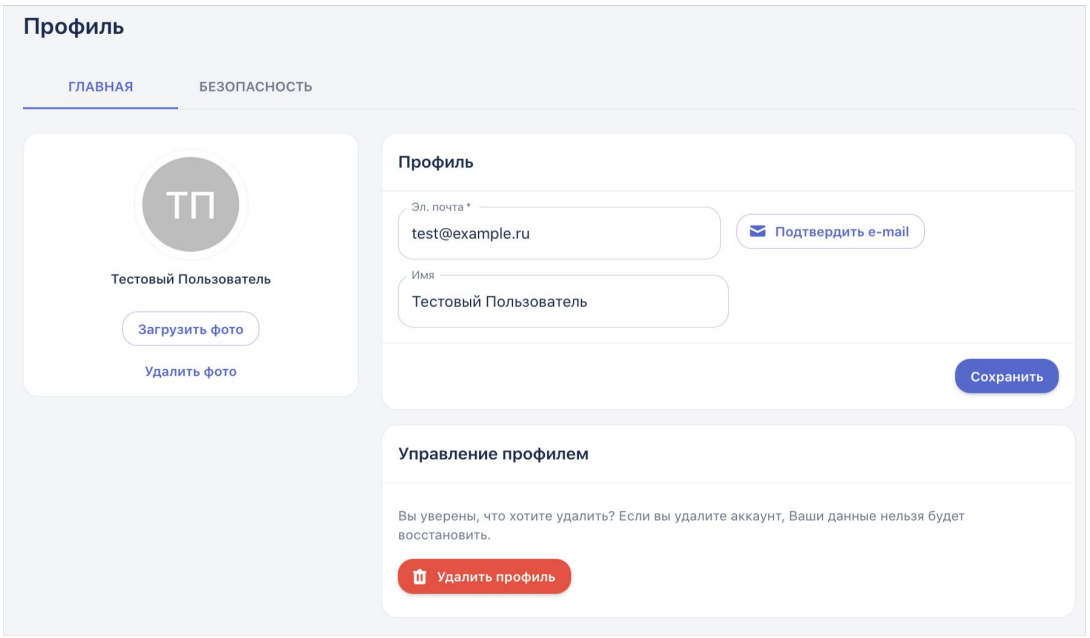


Рисунок 4.3 – Страница «Профиль»

Вкладка «Главная» состоит из:

- аватарки пользователя (по умолчанию это инициалы имени пользователя, загрузка фото не реализована в текущей версии);
- блока для ввода личной информации о пользователе – имени пользователя (логина), адреса электронной почты.
- блока «Управление профилем», в котором реализована возможность удаления своей учетной записи. Перед попыткой выполнить это действие, система выдает предупреждение о последствиях.

Чтобы изменить адрес электронной почты в поле «Электронная почта» укажите новый адрес и нажмите кнопку «Сохранить» (далее сохранение новых настроек в профиле предполагается по умолчанию). Подтверждение e-mail не является обязательным действием при смене адреса.

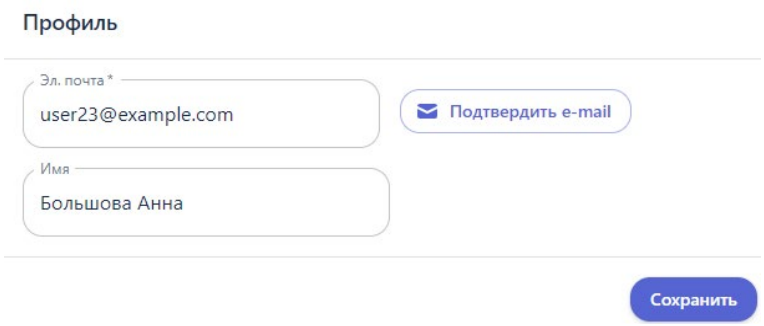


Рисунок 4.4 – Изменение адреса электронной почты в настройках профиля

Чтобы изменить имя пользователя, в поле «Имя» укажите свой новый логин. Тогда следующая авторизация на Платформе будет выполняться уже с новым логином.

5. Интерфейс Платформы

- 5.1. Меню интерфейса

Пункты меню имеют древовидную структуру и представлены в виде вложенных папок:

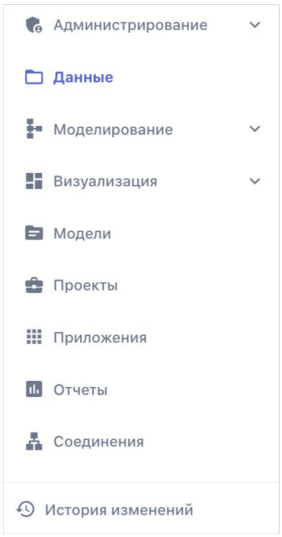


Рисунок 5.1 – Пункты меню программы **BAUM AI PREDICT**

Состав:

Таблица 2 – Описание пунктов меню

	Администрирование	Данный раздел позволяет создавать ролевые модели пользователей - *группы*, согласно которым разделяются уровни доступа к разным модулям системы. Тут же осуществляется управление и назначение ролей всем пользователям системы. В разделе также реализована возможность настройки *каналов уведомлений* (это может быть email или telegram), которые могут быть использованы для получения автоматических оповещений от системы в случае выполнения заданных условий.
	Данные	Данный блок предназначен для загрузки в Систему *входных данных*, это могут быть файлы в различных форматах: ● **Таблицы в форматах csv, txt, xlsx, xls** - табличные данные для создания интеллектуальной базы знаний, в том числе временные ряды. *Временной ряд* – это собранный в разные моменты времени статистический материал о значении каких-либо параметров исследуемого процесса. Возникают временные ряды в результате измерения некоторого показателя. Это могут быть как показатели технических систем, так и показатели природных (погодные условия), социальных, экономических и других систем. Пример временного ряда – показания датчиков на производстве, анализ которых позволяет прогнозировать выход за критические отметки целевых переменных, принять меры и предотвратить возможную поломку оборудования или аварию. Уже загруженные в систему файлы с данными, преобразованные в датасеты, отображаются в формате таблицы (реализовано только для файлов с расширением *.csv*). Эти датасеты используются в качестве входных данных для обучения модели искусственного интеллекта. ● **Изображения в форматах: jpeg, jpg, png.** ● **Видео в форматах: avi, mp4.**

		● **Текстовые файлы в форматах: txt, doc, docx**. Примеры задач для работы с текстом – классификация текста (например, определение авторства), распознавание и оцифровка рукописного текста, чтение и анализ новостного фона, определение тенденций в науке по научным статьям «Скопус», и т.д.
	**Моделирование**	Сердце программы, которое позволяет описывать бизнес-процессы и выполнять целевые действия. ● **Рабочая область**. Для реализации бизнес-процессов заказчика на рабочей области создаются *блок-схемы из элементарных блоков, соединенных стрелками*, которые позволяют выстраивать цепочки, взаимосвязи, условия и т.д. Блок-схема отвечает на вопрос «Что делает процесс». ● **Сохраненные рабочие области**. Рабочие области с ранее созданными блок-схемами. Есть возможность в любое время вернуться и продолжить работу над блок-схемой.
	**Визуализация**	● **Дашборды**. Динамически настраиваемые дашборды. Содержат загруженные и преобразованные массивы данных, представленные в виде таблиц, графиков, гистограмм. ● **Сохранённые дашборды**. Дашборды, созданные пользователем или группой пользователей.
	**Модели**	*Модель на основе машинного обучения* – это абстракция, которая обучена распознаванию определенного типа закономерностей. Хранится в виде файла. Для обучения модели нужны исторические данные (обучающая выборка) и значение целевой переменной (то, что прогнозируем), которое соответствует заданным историческим данным. С помощью алгоритмов машинного обучения модель наблюдает и находит зависимости между данными и целевой переменной. Эти зависимости используются моделью для нового набора данных (тестовой выборки), чтобы прогнозировать целевую переменную, которая неизвестна. Все модели разделяются на обучение с учителем и без учителя. *Обучение с учителем* подразделяется на две подкатегории: регрессия и классификация. В *регрессионных моделях* вывод является непрерывным. Наиболее распространенные типы регрессионных моделей – линейная регрессия, деревья решений, случайный лес, нейронная сеть. В *классификационных моделях* вывод является дискретным. Наиболее распространенные типы классификационных моделей – логистическая регрессия, метод опорных векторов. В отличие от обучения с учителем, *обучение без учителя* используется для того, чтобы сделать выводы из входных данных без отсылок на отмеченные результаты. Два основных метода, используемых в обучении без учителя, включают *кластеризацию* и *снижение размерности*. **Сохранённые модели** – обученные модели, которые используются в моделировании бизнес-процессов с данными в режиме реального времени.
	**Проекты**	Сущность «Проект» объединяет в себе: загруженные в Систему файлы, созданные рабочие области, обученные модели ИИ, визуализацию результатов на дашбордах и сформированные отчеты. Так Система позволяет консолидировать всю информацию по проекту в одном разделе для систематизации и удобного доступа пользователей.
	**Приложения**	Приложение содержит «упакованную» обученную модель ИИ – без необходимости разработки отдельного кода для создания приложения с моделью. Такое приложение можно скачать и развернуть за пределами программы, интегрировать с внешними системами, настроить получение входных данных и выполнить прогнозы.

	Отчеты	Подготовленные по установленной форме отчеты. Могут хранится результаты ИИ исследований, результаты спроектированного бизнес-процесса.
	Соединения	Раздел «Соединения» предназначен для настройки подключения Платформы к внешним источникам данных с целью получения данных из других систем. В этом разделе пользователь может создавать новые и просматривать уже созданные коннекторы. *Коннектор* – это сущность, которая объединяют в себе источник подключения и запрос на получение данных из него.

- 5.2. Окно построения модели ИИ

На примере страницы Платформы, на которой строится конструктор модели искусственного интеллекта, рассматривается, из каких частей состоит интерфейс:

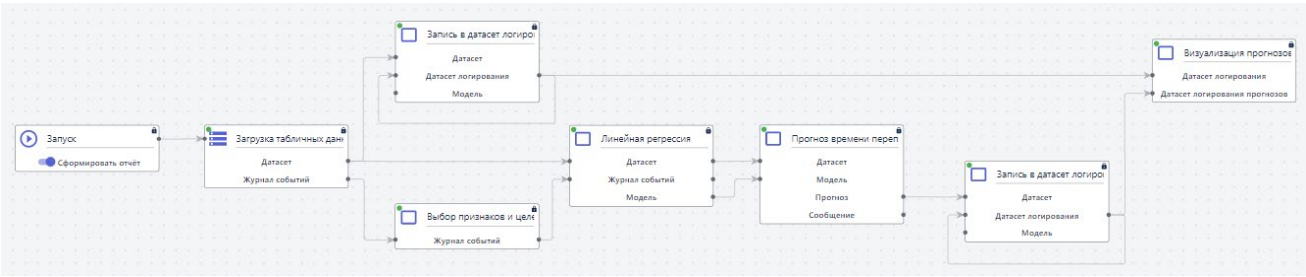
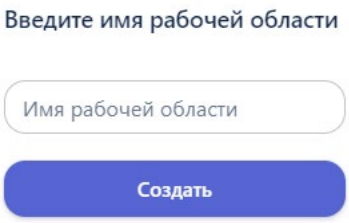
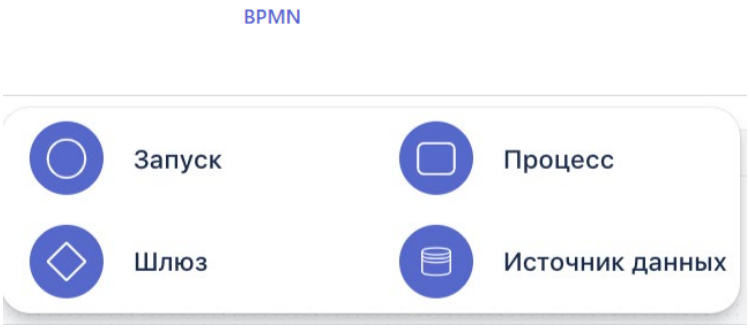


Рисунок 5.2 – Интерфейс **BAUM AI PREDICT**

На верхней панели инструментов рабочей области представлены следующие кнопки:

Таблица 4 – Кнопки панели инструментов рабочей области

	Создание рабочей области	При нажатии на кнопку открывается форма, в которой нужно указать название рабочей области:  Одна рабочая область может содержать несколько блок-схем. Реализована возможность запуска как отдельной блок-схемы, так и всех блок-схем на рабочей области. Рабочую область можно добавить в проект, таким образом организовав доступ к рабочей области для всех пользователей этого проекта.

<table>
<tr><td></td><td>Добавить элемент</td><td>


Чтобы добавить элемент на блок-схему достаточно нажать на кнопку с элементом.
Типы элементов нотации BPMN 2.0:
<table>
<tr><td>Запуск</td><td>Инициирующее событие – главный элемент, обозначающий начало блок-схемы.</td></tr>
<tr><td>Процесс</td><td>Действие, выполняемое в ходе бизнес-процесса. Является основным элементом.</td></tr>
<tr><td>Шлюз</td><td>Действие предназначено для разветвления алгоритма по веткам – прохождение сценария по каждой из веток выполняется при определенных условиях. Когда выполняется раздвоение потока операций, прописывается одно условие, и при его выполнении сценарий проходит по одной из веток.</td></tr>
<tr><td>Источник данных</td><td>Объект данных – это информационный объект (датасет, файл, модель и т.д.), который обрабатывается и передается в ходе выполнения бизнес-процесса. Действие предназначено для выбора уже загруженной и/или сохраненной в Системе сущности.</td></tr>
</table>
</td></tr>
<tr><td></td><td>Графики</td><td>После успешной отработки пайплайна здесь будет отображаться список доступных для визуализации графиков.</td></tr>
<tr><td></td><td>Таблицы</td><td>После успешной отработки пайплайна здесь будет отображаться список доступных для визуализации таблиц.</td></tr>
<tr><td></td><td>Изображения</td><td>После успешной отработки пайплайна здесь будет отображаться список доступных для визуализации изображений.</td></tr>
<tr><td></td><td>Описание</td><td>Отображается описание обученной модели ИИ.</td></tr>
</table>

- **Рабочая область**. Предназначена для построения графической бизнес-модели. С помощью блок-схемы выполняется настройка последовательных операций обработки данных, обучение моделей, выстраивается «pipeline» (цепочка процессов преобразования).

  В правом нижнем углу рабочей области отображается миниатюра бизнес-процесса:

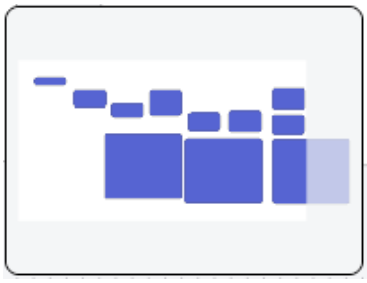


Рисунок 5.3 – Миниатюра моделируемого бизнес-процесса

- **Карточка элемента блок-схемы**. В карточке прописываются условия обработки данных – откуда берутся данные, какие операции над ними выполняются. Для перехода в карточку нажмите кнопку «Настройки» на элементе. Чтобы скрыть карточку элемента, используйте кнопку «Свернуть».

  Карточка содержит *программный код* с основными блоками:

  - входные данные (источник входных данных);
  - преобразование данных;
  - выходные данные – куда и в каком виде передаются преобразованные данные.
- **Вкладки с рабочими областями** - вы можете работать с несколькими рабочими областями одновременно, для удобного перехода между ними используйте вкладки.

5.3. Настройка внешнего вида интерфейса

Чтобы изменить настройки интерфейса, кликните на свою аватарку и перейдите в пункт меню «Настройки». Откроется окно, в котором можно изменить следующее:

- **Тема** – светлая или темная.
- **Адаптивные размеры шрифтов** – адаптация размеров шрифтов программы для работы с ней в мобильных устройствах.
- **Компактный вид -** фиксированная ширина на некоторых экранах.
- **Закругленные углы**. Если выбрать настройку, то окна в программе будут иметь закругленные углы. Иначе окна будут иметь прямые углы.

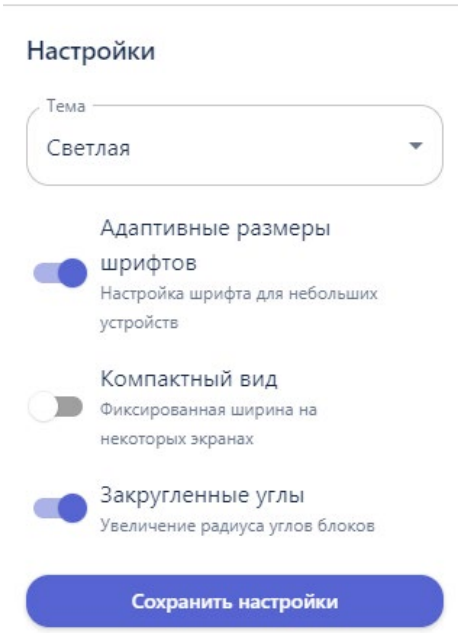


Рисунок 5.4 – Настройки внешнего вида интерфейса программы

Чтобы применить выбранные настройки к интерфейсу нажмите кнопку «Сохранить настройки».

- 5.4. Встроенные функции

Каждая функция относится к одному из модулей Системы: *модулю препроцессинга входных данных*, *модулю машинного обучения*, *модулю нейронных сетей/глубокого обучения*, *модулю анализа данных* или *модулю визуализации*.

Описание всех функций представлено в Приложении 1 настоящего документа.

За каждым типом блока закреплен определенный набор функций:

5.4.1 Функции элемента «Источник данных»

Таблица 5 – Набор функций элемента «Источник данных»

Группа	Функция
Загрузка данных	1. Загрузка табличных данных из коннектора
	2. Загрузка табличных данных
	3. Загрузка текстовых файлов
	4. Загрузка модели
Spark	5. Загрузка табличных данных из файла CSV (Spark)
	6. Загрузка табличных данных из папки CSV (Spark)

Группа	Функция
	7. Загрузка модели
	8. Загрузка табличных данных из коннектора (Spark)

5.4.2 Функции элемента «Процесс»

Таблица 6 – Набор функций элемента «Процесс»

Группа	Подгруппа	Функция
Анализ данных	**Загрузка данных**	1. Преобразование данных во временной ряд
	Препроцессинг	2. Стабилизация дисперсии
		3. Стандартизация
		4. Дифференцирование временного ряда
		5. One-Hot Encoding
		6. Создание признаков для временного ряда
		7. Кодирование целевого признака
		8. Порядковое кодирование категориальных признаков
	Тесты на нормальность распределения	9. Коэффициент асимметрии Skewness
	Тесты на стационарность временного ряда	10. Тест Дики-Фуллера
	-	11. Выбор признаков и целевых признаков
		12. Матрица корреляции
		13. Косинусное расстояние
		14. Поиск пропущенных значений

		15. Анализ временных рядов
		16. Визуализация Real Time
		17. Запись в датасет логирования
Машинное обучение	**Классификация**	18. Логистическая регрессия
		19. Модель XGBClassifier
		20. Дерево решений для классификации
		21. Случайный лес для классификации
		22. Categorial Naive Bayes
		23. Multinomial Naive Bayes
		24. Complement Naive Bayes
		25. Gaussian Naive Bayes
		26. Bernoulli Naive Bayes
	Обучение без учителя	27. Кластеризация DBSCAN
		28. Метод локтя K-Means
		29. Кластеризация K-Means
		30. Агломеративная иерархическая кластеризация
		31. Изоляционный лес
	Регрессия	32. Линейная регрессия
		33. Полиномиальная регрессия
		34. Дерево решений для регрессии

		35. Случайный лес для регрессии
		36. Метод опорных векторов для регрессии
		37. Байесовская гребневая регрессия
		38. Метод k-ближайших соседей для регрессии
	Авторегрессия	39. ARIMA/SARIMAX
Глубокое обучение	**Классификация**	40. Классификация изображений (Валидация)
		41. Классификация (табличные данные)
	Регрессия	42. Регрессия (табличные данные)
Предобработка данных	-	43. Заполнение пропусков
		44. Сглаживание временного ряда
		45. Срез временного ряда по индексу
Отправка уведомлений	-	46. Отправка уведомлений
Spark	-	47. Выбор признаков и целевых признаков
		48. Разделение датасета на обучающую и тестовую выборки
		49. Валидация модели
		50. Сохранение датасета Spark в CSV
		51. Прогноз модели
		52. Косинусное расстояние
	Препроцессинг	53. Порядковое кодирование признаков

		54. Нормализация признаков
	Классификация	55. Модель градиентного бустинга Spark для бинарной классификации
	Кластеризация	56. Кластеризация Spark DBSCAN

“–” в таблице означает, что у функции нет подгруппы, и она напрямую относится к группе функций.

6. Загрузка данных в систему

Сразу после авторизации в Системе открывается её начальная страница – раздел «Данные».

Раздел «Данные» имеет внешний вид аналогичный проводнику файлов в операционной системе компьютера. Пользователи имеют возможность создавать папки, загружать файлы и делать структуры, удобные для личного использования. По умолчанию для нового пользователя раздел Данные выглядит следующим образом:

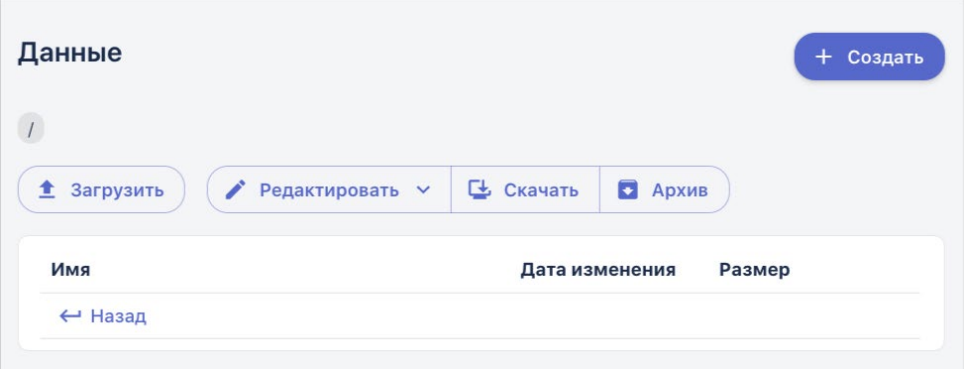


Рисунок 6.1 – Вид раздела Данные

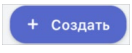
Основные кнопки для работы с разделом:

1. **«Создать»** - позволяет создавать папки и разметки.
2. **«Загрузить»** - осуществление непосредственно загрузки файлов в систему
3. ***«Редактировать»*** *- не работает в данной версии системы*
4. ***«Скачать»*** *- не работает в данной версии системы*
5. ***«Архив»*** *- не работает в данной версии системы*

В данном разделе вы можете создавать папки, добавлять в них файлы, создавая удобную и понятную структуру. Также в разделе осуществляется создание разметки для видео и изображений, и добавление папок классификации для дальнейшего использования в обучении искусственного интеллекта.

Вы можете загружать в систему файлы разных форматов и типов. Данные можно загружать напрямую в основную директорию, или создавая новые папки внутри.

- 6.1. Создание новой папки



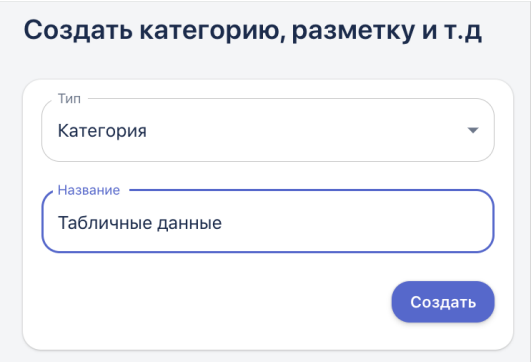


Рисунок 6.2 – Создание новой папки в разделе Данные

После этого папка появится в разделе Данные:

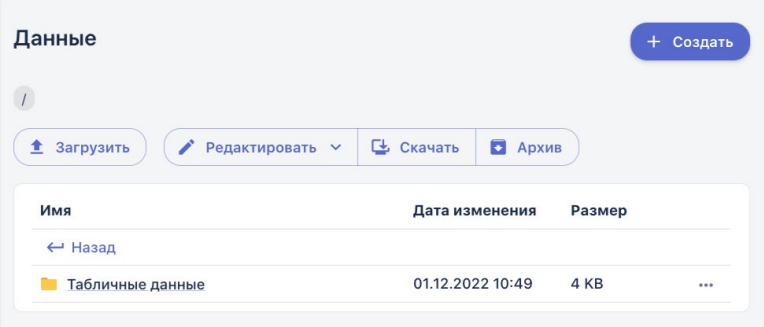


Рисунок 6.3 – Новая папка в разделе Данные

Обратите внимание, что папка будет добавлена в том разделе, из которого вы нажали кнопку «Создать». Т.е. далее вы можете перейти в папку «Табличные данные» и создать внутри еще одну категорию-папку.

- 6.2. Загрузка файлов

Для того чтобы загрузить файлы в папку, кликните на неё и перейдите в ее содержимое. Далее нажмите кнопку «Загрузить»:

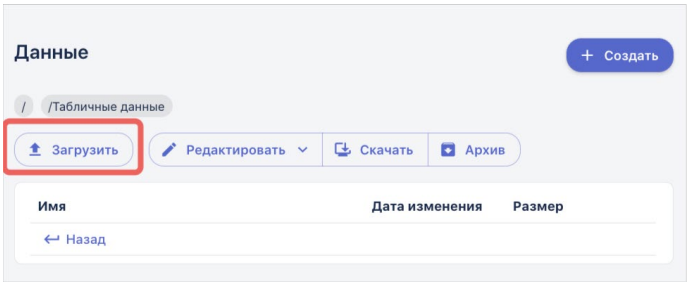


Рисунок 6.4 – Загрузка файлов в папку

В открывшемся окне левой кнопкой мыши нажмите на ссылку выбора файла. Указать путь к файлу для загрузки на вашем ПК. Второй вариант – перенести файлы с локального компьютера в этот раздел по технологии «drag n drop».

Выбранные файлы отобразятся в нижней части окна загрузки:

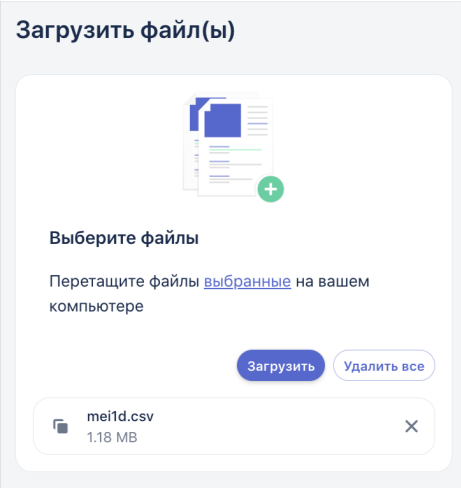


Рисунок 6.5 – Отображение выбранного файла

При необходимости выбранные файлы можно удалить по одному, нажав на крестик, или все вместе, нажав кнопку «Удалить все».

Для того чтобы загрузить выбранные файлы, нажмите на кнопку «Загрузить». Файлы отобразятся в папке:

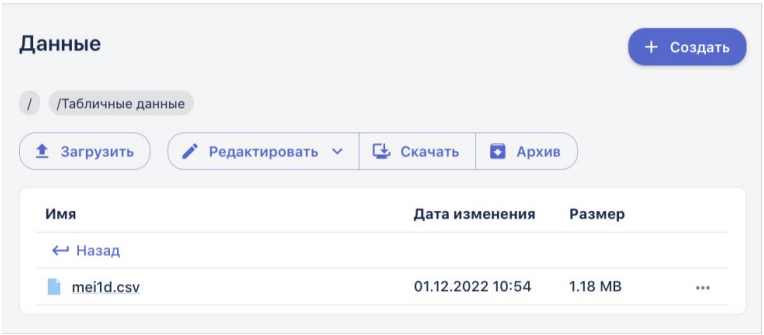


Рисунок 6.6 – Загруженный файл в разделе Данные

- 6.3. Предпросмотр данных

В системе реализована функция предпросмотра загруженных файлов следующих форматов:

- табличные данные формата .csv
- изображения в форматах: jpeg, jpg, png.
- видео в форматах: avi, mp4.

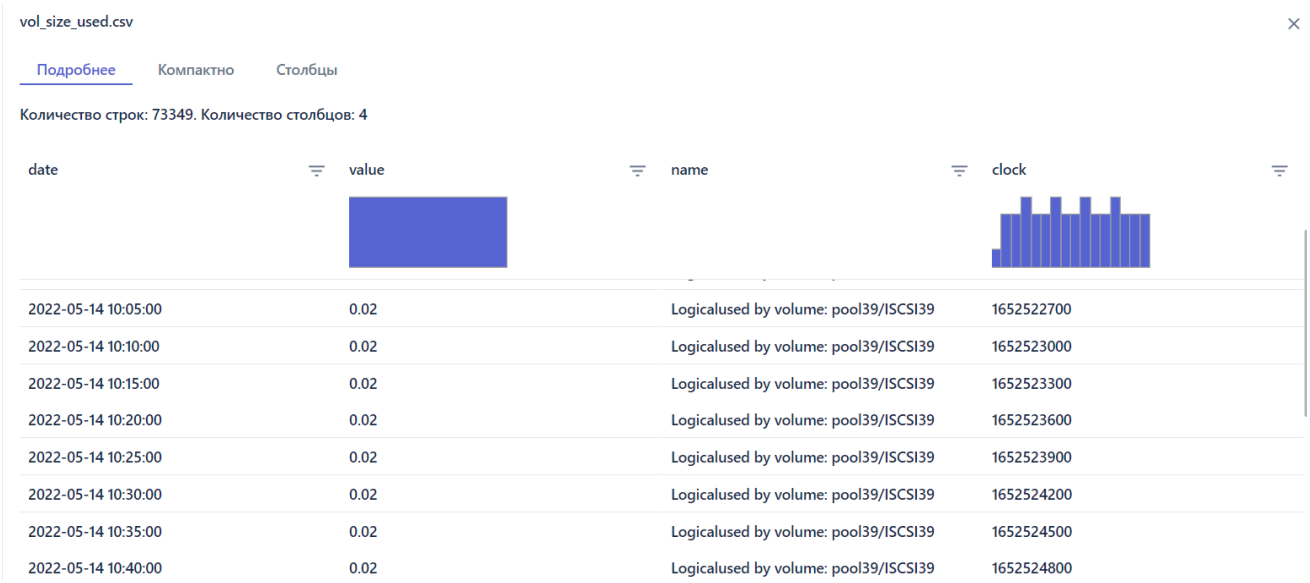
Если кликнуть на файл с изображением или видео - его предпросмотр начнется прямо в окне, чтобы выйти из режима предпросмотра - просто кликните в любую область экрана за пределами медиафайла.

При предпросмотре табличных данных в формате csv, в нижней части экрана отобразится окно с тремя вкладками, содержащими описательно статистический анализ.

На вкладке «Подробнее» (показано на примере файла с временным рядом) пользователь имеет возможность увидеть:

- размер датасета (количество строк, столбцов)

- гистограммы распределения каждого признака (числового)


vol_size_used.csv

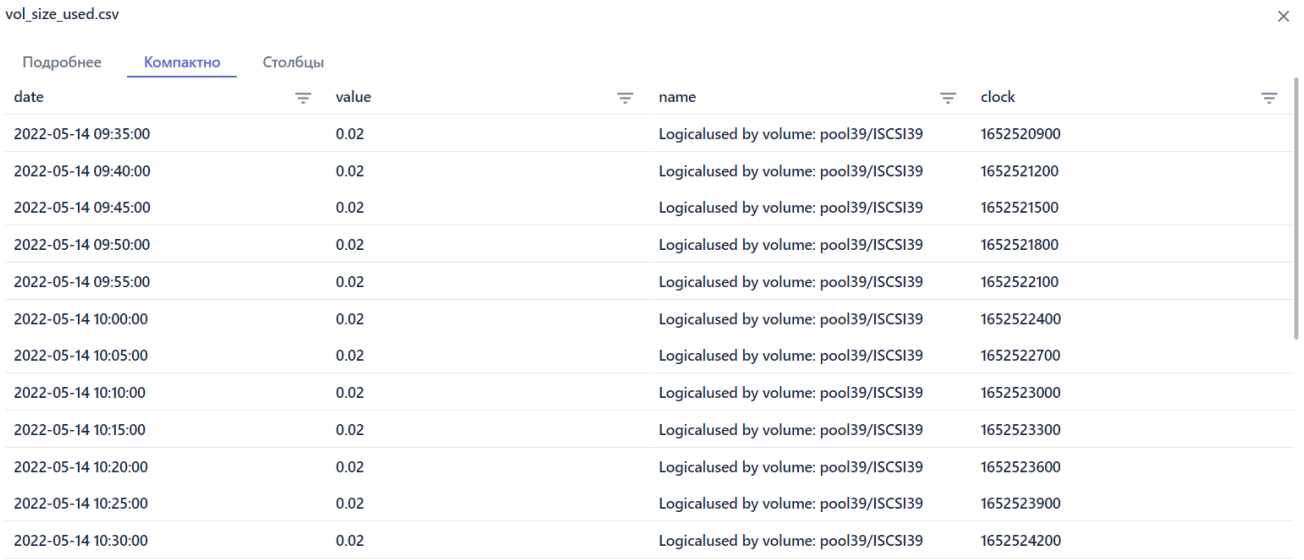
Подробнее Компактно Столбцы

Количество строк: 73349. Количество столбцов: 4

date	value	name	clock
2022-05-14 10:05:00	0.02	Logicalused by volume: pool39/ISCSI39	1652522700
2022-05-14 10:10:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523000
2022-05-14 10:15:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523300
2022-05-14 10:20:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523600
2022-05-14 10:25:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523900
2022-05-14 10:30:00	0.02	Logicalused by volume: pool39/ISCSI39	1652524200
2022-05-14 10:35:00	0.02	Logicalused by volume: pool39/ISCSI39	1652524500
2022-05-14 10:40:00	0.02	Logicalused by volume: pool39/ISCSI39	1652524800

Рисунок 6.7– Вкладка «Подробнее» окна отображения данных о датасете

На вкладке «Компактно» отображается состав строк и столбцов файлов:


vol_size_used.csv

Подробнее Компактно Столбцы

date	value	name	clock
2022-05-14 09:35:00	0.02	Logicalused by volume: pool39/ISCSI39	1652520900
2022-05-14 09:40:00	0.02	Logicalused by volume: pool39/ISCSI39	1652521200
2022-05-14 09:45:00	0.02	Logicalused by volume: pool39/ISCSI39	1652521500
2022-05-14 09:50:00	0.02	Logicalused by volume: pool39/ISCSI39	1652521800
2022-05-14 09:55:00	0.02	Logicalused by volume: pool39/ISCSI39	1652522100
2022-05-14 10:00:00	0.02	Logicalused by volume: pool39/ISCSI39	1652522400
2022-05-14 10:05:00	0.02	Logicalused by volume: pool39/ISCSI39	1652522700
2022-05-14 10:10:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523000
2022-05-14 10:15:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523300
2022-05-14 10:20:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523600
2022-05-14 10:25:00	0.02	Logicalused by volume: pool39/ISCSI39	1652523900
2022-05-14 10:30:00	0.02	Logicalused by volume: pool39/ISCSI39	1652524200

Рисунок 6.8 – Вкладка «Компактно» окна отображения данных о датасете

На вкладке «Столбцы» для каждого признака отображается:

- количество пропусков (Missing) в абсолютном и в процентном выражении, под пропуском понимается пустая ячейка в таблице
- среднее значение (Mean)
- стандартное отклонение (Std. Deviation)
- квантили (quantilies):

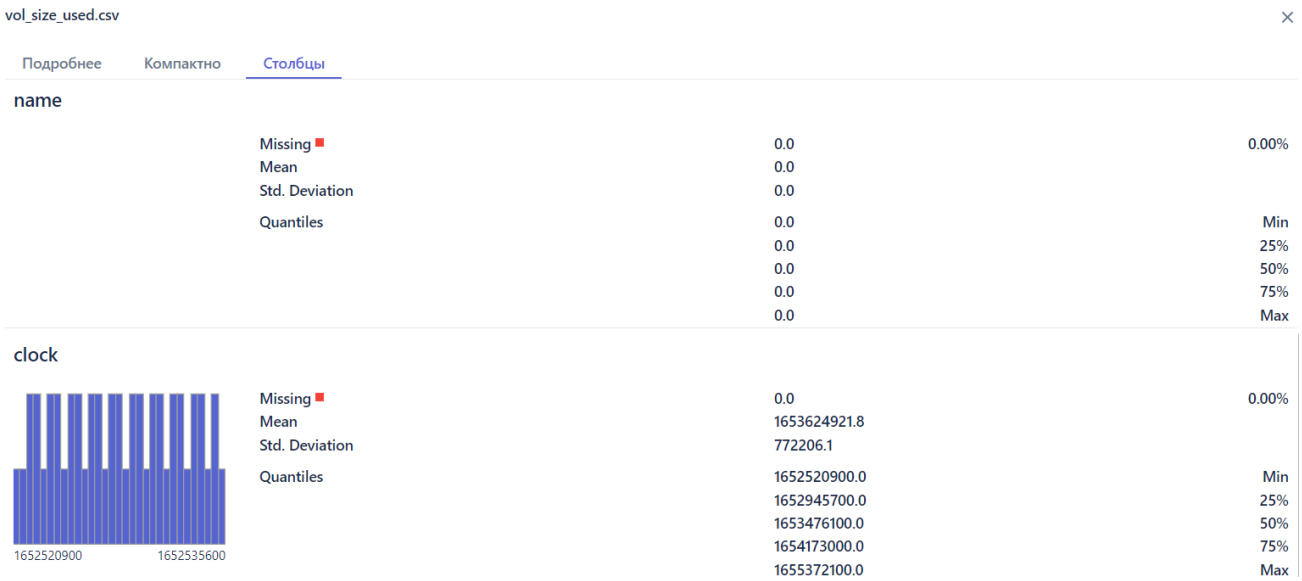


Рисунок 6.7– Вкладка «Столбцы» окна отображения данных о датасете

- 6.4. Взаимодействие с данными

Загруженный файл или созданную папку можно скачать, скачать архивом, переименовать или удалить. Для этого нажмите на три точки в правой части раздела и выберите соответствующую кнопку:

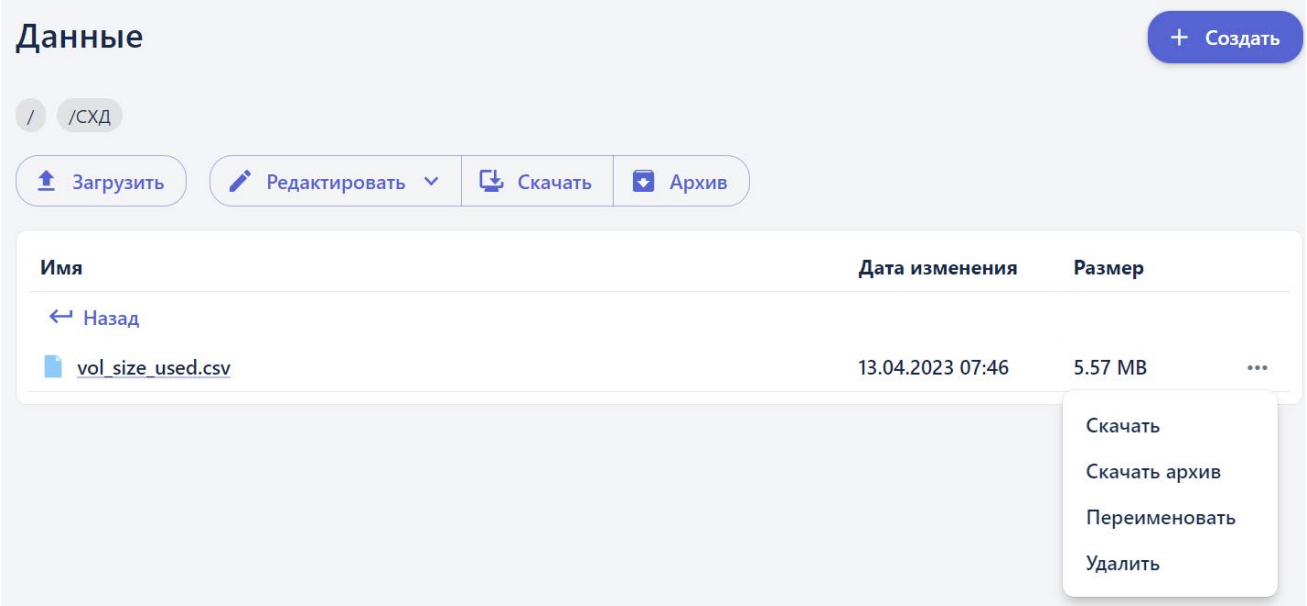
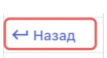


Рисунок 6.8 – Удаление или скачивание файла из раздела Данные

Вы можете удалять файлы по отдельности или сразу целую папку.



7. Создание модели ИИ

В разделе «Моделирование» осуществляется процесс построения блок-схем - соединение последовательных функции процессов анализа, обработки и преобразования исходных данных для построения и обучения моделей искусственного интеллекта. Построение таких блок-схем осуществляется на рабочих областях.

- 7.1 Создание новой и открытие сохраненной рабочей области

■7.1.1. Создание новой рабочей области

1. Перейдите в пункт меню системы Моделирование -> Рабочая область:

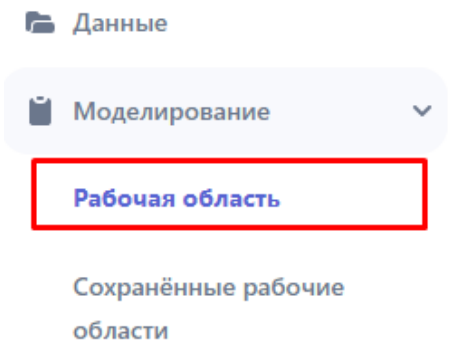


Рисунок 7.1.1 – Переход на рабочую область

Откроется страница с пустой рабочей областью:

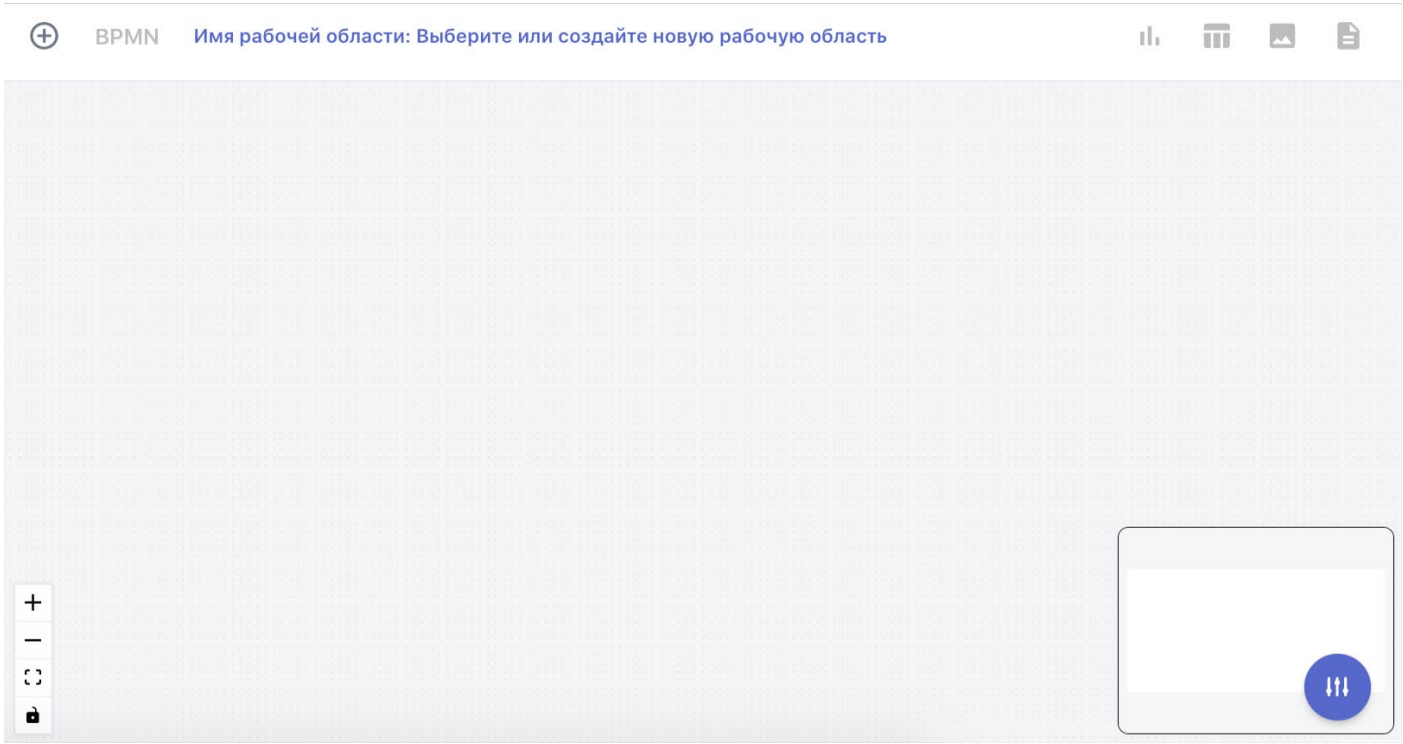


Рисунок 7.1.2 – Пустая рабочая область

В открывшейся форме введите название новой рабочей области и нажмите кнопку «Создать»:

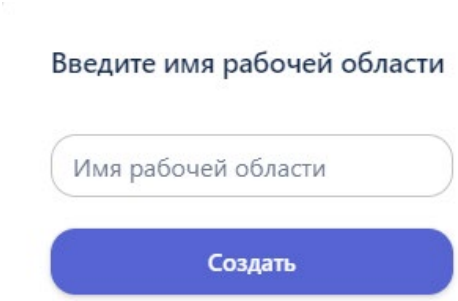


Рисунок 7.1.3 – Создание новой рабочей области

На панели инструментов отобразится название созданной рабочей области:

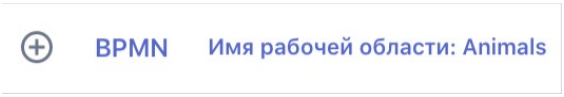


Рисунок 7.1.4 – Отображение названия рабочей области

После присвоения названия рабочая область автоматически сохраняется в раздел «Сохраненные рабочие области».

■ 7.1.2. Открытие сохраненной рабочей области

Для того чтобы открыть ранее созданную рабочую область, перейдите в пункт меню системы Моделирование -> Сохраненные рабочие области. Откроется страница «Рабочие области»:

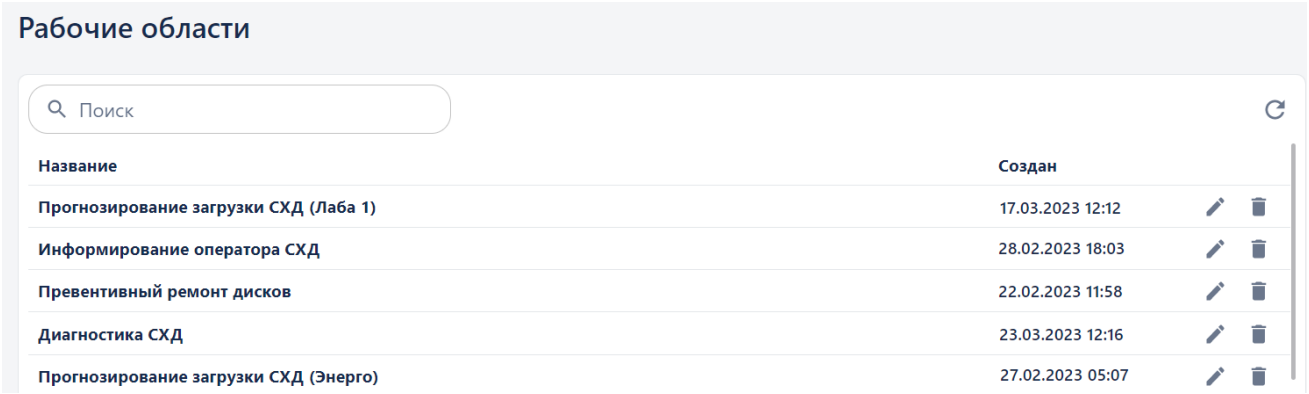


Рисунок 7.1.2 – Страница со списком созданных в Системе рабочих областей

На странице отображаются рабочие области. Чтобы открыть сохраненную рабочую область кликните на её название. Чтобы удалить ненужную рабочую облать - кликните кнопку удалить в правой части строки с названием области.

- 7.2. Построение блок схемы

Предварительным условием для построения блок схемы является:

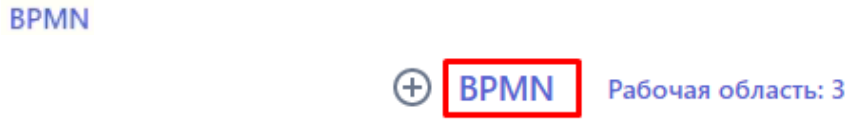
1) Загруженные в систему данные (файлы или датасеты)

2) Созданная рабочая область (на одной рабочей области можно создать неограниченное количество блок-схем.

Построение блок-схемы осуществляется путем добавления на рабочую область элементов (блоков) и соединение их между собой.

■7.2.1 Блок «Запуск»

Блок «Запуск» обозначает начало блок-схемы, и всегда является её первым элементом. Так как на рабочей области может быть несколько блок-схем, именно по блоку «Запуск» определяется их количество, и идентифицируется принадлежность блоков к той или иной блок-схеме.



Откроется меню выбора блоков – библиотека графических элементов нотации BPMN 2.0:

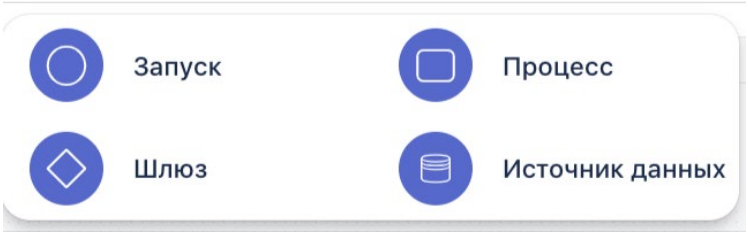
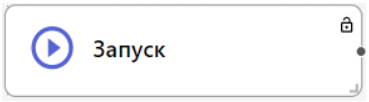


Рисунок 7.2.1 – Меню блоков

Выберите элемент «Запуск». Выбранный элемент будет добавлен на рабочую область:



При необходимости размер блока можно увеличить или уменьшить, потянув за уголок в правой нижней части элемента:

Обратите внимание, что в правом нижнем углу рабочей области отображается уменьшенное изображение создаваемой блок-схемы, эта функция упрощает навигацию по полотну.

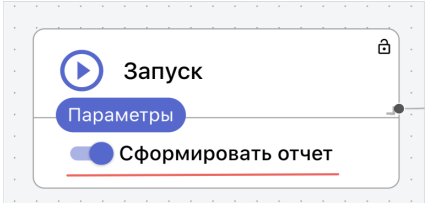


Рисунок 7.1.2 – Опция формирования отчета на блоке «Запуск»

■ 7.2.2. Блок «Источник данных»

Следующим элементом в блок схеме после «Запуска» всегда является «Источник данных» - блок, который определяет какие данные будут использоваться в сценарии.

Для объединения элементов в блок-схему, их требуется соединить между собой. Для этого нажмите на точку выхода блока, которая отображается в виде круглой точки на правой грани блока, и перетащите мышью появившуюся стрелку в сторону нужного блока, как показано на рисунке:

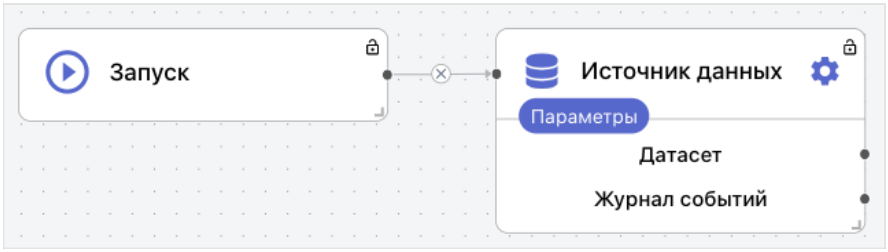


Рисунок 7.2.1 – Соединение блоков между собой

На блоке «Источник данных» отображаются два компонента:

1) «Датасет» - это непосредственно сами данные.
2) «Журнал событий» содержит информацию обо всех преобразованиях с данными, которые выполняются в текущем блоке пайплайна. Ведение журнала позволяет сохранить историю преобразований над данными, и при необходимости выполнить обратное преобразование.

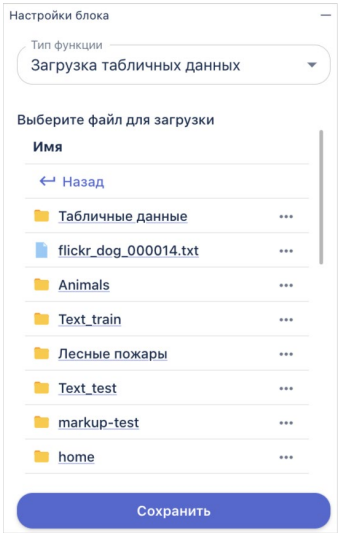


Рисунок 7.2.2 – Настройка параметров блока «Источник данных»

** Дальнейшие действия по настройке блоков указаны для примера. При работе с Системой пользователь должен выбрать необходимые параметры исходя из своей задачи и загруженных данных.*

После добавления на рабочую область, для элемента «Источник данных» по умолчанию выбрана функция: тип функции «Загрузка данных» -> функция «Загрузка табличных данных». Это можно увидеть в верхней части окна настройки параметров элемента:

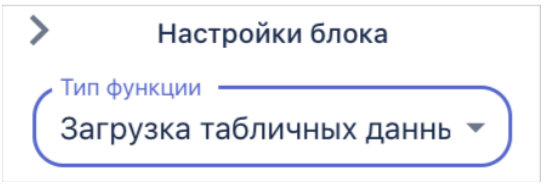


Рисунок 7.2.3 – Отображение типа функции в параметрах блока

Для того чтобы посмотреть другие доступные функции, нужно нажать на выпадающий список. Весь список доступных функций и их описания доступны в Таблице 18.1 – Перечень автоматизированных функций элемента «Источник данных».

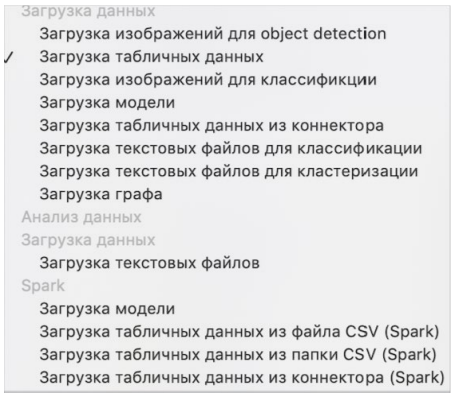


Рисунок 7.2.4 – Список возможных функций элемента «Источник данных»

В разделе «Выберите файл» отображается структура папок из разделе «Данные», чтобы выбрать файл достаточно перейти в нужную папку и кликнуть на три точки в правой части строки с названием файла и нажать «Выбрать»:

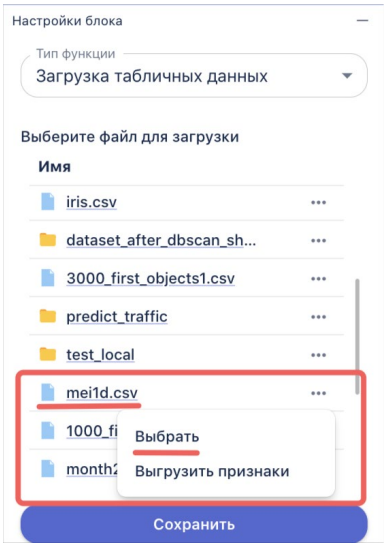


Рисунок 7.2.5 – Отображение папок и файлов из раздела «Данные» в параметрах блока «Источник данных»

После этого в нижней части окна отобразится его название:

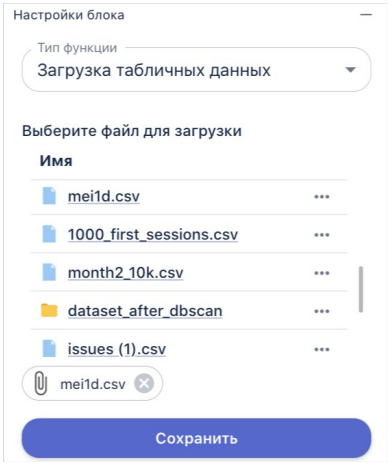


Рисунок 7.2.6 – Отображение выбранного файла, предназначенного для загрузки в блок-схему

Кнопка «Выгрузить признаки» используется для других блоков, где в настройках необходимо указать целевые признаки для конкретной функции.

Если кликнуть на название файла на рабочей области отобразится его предпросмотр:

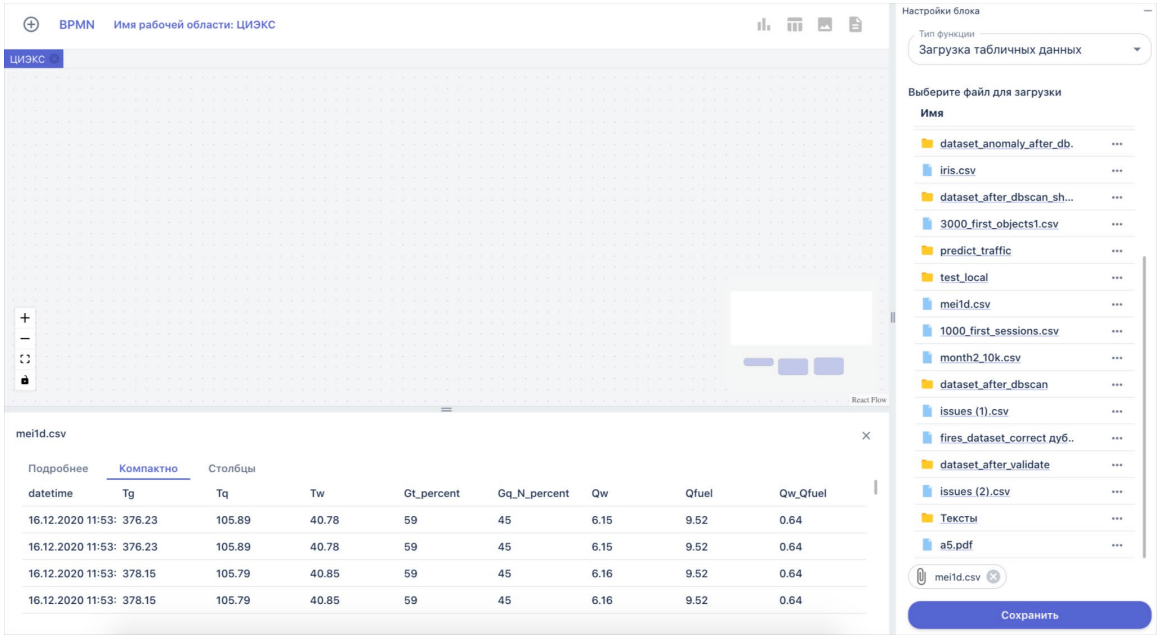


Рисунок 7.2.6.1 – Предпросмотр файла на рабочей области

Для сохранения выбранных настроек нажмите на панели параметров кнопку «Сохранить» (далее сохранение настроек элемента предполагается по умолчанию). Для удаления неверно добавленного на рабочую область элемента предусмотрена кнопка «Удалить блок».

Любой блок можно переименовать, чтобы дать ему понятное название, отображающее суть происходящего процесса. Для этого дважды щелкните левой кнопкой мыши на текущее название элемента в рабочей области и измените его. Чтобы новое название сохранилось достаточно щелкнуть мышью в любом месте на рабочей области.

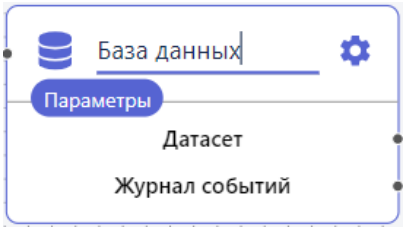


Рисунок 7.2.7 – Ввод названия элемента

Для того чтобы удалить блок, кликните по нему правой кнопкой мыши и нажмите «Удалить».

■ 7.2.3. Блок «Процесс»

Блок «Процесс» предназначен для выполнения операций над данными. Блок-схема может содержать несколько элементов «Процесс», настроенных пользователем для выполнение определенных задач. Весь список доступных функций и их описания доступны в Таблице 18.2 – Перечень автоматизированных функций элемента «Процесс».

Далее будет показан принцип настройки свойств блока на примере одной функции. Выбор функции определяется типом решаемой задачи.

Для примера выберите для элемента функцию: раздел «Машинное обучение» -> функция «Разделение датасета на обучающую и тестовую выборки»:

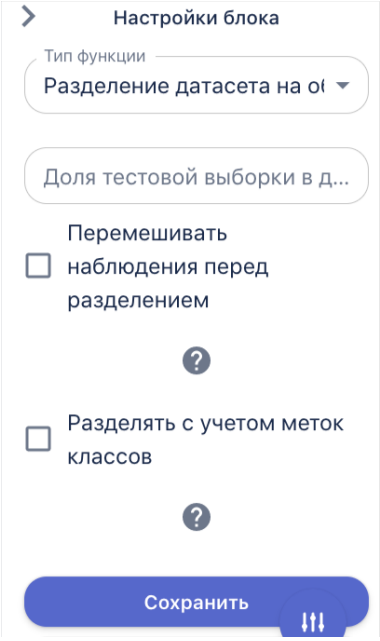


Рисунок 7.2.8 – Панель свойств блока «Процесс»

Далее осуществляется настройка параметров следующим образом:

- В разделе «Параметры» –> в поле «Доля тестовой выборки в датасете» введите значение 0.2;
- Оставьте пустым поле «Перемешивать наблюдения перед разделением». Рядом с полем есть подсказка, что не рекомендуется перемешивать наблюдения во временных рядах (выбирайте действие в зависимости от типа входных данных);
- Установите галочку в поле «Разделять с учетом меток классов» – применяется для задач классификации (выбирайте действие в зависимости от решаемой задачи).

Измените название элемента на «Сплит датасета»:

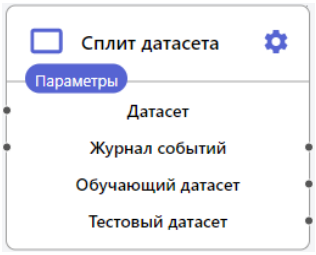


Рисунок 7.2.9 – Отображение блока на рабочей области после настройки его параметров и ввода названия

Обратите внимание, что соединение элемента «Процесс» с другими элементами блок-схемы выполняется только после настройки и сохранения его параметров. Это связано с тем, что каждая функция имеет свой набор компонентов, который отображается на элементе после сохранения его настроек.

Соединить элемент «Процесс» с предыдущими элементами блок-схемы можно следующим образом:

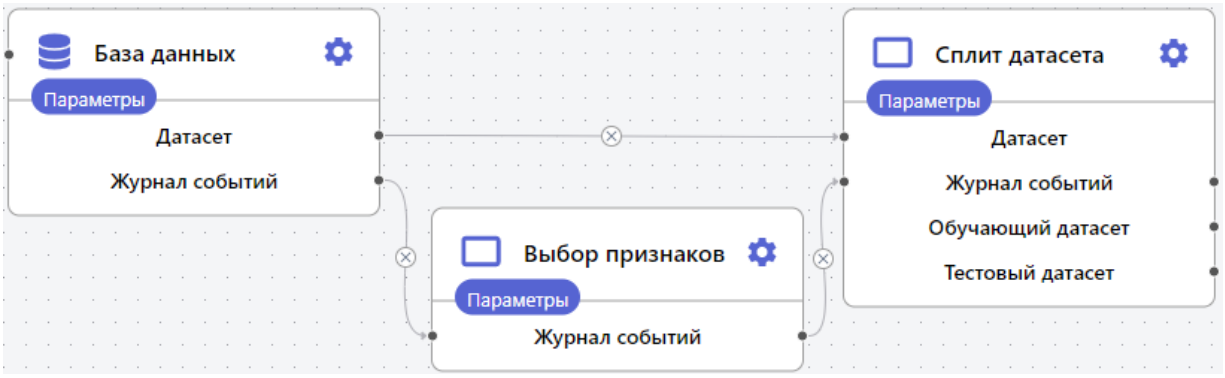


Рисунок 7.2.10 – Соединения между несколькими блоками (организация связей/пробрасывание данных)

Обратите внимание, что соединять можно только идентичные (одноименные) компоненты блоков.

Больше примеров построения блок схем, сценарии сохранения моделей, отображения на рабочих областях таблиц, графиков и изображений можно прочитать в разделе Примеры работы с Платформой.

- 7.3. Запуск блок-схемы на рабочей области

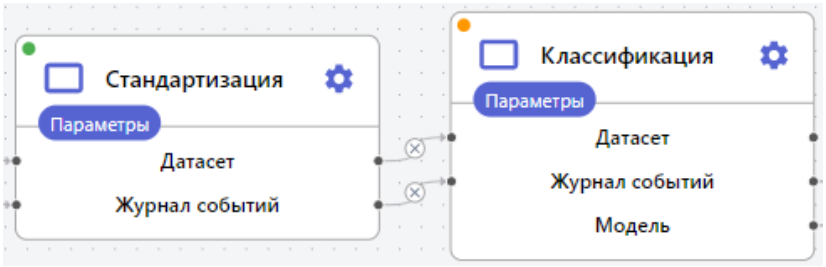


Рисунок 7.3.1 – Последовательная обработка блоков пайплайна

В случае, если блок не отработал (например, вследствие того, что был неправильно настроен или из-за неверных входных данных), на нем появится индикатор красного цвета. Такой блок необходимо проверить, изменить настройки и запустить заново:

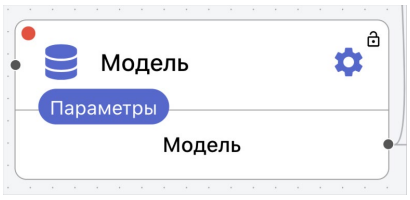


Рисунок 7.3.2 – Индикатор ошибки при обработке блока

8. Сохранение модели ИИ

Для сохранения *модели ИИ*, обученной выполнению какой-либо задачи, на блок-схему необходимо добавить специальный блок Этот блок настраивается для элемента «Процесс» с помощью функции «Управление моделями» → функция «Сохранение модели»:

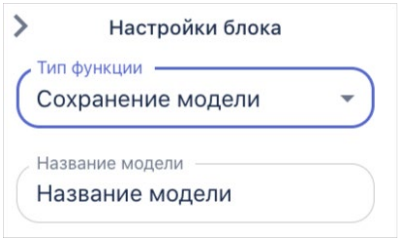


Рисунок 7.3.1 – Настройка блока для сохранения обученной модели ИИ

После успешной отработки блок схемы, которая содержит такой блок, название модели будет добавлено в список сохраненных моделей в меню «Модели»:

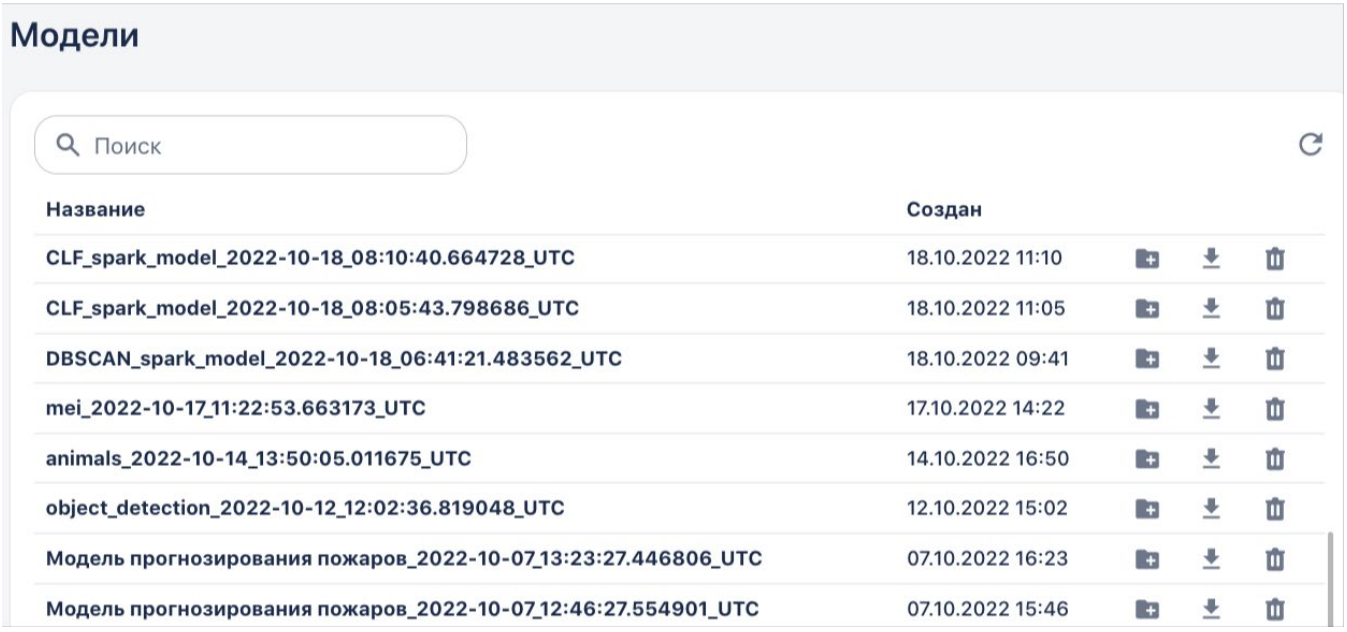


Рисунок 7.3.2 – Вкладка меню «Модели»

Обратите внимание, что модель будет сохраняться столько раз, сколько будет запущена блок схема с элементом «Сохранение модели», при этом в разделе будет меняться временная отметка создания записи.

Модель можно использовать следующими способами:

- **model.pkl** - сама модель.

- **vars_dict.pkl** - словарь преобразований. Преобразования необходимо сохранять, чтобы при анализе новой порции данных над ними выполнялись все те же преобразования, что и при обучении модели.
- **info.json** - служебный файл, куда прописывается тип модели.

3. Использовать при построении новой блок схемы в качестве источника данных (например, для целей прогнозирования). Пример можно посмотреть в разделе 14.4 Работа с данными в режиме реального времени.
4. Создать коннектора с обученной моделью (например, для распознавания объектов на видео или изображениях). Это позволит проверить обучение модели на новой порции данных. Подробней описано в разделе 14.2.1 Проверка обученной модели на локальных данных.

9. Графическое представление информации на рабочей области

После *сборки* и *успешного* запуска блок-схемы на рабочей области есть возможность посмотреть результаты обучения созданной модели в виде графиков, таблиц и изображений, которые можно вывести прямо на рабочую область.

> ***Примечание**: под сборкой имеется в виду, что на рабочую область добавлены все элементы блок-схемы, и они последовательно соединены между собой. А успешным считается запуск блок-схемы, когда все ее элементы отработали с «зелеными»*

Вы можете нажать на те кнопки визуализации, которые подсвечиваются фиолетовым цветом. При этом голубым подсвечиваются только те иконки, которые актуальны для запущенной схемы, т.к. блоки могут содержать разные функции, имеющие разное графическое представление.

- 9.1. Графики

Полный список доступных в Системе графиков с объяснением интерпретации результатов доступен в **Базе знаний.** За каждым типом блока закреплен определенный набор графиков, например, для блока Анализ данных -> Анализ временных данных доступны следующие графики: Линейный график, ACF/PACF, Декомпозиция, Свечной график, Time profile, Extended, Bollinger Bands Stochastic Oscillator.

Для того чтобы добавить график на рабочую область из выпадающего списка выберите нужное название. Например, «Time profile временного ряда» в анализе временных рядов:

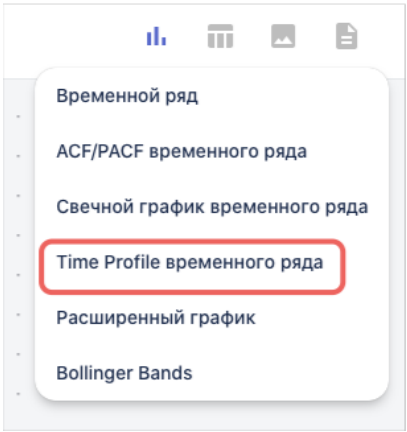


Рисунок 9.1.1 – Список доступных графиков для анализа временных рядов

В результате на рабочую область будет добавлен график:

Рисунок 9.1.2 – Time profile временного ряда на рабочей области

Для ряда графиков доступен выбор из выпадающего списка признака, для которого составляется визуализация. Например, можно два раза выбрать график Time profile и для одного указать признак Tq, а для другого Tw и сопоставлять их значения одновременно на рабочей области:

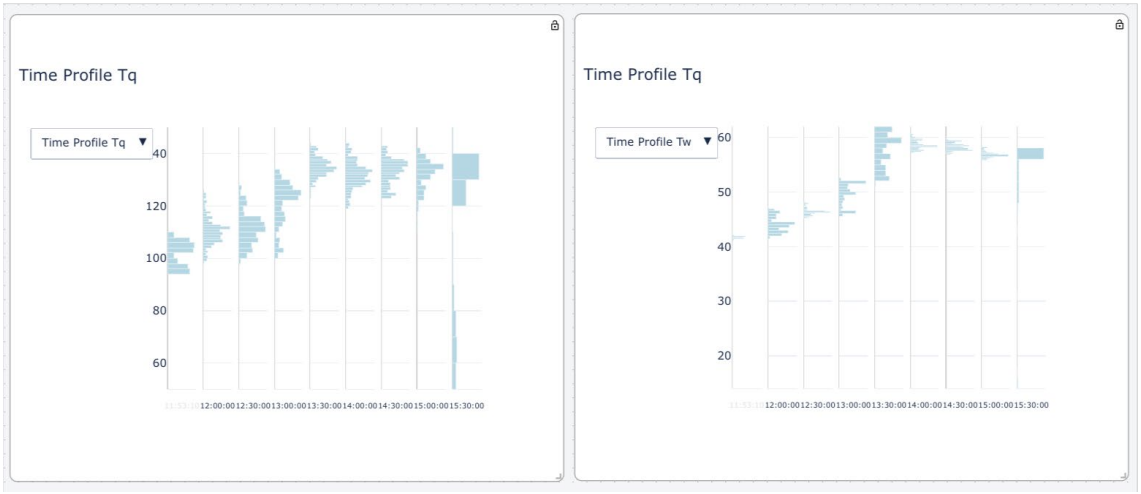


Рисунок 9.1.3 – Отображение одного типа графика для разных признаков

Также в правом углу рамки графической визуализации, при наведении на неё курсора, отображаются следующие кнопки:

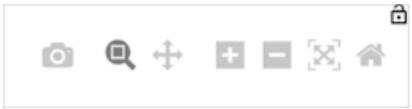


Рисунок 9.1.4 – Кнопки для работы с визуализацией

где:

	- скачать график, как рисунок в формате png
	- при нажатии лупы, можно увеличить любой участок графика
	- при нажатии данной кнопки, можно передвигаться по плоскости графика в любую сторону
	- приблизить, увеличить
	- уменьшить
	- автоматически подогнать размер под границы
	- вернутся к исходному виду

Таблица 9.1 - Функционал кнопок работы с визуализацией

В программе также реализована возможность создания графиков с данными, получаемыми в режиме реального времени. Для таких графиков существует отдельный блок, находящийся в разделе «Анализ данных» -> «Визуализация Real Time». Главное отличие от стандартного блока «Визуализация» в том, что для каждого графика необходимо задавать число периодов в окне и период окна - параметры, которые определяют интервал, который будет отображаться на графике в рабочей области.

- 9.2. Таблицы

Например:

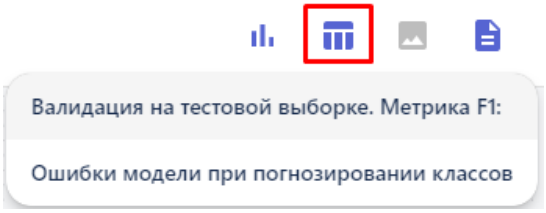
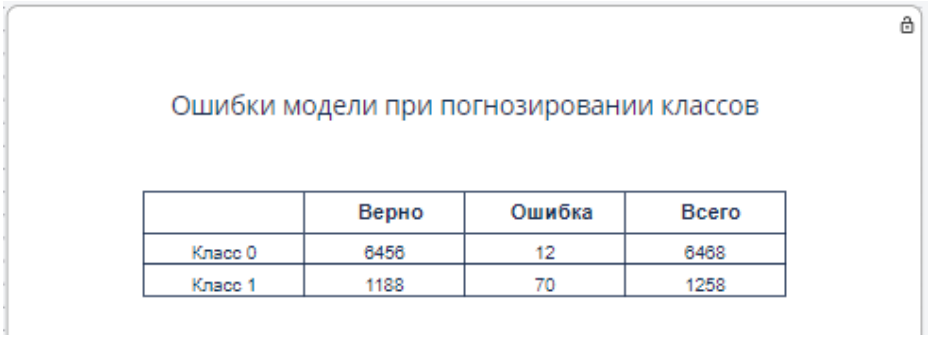


Рисунок 9.2.1 – Список таблиц для визуализации

Чтобы отобразить таблицу на рабочей области, нажмите на её название в выпадающем списке и она появится на экране, например:



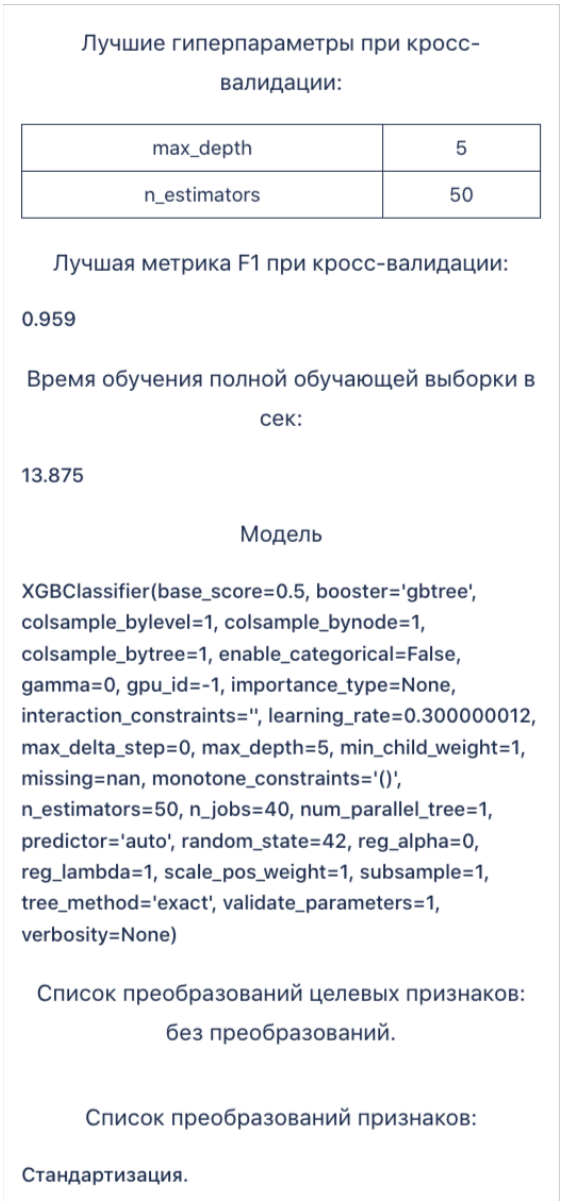
Ошибки модели при погнозировании классов

	Верно	Ошибка	Всего
Класс 0	6456	12	6468
Класс 1	1188	70	1258

Рисунок 9.2.2 – Блок визуализации «Ошибки модели при прогнозировании классов»

- 9.4. Описание модели

Описание варьируется в зависимости от функций, которые были применены в модели. Например, для блок схемы, где присутствовали элементы «Стандартизация», «Валидация» и «XGB классификации» описание будет выглядеть следующим образом:


Лучшие гиперпараметры при кросс-валидации:

max_depth	5
n_estimators	50

Лучшая метрика F1 при кросс-валидации:

0.959

Время обучения полной обучающей выборки в сек:

13.875

Модель

XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1, enable_categorical=False, gamma=0, gpu_id=-1, importance_type=None, interaction_constraints='', learning_rate=0.300000012, max_delta_step=0, max_depth=5, min_child_weight=1, missing=nan, monotone_constraints='()', n_estimators=50, n_jobs=40, num_parallel_tree=1, predictor='auto', random_state=42, reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact', validate_parameters=1, verbosity=None)

Список преобразований целевых признаков:

без преобразований.

Список преобразований признаков:

Стандартизация.

Рисунок 9.4.1 – Вариант описания модели

Если на рабочей области размещены несколько блок схем, при нажатии на описание вы увидите информацию по каждой из них.

10. Работа с Дашбордами. Раздел «Визуализация».

Дашборд – это интерактивная рабочая область, которая наглядно представляет, визуализирует, объясняет и анализирует данные. На рабочую область пользователь может добавлять графики, таблицы, диаграммамы, визуализацию пайплайнов для последующей работы с ними.

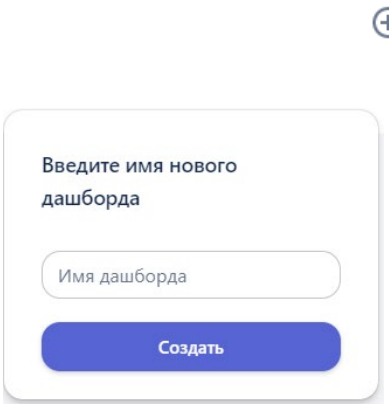


Рисунок 10.1 – Создание нового дашборда

В результате отобразится название дашборда (это текущий дашборд, с которым работает пользователь):

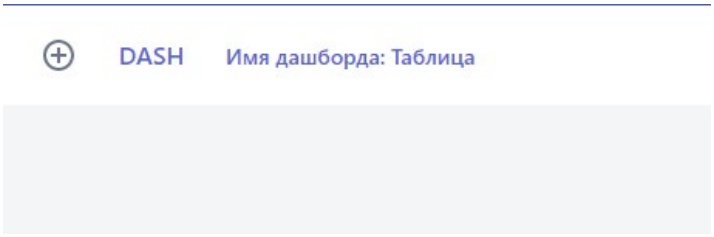


Рисунок 10.2 – Отображение наименования дашборда

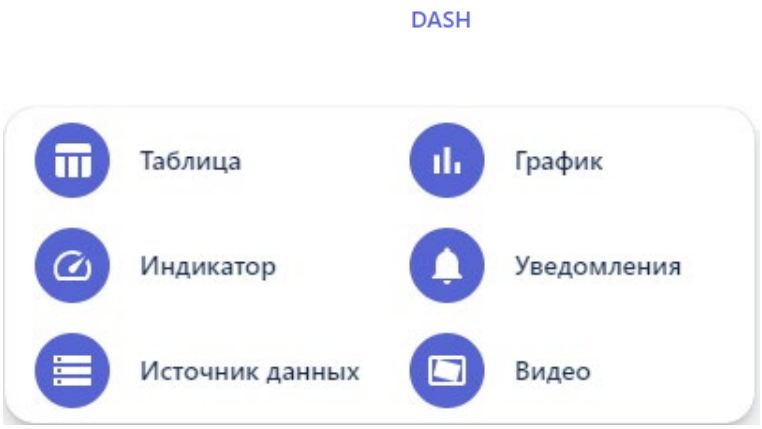


Рисунок 10.3. – Добавление интерактивного блока

В текущей версии Системы реализована работа с блоками: «Таблица», «График» и «Видео». Предварительным условием для добавления любого из типов блоков является создание коннектора в разделе «Соединения», в котором настроено получение данных. При этом это могут быть как данные,

получаемые из внешних источников, так и сгенерированные внутри Системы. Подробней о создании коннекторов вы можете посмотреть в разделе Работа со всеми типами коннекторов.

- 10.1. Таблица

Интерактивный блок «Таблица» предназначен для:

- Отображения подключения к внешним базам данных, в виде табличных данных, обновляющихся в режиме реального времени. В таком случае настраивается подключение к коннекторам с названиями типов баз данных (clickhouse, postgresql, mongo);
- Записи и сохранения в Системе информации, получаемой из внешних баз данных. Используется коннектор с типом «save_table»;
- Прогнозирования целевых событий, когда данные для анализа поступают из внешних баз данных. Используется коннектор с типом «table_app»;
- Отображение таблиц, полученных в результате отработки блок-схем, которые содержат блоки, имеющие в качестве выходной информации визуализации в виде таблиц. Используется коннектор с типом «constructor». Такой тип коннектора создается автоматически, после успешного запуска блок-схемы.

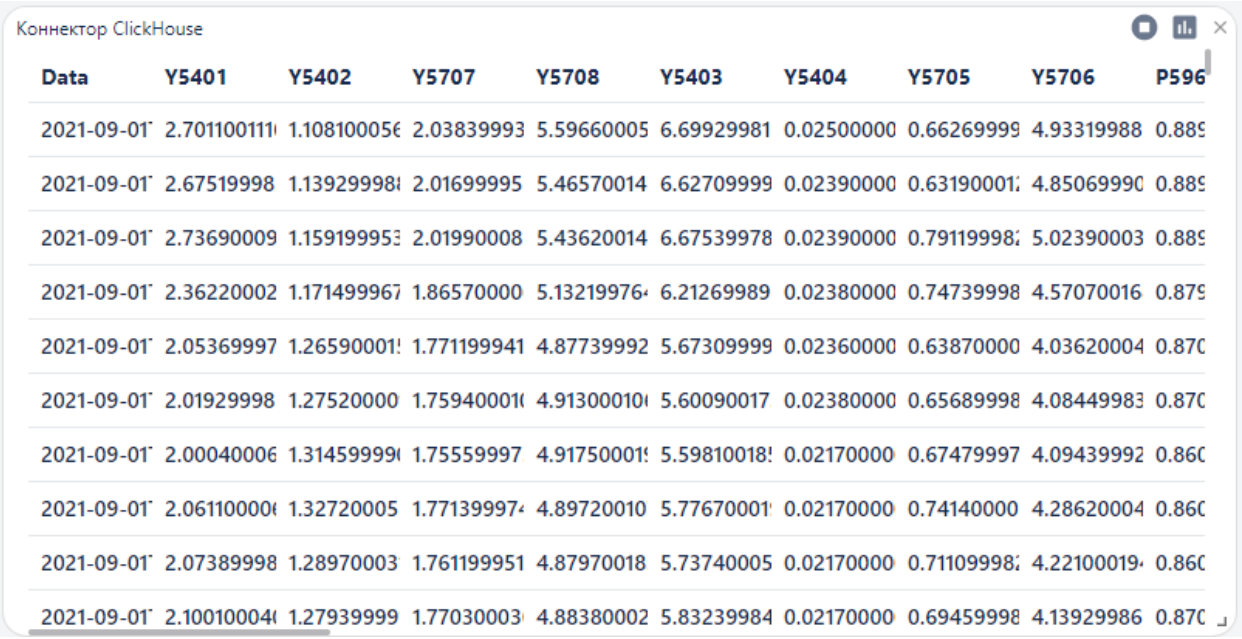
Например, чтобы настроить дашборд с подключением к коннектору с типом «clickhouse»:

1. Создайте новый дашборд.
2. На дашборд добавьте интерактивный блок с типом «Таблица».

Рисунок 10.4. – Список коннекторов

В открывшемся окне для коннекторов отображаются их состояния (Запущен (Started) / Остановлен(Stopped)). Если статус коннектора «остановлен», данные из внешних источников не поступают в Систему. Из этого окна можно запустить/остановить коннектор, выполнить его редактирование при необходимости, или удалить его.

5. Выбрать коннектор из списка нажатием на него левой кнопкой мыши.
6. На дашборде отобразится результат подключения к БД «ClickHouse» в виде таблицы с данными:



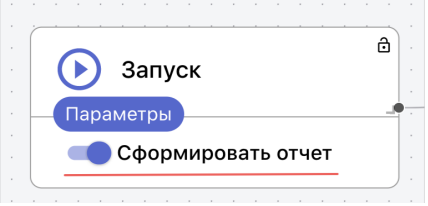
Data	Y5401	Y5402	Y5707	Y5708	Y5403	Y5404	Y5705	Y5706	P596
2021-09-01	2.7011001111	1.108100056	2.03839993	5.59660005	6.69929981	0.02500000	0.66269999	4.93319988	0.889
2021-09-01	2.67519998	1.139299981	2.01699995	5.46570014	6.62709999	0.02390000	0.63190001	4.85069990	0.889
2021-09-01	2.73690009	1.159199953	2.01990008	5.43620014	6.67539978	0.02390000	0.79119998	5.02390003	0.889
2021-09-01	2.36220002	1.171499967	1.86570000	5.13219976	6.21269989	0.02380000	0.74739998	4.57070016	0.879
2021-09-01	2.05369997	1.265900015	1.771199941	4.87739992	5.67309999	0.02360000	0.63870000	4.03620004	0.870
2021-09-01	2.01929998	1.27520000	1.759400016	4.91300010	5.60090017	0.02380000	0.65689998	4.08449983	0.870
2021-09-01	2.00040006	1.314599990	1.75559997	4.91750015	5.59810185	0.02170000	0.67479997	4.09439992	0.860
2021-09-01	2.06110000	1.32720005	1.771399974	4.89720010	5.77670015	0.02170000	0.74140000	4.28620004	0.860
2021-09-01	2.07389998	1.289700033	1.761199951	4.87970018	5.73740005	0.02170000	0.71109998	4.22100019	0.860
2021-09-01	2.10010004	1.27939999	1.77030003	4.88380002	5.83239984	0.02170000	0.69459998	4.13929986	0.870

Рисунок 10.5 – Отображение табличных данных из коннектора clickhouse

Коннектор можно остановить или запустить прямо на дашборде, для этого нажмите кнопку «остановить»/«запустить». Для того чтобы удалить дашборд - нажмите на крестик в правом верхнем углу блока.

11. Создание отчета с результатами анализа данных

Более подробную информацию с результатами обучения модели можно просмотреть в отчете, который создается также после сборки блок-схемы. Для формирования отчета на первом блоке необходимо перевести бегунок вправо:



В названии отчета будет указано название блок-схемы, по которой формируется отчет и временная отметка его формирования. Для просмотра сформированного отчета после запуска блок-схемы нужно перейти в пункт меню «Отчеты» и выбрать из списка отчет, нажав на его название:

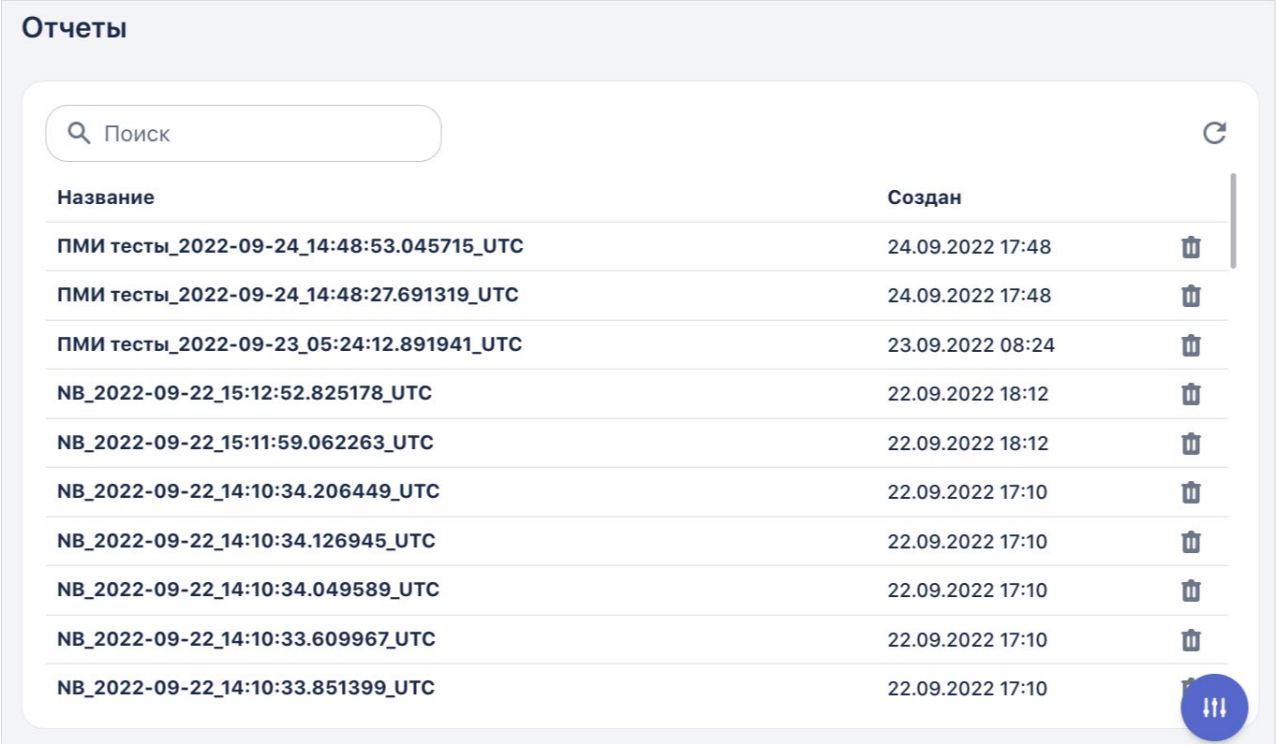


Рисунок 11.1 – Список автоматически сформированных отчетов

После нажатия на название отчета открывается отдельная вкладка с отчетом. В отчете кроме результатов обучения созданной модели (см. «Визуализацию») отображаются также: входные данные, отдельно выборки – обучающая и тестовая, датасет после стандартизации признаков, и т.д. Состав отчета отличается в зависимости от метода, который использовался в решении задачи ИИ.

12.Конвейер приложений

Конвейер приложений позволяет создать приложение на основе обученной модели. В таком приложении заложен шаблон, умеющий предсказывать наступление интересующих событий. Приложение можно развернуть отдельно за пределами системы, интегрировать с внешними системами, настроить получение входных данных и выполнять прогнозы.

Для того чтобы сформировать приложение необходимо перейти в раздел “Модели”:

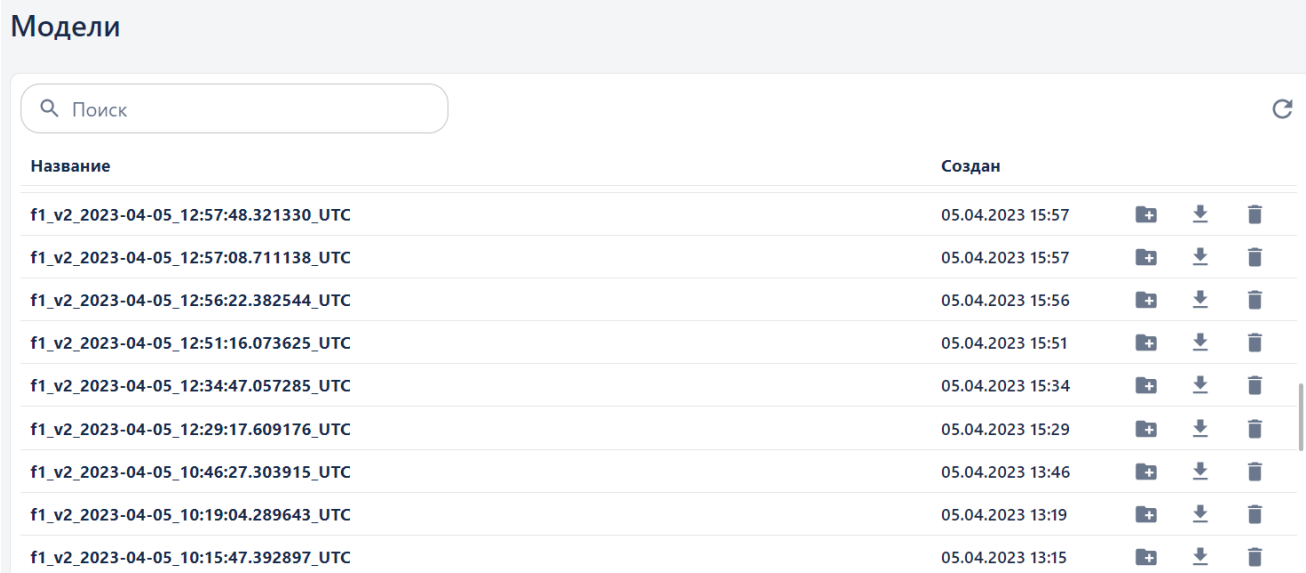


Рисунок 12.1 – Список сохраненных моделей

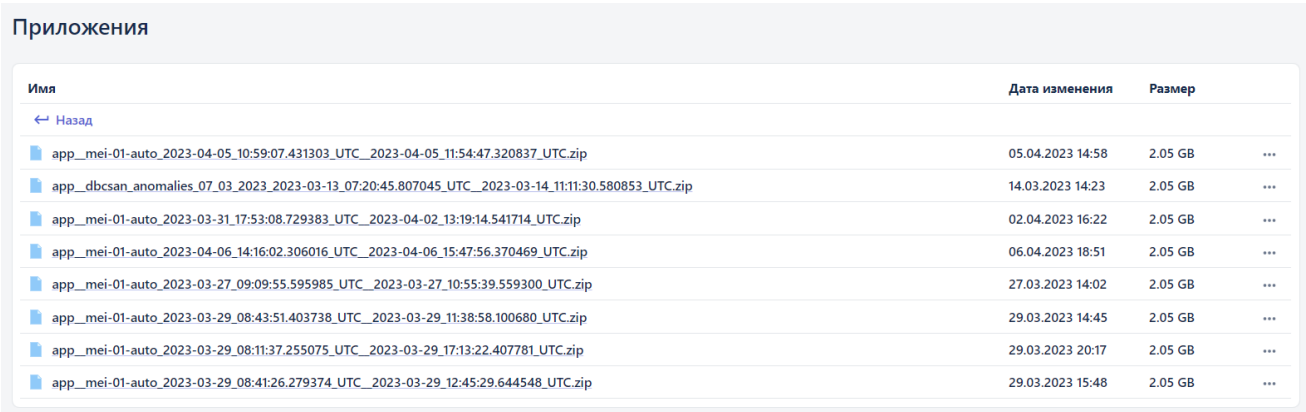


Рисунок 12.2 – Страница «Приложения»

Приложение упаковано в docker-контейнер и доступно для скачивания. Для того чтобы скачать приложение - нажмите на три точки в строке с названием приложения и скачайте его.

Комплектность приложения после создания и скачивании:

- **app.py** - файл приложения, которое принимает данные, вычисляет и возвращает результат прогнозирования на основе обученной модели
- **const.py** - переменные в приложении.
- **methods.py** - методы, используемые в модели

- **model.py** - загружает модель из файлов model.pkl, model_vars_dict.pkl, в которых хранится модель и параметры.
- **preprocess_and_predict.py** - подготовка и вычисление переменных для прогнозирования.
- **run.sh** - командный интерпретатор для Linux.
- **run.bat** - командный интерпретатор для Windows.
- **requirements.txt** - зависимости для сборки приложения.
- **dockerfile** - файл конфигурации, в котором расписано пошаговое создание среды для работы приложения.
- **docker-compose.yml** - файл с командами для запуска среды приложения.

Приложение позволяет решать задачи предиктивной аналитики для новых данных с использованием обученной модели. Предназначено для использования в сторонних системах.

13. Работа с проектом

Сущность «Проект» реализована с целью объединить *группу пользователей* для работы над одним проектом. При этом проект может объединять в себе такие сущности как: «модель», «рабочая область», «дашборд», «отчет», «файл», «коннектор». О том, как добавить каждую из этих сущностей в проект, написано в рамках данного раздела.

- 13.1. Создание нового проекта

1. Перейдите в пункт меню «Проекты»:

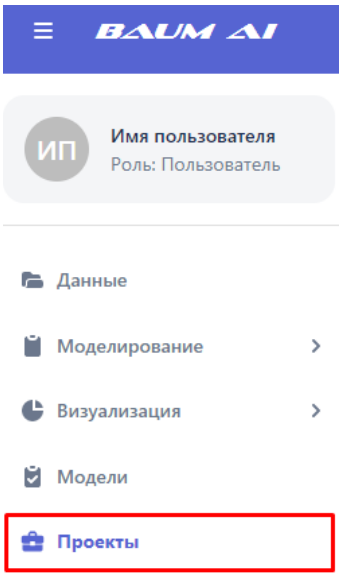


Рисунок 13.1 –Пункт меню «Проекты»

2. Откроется страница «Проекты», на которой отображаются все *проекты*, созданные *пользователем*:

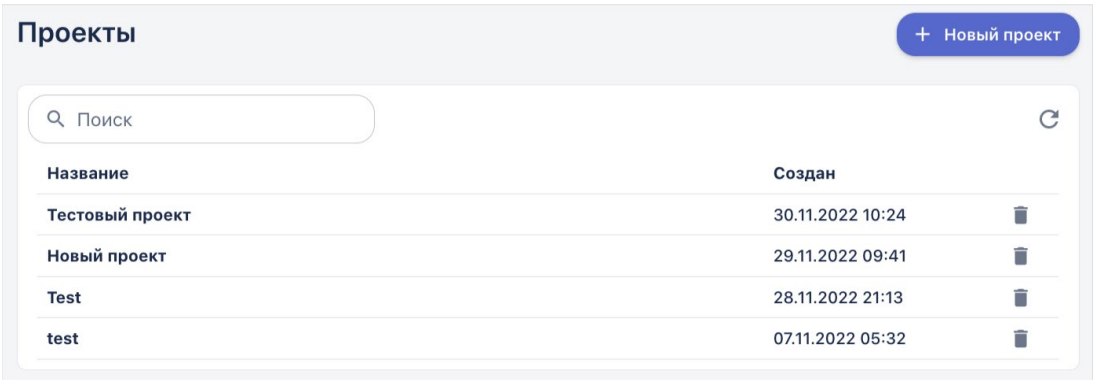


Рисунок 13.2 – Страница с проектами, доступными пользователю

3. Нажмите на кнопку «Новый проект», откроется окно создания нового проекта:

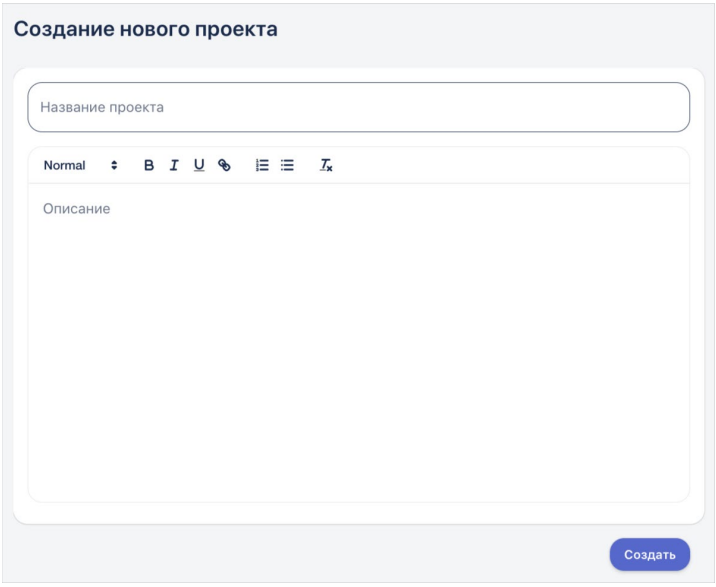


Рисунок 13.3 – Создание проекта

4. Задайте название проекта, например, «Проект тест» (обязательное поле)
5. Задайте описание проекта (необязательное поле), вы можете использовать средства форматирования текста:
    - Заголовки и подзаголовки
    - Жирный текст
    - Курсив
    - Подчеркивание
    - Вставка ссылки
    - Список
    - Нумерованный список
    - Кнопка очистки форматирования
6. Нажмите кнопку «Создать».
7. На страницу «Проекты» добавится новый проект.

- 13.2. Редактирование проекта

Пользователь имеет возможность отредактировать название проекта и его описание:

1. На странице «Проекты» перейдите в проект, который требует редактирования
2. В открывшемся окне в верхнем правом углу нажмите кнопку «Редактировать» :

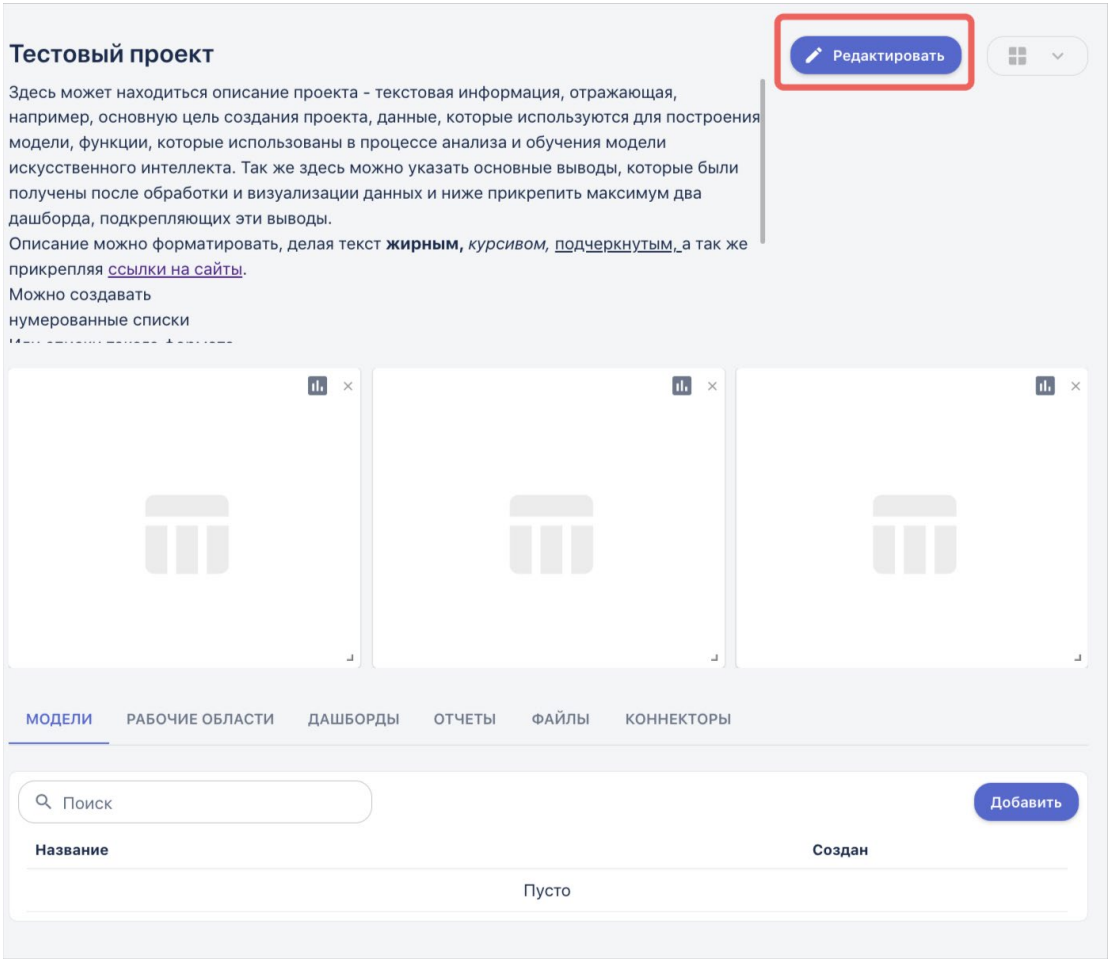


Рисунок 13.4 – Переход к редактированию существующего проекта

3. Задайте новое название проекта и его описание, затем нажмите кнопку «Сохранить»

4. Система вернется на страницу со списком проектов, для того чтобы посмотреть обновленное описание, перейдите в проект, нажав на его наименование.

- 13.3. Наполнение проекта

После того, как проект создан, его можно наполнить *содержимым* – теми сущностями, над которыми предстоит совместно работать группе пользователей. Для этого:

1. На странице «Проекты» перейдите по ссылке с названием созданного проекта, кликнув на его название:

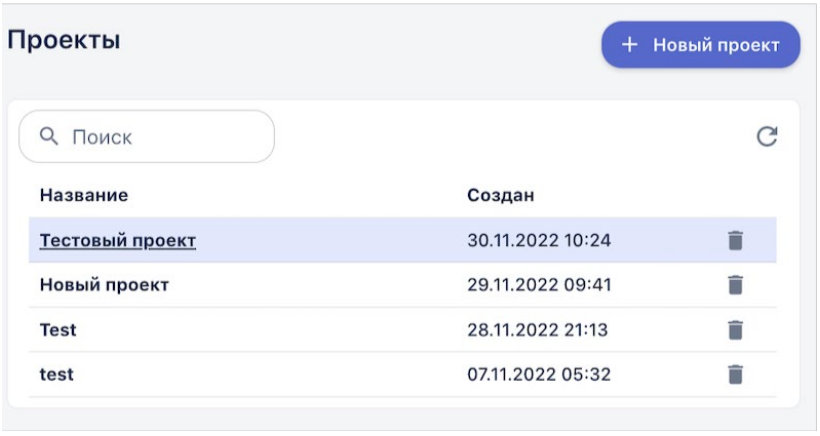


Рисунок 13.5 – Переход на страницу проекта для его просмотра и редактирования

2. Откроется *страница проекта* на первой вкладке «Модели»:

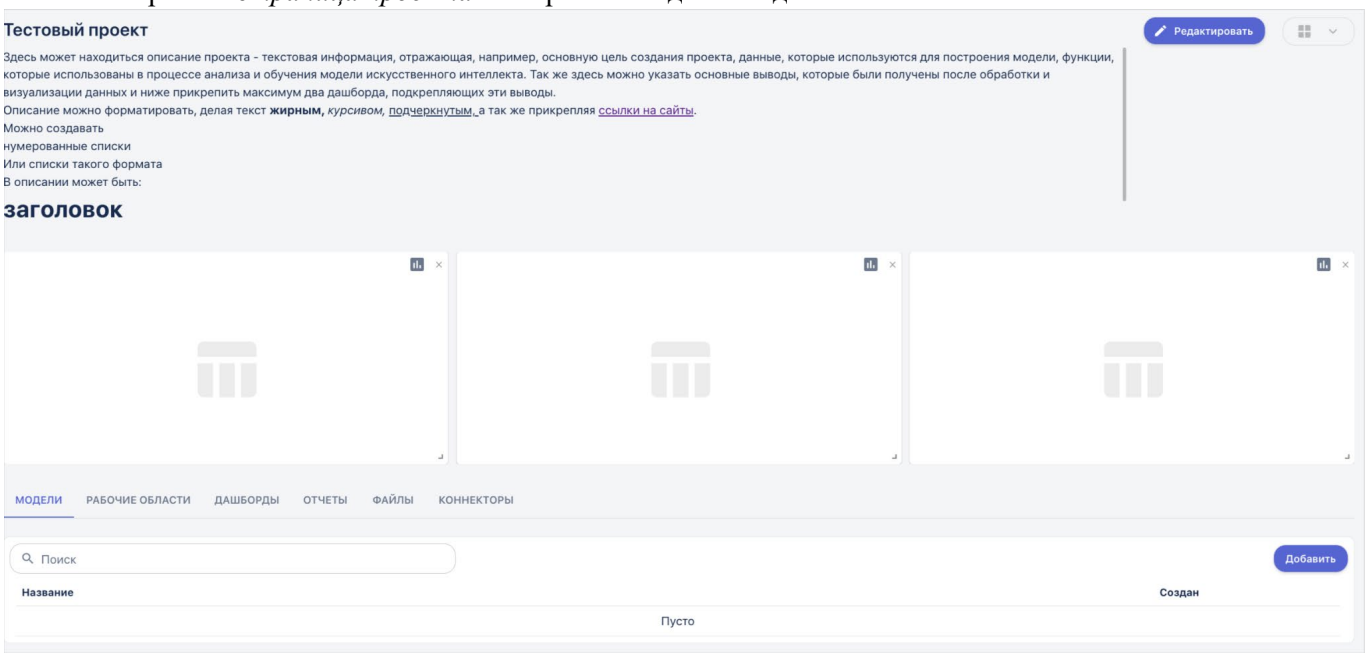
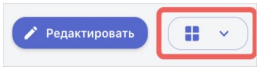


Рисунок 13.6 – Страница проекта

Целиком страница проекта состоит из вкладок с одноименными сущностями: модели, рабочие области, дашборды, отчеты, файлы, коннекторы.

3. **Работа с дашбордами.** Под описанием по умолчанию находятся три пустых дашборда. Вы можете сделать следующее в данном разделе:

   a. Удалить ненужные дашборды, для этого нажмите “x” в правом верхнем углу дашборда.



   b. Добавить новый дашборд. Для этого нажмите на кнопку «Добавить дашборд» и в выпадающем списке выберите нужный тип визуализации:

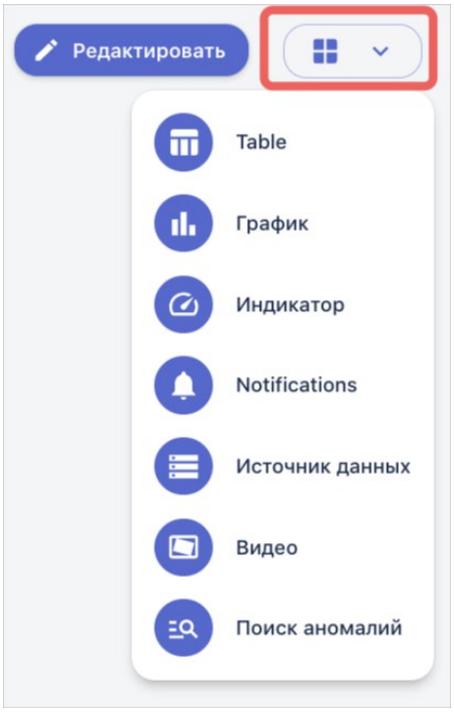


Рисунок 13.7 – Выбор типа визуализации

После этого под описание проекта добавится новый пустой дашборд выбранного типа

Рисунок 13.8 – Выбор коннектора

После этого визуализация отобразится на дашборде.

4. **Добавление сущности «Модель»**. На вкладке «Модели» пользователь нажимает кнопку «Добавить», и открывается выпадающий список со всеми моделями ИИ, созданными в Системе. Выбирается модель и добавляется в проект.

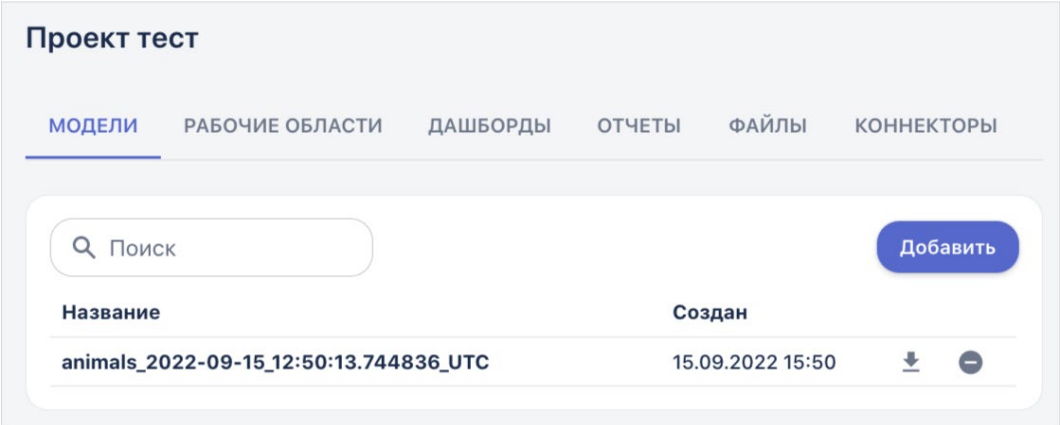


Рисунок 13.9 – Загруженная в проект модель

5. **Добавление сущности «Рабочая область»**. Перейдите на вкладку «Рабочие области» и нажмите кнопку «Добавить». Из выпадающего списка выберите рабочую область для добавления в проект:

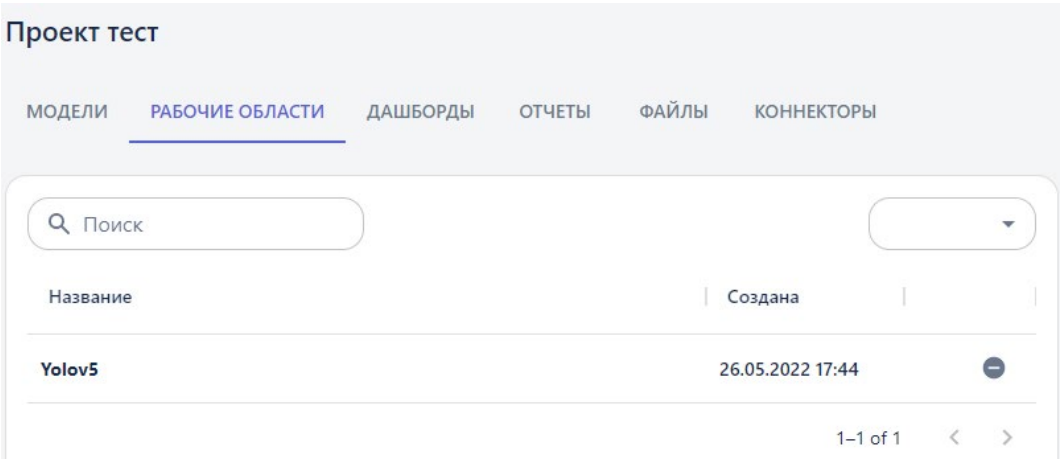


Рисунок 13.10 – Вкладка «Рабочие области»

6. **Добавление сущности «Дашборд»**. Перейдите на вкладку «Дашборды» и нажмите кнопку «Добавить», после чего выберите дашборд из выпадающего списка:

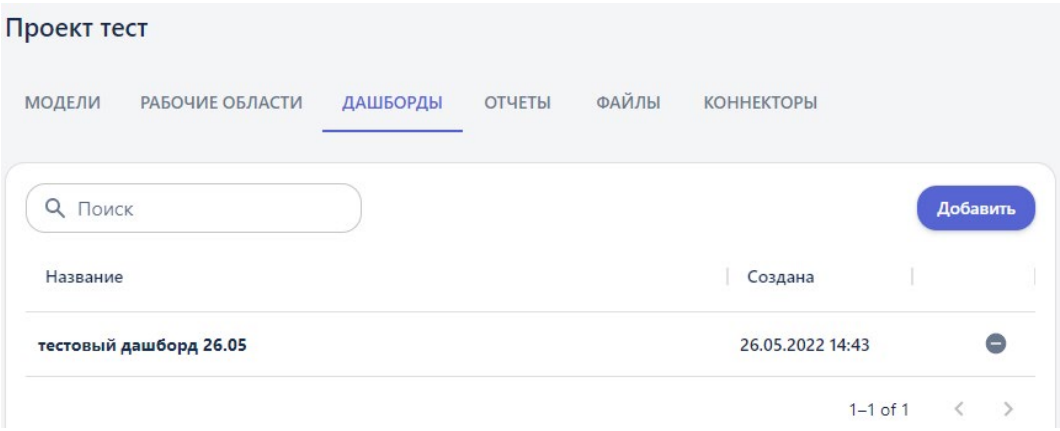


Рисунок 13.11 – Вкладка «Дашборды»

7. **Добавление сущности «Отчет»**. Перейдите на вкладку «Отчеты» и нажмите кнопку «Добавить». Из выпадающего списка выберите отчет для добавления в проект.

8. **Добавление сущности «Файл»**. Перейдите на вкладку «Файлы» и нажмите кнопку «Добавить». Из выпадающего списка выберите файл для добавления в проект:

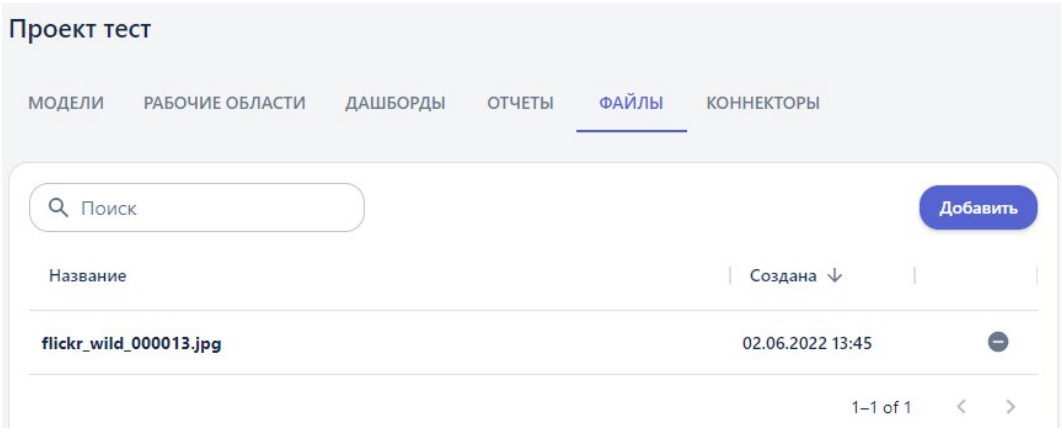


Рисунок 13.12 – Вкладка «Файлы»

9. **Добавление сущности «Коннектор»**. Перейдите на вкладку «Коннекторы» и нажмите кнопку «Добавить». Из выпадающего списка выберите коннектор для добавления в проект:

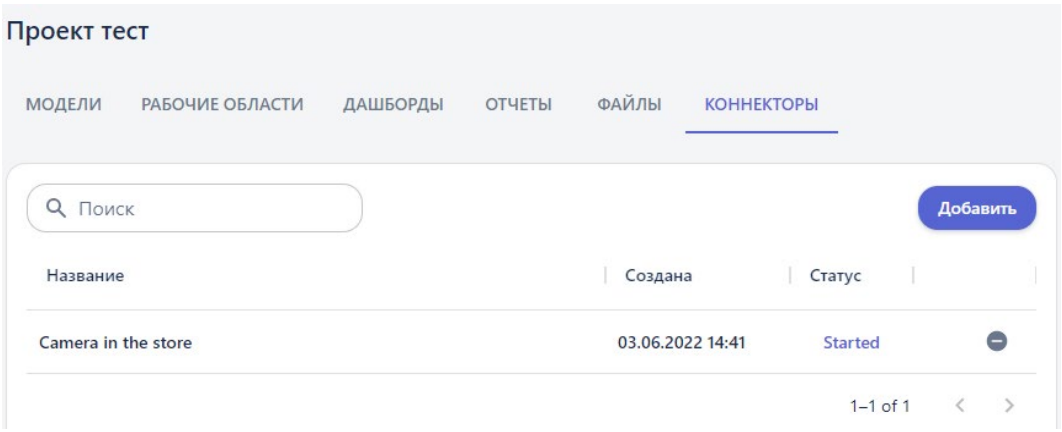


Рисунок 13.13 – Вкладка «Коннекторы»

14. Настройка подключения к источникам данных

В платформе «BAUM AI PREDICT» реализована возможность подключения к *внешним системам*, выступающим в качестве *источников данных* для Платформы. При этом данные в режиме реального времени могут поступать из следующих источников: БД (ClickHouse, PostgreSQL, MongoDB) и камеры видеонаблюдения. В данном разделе рассказывается, как настроить в Системе такие подключения. Подробно о визуализации информации из коннекторов на дашбордах написано в разделе «Работа с дашбордами»

Все подключения настраиваются в пункте меню «Соединения», где создаются сущности «Коннектор». *Коннекторы* объединяют в себе источник подключения и запрос на получение данных из него. В данном разделе описаны все типы коннекторов на платформе и сценарии работы с ними на примерах с тестовыми данными.

- 14.1 Типы коннекторов

1. **«ClickHouse»**

Данный коннектор предназначен для подключения к БД «ClickHouse». Настройка подключения описана выше в разделе Настройка подключения на примере ClickHouse.

2. **«PostgreSQL»**

Данный коннектор предназначен для подключения к БД «PostgreSQL». Настраивается по аналогии с коннектором «ClickHouse».

3. **«Mongo»**

Данный коннектор предназначен для подключения к БД «MongoDB». Настраивается по аналогии с коннектором «ClickHouse».

4. **«Table_app»**

На вход коннектора поступают *табличные данные* в онлайн режиме, например из БД «PostgreSQL». Чтобы анализировать входные данные используется обученная в системе *модель* и используется тип коннектора table_app.

5. **«Save_table»**

Данный коннектор предназначен для сохранения в Системе в виде файлов (на данный момент реализовано сохранение файлов в формате csv) табличных данных, поступающих из сторонних систем. Директория для сохранения файлов в Системе – это раздел «Данные».

В поле «Коннектор» указывается коннектор для подключения к таблице внешнего источника. Устанавливается галочка в поле «Постоянное обновление».

Для такого типа коннектора обязательно указать на выбор:

- Количество строк данных - количество строк из таблицы, которые будут сохранены.
- Интервал - промежуток времени в секундах, в течение которого коннектор должен собирать информацию из таблицы, по истечению этого времени загрузка прекратится и файл будет сохранен.

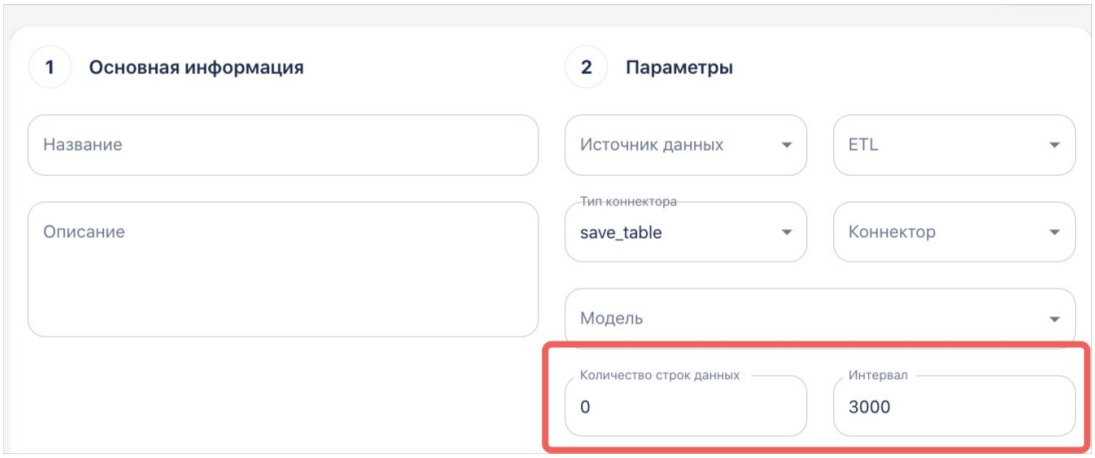


Рисунок 14.2 – Настройка коннектора «save_table»

6. «Classification_app»

Данный коннектор используется в задаче *классификации изображений*, где на вход коннектора для анализа подается серия из нескольких изображений. Пример настройки коннектора:

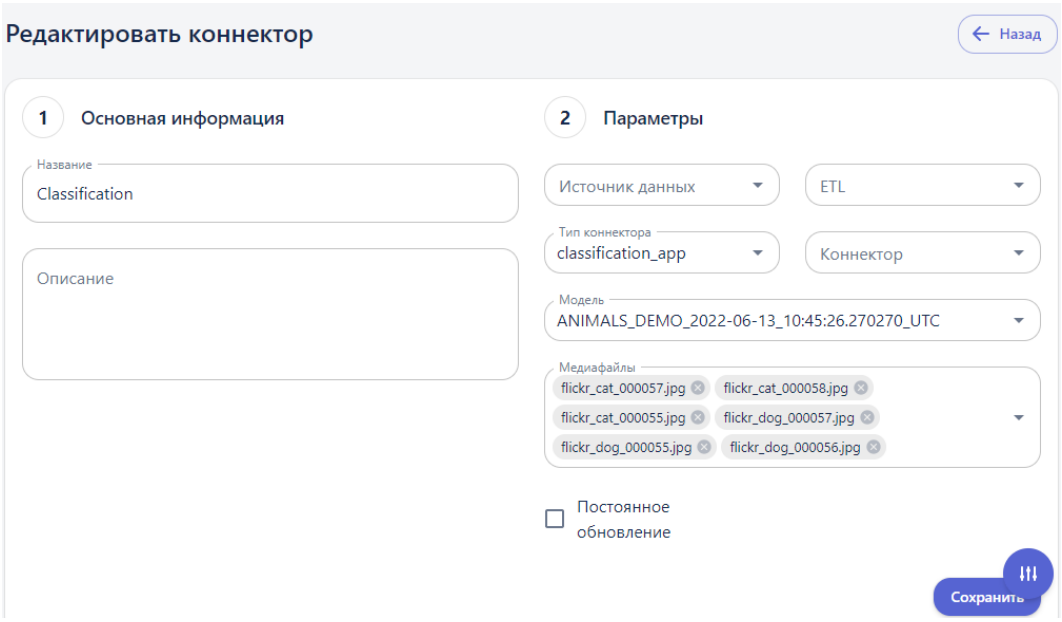


Рисунок 14.4 – Настройка коннектора «Classification_app»

В поле «Модель» выбирается обученная модель классификации изображений. В поле «Медиафайлы» – серия изображений, которые необходимо классифицировать с использованием обученной модели.
*Для данного типа коннектора не нужно устанавливать признак «Постоянное обновление», так как анализируются данные, загружаемые с локального устройства.

7. «Constructor» (автоматически создаваемый)

Условием создания коннектора является следующее: на Платформе создается блок-схема, где один из элементов имеет на выходе *визуализацию* – выходным параметром элемента является *таблица*, *график* и т.д. Пользователь запускает такую блок-схему, и после успешной отработки элемента с визуализацией в Системе создается коннектор. Число создаваемых коннекторов при запуске блок-схемы соответствует числу элементов с визуализацией на этой блок-схеме.

Название коннектора формируется из названия элемента и названия рабочей области, и должно являться уникальным в рамках Системы. Пример – *yolov5_train_(yolov5_noses_eyes)*, где название рабочей области указано в скобках.

Такой коннектор автоматически создается в статусе «Started» – пользователь не должен запускать коннектор, и может сразу же перейти к просмотру данных коннектора в окне дашборда.

- 14.2 Порядок работы с коннекторами

Предварительно коннектор должен быть создан и запущен (за исключением служебных коннекторов, которые создаются и запускаются автоматически). Только после этого выполняется подключение к нему через окно дашборда.

■ 14.2.1 Создание коннектора

Создание коннекторов осуществляется в разделе меню «Соединения». Для коннекторов, в которых настраивается подключение к внешним источникам данных (к внешним базам данных, к камере видеонаблюдения), дополнительно создаются сущности – *источник данных*, и *ETL*. Для остальных типов, эти сущности не создаются, а сразу создается сущность «Коннектор».

Для того чтобы создать новый коннектор, перейдите на вкладку «Коннекторы» и нажмите на кнопку «Создать коннектор»:

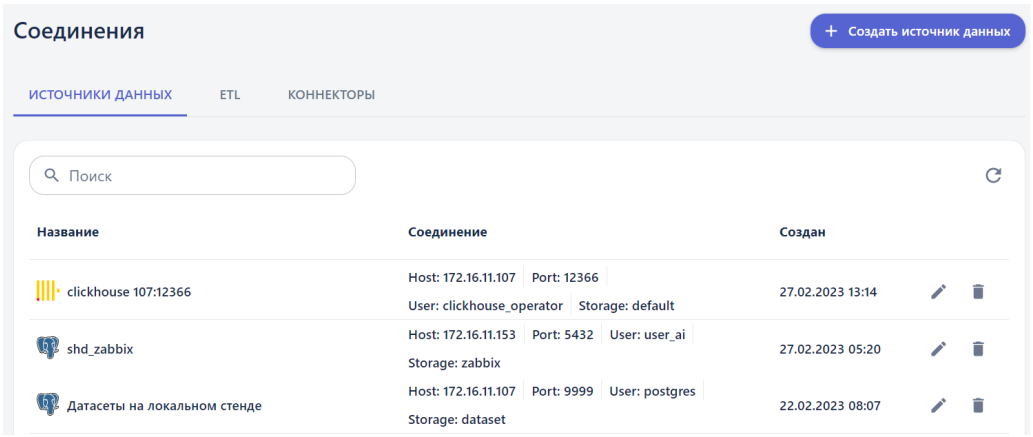


Рисунок 14.5 – Переход к созданию коннектора на вкладке «Коннекторы»

В открывшейся форме выберите тип коннектора в одноименном поле «Тип коннектора», и заполните поля:

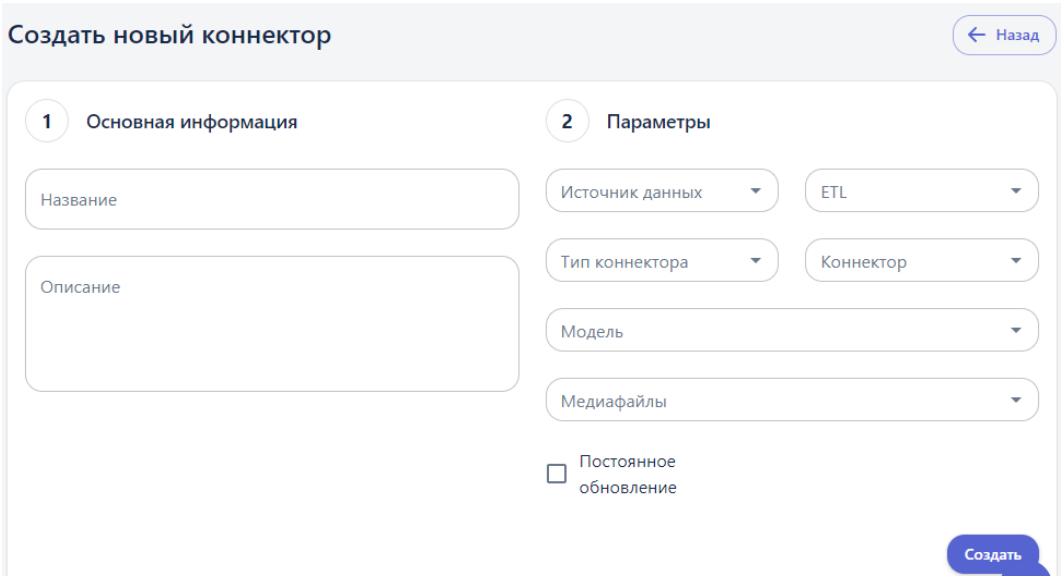


Рисунок 14.6 – Форма создания нового коннектора

После заполнения формы нажмите кнопку «Создать».

■ 14.2.2 Запуск коннектора

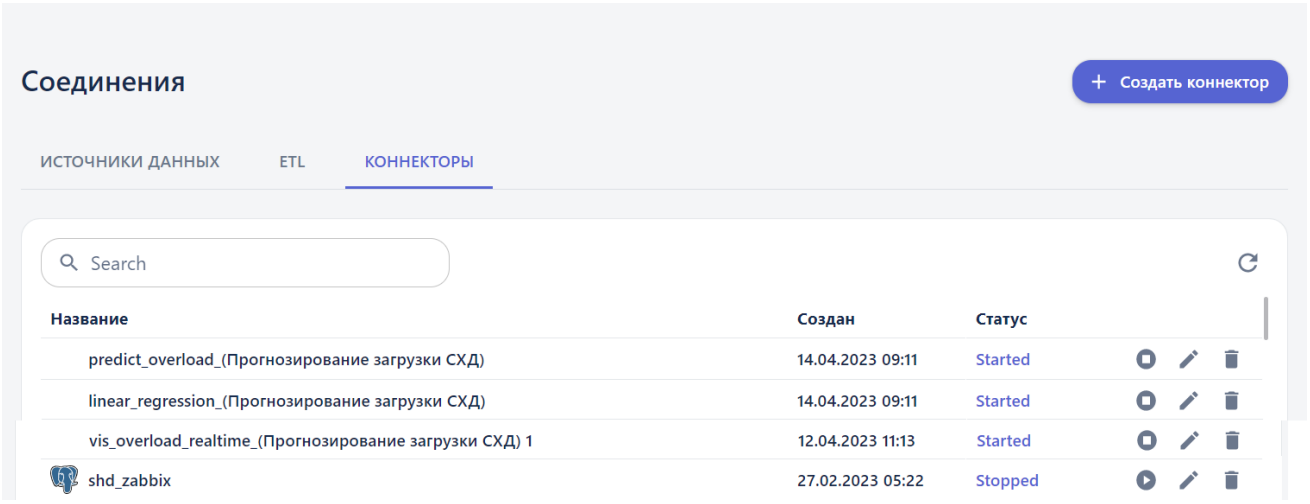


Рисунок 14.7 – Запуск коннектора

В результате коннектору присваивается статус «Started», и он готов к визуализации на дашборде.

■ 14.2.3 Подключение к коннектору на дашборде

Откроется модальное окно со списком коннекторов для подключения.

- 14.3 Настройка подключения на примере ClickHouse

Действия:

1. Перейдите в пункт меню «Соединения»:

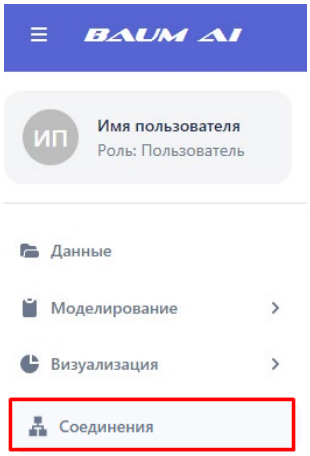


Рисунок 14.8 – Пункт меню Соединения

Откроется страница «Соединения» на первой вкладке «Источники данных», на которой отображаются все ранее созданные источники:

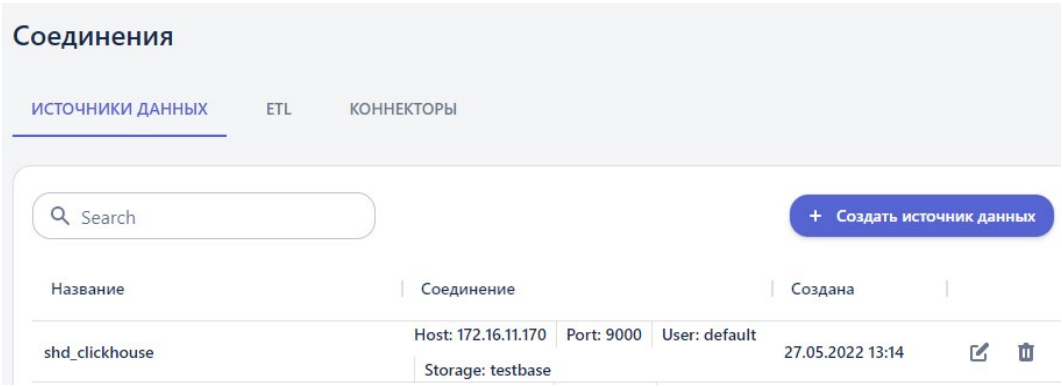


Рисунок 14.9 – Вкладка «Источники данных»

2. **Создание нового источника**. Нажмите на кнопку «Создать источник данных». Откроется окно «Создание нового источника данных»:

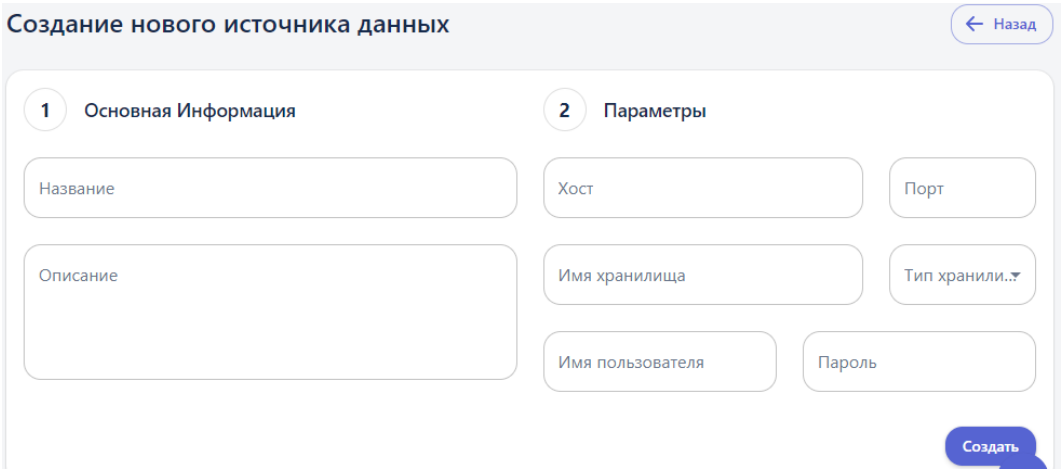


Рисунок 14.10 – Окно настройки источника данных

Заполните поля:

- *Название*. Пользователь задает название источника «TEST DATA SOURCE (clickhouse)», к которому будет настраиваться подключение.
- *Хост*. Указывается хост протокола TCP/IP, т.е. сетевой интерфейс устройства, предоставляющего сервис формата «клиент-сервер», где сервером выступает БД **ClickHouse**, а клиентом – платформа **BAUM AI PREDICT**. По сути это IP-адрес подключаемой БД. Необходимо указать «172.16.11.116».
- *Порт* – номер порта, по которому устанавливается соединение с сервером, на котором установлена БД **ClickHouse**. Указать «12366».
- *Имя хранилища* – название базы данных, которое указано на подключаемом сервере. Указать «default».
- *Тип хранилища*. Из выпадающего списка необходимо выбрать тип «clickhouse»:

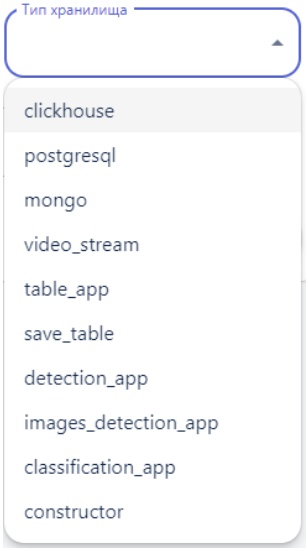


Рисунок 14.11 – Список возможных типов хранилища

Описание всех типов коннекторов представлено в разделе Классификация коннекторов

- *Имя пользователя, пароль* – параметры учетной записи администратора внешнего сервера для разрешения доступа к данным. Указать пользователя «clickhouse_operator», и пароль для него «clickhouse_operator_password».
- *Описание.* Вводится дополнительная информация по источнику, необязательное поле.

Для регистрации в Системе источника нажмите кнопку «Создать».

3. **Создание нового ETL**. Сущность «ETL» (дословно Extract, Transform, Load – с англ. извлечение, преобразование загрузка) содержит в себе sql запрос для извлечения данных из источника. То есть в шаге 2 создается источник и прописывается запрос для извлечения данных из него.

   3.1. Перейдите на вкладку «ETL»:

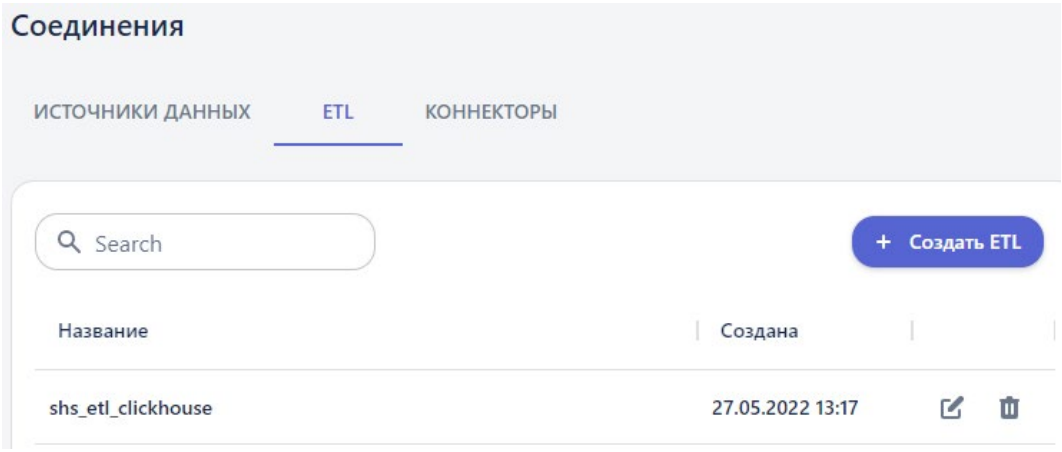


Рисунок 14.12 – Вкладка ETL

3.2. Нажмите на кнопку «Создать ETL». Откроется окно «Создать новый ETL»:

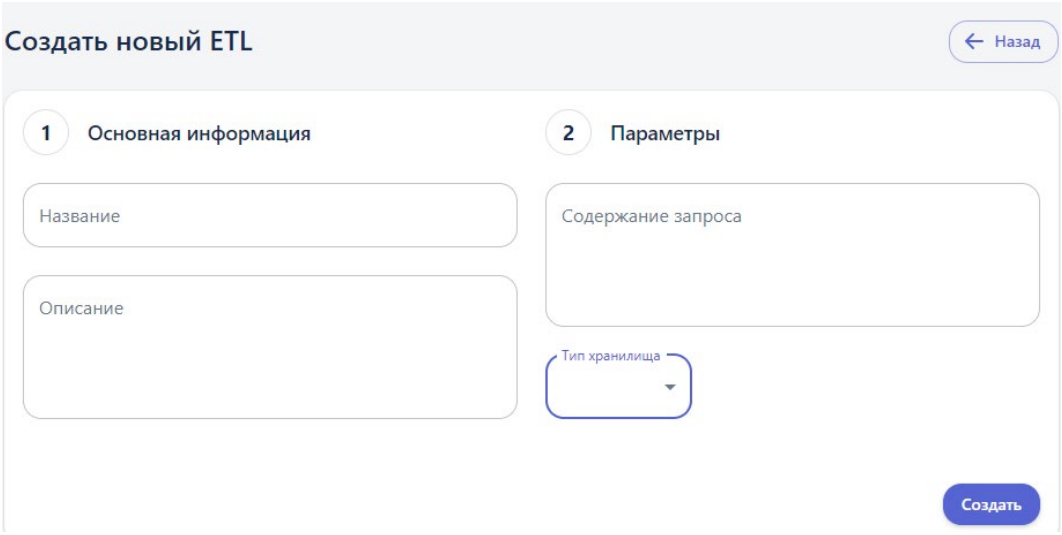


Рисунок 14.13 – Окно настройки ETL

3.3. Заполните поля:

- ➔ *Название. Пользователь вручную задает название ETL «TEST ETL (clickhouse)» – запрос на извлечение данных.*
- ➔ *Содержание запроса. Прописывается непосредственно sql запрос для извлечения данных из внешнего сервера. При этом указывается название таблицы, из которой данные извлекаются, в запросе «SELECT * FROM stock». Чтобы извлечь данные только из первых ста строк этой таблицы используется запрос «SELECT * FROM stock LIMIT 100».*
- ➔ *Тип хранилища. Выбирается тип «clickhouse».*
- ➔ *Описание (необязательное поле).*

3.4. Нажмите кнопку «Создать».

4. **Создание нового коннектора**:

4.1. Перейдите на вкладку «Коннекторы»:

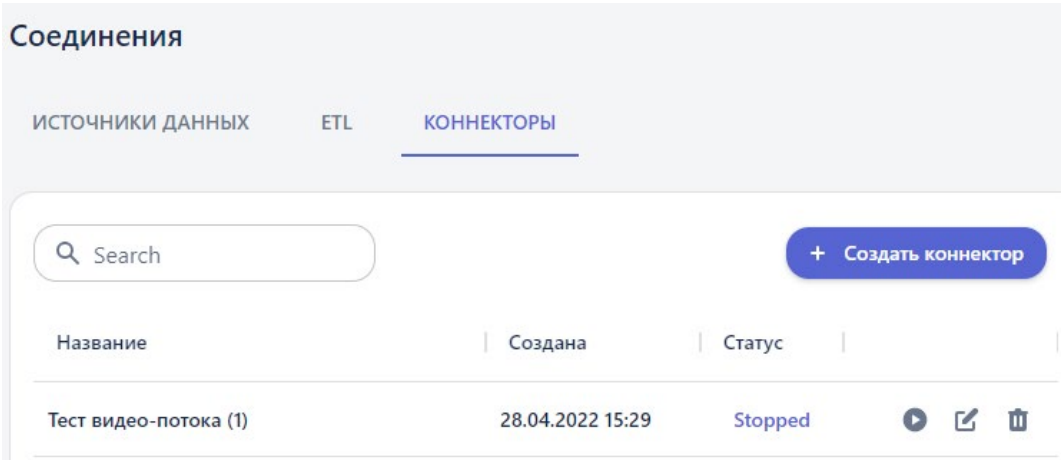


Рисунок 14.14 – Вкладка коннекторов

4.2. Нажмите на кнопку «Создать коннектор». Откроется окно «Создать новый коннектор»:

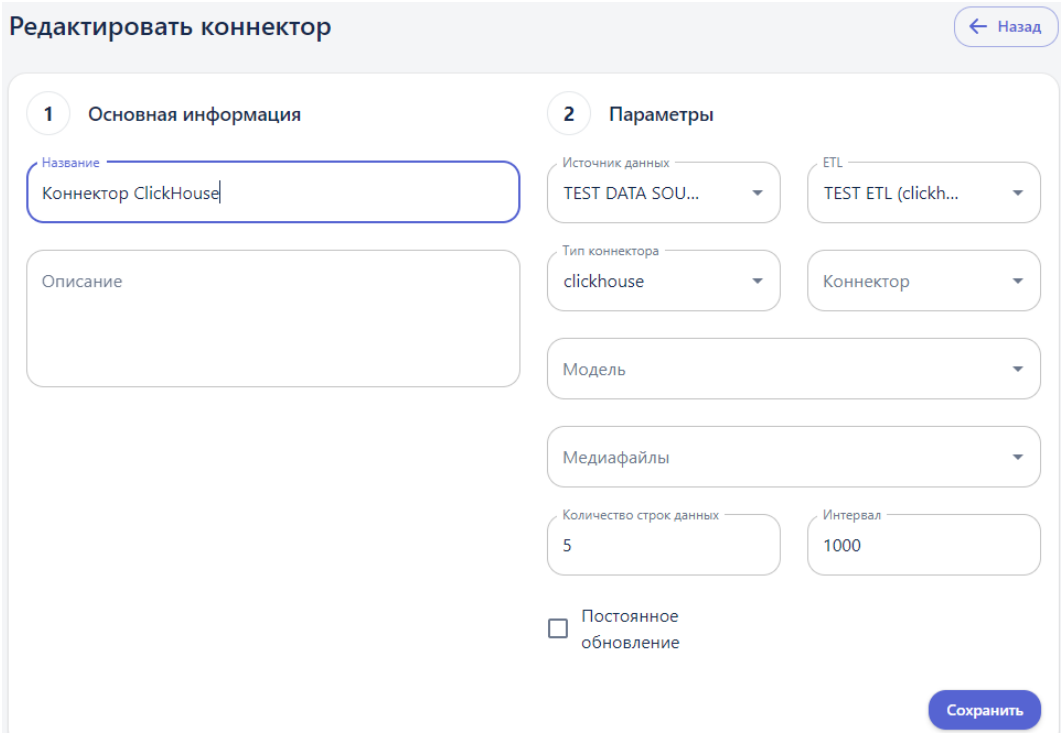


Рисунок 14.15 – Окно настройки коннектора

4.3. Заполните поля:

➔ *Название.* Пользователь вручную задает название создаваемого коннектора «Коннектор ClickHouse».

➔ *Источник данных.* Из списка выбирается источник «TEST DATA SOURCE (clickhouse)», созданный в шаге 2.

➔ *ETL.* Из списка выбирается ETL «TEST ETL (clickhouse)», созданный в шаге 3.

➔ *Тип коннектора.* По умолчанию выбирается первый тип из списка – clickhouse, оставить выбранное значение.

➔ *Количество строк данных.* Данные из внешней БД поступают порциями, по указанному или меньшему количеству строк за раз.

➔ *Интервал* – периодичность, с которой выполняются запросы во внешнюю БД. Заполняется числовым значением (в миллисекундах), или значением в формате «Дата» (*второй вариант не реализован в текущей версии*). Если задать интервал 1000 мс, и указать количество строк пять, каждую секунду будет запрашиваться пять записей.

➔ *Постоянное обновление*. Признак устанавливается, когда данные ожидаются бесконечно. Если признак не установить, запрос данных завершится, при получении их в полном объеме.

➔ *Описание.*

Остальные поля на форме создания коннектора для текущего сценария не заполняются, они используются при создании других типов коннекторов. Нажмите кнопку «Создать».

4.4. Сразу после создания коннектору присваивается статус «Stopped»:

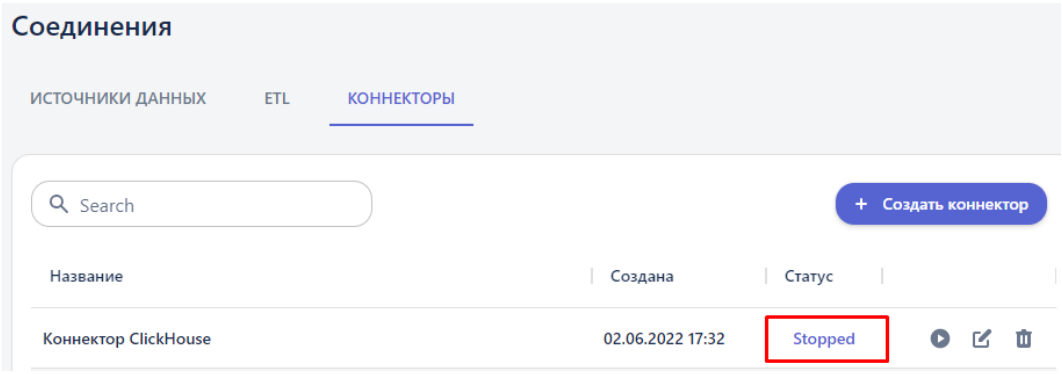


Рисунок 14.16 – Статус запуска коннектора

Вы так же можете отображать табличные данные ClickHouse в режиме реального времени, используя для визуализации сущность «Дашборд» и подключаясь к запущенному коннектору.

15. Примеры рабочих областей

15.1.1. Описание проблемы

В работе оператора систем хранения данных возникает необходимость отслеживания состояния заполнения томов данных. При приближении заполнения тома к задаваемому заранее пороговому значению, оператор должен принимать решение об удалении ненужных данных или о вводе в строй новых томов для сохранения информации.

■

15.1.2. Исходные данные

15.1.2.1. Формат данных.

■ Формат входных данных алгоритма - поток данных с СХД по SNMP 123 .

Формат: Дата, Заполнение тома, Том, Время в сек (с 00:00 01.01.1970 года). Ключевые поля подчеркнуты

15.1.2.2. Пример данных из СХД

2022-06-02 20:15:00,0.9748873805999756,Logicalused by volume: pool1/vol1,1654200900
2022-06-02 20:20:00,0.974887731552124,Logicalused by volume: pool1/vol1,1654201200
2022-06-02 20:25:00,0.9748881130218505,Logicalused by volume: pool1/vol1,1654201500
2022-06-02 20:30:00,0.9748884716033936,Logicalused by volume: pool1/vol1,1654201800
2022-06-02 20:35:00,0.9748888301849366,Logicalused by volume: pool1/vol1,1654202100
2022-06-02 20:40:00,0.9748892040252686,Logicalused by volume: pool1/vol1,1654202400
2022-06-02 20:45:00,0.9748895473480225,Logicalused by volume: pool1/vol1,1654202700

15.1.3. Постановка задачи

Необходимо на базе платформы реализовать механизм для автоматизации процесса отслеживания состояния заполнения тома и уведомления оператора в случае заполнения тома выше определенного значения. При этом должны быть соблюдены следующие условия:

1) Для решения задачи отсутствует необходимость владения навыками программирования для обучения модели классификации. Специалист должен иметь возможность использования предобученной модели в режиме no-code.
2) Должен быть предусмотрен интерфейс для доступа к СХД через коннектор. который позволит получать данные в автоматическом режиме.

Этапы решения проблемы:

Этап 1

1. Создать коннектор для подключения к Zabbix в режиме реального времени
2. Создать рабочую область
3. Создать дашборд для отображения результата

Результат

1. Создан коннектор для подключения к Zabbix “**shd_zabbix**”;
2. Создана рабочая область “**СХД**”;
3. Создан дашборд “**СХД_zabbix**”.
4. Получен датасет логирования (буфер)

- 15.2 Практическая часть

15.1.1. Функциональные требования

Сервер с доступом в Интернет.

15.1.2. Проект

Для удобства организации и работы с сохраняемыми объектами, на платформе можно создать проект, к которому прикрепляются все относящиеся к нему объекты. При нажатии на название проекта откроется окно, в котором можно работать над сущностями внутри проекта: модели, рабочие области, дашборды, отчеты, файлы и коннекторы (см. рис. 2-1). Для добавления сущности в проект необходимо сначала создать её в соответствующем разделе системы.

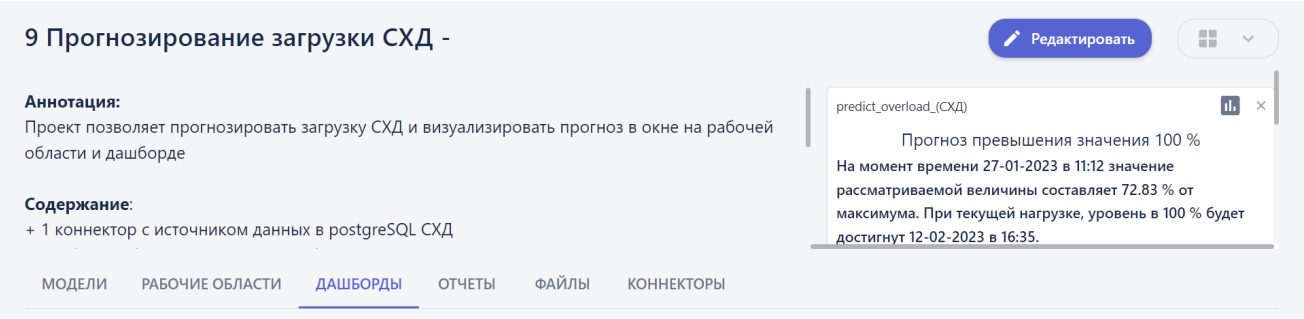


Рисунок 2-1 - Проект и его сущности

■

15.1.2.1. Рабочие области

15.1.2.1.1. Пайплайн для прогнозирования загрузки

Пайплайн предназначен для обучения модели прогнозировать нагрузку на СХД на основе анализа данных, принятых из коннектора, применяя функцию линейной регрессии.

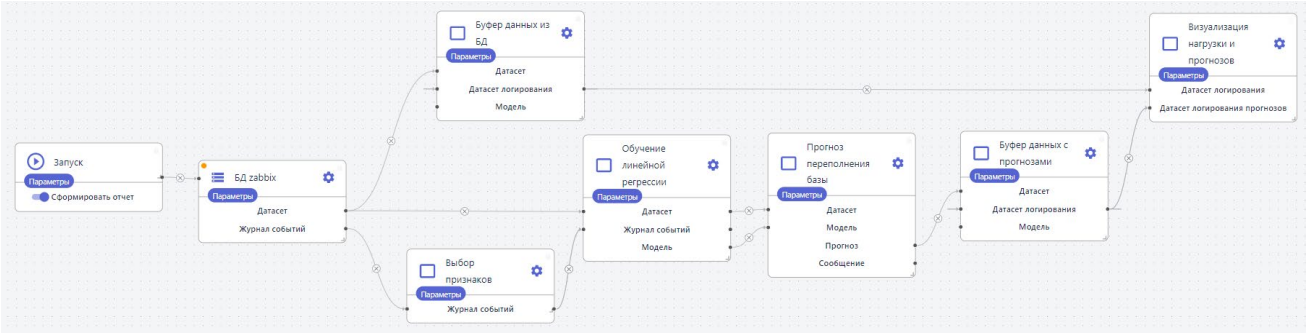


Рисунок 2.1.1-1 - Пайплайн для создания модели

Подробное описание блоков находится в Базе Знаний раздел “Моделирование”: [База знаний BAUM AI](#).

Параметры блоков:

- Блок “**Запуск**”. Блок является начальной точкой для пайплайна.

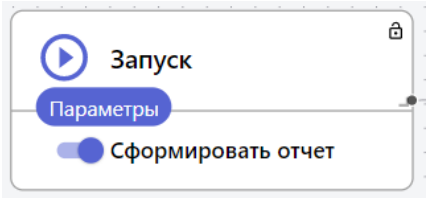


Рисунок 2.1.1-2 - Блок “Запуск”

Комментарий:

Кнопка “Сформировать отчет” появится после первого нажатия на запуск. Если активировать данную опцию, будет сформирован отчет об обучении модели предиктивной аналитики.

- Блок “**Загрузка табличных данных их коннектора**”. Блок реализует загрузку данных из коннектора **shz_zabbix**. При выборе опции “Сохранить датасет” - обязательно выбрать ниже директорию для сохранения (в данном случае **shd_test.csv**).

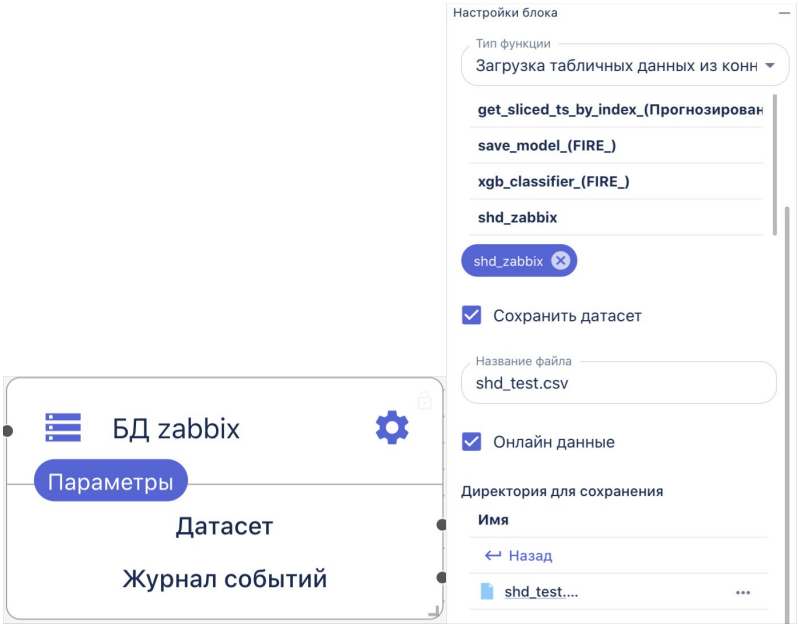


Рисунок 2.1.1-3 - Блок “Загрузка табличных данных их коннектора”

- Блок “**Выбор признаков**”. В блоке происходит выбор целевого признака (значение, которое предстоит научиться предсказывать модели) и признака (параметр, который будет исследоваться для выявляется корреляция между ним и рассматриваемым целевым признаком).

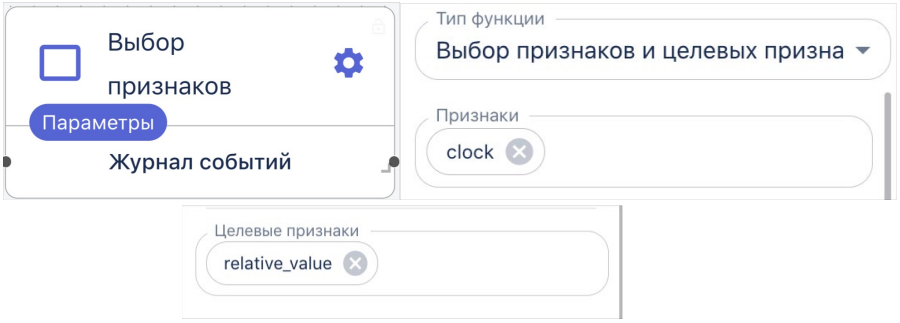


Рисунок 2.1.1-4 - Блок “Выбор признаков”

- Блок “**Обучение линейной регрессии**”. Блок прогнозирует целевую переменную (признак) на основании независимой переменной (признака).

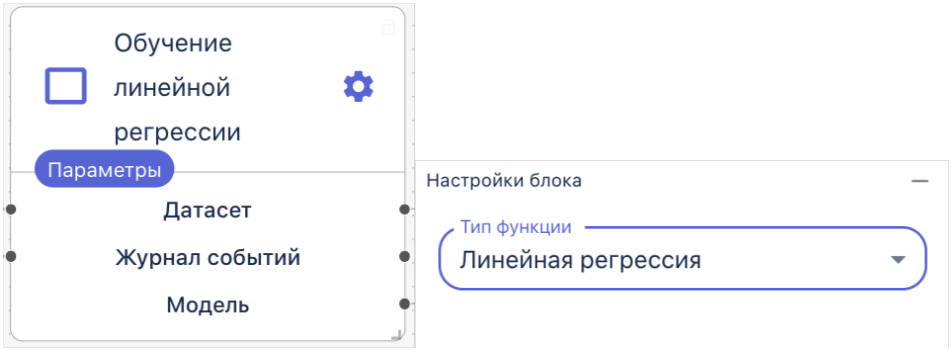


Рисунок 2.1.1-5 - Блок “Обучение линейной регрессии”

- Блок “**Буфер данных из БД**”. В данном блоке будет осуществляться логирование новой поступающей информации (**Датасет**) из коннектора в “растущий” буфер “**Датасет логирования**”.

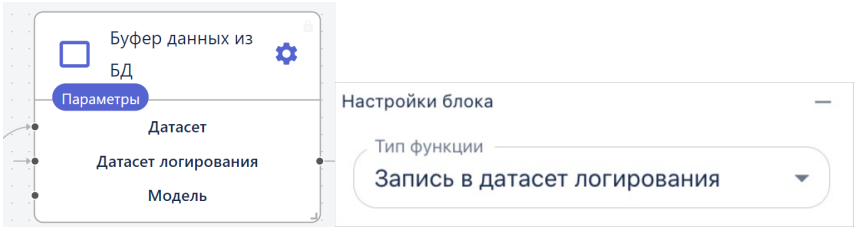


Рисунок 2.1.1-6 - Блок “Буфер данных из БД”

- Блок “**Прогноз переполнения базы**”. Блок будет осуществлять прогнозирование времени переполнения объема памяти с учетом заданного уровня, где 1 = 100%, то есть можно спрогнозировать, когда память заполнится полностью.

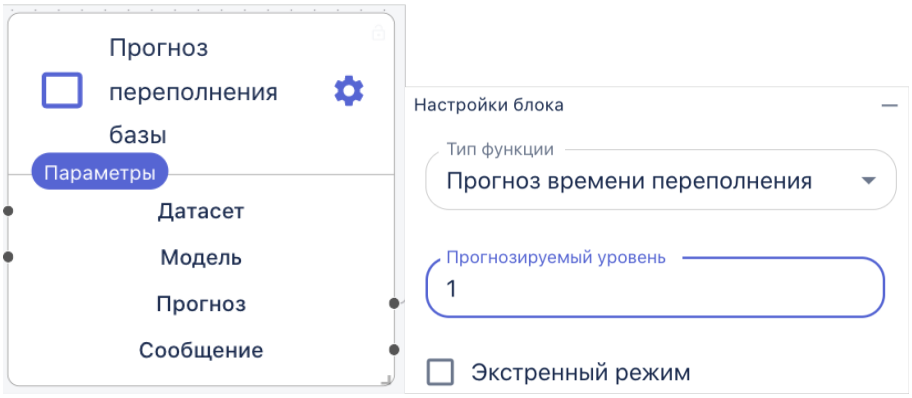


Рисунок 2.1.1-7 - Блок “Прогноз переполнения базы”

- Блок “**Буфер данных с прогнозами**”. В данном блоке будет

осуществляться логирование прогнозных значений в датасет.

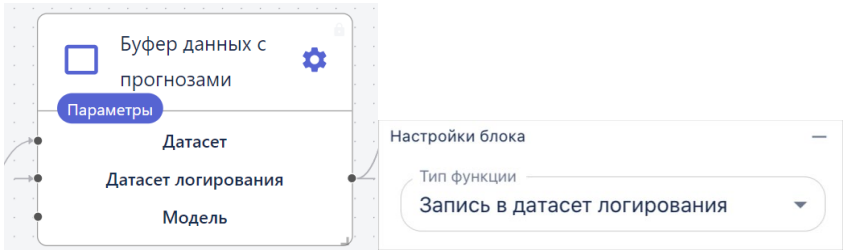


Рисунок 2.1.1-8 - Блок “Буфер данных с прогнозами”

- Блок “**Визуализация нагрузки и прогнозов**”. Блок позволяет создать визуализацию с помощью графика, на котором будут отображаться фактические и прогнозные значения заполнения памяти.

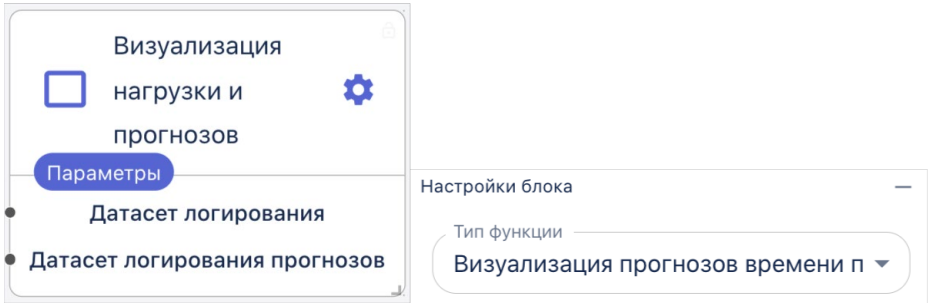


Рисунок 2.1.1-9 - Блок “Визуализация нагрузки и прогнозов”

Рисунок 2.1.1-10 - Блок “Визуализация нагрузки и прогнозов”. График “Прогноз превышения значения”

15.1.2.1.2. Визуализация результатов

После того как все элементы схемы будут успешно обработаны на панели инструментов появляются кнопки (см. рис. 2.1.2-1):

Рисунок 2.1.2-1 - Меню результатов работы

Последовательно нажимая на кнопки визуализации, можно отобразить на рабочей области следующую информацию:

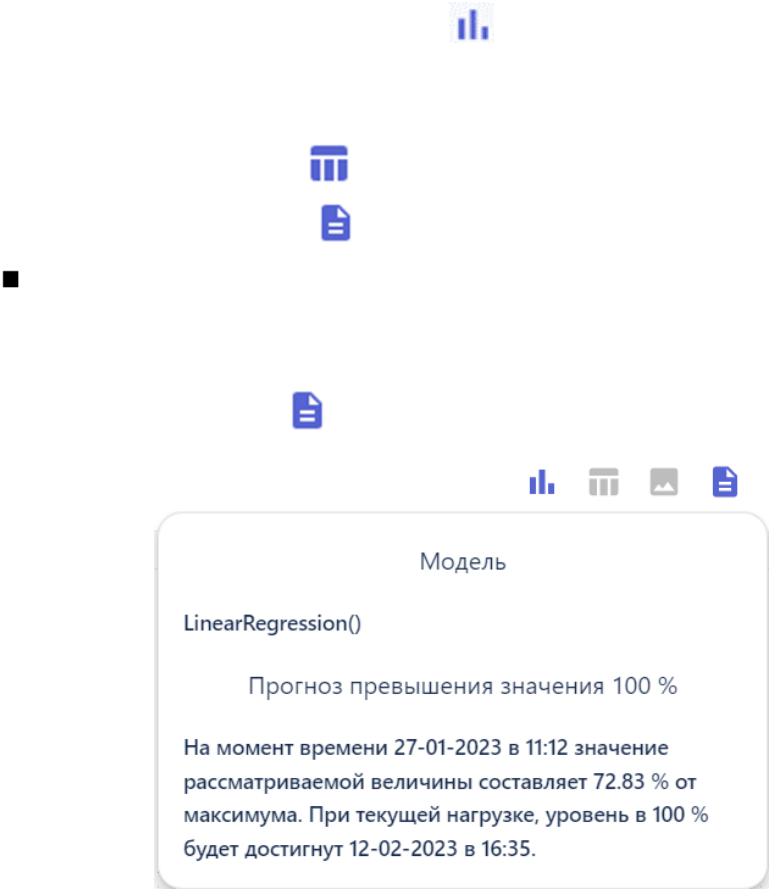


Рисунок 2.1.2-2 - Автогенерируемое описание

Более подробный отчет, включающий в себя дополнительно визуализацию результатов обучения модели, информацию по датасету и валидацию на тестовой выборке можно посмотреть в разделе “**Отчеты**”.

15.1.2.2. Дашборды

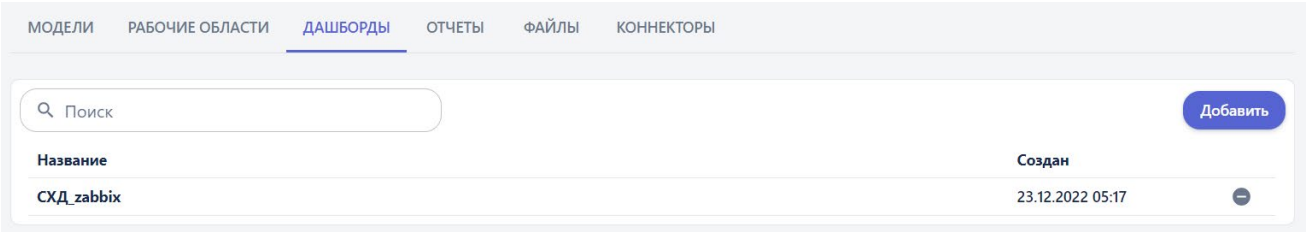


Рисунок 2.2-1 - Дашборд в проекте

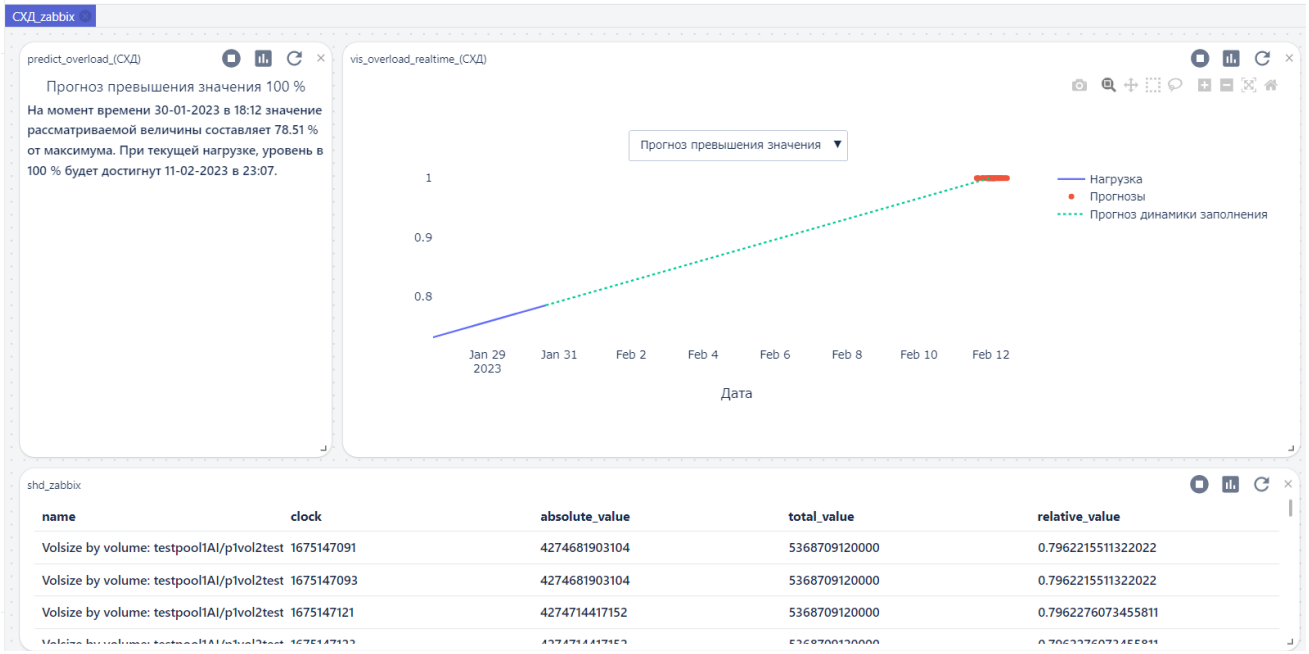


Рисунок 2.2-1 - Отображение информации на дашборде

15.1.2.3. Файлы

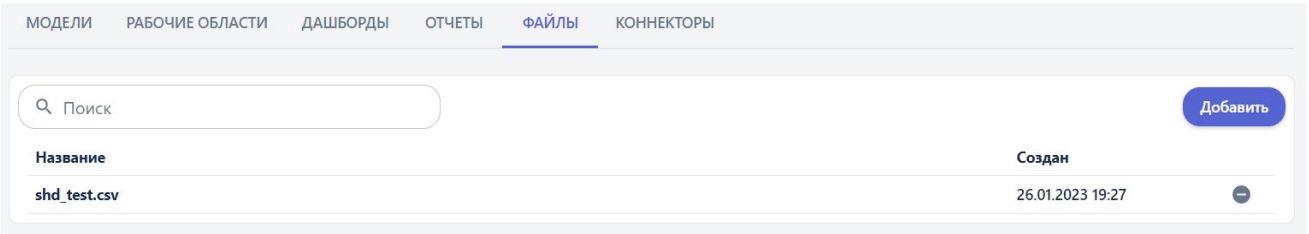


Рисунок 2.4-1 - Сохраняемый датасет в проекте

15.1.2.4. Коннекторы

Коннектор предназначен для создания соединения с внешней базой данных и позволяет получать информацию напрямую из системы по заданному интервалу. Соединение состоит из трех последовательно настраиваемых элементов: источник данных, ETL и коннектор.

15.1.2.5.1. Источник данных

Здесь настраивается соединение с сервером с указанием его параметров, а именно IP-адрес хоста, порт, имя хранилища, тип хранилища, имя пользователя и пароль (скрыт на рис. 2.5.1-1).

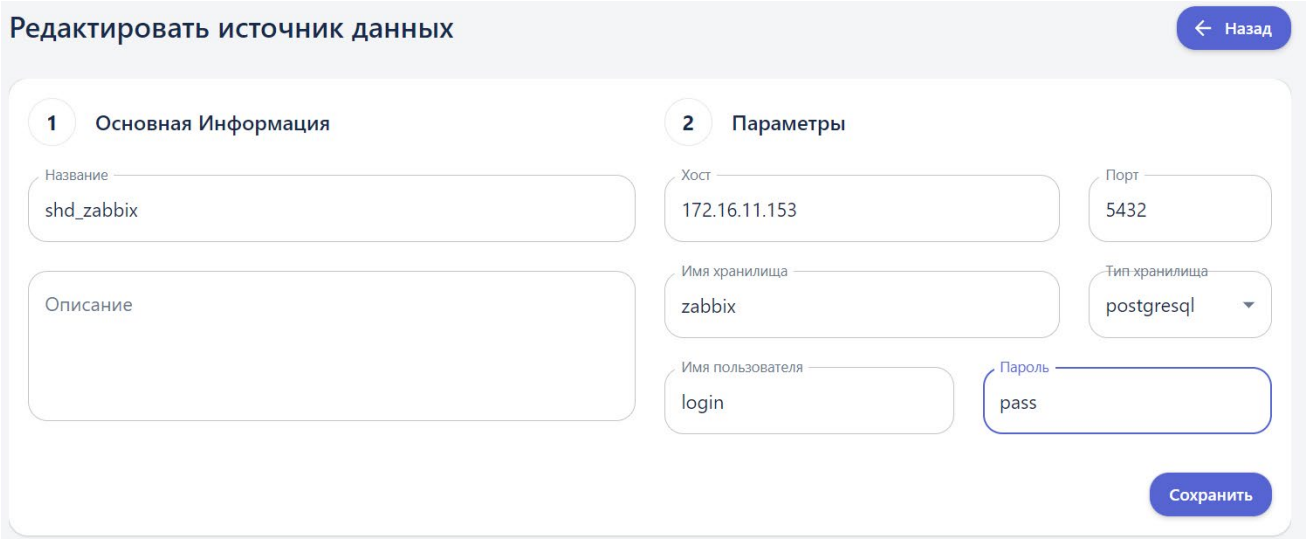


Рисунок 2.5.1-1 - Настройки “Источника данных”

15.1.2.5.2. ETL

Здесь прописывается SQL запрос для извлечения данных из источника, и указывается тип хранилища.

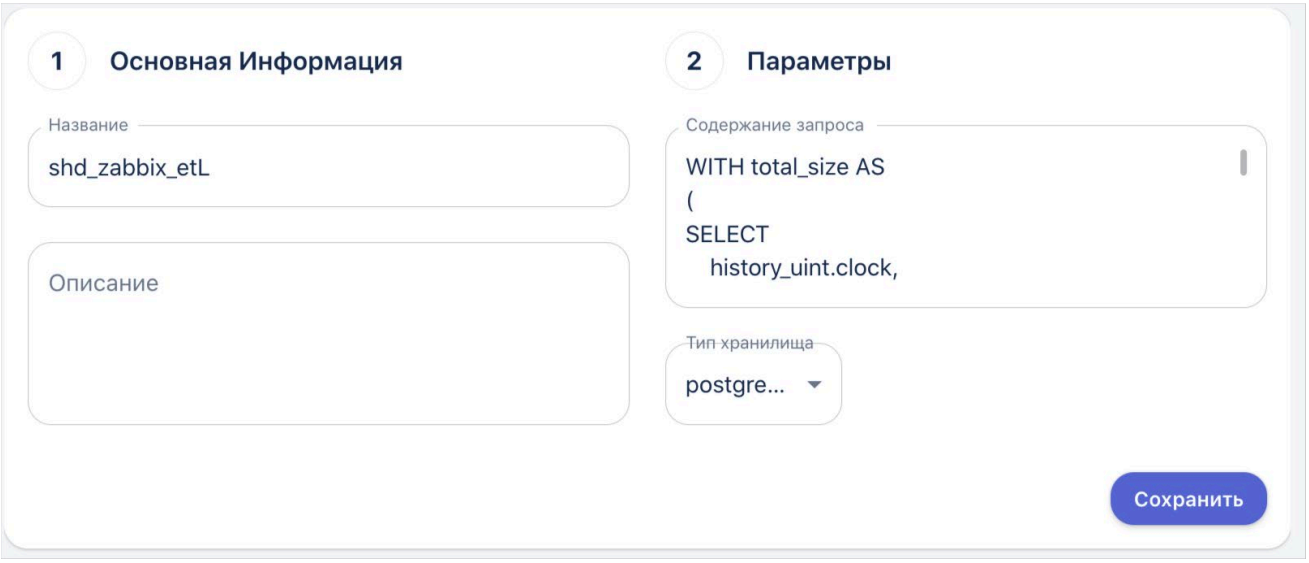


Рисунок 2.5.2-1 - Настройки “ETL”

Содержание запроса (название таблиц в запросе может изменяться в зависимости от названия таблиц у заказчика):

```
WITH total_size AS
(
SELECT
   history_uint.clock,
   history_uint.value AS total_value,
   items.name
FROM history_uint
JOIN items ON (history_uint.itemid = items.itemid)
```

```
        AND items.snmp_oid LIKE '.1.3.6.1.4.1.91919191.1.3.6.4.2.1.8.%'
        WHERE name LIKE '%testpool1AI/p1vol2testAI'
        LIMIT 1
        )

        SELECT
           total_size.name,
           used_size.clock,
           used_size.absolute_value,
           total_size.total_value,
           absolute_value / total_size.total_value AS relative_value
        FROM total_size, (
                        SELECT
                           history_uint.clock,
                           history_uint.value AS absolute_value,
                           items.name
                        FROM history_uint
                        JOIN items ON (history_uint.itemid = items.itemid)
                        AND items.snmp_oid LIKE '.1.3.6.1.4.1.91919191.1.3.6.4.2.1.9.%'
                        WHERE name LIKE '%testpool1AI/p1vol2testAI' AND clock > 1668508000) AS
used_size
        ORDER BY clock ASC
```

15.1.2.5.3. Коннектор

В коннекторе указывается уже настроенный ранее источник соединения данных и ETL и указывается тип хранилища.

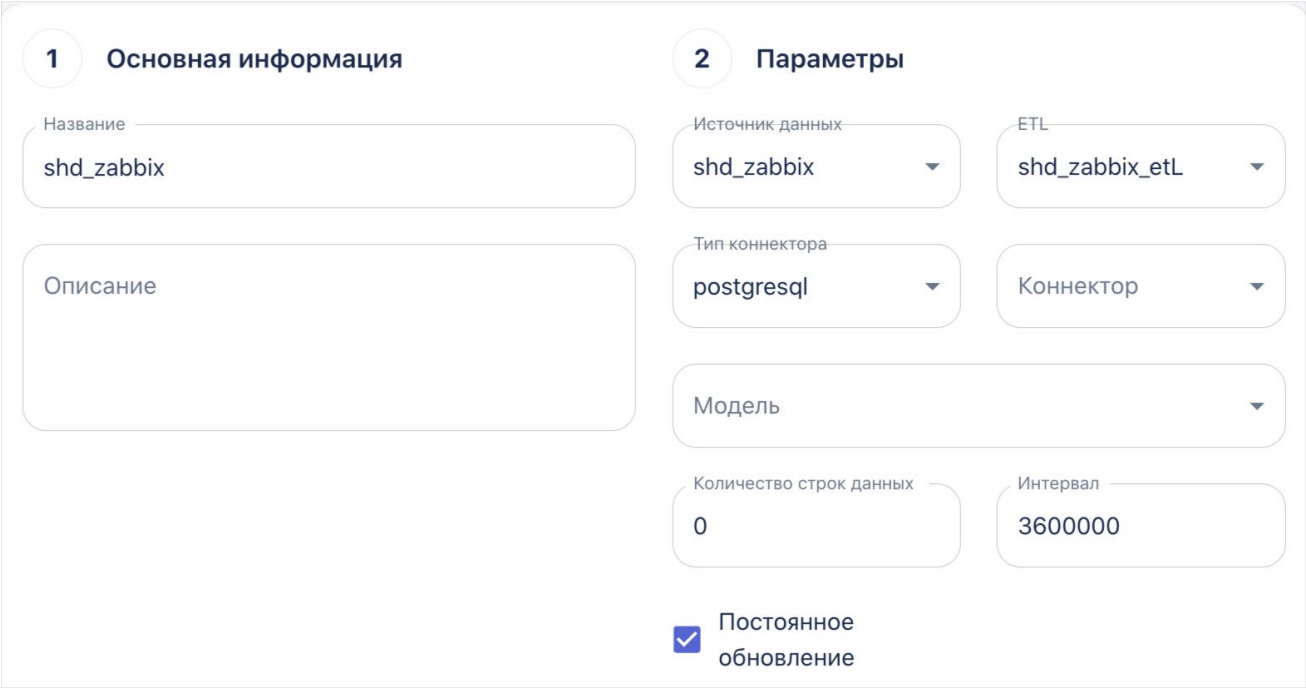


Рисунок 2.5.3-1 - Настройки “Коннектора”

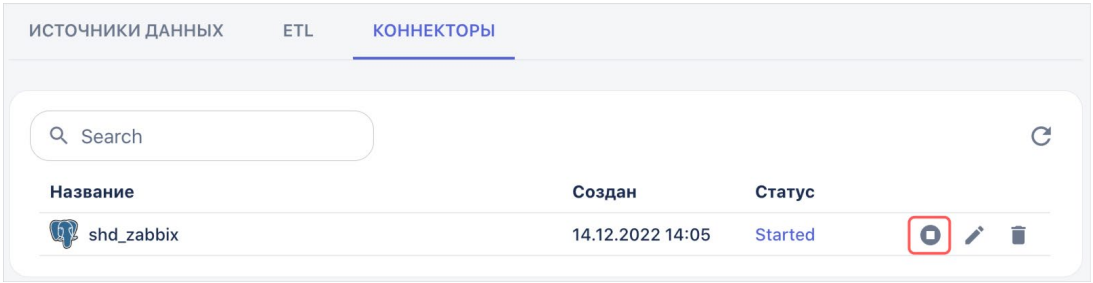


Рисунок 2.5.3-2 - Коннектора shd_zabbix

15.1.3. Результаты

На базе платформы создана рабочая область с пайплайном и дашбордом для отслеживания состояния загрузки СХД и превентивного принятия решения оператором.

15.1.4. Сохраняемые объекты

1. Коннектор;
2. Рабочая область;
3. Дашборд;
4. Датасет логирования.

16. Администрирование Платформы

- 16.1 Пользователи и группы

Платформа позволяет разделять уровни доступа к разделам меню для разных пользователей в зависимости от требования проекта. Для этого создаются Группы, которые впоследствии назначаются отдельным пользователям.

Создание новой Группы или редактирование уже созданной группы осуществляется в разделе Администрирование -> Группы:

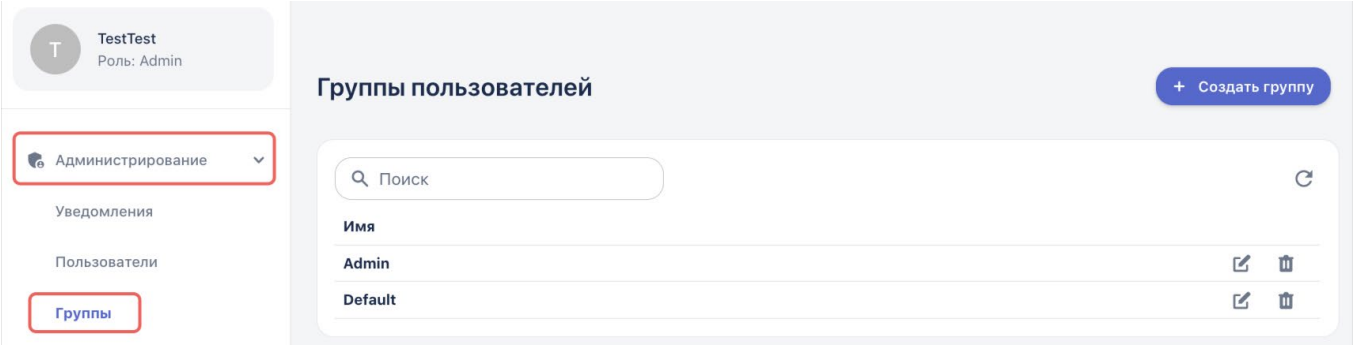


Рисунок 16.1 – Раздел Группы меню Администрирование

В списке отображаются уже созданные группы. Для создания новой группы, нажимается кнопка «Создать группу» в правом верхнем угле, в открывшемся окне отобразятся все доступные для настройки параметров:

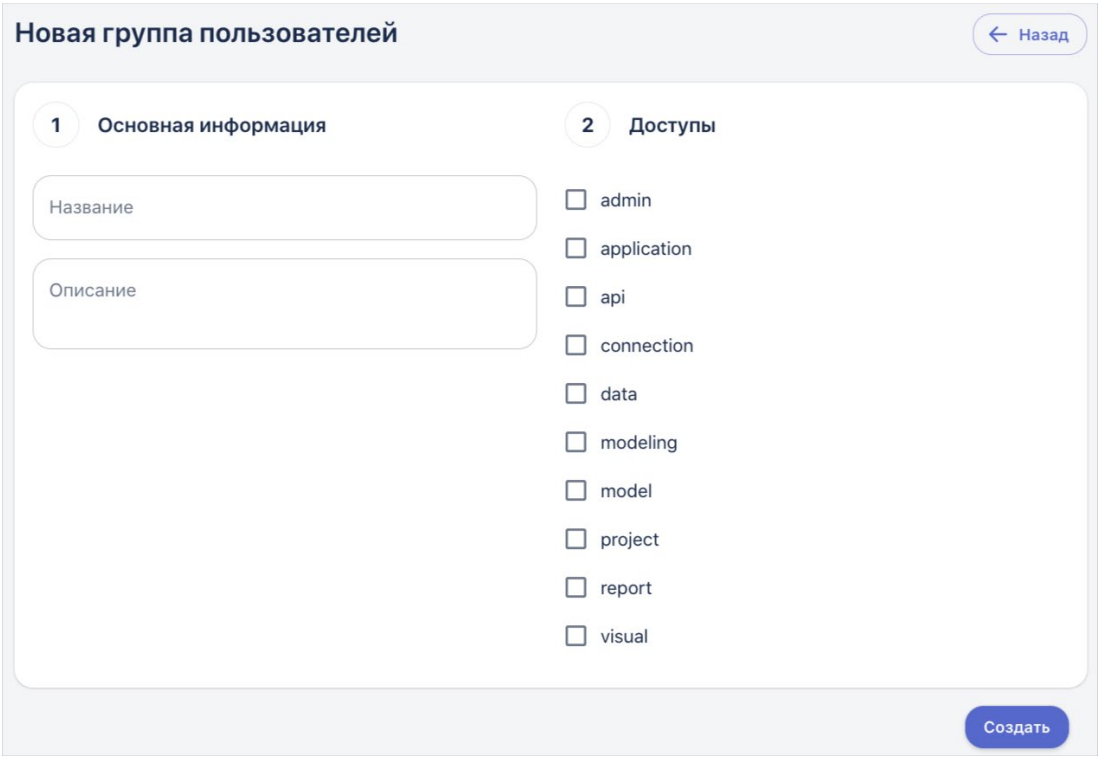


Рисунок 16.2 – Настройка новой группы пользователей

Сначала заполняется основная информация:

- *Название* - задать уникальное название для Группы (обязательное поле)

- *Описание* - внести краткую информацию о применимости данной роли (необязательное поле)

Далее из списка доступов выбираются разделы данных, к которым у пользователей, принадлежащих к этой Группе, должны быть доступны:

- admin - Администрирования
- application - Приложения
- api - API
- connection - Соединения
- data - Данные
- modeling - Моделирование
- model - Модели
- project - Проекты
- report - Отчеты
- visual - Визуализация

После того, как галочки для соответствующих разделов проставлены, нажимается кнопка «Создать». Новая группа появится в списке.

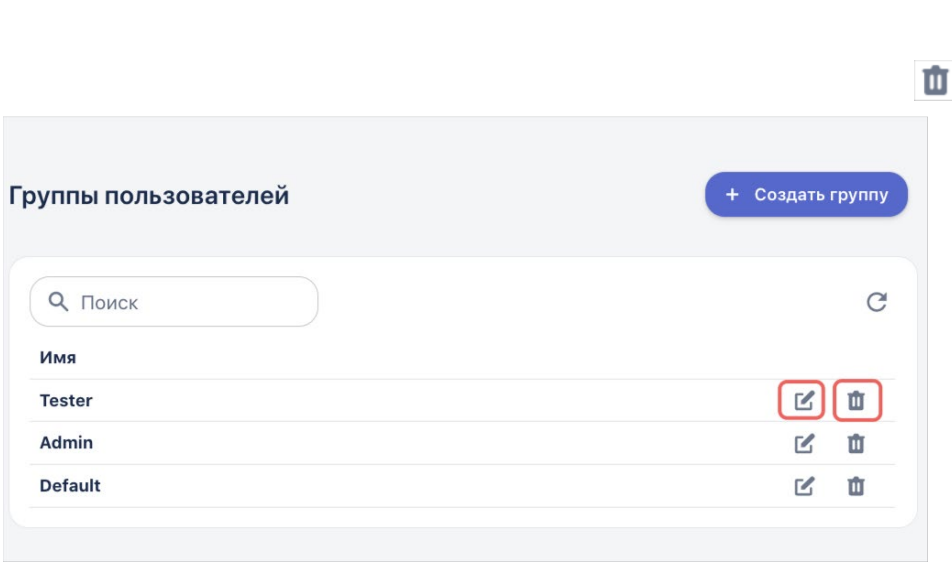


Рисунок 16.3 – Кнопки удаления и редактирования группы пользователей

Список всех пользователей представлен в разделе Администрирование -> Пользователи:

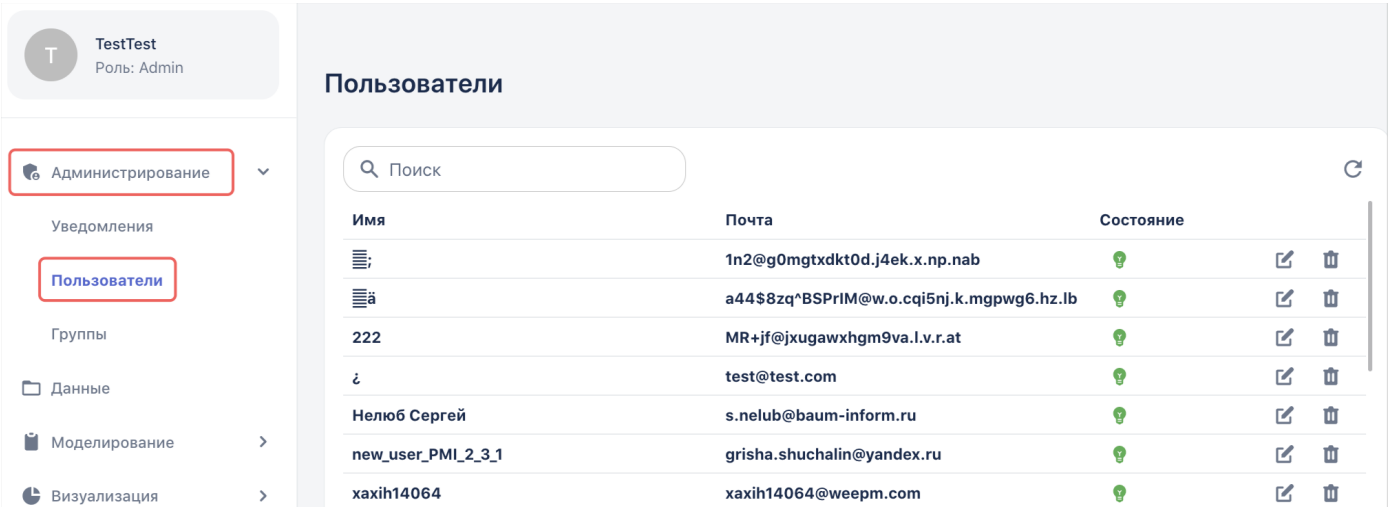


Рисунок 16.4 – Список пользователей в меню Администрирование

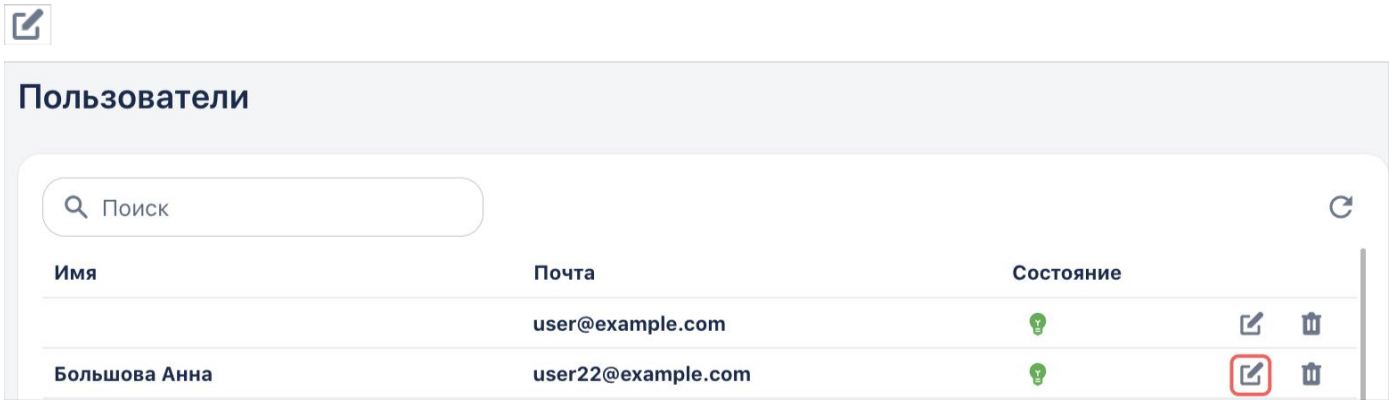


Рисунок 16.5 – Редактирование данных пользователя

В открывшемся окне есть возможность изменить имя пользователя, адрес электронной почты, а также назначить Группу:

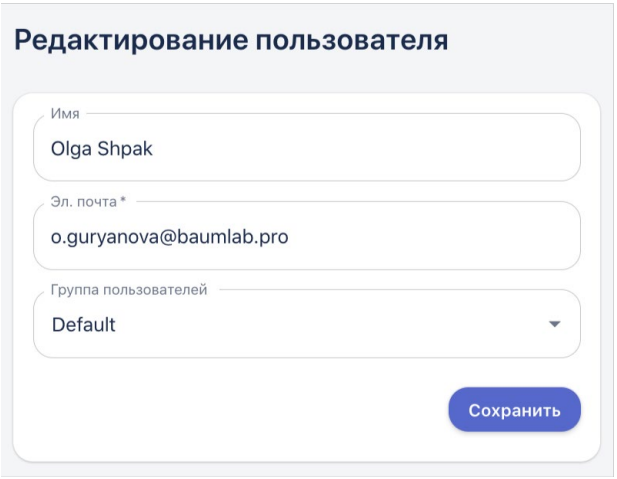


Рисунок 16.6 – Параметры для редактирования

Обратите внимание, что всем новым пользователям по умолчанию присваивается Группа Default (группа по умолчанию).

Рекомендуется оставлять одного-двух пользователей Администраторов, которые смогут управлять доступами и отвечать за настройки.

- 16.2 Настройка отправки уведомлений

На платформе реализована возможность отправки уведомлений - сообщений в телеграм или на почту. Этот функционал позволяет пользователям получать автоматические оповещения о результатах работы пайплайнов, при выполнении заданных условий. Создание и настройка каналов осуществляется следующим образом:

1. Перейти в раздел Администрирование -> Уведомления:

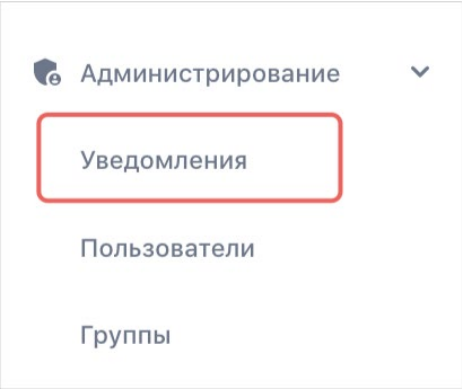


Рисунок 16.6 - Раздел Уведомления

2. В открывшемся окне отобразится список всех существующих каналов уведомлений:

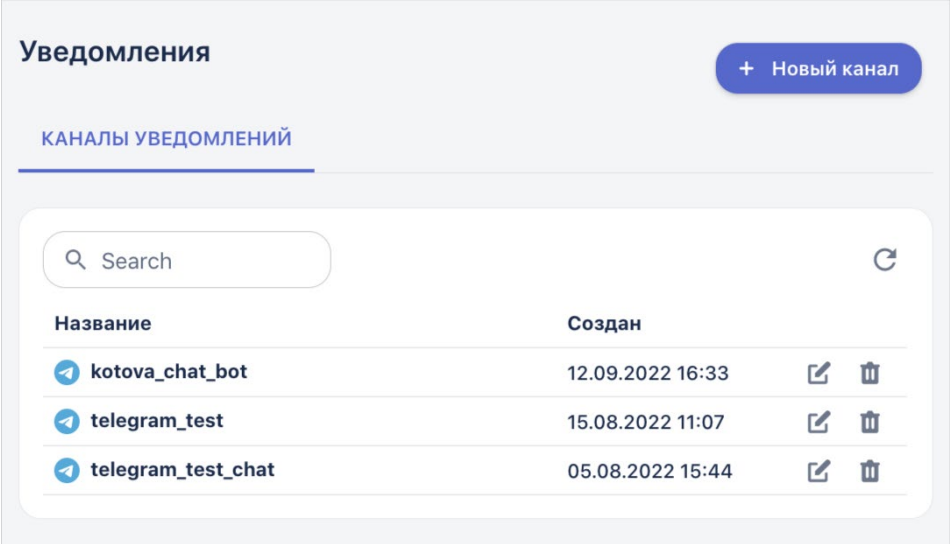


Рисунок 16.6 - Список созданных каналов

3. Для создания нового канала уведомлений нажимается кнопка «Новый канал» в правом верхнем углу
4. Открывается окно создания нового канала уведомлений:

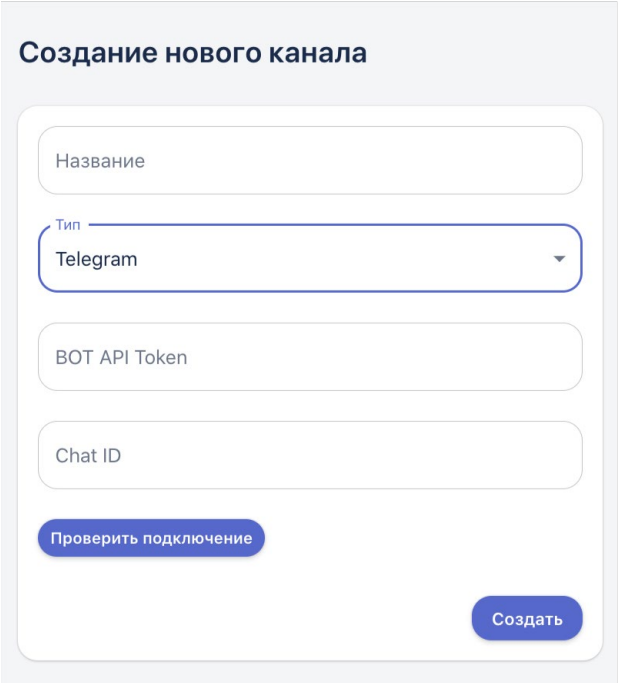


Рисунок 16.7 - Окно создания нового канала

5. Уведомления можно отправлять в телеграм бот или на почту.

5.1. Для создания уведомления с использованием **телеграм бота** поля заполняются следующим образом:

- *Название* - пользователь задаёт название бота из телеграмм
- *Тип* - Telegram
- *BOT API Token* - токен бота, полученный в телеграм при создании бота
- *Chat ID* - уникальный численный идентификатор чат бота

> ***Примечание**: сначала пользователь должен создать чат бота в телеграм или получить его*
> *токен и id для настройки канала уведомлений. Создание бота описано*

5.2. Для создания уведомления **на почту** поля заполняются следующим образом:

- *Название* - пользователь задает название канала
- *Тип* - Почта
- *Почты* - указываются адреса, на которые должны быть отправлены уведомления. При этом после ввода первого адреса, нужно нажать Enter, потом перейти к указанию следующего адреса и т д.

> ***Примечание**: уведомления на почту не реализованы в текущей версии системы*

6. После того, как все поля заполнены, нажимается кнопка «Проверить подключение», если все настройки были указаны верно, система выдаст сообщение от успешности подключения в верхнем правом углу окна. В противном случае, отобразится сообщение «Не удалось подключиться»
7. Завершающим этапом нажимается кнопка «Создать»
8. Новый канал отобразится в списке.

Для того чтобы начать получать уведомления в телеграм, пользователь должен создать пайплайн, который будет содержать блок «Процесс» с функцией «Отправка уведомлений», где указывается необходимый канал:

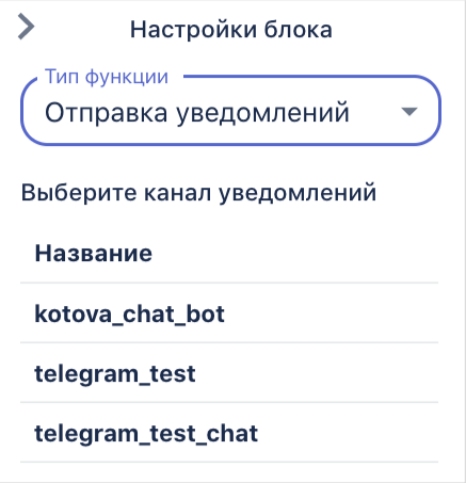


Рисунок 16.8 - Настройка блока «Отправка уведомлений»

Условием для отправки будет служить блок, идущий перед блоком уведомлений. Например, это может быть шлюз, где задаётся параметр, при котором уведомление должно быть отправлено:

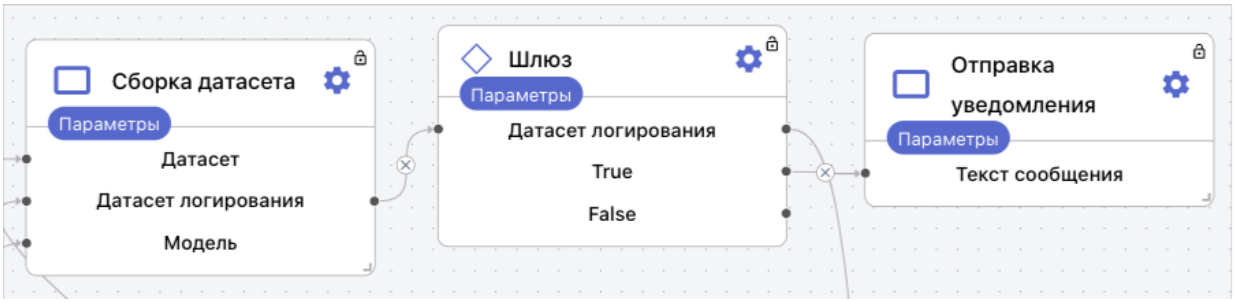


Рисунок 16.9 - Пример построения пайплайна с блоком уведомления

Например, в результате отработки такого пайплайна, пользователи будут получать следующую информацию:

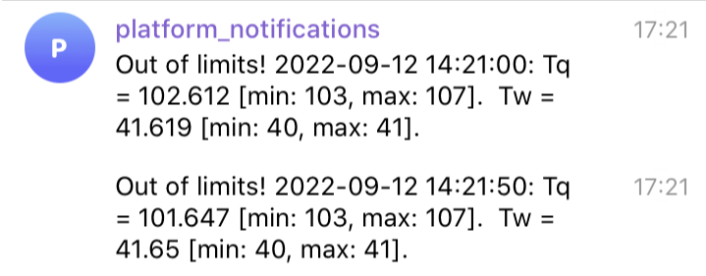


Рисунок 16.10 - Пример уведомлений в телеграм канале

17. Дополнительные возможности Платформы

- 17.1. Обращение в службу поддержки

В верхнем правом углу окна системы находится кнопка «Сообщить об ошибке»:

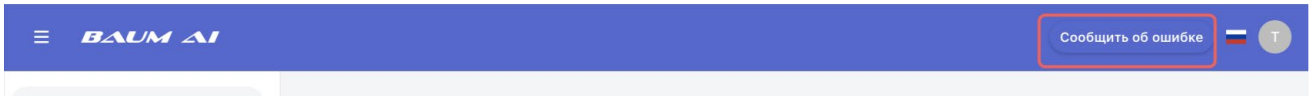


Рисунок 17.1 – Кнопка «Сообщить об ошибке»

В случае, если пользователь сталкивается с какой-то проблемой в работе платформы, поведением, не описанным в данном руководстве, есть возможность связаться с командой поддержки системы и сообщить о проблеме.

Пользователю необходимо максимально подробно описать проблему и шаги, которые привели к её появлению в текстовом окне сообщения об ошибке, а после нажать кнопку «Отправить»:

Рисунок 17.2 – Описание найденной в Системе ошибки, и отправка на обработку Разработчику

Ограничение размера сообщения составляет 256 символов.

- 17.2. История изменений

История изменений - это раздел системы, отображающий информацию о том в каком блоке и в какое время было совершено какое либо изменение. Раздел содержит информацию о всех запусках пайплайнов, сформированных отчетах, названия блоков, дату и время каждого изменения.

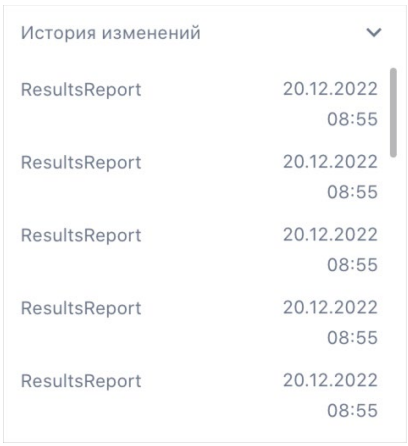


Рисунок 17.3 – История изменений

18. Приложения

- Приложение 1. Автоматизированные функции

*_Журнал преобразований_ – технический объект, который позволяет пробрасывать данные между элементами блок-схемы.

**Выходным параметром для всех функций является _словарь с данными_, который может включать в себя: таблицы, графики, текстовое описание.

Таблица 18.1 – Перечень автоматизированных функций элемента «Источник данных»

Функция	Назначение	Параметры	Выходная информация
Группа «Загрузка данных»			
Загрузка табличных данных	Преобразует загруженные данные во временной ряд. При этом автоматически определяет: формат файла (csv, txt, xls, xlsx), разделитель, размерность, где *разделитель* – это символ в датасете с временным рядом, обозначающий деление на ячейки. Для *ресемплирования* использует агрегирующую функцию	**Выберите файл** – Название файла, содержащего временной ряд для анализа	**
Загрузка изображений для классификации	Загружаются изображения с объектами для дальнейшей классификации этих объектов на новых данных. При этом предусмотрено, что изображения с объектами	**Группа обучающих изображений** – Выбрать папку для *обучения* нейронной сети, который представляет собой каталог с папками, где каждая папка содержит изображения с объектами одного класса.	В БД сохраняются каталоги датасета обучения и валидации, прошедшие предварительную обработку перед загрузкой.

<table>
<tr>
<td></td>
<td>одного класса распределены по отдельным папкам. При загрузке выполняется ресайз изображений – изменение (чаще уменьшение) размера изображений до заданного формата. Данные загружаются маленькими порциями, так называемыми мини-батчами (например, за один раз подается два изображения).</td>
<td>
<table>
<tr><td></td><td>Например – папки с изображениями собак, кошек и диких животных (три класса).</td></tr>
<tr><td>Группа тестовых изображений</td><td>Выбрать папку для валидации обученной нейронной сети, который также представлен в виде каталога с папками отдельных классов.</td></tr>
<tr><td>Размер мини-батча</td><td>Указывается количество изображений, которое за один раз подается на вход нейронной сети для её обучения.</td></tr>
<tr><td>Новая высота</td><td>Новая высота изображения</td></tr>
<tr><td>Новая ширина</td><td>Новая ширина изображения</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Загрузка табличных данных из коннектора</td>
<td>Функция предназначена для подключения к источникам данным в виде баз данных «ClickHouse» или «Postgresql». При этом используется сущность «Коннектор», в которой прописываются настройки для подключения к этим базам данных. Реализована возможность формирования датасета на основании полученных данных. Для этого в настройках функции активируется галочка в поле «Сохранить датасет», и в поле «Название файла»</td>
<td>
<table>
<tr><td>Выберите файл</td><td>Из списка выбирается коннектор – источник данных, подключение к которому необходимо выполнить. Источником выступает сторонняя база данных – ClickHouse или Postgresql (соответственно из списка выбирается коннектор с таким типом).</td></tr>
<tr><td>Сохранить датасет</td><td>Чтобы сформировать бэкап таблицы внешней БД в поле «Сохранить датасет» устанавливается галочка.
Иначе, если не установить галочку в поле «Сохранить датасет», выполняется подключение к внешней БД в её состоянии на текущий момент времени, без</td></tr>
</table>
</td>
<td>**</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>указывается имя для бэкапа таблицы базы данных в настоящий момент времени. В результате в разделе «Данные» сохраняется файл в формате .csv с данными из коннектора.
Важно – Подключение выполняется к БД в её состоянии на текущий момент времени.</td>
<td>
<table>
<tr><td></td><td>дополнительного формирования датасета во внутренней БД.</td></tr>
<tr><td>Название файла</td><td>Указывается название файла бэкапа таблицы для сохранения во внутренней БД.</td></tr>
<tr><td>Онлайн данные</td><td>Указывается, осуществляется ли получение данных в режиме он лайн.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Загрузка модели</td>
<td>Функция предназначена для использования в качестве источника данных ранее обученной модели. При этом система при обработке данных пайплайна применяет ранее полученные знания для построения прогнозов.</td>
<td>
<table>
<tr><td>Модель</td><td>Выбор из списка ранее сохраненных моделей</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Загрузка текстовых файлов для классификации</td>
<td>Данная функция предназначена для загрузки текстов, принадлежащих к тем или иным классам, для обучения нейронной сети определять эти классы на новых данных. Функция обязательно используется при решении задач классификации текстов.</td>
<td>
<table>
<tr><td>Группа обучающих текстов</td><td>Выбор папки для обучения нейронной сети, которая должна содержать в себе подпапки с названиями классов объектов. Данные подпапки содержат тексты, принадлежащие определенному классу. Например, это могут быть: «Пушкин»,«Лермонтов»,«Толстой».</td></tr>
<tr><td>Группа тестовых текстов</td><td>Выбор папки для валидации обученной нейронной сети. Папка должна иметь такую же структуру, как и обучающая.</td></tr>
</table>
</td>
<td></td>
</tr>
</table>

		Группа текстов для классификации — Здесь можно сразу выбрать файл или папку с файлами, которые необходимо классифицировать с применением обученной модели.	
Загрузка текстовых файлов для кластеризации	Функция обязательно используется при решении задач кластеризации текстов, когда необходимо определить кластеры к которым принадлежат тексты.	**Группа текстов для кластеризации** — Выбор файла, содержащего однотипные данные, подлежащие разделению на кластеры.	
Загрузка графа	Функция предназначена для загрузки и дальнейшего преобразования файлов с форматом .graphml в переменную graph_out с типом данных networkx.MultiDiGraph, предназначенных для решения задач с применением теории графов. Граф — это геометрическая фигура, которая состоит из точек и линий, которые их соединяют. Точки называют вершинами графа, а линии — ребрами. Графы имеют очень широкое применение: с их помощью выбирают наиболее выгодное расположение зданий, графами представлены схемы	**Выберите файл с графом** — Выбирается ранее загруженный в систему файл в формате .graphml.	

<table>
<tr><td></td><td>метро, маршруты, схемы игр, блок схемы процессов и т д.</td><td></td><td></td></tr>
<tr><td colspan="4">Группа «Spark»</td></tr>
<tr><td>Загрузка табличных данных из файла CSV (Spark)</td><td>При помощи данной функции осуществляется загрузка в систему табличных данных с помощью фрйемворка для распределенных вычислений Apache Spark, конкретно, с помощью библиотеки PySpark для Python. Датафрейм в PySpark — это таблица, строки которой хранятся в RDD (Отказоустойчивый распределенный набор данных (англ. Resilient Distributed Dataset, RDD) — тип структуры данных, который можно распределить между несколькими узлами в кластере). Работа с датафреймами ведётся по принципу «ленивых вычислений» (англ. lazy evaluations). Это вычисления, которые откладываются до тех пор, пока пользователь не запросит их результат.
Данная функция работает только для файлов в формате csv, содержащих big data.</td><td><table><tr><td>Выберите файл для загрузки</td><td>Выбор из списка файла для дальнейшего анализа.</td></tr></table></td><td></td></tr>
</table>

Загрузка табличных данных из папки CSV (Spark)	Фреймворк «Apache Spark» распределяет хранимые данные по серверам и директориям. Чтобы обратиться к файлам на уровне папки, в которой они хранятся, используется данный метод.	**Выберите директорию с датасетом для загрузки** — Выбор из списка *папки* для дальнейшего анализа.	
Загрузка модели	Функция предназначена для использования в качестве источника данных ранее обученной модели ИИ «Spark».	**Модель** — Выбор из списка ранее сохраненных моделей Spark (в разработке отдельное *API* для моделей Spark. Модели будут объединяться в одну группу по семантическому типу).	
Загрузка табличных данных из коннектора (Spark)	Функция предназначена для получения табличных данных через коннектор с типом «ClickHouse», с использованием библиотеки «Spark».	(*в разработке*)	

Таблица 18.2 – Перечень автоматизированных функций элемента «Процесс»

Функция	Назначение	Параметры	Выходная информация
1.Группа «Анализ данных»			

<table>
<tr>
<td>Выбор признаков и целевых признаков</td>
<td>В датасете выбираются: признаки – измеримые характеристики исследуемого объекта/ процесса, и целевые (зависимые) переменные, значения которых предстоит предсказывать модели.</td>
<td>
<table>
<tr><td>Признаки</td><td>Характеристики, которые исследуются и выявляется корреляция между ними и рассматриваемым целевым признаком</td></tr>
<tr><td>Целевые признаки</td><td>Предсказываемые переменные</td></tr>
</table>
</td>
<td>Датасет с размеченными признаками и целевыми признаками</td>
</tr>
<tr>
<td>Матрица корреляции</td>
<td>1.Алгоритм сначала рассчитывает коэффициенты корреляции по всем признакам (общая матрица корреляции).
2.Затем в этой матрице отбираются топ-k максимальных (ближе к 1) значений коэффициентов корреляции.
3.Строится новая матрица корреляции, состоящая из признаков, для которых найдены максимальные значения коэффициентов.</td>
<td>
<table>
<tr><td>Топ k-значений для корреляции</td><td>Количество максимальных значений корреляции (int)</td></tr>
</table>
</td>
<td>1. Матрица корреляции с топ-k признаков, имеющих максимальные значения корреляции.
2.Матрица корреляции по всем признакам.</td>
</tr>
</table>

<table>
<tr>
<td>Косинусное расстояние</td>
<td>Вычисляется косинусное расстояние между значениями во входном векторе и значениями выбранных столбцов в наблюдениях.</td>
<td>
<table>
<tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr>
<tr><td>Признаки</td><td>Признаки для анализа.</td></tr>
<tr><td>Вектор</td><td>Вектор такой же длины, что и количество выбранных в датасете признаков (1D-array).</td></tr>
</table>
</td>
<td>Таблица с наблюдениями, наиболее схожими с входным вектором, где первые топ-5 наблюдений выделены жирным шрифтом (по ним рассчитанная косинусная мера имеет значение, наиболее близкое к 0).</td>
</tr>
<tr>
<td>Поиск пропущенных значений</td>
<td>Для каждого выбранного признака метод находит пропущенные значения в наблюдениях.</td>
<td>
<table>
<tr><td>Признаки</td><td>Выбираются признаки, в которых необходимо найти пропущенные значения.</td></tr>
</table>
</td>
<td>Словарь, в котором по каждому выбранному признаку отображается количество пропущенных значений в наблюдениях, и процент пропусков.</td>
</tr>
<tr>
<td>Анализ временных рядов</td>
<td>Совокупность математико-статистических методов анализа, предназначенных для выявления структуры временных рядов и для их прогнозирования</td>
<td>
<table>
<tr><td>Список графиков</td><td>Выбираются графики, которые будут использованы для визуализации анализа временных рядов и задаются параметры для них</td></tr>
</table>
</td>
<td>Линейный график, ACF/PACF, Декомпозиция, Свечной график, Time profile, Extended, Bollinger Bands, Stochastic Oscillator</td>
</tr>
<tr>
<td>Запись в датасет логирования</td>
<td>Данная функция применяется при построении пайплайнов в режиме реального времени. В процессе логирования записываются все новые поступающие значения в датасет для дальнейшего использования при валидации и работе с моделью. При этом в</td>
<td>-</td>
<td>Датасет с данными</td>
</tr>
</table>

<table>
<tr><td></td><td>процессе логирования записываются:
- фактические значения
- промежуточные транформации при препроцессинге до подачи в модель (если в качестве источника данных выбрана модель)
- прогнозные значения (если в пайплайне настроен прогноз)</td><td></td><td></td></tr>
<tr><td>Визуализация Real-time</td><td>Представление в виде графиков и диаграмм результатов работы пайплайна, содержащего данные, получаемые в режиме реального времени</td><td><table><tr><td>Список графиков</td><td>Выбираются графики, которые будут использованы для визуализации данных в режиме он лайн и задаются параметры для них</td></tr></table></td><td>Линейный график, Свечной график, Time Profile, Extended, Bollinger Bands, Stochastic Oscillator</td></tr>
<tr><td colspan="4">Загрузка данных</td></tr>
<tr><td>Преобразование данных во временной ряд</td><td>Метод редактирует исходные данные, исключая в них аномалии и искаженные наблюдения, которые могли быть зафиксированы в результате помех. Далее выполняется дискретизация – определяются точки (моменты времени), в которых должны быть произведены выборки значений. Дискретизация</td><td><table><tr><td>Шаг ресемплирования</td><td>Дискретность для временного ряда – частота фиксирования наблюдений, значения начиная с нано-, микро-, милли-, секунд и заканчивая годами. Указывается оптимальный интервал дискретности.</td></tr><tr><td>Частота ресемплирования</td><td>Единица измерения, в которой фиксируются наблюдения.</td></tr><tr><td>Агрегирующая функция</td><td>Функция, вычисляющая результат по набору значений группы, где группа – наблюдения в пределах шага</td></tr></table></td><td>Создается временной ряд, с заданным шагом ресемплирования</td></tr>
</table>

<table>
<tr><td></td><td>производится через равные промежутки времени.</td><td><table>
<tr><td></td><td>ресемплирования. По умолчанию, значение вычисляется функцией медианы.</td></tr>
<tr><td>Столбец с временной меткой</td><td>Время фиксирования наблюдения. По умолчанию, нулевой столбец в датасете.</td></tr>
</table></td><td></td></tr>
<tr><td colspan="4">Препроцессинг</td></tr>
<tr><td>Стабилизация дисперсии</td><td>Уменьшает разброс исследуемых данных, чтобы сделать их более компактными и пригодными для работы.</td><td><table>
<tr><td>Замена значений столбцов</td><td>Преобразование оригинального временного ряда, загруженного в систему. Позволяет заменять трансформируемые столбцы или добавлять новые</td></tr>
<tr><td>Стандартизация</td><td>Преобразование значений признака, адаптирующая признаки с разными диапазонами значений к моделям машинного обучения</td></tr>
<tr><td>Метод</td><td>Выбирается метод, с помощью которого проводится стабилизация дисперсии – приведение данных к нормальному распределению. На выбор два метода – yeo-johnson и box-cox. Метод yeo-johnson работает как с отрицательными, так и с положительными значениями, а метод box-cox только с положительными</td></tr>
<tr><td>Флаг признака</td><td>Показатели датасета, значения которых предстоит предсказывать модели машинного обучения</td></tr>
</table></td><td>Преобразованный датасет. При этом преобразования над целевыми признаками проводятся отдельно.</td></tr>
</table>

<table>
<tr>
<td>Стандартизация</td>
<td>Чтобы сгладить большие различия между диапазонами признаков датасета и предотвратить искаженное восприятие данных моделью машинного обучения выполняется стандартизация – преобразование и приведение признаков датасета к единому формату</td>
<td>
<table>
<tr><td>Замена значений столбцов</td><td>Подтверждение преобразования оригинального временного ряда (заменой трансформируемых столбцов или добавлением новых)</td></tr>
<tr><td>Флаг признака</td><td>Выбрать столбцы для преобразования – все, кроме столбцов с датой и целевым признаком</td></tr>
</table>
</td>
<td>Преобразованные значения показателей временного ряда, кроме целевого признака</td>
</tr>
<tr>
<td>Дифференцирование временного ряда</td>
<td>Выполняется дифференцирование целевых признаков (таргетов) временного ряда. При этом временной ряд сдвигается на указанное число шагов в разрезе каждого целевого признака. Если есть сезонность, сначала проводится сезонное дифференцирование.
Желательно дифференцировать ряд как можно меньше раз, потому что с увеличением количества дифференцирований растет дисперсия ошибки прогноза</td>
<td>
<table>
<tr><td>Шаг дифференцирования для каждого целевого признака</td><td>Есть возможность задать шаг дифференцирования для каждого таргета, в формате [сдвиг для таргета 1, свиг для таргета 2, …], где сдвиг на один шаг применяется для обычного (для избавления от тренда) дифференцирования, сдвиг на несколько шагов – для сезонного, а сдвиг, равный нулю означает, что дифференцирование для данного таргета не проводится.
Например, [1, 0, 3].</td></tr>
</table>
</td>
<td>К датасету временного ряда добавляются новые столбцы с окончанием '_diff' для каждого указанного таргета. При этом замена колонок не предусмотрена – оригинальные колонки сохраняются для задачи обратного дифференцирования. Отображаются графики настоящего и сдвинутого временных рядов.</td>
</tr>
<tr>
<td>One-Hot Encoding</td>
<td>Метод One Hot Encoding (OHE) применяется, когда в датасете необходимо</td>
<td>
<table>
<tr><td>Флаг удаления</td><td>Устанавливается, чтобы удалить из итоговой таблицы столбец с признаком, над</td></tr>
</table>
</td>
<td>Таблица с новыми столбцами, в которых отражается принадлежность наблюдений к</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>закодировать категориальные признаки (текстовые), перед подачей в модель. Для кодируемого категориального признака создаются N новых столбцов в датасете, где N – количество уникальных категорий. Значения в новых столбцах – 0 или 1, в зависимости от принадлежности к категории. Так каждый новый признак – бинарный характеристический признак категории.</td>
<td>
<table>
<tr><td>первого признака</td><td>которым были выполнены преобразования. Так как новые столбцы отражают принадлежность наблюдения к той или иной категории признака, удаление первого признака не повлияет на результат.</td></tr>
<tr><td>Флаг признака</td><td>Выбираются столбцы, над которыми будут осуществляться преобразования. Значения на выбор – признаки или таргеты.</td></tr>
</table>
</td>
<td>тем или иным категориям преобразованных признаков.</td>
</tr>
<tr>
<td>Создание признаков для временного ряда</td>
<td>Для временного ряда создаются новые признаки, в которых значения таргетов сдвигаются на указанное число шагов. Например, если для одномерного (с одним таргетом) временного ряда задать сдвиг в один шаг, создается новая колонка y, в которой значение первой строки равно значению второй строки в колонке с таргетом x (категориальный признак), т.е. значения сдвигаются на один шаг вперед. Если ряд многомерный – состоящий из нескольких таргетов, то для каждого из них</td>
<td>
<table>
<tr><td>Максимальное количество лагов</td><td>Указывается, на какое количество шагов может быть сдвинут временной ряд.</td></tr>
</table>
</td>
<td>Создается таблица, график со смещенным временным рядом. Процесс создания признаков-лагов сохраняется в журнале преобразований, и далее отрабатывается при препроцессинге.</td>
</tr>
</table>

<table>
<tr><td></td><td>передается общий массив признаков, с учетом лагов всех таргет-рядов.
Такое действие является предварительным перед тем, как подавать данные в модель ИИ, чтобы у модели были не только фактические значения таргета, но и прогнозные.</td><td></td><td></td></tr>
<tr><td>Препроцессинг текстовых данных</td><td>Алгоритмы машинного обучения не работают с «сырыми данными». Большая часть процесса – это подготовка текста, преобразование ее в вид, доступный для восприятия компьютером.
В первую очередь выполняется очистка текста. Из текста удаляются бесполезные для машины данные – это большинство знаков пунктуации, особые символы, скобки, теги и т.д.
Дальше наступает большой этап предварительной обработки – препроцессинга. Это приведение информации к виду, в котором она более понятна алгоритму. Методы препроцессинга:</td><td><table><tr><td>Метод векторизации</td><td>1. TF-IDF. С англ. TF – term frequency (частота слова), IDF – inverse document frequency (обратная частота документа). Это мера, используемая для оценки важности слова в контексте документа, являющегося частью коллекции документов. Вес некоторого слова пропорционален частоте употребления этого слова в документе, и обратно пропорционален частоте употребления слова во всех документах коллекции.
Метод TF-IDF используется в задачах анализа текстов, и представляет собой линейный классификатор с разреженными признаками, взвешенными по частоте.
Этот метод выбирается по умолчанию.
2. Word2vec. Принимает большой текстовый корпус в качестве входных данных, и сопоставляет каждому слову вектор, выдавая координаты слов на выходе. Сначала он генерирует словарь корпуса, а</td></tr></table></td><td>Числовые векторы, созданные на основе исходной текстовой информации, которые отражают важность использования каждого слова.</td></tr>
</table>

	-приведение символов к одному регистру; -*токенизация* – разбиение текста на токены. Так называют отдельные компоненты – слова, предложения или фразы; -*лемматизация* – приведение слов к изначальным словоформам, часто с учетом контекста; -удаление *стоп-слов* – артиклей, междометий и пр.; После предподготовки на выходе получается набор подготовленных слов. Но алгоритмы работают с числовыми данными, поэтому из входящей информации создают *векторы* – представляют ее как набор числовых значений. На Платформе используются следующие методы векторизации – TF-IDF, и Word2vec.		затем вычисляет векторное представление слов, ‘обучаясь’ на входных тестах. Векторное представление основывается на контекстной близости: слова, встречающиеся в тексте с одинаковыми словами (а следовательно, имеющие схожий смысл), будут иметь близкие (по *косинусному расстоянию*) векторы. Полученные векторные представления слов используются для обработки естественного языка и машинного обучения.	
		Использование GPU и нейронной сети	В данном поле устанавливается галочка, когда обучение модели ИИ происходит с использованием нейронной сети.	
Кодирование целевого признака	Данная функция применяется, когда необходимо преобразовать *категориальный целевой признак в датасете* в числовое значение. Такое преобразование	Для данной функции не предусмотрен ручной ввод параметров пользователем. На вход функции подается *датасет* с входными данными, над которыми уже выявлены некоторые преобразования (очистка, выделение целевого признака, и т.д.), и файл *vars_dict*, в котором содержится информация о выполненных преобразованиях над данными.		Закодированный целевой признак в датасете.

	выполняется перед подачей входных данных в алгоритм. Правила перевода категориальный признаков в числовые прописываются в *кодировщике*. Данная функция представляет собой первый тип кодирования – **Label Encoder**. Выполняется *порядковое кодирование* всех уникальных значений категориального признака: первое (выбранное каким-то образом) уникальное значение кодируется нулем, второе единицей, и так далее, последнее кодируется числом, равным количеству уникальных значений минус единица.		
Порядковое кодирование категориальных признаков	Отличие данной функции в том, что она выполняет преобразование всех *категориальных признаков датасета* в числовые значения. При этом кодировщик используется тот же, что и в предыдущей функции – Label Encoder, но кодируются признаки. Выполняется порядковое кодирование каждой категориальной переменной (кроме целевой).	Для данной функции не предусмотрен ручной ввод параметров.	Закодированные категориальные признаки в датасете.

<table>
<tr><th colspan="4">Предобработка данных</th></tr>
<tr>
<td>Заполнение пропусков</td>
<td>Позволяет заполнять пропущенные значения в датасете одним из следующих способов: среднее, мода, медиана, квантили 0.25, 0.5, 0.75 или по выбору, min, max. Функция применяется для датасетов, созданных из табличных данных. При этом при загрузке файла в систему можно посмотреть количество пропусков (пустых ячеек) в датасете, если для одного из признаков пропусков слишком много, предусмотрена возможность удаления его полностью.</td>
<td>
<table>
<tr><td>Индекс столбца</td><td>номер столбца</td></tr>
<tr><td>Способ заполнения пропусков</td><td>mean, mode, median, max, min, quantile25, quantile50, quantile75
drop - удаление</td></tr>
</table>
</td>
<td>Датасет с заполненными пропусками</td>
</tr>
<tr>
<td>Сглаживание временного ряда</td>
<td>Позволяет исключить влияние шума в данных и увидеть структуру временного ряда. Для сглаживания применяется метод центрированного скользящего среднего: по временному ряду «скользит окно» определенного размера, в рамках окна значения группируются и по ним рассчитываются средние значения.</td>
<td>
<table>
<tr><td>Список признаков</td><td>Выбираются признаки для расчета скользящего среднего</td></tr>
<tr><td>Размер окна для сглаживания</td><td>Временное окно анализа, определяется количеством входящих в него наблюдений. Например, размер окна три, тогда берутся первые три наблюдения и по ним считаются средние значения признаков</td></tr>
</table>
</td>
<td>1.Таблица, содержащая сглаженные значения признаков.
2.График исходных данных и сглаженного временного ряда.</td>
</tr>
</table>

<table>
<tr>
<td>Срез временного ряда по индексу</td>
<td>Позволяет создавать выборки данных за период времени, используя временные метки или временные диапазоны.</td>
<td>
<table>
<tr><td>Дата начала</td><td>Дата начала среза</td></tr>
<tr><td>Дата окончания</td><td>Дата окончания среза</td></tr>
</table>
</td>
<td>Временной ряд после применения фильтра.</td>
</tr>
<tr>
<td>Фильтрация текстового шума</td>
<td>Данная функция позволяет очистить текст от шумов: из текста убираются знаки препинания, заглавные буквы (они заменяются на строчные) и стоп-слова (различные служебные части речи - союзы, предлоги, частицы и т.д.)</td>
<td>-</td>
<td>Текст без шумов</td>
</tr>
<tr>
<td>Лемматизация текста</td>
<td>Лемматизация - это процесс приведения всех встречающихся форм слова к одной, нормальной словарной форме. В процессе лемматизации платформа использует словарь и морфологический анализ, чтобы привести слово к его канонической форме – т.н. «лемме», в итоге получается текст, состоящий из слов приведенных к единственному числу, мужскому роду, именительному падежу и инфинитиву (в зависимости от части речи).</td>
<td>-</td>
<td>Нормализованный текст</td>
</tr>
</table>

<table>
<tr>
<td>Векторизация текста</td>
<td>Векторизация текста - это процесс преобразования слов в векторы (числа), которые являются «читаемым» форматом для алгоритмов машинного обучения.</td>
<td>
<table>
<tr><td>Метод векторизации</td><td>1.TD IDF - метод, используемый для оценки важности слова в контексте документа. Вес некоторого слова пропорционален частоте употребления этого слова в документе и обратно пропорционален частоте употребления слова во всех документах коллекции.
2.Word to Vec - данный метод использует контекст, чтобы сформировать численные представления слов, в результате слова, используемые в одном и том же контексте, имеют похожие векторы.</td></tr>
<tr><td>Максимальная размерность текста</td><td>Указывается примерное количество уникальных слов в тексте.</td></tr>
<tr><td>Количество признаков</td><td>Указывается количество столбцов таблицы, которая получится в результате преобразования текста в числовой вид</td></tr>
<tr><td>Сгенерировать тензор для GPU</td><td>Выбирается в случае если предполагается что дальше будет использоваться графический процессор. Тензор - это просто таблица особого вида</td></tr>
</table>
</td>
<td></td>
</tr>
<tr><td colspan="4">Тесты на нормальность распределения</td></tr>
<tr>
<td>Коэффициент асимметрии Skewness</td>
<td>Данный метод проверяет выборку на нормальность распределения путем расчета асимметрии данных. Если правый хвост асимметрии длиннее левого,</td>
<td>
<table>
<tr><td>Признаки</td><td>Выбираются все признаки в датасете для расчета коэффициента асимметрии.</td></tr>
</table>
</td>
<td>Словарь с данными.</td>
</tr>
</table>

	то коэффициент положителен, иначе – отрицателен. Если распределение симметрично (в форме ‘колокола’), коэффициент равен нулю.		
Тесты на стационарность временного ряда			
Тест Дики-Фуллера	Проверяется, является ли временной ряд *стационарным* – не влияют ли на него тренды и сезонность. Для такого ряда суммарные статистические данные согласованы по времени, например, *среднее значение* и *дисперсия наблюдений*. Стационарность влияет на легкость моделирования – часто требуется, чтобы временной ряд был стационарным, чтобы быть эффективным.	**Пороговое значение alpha**: Задается пороговое значение *p* из теста Дики-Фуллера, с использованием которого интерпретируются результаты гипотез: ● *Нулевая гипотеза* – временной ряд имеет единичный корень, то есть он нестационарный; ● *Альтернативная гипотеза* – нулевая гипотеза отвергается, и предполагается, что временной ряд не имеет единичного корня, то есть он является стационарным. Значение p ниже порогового значения означает, что отвергается нулевая гипотеза и временной ряд *стационарный*. Значение p выше порогового значения означает, что подтверждается нулевая гипотеза и временной ряд *нестационарный*. Значение *p* задается в формате числа с плавающей точкой (float).	Результаты теста Дики-Фуллера.
2.Группа «Машинное обучение»			
Валидация модели	На тестовой выборке данных (обычно это 20% датасета) проверяется правильность работы (предсказательная	**Метрика**: Из списка выбирается название метрики для валидации. Для задачи *классификации*: Accuracy, F1, Precision, Recall, AUC_ROC.	Таблица со значением выбранной метрики, отражающей количество правильных ответов обученной модели на тестовой выборке

<table>
<tr><td></td><td>способность) модели ИИ, построенной на основе машинного обучения.</td><td><table>
<tr><td></td><td>Для задачи регрессии: RMSE, MAE, WMAPE.</td></tr>
<tr><td>–</td><td>Журнал преобразований над данными.</td></tr>
<tr><td>–</td><td>Обученная модель ИИ.</td></tr>
</table></td><td>данных (максимальное значение метрики равно 1).</td></tr>
<tr><td>Прогноз модели</td><td>Выполняется последовательность действий по прогнозированию будущих значений целевых признаков.</td><td>-</td><td>1.Точность прогноза.
2.Словарь с данными.
3.Датасет логирования.</td></tr>
<tr><td>Разделение датасета на обучающую и тестовую выборки</td><td>Разделение выборки данных на две категории: для обучения модели ИИ, и для проверки результатов обучения.</td><td><table>
<tr><td>Доля тестовой выборки в датасете</td><td>Обычно на 80% датасета выполняется обучение модели, а на оставшихся 20% – ее валидация. Значение указывается в формате 0.2.</td></tr>
<tr><td>Перемешивать наблюдения перед разделением</td><td>Необходимо установить галочку в поле, чтобы случайным образом поменять местами наблюдения в датасете. Не рекомендуется перемешивать временные ряды, т.к. наблюдения в них упорядочены и зафиксированы последовательно по времени.</td></tr>
<tr><td>Разделять с учетом меток классов</td><td>Выбирается, учитывать ли долю таргетов при разделении датасета. Используется только для задач классификации, когда объекты распределяются по категориям согласно определенным и заданным заранее признакам.</td></tr>
</table></td><td>1.Отдельно обучающая и тестовая выборки.
2.Журнал преобразований.</td></tr>
<tr><td colspan="4">Классификация решает задачу разделения множества наблюдений (объектов) на группы, называемые классами, на основе анализа их формального описания. При классификации каждое наблюдение относится к определенной группе на основе некоторого качественного свойства. Пусть X – множество описаний</td></tr>
</table>

<table>
<tr><td colspan="4">объектов, Y – конечное множество номеров/имен/меток классов. Существует неизвестная целевая зависимость отображения $y^*: X \to Y$, значения которой известны только на объектах обучающей выборки $X^m = (x_1, y_1), \ldots, (x_m, y_m)$. Строится алгоритм, способный классифицировать произвольный объект $x \in X$.
.</td></tr>
<tr><td>Логистическая Регрессия</td><td>Используется логистическая функция для моделирования зависимости выходной переменной y от набора входных переменных x, в случае когда первая является бинарной. Например, с помощью логистической регрессии можно оценивать вероятность наступления/или не наступления некоторого события.
Предсказывается непрерывная переменная – коэффициент логистической регрессии, принимающий значение от 0 до 1:
● если значение коэффициента больше порогового значения, то вероятность наступления события равна 1;
● иначе вероятность наступления события равна 0.</td><td>
<table>
<tr><td>Коэффициент регуляризации</td><td>Указывается значение строго больше нуля – положительное вещественное число, с помощью которого добавляется дополнительное ограничение к условию с целью предотвратить переобучение модели.</td></tr>
<tr><td>Порог классификации</td><td>Значение вещественного типа от 0 до 1, определяющее принадлежность объекта к тому или иному классу.</td></tr>
<tr><td>Флаг возврата вероятности при прогнозе</td><td>Возвращает вероятность или метки классов для дальнейшего прогноза после обучения модели. Используется для решения задач бинарной классификации, когда выходная переменная может принимать только два значения – решается вопрос о принадлежности объекта к одному из двух классов.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Флаг подбора гиперпараметров. Флаг активируется, когда указывается несколько гиперпараметров.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Критерий остановки итераций. Настройка, позволяющая определить точность нахождения минимума функции ошибки.</td></tr>
</table>
</td><td>1.Модель бинарной классификации.
2.Словарь с данными.
3.Точность модели.
4.Журнал преобразований.</td></tr>
</table>

<table>
<tr>
<td></td>
<td></td>
<td>
<table>
<tr><td>Количество фолдов для оптимизации</td><td>Датасет делится на фолды – на указанное количество равных частей. При обучении модели каждый фолд становится валидационным один раз, при этом на остальных фолдах выполняется обучение. Каждый раз рассчитывается значение метрики. Затем рассчитывается усредненная метрика, которая характеризует точность модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Модель XGB Classifier</td>
<td>Алгоритм XGBClassifier анализирует связь между признаками и целевым признаком. На обучающей выборке модель обучается соотносить наблюдение к аномалиям, а на тестовой выборке выполняется валидация ответов обученной модели.</td>
<td>
<table>
<tr><td>Глубина дерева</td><td>Заданное максимальное число разбиений в ветвях, по достижению которого обучение модели ИИ останавливается.</td></tr>
<tr><td>Количество базовых моделей</td><td>Определяет сколько независимых моделей будет работать над обучением.</td></tr>
<tr><td>Порог классификации</td><td>Значение от 0 до 1, указывающее на верхнюю границу вероятности причисления объекта к классу.</td></tr>
<tr><td>Флаг возврата вероятности при прогнозе</td><td>Возвращает вероятность или метки классов для дальнейшего прогноза после обучения модели. Используется для решения задач бинарной классификации, когда выходная переменная может принимать только два значения – решается вопрос о принадлежности объекта к одному из двух классов.</td></tr>
</table>
</td>
<td>1.Таблица с матрицей ошибок.
2.Таблица верных и ошибочных прогнозов модели в разрезе классов.
3.Модель бинарной классификации.</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td></td>
<td>
<table>
<tr><td>Оптимизация гиперпараметров</td><td>Флаг подбора гиперпараметров. Флаг активируется, когда указывается несколько гиперпараметров.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Критерий остановки итераций. Настройка, позволяющая определить точность нахождения минимума функции ошибки.</td></tr>
<tr><td>Количество фолдов для оптимизации.</td><td>Датасет делится на фолды – на указанное количество равных частей. При обучении модели каждый фолд становится валидационным один раз, при этом на остальных фолдах выполняется обучение. Каждый раз рассчитывается значение метрики. Затем рассчитывается усредненная метрика, которая характеризует точность модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Дерево решений для классификации</td>
<td>Предсказывает, к какому классу принадлежит объект из обучающего массива данных. Для этого строится дерево решений: древовидная структура, где моменты принятий решений соответствуют узлам, в узлах происходит ветвление процесса на ветки в зависимости от сделанного выбора, и конечные узлы (листья) – конечные результаты последовательного</td>
<td>
<table>
<tr><td>Глубина дерева</td><td>Заданное максимальное число разбиений в ветвях, по достижению которого обучение модели ИИ останавливается.</td></tr>
<tr><td>Порог классификации</td><td>Значение порога определяет принадлежность объекта к одному из классов: к положительному – если порог выше указанного значения, к отрицательному – если порог ниже.</td></tr>
<tr><td>Флаг возврата вероятности при прогнозе</td><td>Возвращает вероятность или метки классов для дальнейшего прогноза после обучения модели.</td></tr>
</table>
</td>
<td>1.Датасет с меткой класса, определяющей принадлежность объекта к одному из классов.
2.Модель ИИ, обученная классифицировать данные по заданным критериям.
3.Журнал преобразований над данными.
4.Словарь с переменными (описание модели, таблицы, графики) для отображения в интерфейсе Программы.</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>принятия решений. В узлах, начиная с корневого, выбирается признак, значение которого используется для разбиения всех данных на два класса. Процесс продолжается до тех пор, пока не выполнится критерий остановки – дерево превысило заранее заданный «лимит роста» (достигнута глубина дерева). При этом разбиения выполняются таким образом, чтобы уменьшить выбранный критерий, например энтропию – степень неопределенности в разбиении на классы.</td>
<td>
<table>
<tr><td>Оптимизация гиперпараметров</td><td>При выборе флага оптимизации не нужно вручную задавать глубину дерева, или можно задать несколько значений на выбор. Алгоритм подбирает глубину дерева из расчета получения максимального значения метрики.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Метод, который рассчитывает точность обученной модели. Выбирается один из предлагаемых методов.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Датасет делится на фолды – на указанное количество равных частей. При обучении модели каждый фолд становится валидационным один раз, при этом на остальных фолдах выполняется обучение. Каждый раз рассчитывается значение метрики. Затем рассчитывается усредненная метрика, которая характеризует точность модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Случайный лес для классификации</td>
<td>Строится множество решающих деревьев, и в задаче классификации принимается решение голосованием по большинству. Все деревья строятся независимо по схеме:
1.Выбирается подвыборка обучающей выборки и по ней строится дерево.</td>
<td>
<table>
<tr><td>Глубина дерева</td><td>Максимальная глубина для деревьев решений.</td></tr>
<tr><td>Количество деревьев</td><td>Число деревьев в «лесу».</td></tr>
<tr><td>Порог классификации</td><td>–</td></tr>
</table>
</td>
<td>**</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>2.Для построения каждого расщепления в дереве просматривается максимальное количество случайных признаков.
3.Выбирается наилучший признак и расщепление по нему (по заранее заданному критерию). Дерево строится, до достижения параметра, ограничивающего его высоту.
Таким образом деревья обучаются не только на разных наборах данных, но и используют разные признаки для принятия решений – это создает некоррелированные деревья, которые и защищают друг друга от своих ошибок. Прогноз получается точнее, чем у любого отдельного дерева.</td>
<td>
<table>
<tr><td>Флаг возврата вероятности при прогнозе</td><td>–</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Необходимо активировать галочку в поле, чтобы подобрать гиперпараметры – глубину и количество деревьев. Гиперпараметры подбираются таким образом, чтобы получить максимальное значение метрики.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Categorical Naive Bayes</td>
<td>Группа байесовских классификаторов позволяет определить к какому классу принадлежит объект на основе теоремы Байеса с допущением о независимости признаков. Категориальный наивный байесовский классификатор применяется для признаков</td>
<td>
<table>
<tr><td>Параметр сглаживания Лапласа</td><td>Значение для аддитивного сглаживания Лапласа во избежание проблемы нулевой вероятности. Если равен нулю, то сглаживания нет.</td></tr>
<tr><td>Априорные вероятности классов</td><td>Определяет, будут ли взяты в расчет априорные вероятности классов. Если не активирован, то применяются значения вероятностей для равномерного</td></tr>
</table>
</td>
<td>1.Датасет с меткой класса, определяющей принадлежность объекта к одному из классов.
2.Модель ИИ, обученная классифицировать данные по заданным критериям.
3.Журнал преобразований над данными.
4.Словарь с переменными (описание модели, таблицы,</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>с категориальным распределением.</td>
<td>
<table>
<tr><td></td><td>распределения.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Определяет, будет ли проводиться оптимизация гиперпараметров при обучении модели</td></tr>
<tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr>
</table>
</td>
<td>графики) для отображения в интерфейсе Программы.</td>
</tr>
<tr>
<td>Multinomial Naive Bayes</td>
<td>Мультиномиальный классификатор применяется для признаков с полиномиальным распределением. Пример: классификация текстов, где каждый текст представлен вектором слов (например, мешок слов или tf-idf).</td>
<td>
<table>
<tr><td>Параметр сглаживания Лапласа</td><td>Значение для аддитивного сглаживания Лапласа во избежание проблемы нулевой вероятности. Если равен нулю, то сглаживания нет.</td></tr>
<tr><td>Априорные вероятности классов</td><td>Определяет, будут ли взяты в расчет априорные вероятности классов. Если не активирован, то применяются значения вероятностей для равномерного распределения.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Определяет, будет ли проводиться оптимизация гиперпараметров при обучении модели</td></tr>
<tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr>
</table>
</td>
<td>1.Датасет с меткой класса, определяющей принадлежность объекта к одному из классов.
2.Модель ИИ, обученная классифицировать данные по заданным критериям.
3.Журнал преобразований над данными.
4.Словарь с переменными (описание модели, таблицы, графики) для отображения в интерфейсе Программы.</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr></table></td><td></td></tr>
<tr><td>Complement Naive Bayes</td><td>Представляет собой вариант адаптации Multinomial Naive Bayes для датасетов с несбалансированными классами.
Вместо вычисления вероятностей принадлежности объекта к конкретному классу для каждого класса вычисляются вероятности того, что объект им не принадлежит.
Выбирается наименьшая вероятность "непринадлежности" к классу, так как это означает, что объект с наибольшей вероятностью принадлежит к данному классу.</td><td><table><tr><td>Параметр сглаживания Лапласа</td><td>Значение для аддитивного сглаживания Лапласа во избежание проблемы нулевой вероятности. Если равен нулю, то сглаживания нет.</td></tr><tr><td>Априорные вероятности классов</td><td>Определяет, будут ли взяты в расчет априорные вероятности классов. Если не активирован, то применяются значения вероятностей для равномерного распределения.</td></tr><tr><td>Оптимизация гиперпараметров</td><td>Определяет, будет ли проводиться оптимизация гиперпараметров при обучении модели</td></tr><tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr><tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr></table></td><td>1.Датасет с меткой класса, определяющей принадлежность объекта к одному из классов.
2.Модель ИИ, обученная классифицировать данные по заданным критериям.
3.Журнал преобразований над данными.
4.Словарь с переменными (описание модели, таблицы, графики) для отображения в интерфейсе Программы.</td></tr>
<tr><td>Gaussian Naive Bayes</td><td>Для значений признаков для каждого класса строится распределение Гаусса. В качестве значений</td><td></td><td>1.Датасет с меткой класса, определяющей принадлежность объекта к одному из классов.</td></tr>
</table>

<table>
<tr>
<td></td>
<td>правдоподобия для признаков берутся значения функции Гаусса из конкретного распределения (соответствующее признаку и классу).</td>
<td>
<table>
<tr><td>Параметр сглаживания Лапласа</td><td>Значение для аддитивного сглаживания Лапласа во избежание проблемы нулевой вероятности. Если равен нулю, то сглаживания нет.</td></tr>
<tr><td>Априорные вероятности классов</td><td>Определяет, будут ли взяты в расчет априорные вероятности классов. Если не активирован, то применяются значения вероятностей для равномерного распределения.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Определяет, будет ли проводиться оптимизация гиперпараметров при обучении модели</td></tr>
<tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr>
</table>
</td>
<td>2.Модель ИИ, обученная классифицировать данные по заданным критериям.
3.Журнал преобразований над данными.
4.Словарь с переменными (описание модели, таблицы, графики) для отображения в интерфейсе Программы.</td>
</tr>
<tr>
<td>Bernoulli Naive Bayes</td>
<td>Применяется для признаков с биномиальным распределением. Пример: классификация текстов, где каждый текст представлен вектором наличия слов из словаря (1 - есть слово, 0 - нет).</td>
<td>
<table>
<tr><td>Параметр сглаживания Лапласа</td><td>Значение для аддитивного сглаживания Лапласа во избежание проблемы нулевой вероятности. Если равен нулю, то сглаживания нет.</td></tr>
<tr><td>Априорные вероятности классов</td><td>Определяет, будут ли взяты в расчет априорные вероятности классов. Если не активирован, то применяются значения</td></tr>
</table>
</td>
<td>1.Датасет с меткой класса, определяющей принадлежность объекта к одному из классов.
2.Модель ИИ, обученная классифицировать данные по заданным критериям.
3.Журнал преобразований над данными.
4.Словарь с переменными (описание модели, таблицы,</td>
</tr>
</table>

			вероятностей для равномерного распределения.		графики) для отображения в интерфейсе Программы.
			Оптимизация гиперпараметров	Определяет, будет ли проводиться оптимизация гиперпараметров при обучении модели	
			Метрика для оптимизации	Выбирается одна из предлагаемых метрик для оценки работы модели.	
			Количество фолдов для оптимизации	Указывается, на сколько равных частей разбивается входной датасет при обучении модели.	

Кластеризация – это задача группировки множества объектов на подмножества (кластеры) так, чтобы объекты одного кластера были более похожи друг на друга, чем на объекты других кластеров по какому-либо критерию. Относится к классу задач *обучения без учителя*.

<table>
<tr>
<td>Алгоритм кластеризации DBSCAN</td>
<td>Алгоритм DBScan формирует группы коренных соседей/кластеры, объединяя точки, расположенные рядом. А точки, которые не попадают ни в одну из групп, отмечаются меткой -1 и приравниваются к аномалиям.</td>
<td>
<table>
<tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr>
<tr><td>Журнал преобразований</td><td>–</td></tr>
<tr><td>Радиус</td><td>Радиус в единицах расстояния, в рамках которого выполняется поиск потенциальных соседей (float/list/tuple).</td></tr>
<tr><td>Число соседей</td><td>Минимальное число ближайших соседей в указанном радиусе для формирования группы коренных соседей (int/list/tuple).</td></tr>
<tr><td>Метрика расстояния</td><td>Метрика расстояния (str/list): расстояние Евклида, косинусное расстояние. По умолчанию «Евклидово расстояние» – используется при кластеризации данных в текущем датасете, а также при отнесении нового объекта к кластеру.</td></tr>
<tr><td>Оптимизация гиперпарамет ров</td><td>Флаг подбора гиперпараметров. При значении «true» выполняется ручной ввод следующих гиперпараметров: радиус, число соседей, метрика расстояния. При значении «false» эти гиперпараметры подбираются автоматически.</td></tr>
</table>
* Параметры: датасет и журнал преобразований являются входными и выходными параметрами для всех алгоритмов.
</td>
<td>1.Модель кластеризации.
2.Выходной датасет, дополненный меткой кластера и/или флагом аномалии.
3. Словарь, содержащий информацию (графики, таблицы, текст) для отображения в пользовательском интерфейсе.
4.Журнал преобразований.
**Для алгоритмов кластеризации, регрессии, классификации наборы выходных параметров идентичны, отличие заключается в их содержимом.</td>
</tr>
</table>

<table>
<tr>
<td>Изоляционный лес</td>
<td>Алгоритм поиска аномалий (выбросов) методом «Изоляционный лес»: Изолирует наблюдения, случайным образом выбирая объект, а затем случайным образом выбирая разделения между максимальным и минимальным значениями объекта. Разбиение представлено древовидной структурой, количество разбиений, необходимое для изоляции выборки, равно длине пути от корневого до конечного узла. Эта длина пути является мерой нормальности и функции принятия решений. Когда лес случайных деревьев создает более короткие пути для отдельных объектов, они, скорее всего, являются аномалиями.</td>
<td>
<table>
<tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr>
<tr><td>Журнал преобразований</td><td>–</td></tr>
<tr><td>Количество деревьев</td><td>По умолчанию устанавливается значение, равное 2</td></tr>
</table>
</td>
<td>**</td>
</tr>
<tr>
<td>Кластеризация K-Means</td>
<td>Алгоритм кластеризации K-средних:
1.Из исходного множества случайным образом выбирается K наблюдений, равное заданному количеству кластеров.
2.Для каждого наблюдения определяется ближайший к нему центр кластера</td>
<td>
<table>
<tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr>
<tr><td>Журнал преобразований</td><td>–</td></tr>
<tr><td>Число кластеров</td><td>По умолчанию устанавливается значение, равное 2.</td></tr>
</table>
</td>
<td>**</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>(измеряется Евклидового расстояние до центра). Образуются начальные кластеры.
3.Вычисляются центры тяжести кластеров – вектора, элементы которых представляют собой среднее арифметическое значение признаков кластера.
4.Центры кластеров смещаются и объединяют вокруг себя наблюдения, пока центры и границы кластеров не перестанут изменяться.</td>
<td>
<table>
<tr><td>Оптимизация гиперпараметров</td><td>Флаг подбора гиперпараметров.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Агломеративная иерархическая кластеризация</td>
<td>Последовательно объединяет объекты во все более крупные подмножества, в результате образуется древовидная структура. Отдельные версии иерархии отличаются правилами вычисления расстояния между кластерами. Например, алгоритм средней связи на каждом шаге объединяет два ближайших кластера, рассчитывая среднюю арифметическую</td>
<td>
<table>
<tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr>
<tr><td>Журнал преобразований</td><td>–</td></tr>
<tr><td>Число кластеров</td><td>Задается оптимальное количество кластеров.</td></tr>
<tr><td>Метрика расчета расстояния</td><td>Используется при кластеризации данных в текущем датасете, а также при отнесении нового объекта к кластеру. Значения: евклидово расстояние, косинусная мера, расстояние городских кварталов, расстояние Чебышева.</td></tr>
</table>
</td>
<td>**</td>
</tr>
</table>

	дистанцию между всеми парами объектов.	**Критерий связи для расчета расстояния**	Правила вычисления расстояния между кластерами при каждой итерации. Значения: -алгоритм средней связи, -алгоритм одиночной связи или ближайшего соседа, -алгоритм полной связи или дальнего соседа, -метод минимума дисперсии Уорда. *Дисперсия* объединяет кластеры с минимальной общей внутрикластерной дисперсией после слияния, в качестве метрики расстояния используется евклидово расстояние. Минимальный использует самые близкие точки в обоих кластерах для расчета расстояния.	
		Оптимизация гиперпараметров	Флаг подбора гиперпараметров.	
Метод локтя K-Means	Метод локтя позволяет вычислить правильное значение k (количество кластеров) и повысить производительность модели. Вычисляется сумма квадратов расстояний между точками, и среднее арифметическое значение (Mean) – сумма элементов датасета, разделенная на их количество. Когда значение	**Число кластеров**	Задается оптимальное количество кластеров.	**

<table>
<tr><td></td><td>k равно 1, сумма квадрата внутри кластера будет большой. По мере увеличения значения k сумма квадратов внутри кластера будет уменьшаться. Наконец будет построен график между значениями k и суммой квадрата внутри кластера. В момент, когда значение k резко уменьшится будет считаться оптимальным числом кластеров.</td><td></td><td></td></tr>
<tr><td colspan="4">Регрессия – математическое выражение, отражающее связь между зависимой переменной y и независимыми переменными x. Алгоритмы регрессии используются для контролируемого обучения моделей ИИ – так называемого обучения «с учителем», когда данные размечаются для помощи в прогнозировании. Сопоставляя входные данные и полученные результаты на точность, модель постепенно обучается прогнозировать числовые значения целевых переменных.</td></tr>
<tr><td>Линейная регрессия</td><td>Прогнозирует целевую переменную Y на основе одной или нескольких независимых переменных X. Для этого между X и Y строится линейная связь.</td><td><table><tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr><tr><td>Журнал преобразований</td><td>–</td></tr></table></td><td>**</td></tr>
<tr><td>Дерево решений для регрессии</td><td>Предсказывает значение целевой переменной, изучая простые правила принятия решений, выведенные из характеристик данных. Представляет собой древовидный граф с узлами, где атрибут – вопрос, ребро – ответ на вопрос, а</td><td><table><tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr><tr><td>Журнал преобразований</td><td>–</td></tr><tr><td>Глубина дерева</td><td>Заданное максимальное число разбиений в ветвях, по достижению</td></tr></table></td><td>**</td></tr>
</table>

<table>
<tr>
<td></td>
<td>листья – фактический результат. Наблюдения классифицируются сверху вниз от корня до листьев.</td>
<td>
<table>
<tr><td></td><td>которого обучение останавливается.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>При выборе флага оптимизации не нужно вручную задавать глубину дерева, или можно задать несколько значений на выбор. Алгоритм подбирает глубину дерева из расчета получить максимальное значение метрики.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Метод, который рассчитывает точность обученной модели. Выбирается один из предлагаемых методов.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Датасет делится на фолды – на указанное количество равных частей. При обучении модели регрессии каждый фолд становится валидационным один раз, при этом на остальных фолдах выполняется обучение. Каждый раз рассчитывается значение метрики. Затем рассчитывается усредненная метрика, которая характеризует точность модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Случайный лес для регрессии</td>
<td>В отличие от предыдущего алгоритма здесь строится ансамбль решающих деревьев. При этом большое количество некоррелированных моделей (деревьев)</td>
<td>
<table>
<tr><td>Датасет</td><td>Датасет с исходными данными.</td></tr>
<tr><td>Журнал преобразований</td><td>–</td></tr>
<tr><td>Глубина дерева</td><td>–</td></tr>
</table>
</td>
<td>**</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>превосходит любую из отдельных моделей.</td>
<td>
<table>
<tr><td>Количество деревьев</td><td>–</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Алгоритм подбирает гиперпараметры: глубину и количество деревьев из расчета получить максимальное значение метрики.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td>Полиномиальная регрессия</td>
<td>Метод регрессионного анализа, в которой взаимосвязь между независимыми переменными x и зависимой переменной y моделируется как полином n-ой степени от x. Полиномиальная регрессия соответствует нелинейной зависимости между значением x и соответствующим условным средним y, обозначающим E(y|x).
В отличие от линейной регрессии моделирует нелинейно разделенные данные – более гибкая и может моделировать сложные взаимосвязи.</td>
<td>
<table>
<tr><td>Степень полинома</td><td>Степень уравнения полиномиальной регрессии, которая определяет линию наилучшего соответствия. При неправильном выборе степени, модель может быть перенасыщена. Значение по умолчанию – 2.</td></tr>
<tr><td>Только произведение</td><td>Если установить галочку в поле, то не выполняется возведение в степень, а только перемножение.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Нужно активировать галочку в поле, когда выбирается наиболее подходящая степень полинома из нескольких предложенных. А подбирается гиперпараметр так, чтобы получить наилучшее значение метрики.</td></tr>
</table>
</td>
<td>1.Модель полиномиальной регрессии.
2.Словарь с переменными для отображения в интерфейсе.
3.Словарь с преобразованиями данных.
4.Выходной датасет.</td>
</tr>
</table>

		Метрика для оптимизации	Значения на выбор: RMSE, MAE, WMAPE, где RMSE – среднеквадратическая ошибка, MAE – средняя абсолютная ошибка, а ошибка – разница между значениями, предсказанными моделью, и фактическими значениями переменной. Эти метрики используются для оценки работы модели регрессии – проверяют точность прогноза и измеряют величину отклонения от фактических значений.	
		Количество фолдов для оптимизации	Указывается, на сколько равных частей разбивается входной датасет при обучении модели.	
Метод опорных векторов для регрессии	В основе регрессии опорных векторов (с англ. **SVR** – Support Vector Regression) лежит поиск гиперплоскости, при которой риск в многомерном пространстве будет минимальным. SVR оценивает коэффициенты путем минимизации квадратичных потерь: считается сумма квадратов ошибок (между прогнозом и фактом), и к ней прибавляется штраф в виде произведения *коэффициента регуляризации* и суммы квадратов весов.	**Тип ядра**	Функция ядра (**kernel**) может принимать значения: {‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’}.	**
		Степень для ядра полинома	Если в качестве функции ядра используется полиномиальная функция (‘poly’), которая является методом нелинейной регрессии, то зависимая переменная связана с независимыми переменными n-ой степени. В поле указывается степень этого ядра.	
		Коэффициент регуляризации	Мера степени наказания модели за каждую неверно спрогнозированную точку.	
		Оптимизация	Флаг подбора гиперпараметров.	

	*Вместо квадратичной функции используется кусочно-линейная, и задается отступ *eps* (по умолчанию, равная 0.1): Если разница между прогнозируемым и истинным значением меньше eps (прогнозное значение попадает в пространство гиперплоскости), модель не считает это за ошибку, иначе – берется модуль разницы.	**гиперпараметров** |	
		Метрика для оптимизации | Выбирается одна из предлагаемых метрик для оценки работы модели.	
		Количество фолдов для оптимизации | Указывается, на сколько равных частей разбивается входной датасет при обучении модели.	
Байесовская гребневая регрессия	В основе метода лежит формула Байеса, которая дает возможность оценить вероятность событий эмпирическим путем. *Гребневая* регрессия – один из методов снижения размерности. Для гребневой регрессии к функции потерь прибавляется параметр **lambda**, обозначающий размер штрафа. Чем меньше **lambda**, тем выше *дисперсия* и ниже *смещение*. Смещение – это погрешность оценки, возникающая в результате ошибочного	**alpha_1**, **alpha_2** | Допустимые максимальные расстояния графика регрессии до верхнего и нижнего доверительного интервала.	**
		lambda_1, **lambda_2** | Размеры штрафов при выходе прогнозируемых значений за пределы верхнего и нижнего доверительного интервала.	
		Оптимизация гиперпараметров | Флаг подбора гиперпараметров.	
		Метрика для оптимизации | Выбирается одна из предлагаемых метрик для оценки работы модели.	
		Количество фолдов для | –	

	предположения в алгоритме обучения. В результате большого смещения алгоритм может пропустить связь между признаками и выводом (недообучение). Дисперсия – это ошибка чувствительности к малым отклонениям в тренировочном наборе. При высокой дисперсии алгоритм может трактовать случайный шум в тренировочном наборе, а не желаемый результат (переобучение).	**оптимизации**	
Метод k-ближайших соседей для регрессии	Для регрессии объекту присваивается среднее значение по *k* ближайшим к нему объектам, значения которых уже известны. Алгоритм применяется к выборке с большим количеством атрибутов (многомерной). Для этого перед применением определяется функция расстояния, классический вариант такой функции – *евклидова метрика*. Разные признаки могут иметь разный диапазон	**Количество ближайших соседей**: Число *k*, характеризующее количество соседей в кластере. **Тип веса для соседей**: Задается одно из значений: ‘uniform’ (единый – всем признакам присваивается единый вес), или ‘distance’ (по расстоянию). Значение по умолчанию – единый. **Метрика расстояния**: Задается одно из значений: ‘chebyshev’ (Чебышева), ‘euclidean’ (Евклидова), ‘cosine’ (Косинусное), ‘cityblock’ (Манхэттенское).	**

<table>
<tr>
<td></td>
<td>представленных значений в выборке, поэтому выполняется нормализация данных.
Некоторые значимые признаки могут быть важнее остальных, поэтому для каждого признака задается определенный вес.
Алгоритм предполагает, что похожие наблюдения существуют в непосредственной близости: улавливается идея сходства (иногда называемого расстоянием или близостью) благодаря вычислению Евклидова расстояния между точками.</td>
<td>
<table>
<tr><td></td><td>Значение по умолчанию – евклидово расстояние, когда вычисляется расстояние между всеми точками попарно.</td></tr>
<tr><td>Оптимизация гиперпараметров</td><td>Флаг подбора гиперпараметров.</td></tr>
<tr><td>Метрика для оптимизации</td><td>Выбирается одна из предлагаемых метрик для оценки работы модели.</td></tr>
<tr><td>Количество фолдов для оптимизации</td><td>Указывается, на сколько равных частей разбивается входной датасет при обучении модели.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td colspan="4">Авторегрессия – модель временного ряда, в которой ее текущее значение линейно зависит от предыдущих (ретроспективных) значений этого же ряда. Линейная зависимость означает, что текущее значение равно взвешенной сумме нескольких предыдущих значений ряда.
Зная параметры модели и соответствующие ретроспективные значения временного ряда, можно предсказать его будущие значения. Основное назначение авторегрессионной модели – прогнозирование. Также с ее помощью можно производить анализ временных рядов – выявлять тенденции, сезонность, и другие особенности.</td>
</tr>
<tr>
<td>ARIMA/ SARIMAX</td>
<td>Авторегрессионное интегрированное скользящее среднее (с англ. ARIMA – autoregressive integrated moving average) используется при работе с временными рядами для более глубокого понимания данных, или предсказания</td>
<td>
<table>
<tr><td>Число шагов для прогноза</td><td>Количество шагов, на которые модель будет предсказывать.</td></tr>
<tr><td>Порядок авторегрессии, p</td><td>Количество запаздывающих наблюдений, включенных в модель, также называется лаговый порядок. P помогает настроить линию для</td></tr>
</table>
</td>
<td>**</td>
</tr>
</table>

	будущих точек ряда. Упоминается как **ARIMA (p, d, q)**, где *p, d* и *q* – целые неотрицательные числа, характеризующие порядок для частей модели (соответственно – авторегрессионной, интегрированной и скользящего среднего). **Авторегрессия**. Модель, использующая зависимую связь между наблюдением и некоторым количеством запаздывающих наблюдений. **Интегрированный**. Использование разности необработанных наблюдений (например, вычитание наблюдения из наблюдения на предыдущем временном шаге), чтобы сделать временной ряд стационарным. **Скользящая средняя**. Модель, в которой используется зависимость между наблюдением и остаточной ошибкой из модели скользящего среднего, применяемая к запаздывающим наблюдениям.				

	прогнозирования серии. Чисто авторегрессионные модели напоминают линейную регрессию, где прогностическими переменными являются p числа предыдущих периодов.
Порядок интегрирования, d	Число обычных дифференцирований – количество раз, когда необработанные наблюдения различаются, также называется степенью различия. В модели ARIMA временные ряды преобразуются в *стационарные* (серии без тренда и сезонности), используя дифференцирование. Стационарный ряд – это когда среднее значение и дисперсия постоянны во времени.
Порядок скользящего среднего, q	Размер окна скользящей средней.
Параметры модели SARIMAX:	
Порядок авторегрессии	–
Порядок интегрирования	Число сезонных дифференцирований.
Порядок скользящего среднего	–
Сезонный	Число наблюдений за сезон.

	Модель **SARIMAX** используется для временных рядов с учетом сезонности.	**период** **Оптимизация гиперпараметров** — Флаг подбора гиперпараметров.	
Группа «Работа с текстами»			
Автореферирование текста	Данная функция представляет собой автоматический процесс выделения краткого содержания текста с помощью модели машинного обучения. На выходе получается датасет заданного объема, который можно представить в виде таблицы.	**Объем автореферата** — Максимальное количество символов в выходном результате	Таблица с кратким содержанием
Группа «Управление моделями»			
Сохранение модели	Сохраняет модель по настроенному в системе пути и названию файла, а также сохраняет словарь с переменными. В этом словаре содержится отдельно список независимых переменных и список целевых признаков, с указанием выполненных над ними преобразований. Сохраняет шаг ресемплирования	**Название модели** — Пользователь задает название для обучаемой модели.	Созданная модель сохраняется в пункте меню системы **Модели** –> **Сохранённые модели**
Классификация			

Сохранение модели классификации изображений	Функция предназначена для сохранения в системе модели классификации изображений.	**Название модели** — Пользователь задает название для обучаемой модели.	**
Обнаружение объектов			
Сохранение модели YOLO v5	**** данный функционал находится в разработке, в текущей версии 2.3.3 применение функции недоступно.*** Функция предназначена для сохранения в системе модели распознавания изображений «YOLO v5». Сохранив модель, можно создать на ее основе приложение для последующей интеграции со сторонними системами. Также обученная модель может использоваться повторно для анализа онлайн данных.	**Название модели** — Пользователь задает название для обучаемой модели.	**
Spark			
Сохранение модели Spark	Функция предназначена для сохранения в системе моделей, собранных с применением технологии Spark.	**Названи е модели** — Пользователь задает название для обучаемой модели.	
Группа «Глубокое обучение»			

<table>
<tr>
<td>Валидация модели классификации изображений</td>
<td>После того, как модель нейронной сети обучена, натренирована и для нее выбраны оптимальные гиперпараметры,
необходимо проверить ее точность и адекватность. Для этого выполняется валидация итоговой модели нейронной сети на тестовой выборке.
В качестве входных данных для функции используются:
-тестовый датасет с изображениями;
- обученная модель;
-словарь с преобразованиями данных;
-выбранная метрика валидации.
Рассчитывается сколько изображений тестовой выборки попадают в каждую ячейку матрицы ошибок.
Оценивается качество классификации.</td>
<td>
<table>
<tr>
<td>Метрика</td>
<td>Метрика, которая оценивает работу обученной модели нейронной сети. Применяется к типу данных – изображения.
Чтобы оценить качество модели классификации используются следующие метрики:
1. Accuracy – оценивает долю правильных ответов модели.
2. F1 (среднее гармоническое) – агрегированная функция, которая позволяет вместо точности и полноты использовать только один параметр качества классификации. Формула:
$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$
Чем ближе F1 к 1, тем лучше.
*В задачах, в которых точность и полнота не равноценны, применяется взвешенное значение F_β.
3. Precision. Метрика, которая оценивает точность модели. Рассчитывается по формуле:
$$precision = \frac{TP}{TP + FP},$$
здесь считается точность для класса 1 (для класса 0 считается аналогично).
4. Recall – оценивает полноту модели:
$$recall = \frac{TP}{TP + FN}.$$
Идеально, чтобы точность и полнота были равны 1 (100%).</td>
</tr>
</table>
</td>
<td>1.Строится матрица ошибок. Пример для бинарной классификации:
<table>
<tr><td></td><td colspan="2">Истинный класс</td></tr>
<tr><td></td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>TN</td><td>FN</td></tr>
<tr><td>1</td><td>FP</td><td>TP</td></tr>
</table>
где столбцы – истинные классы, а строки – предсказанные классы.
Обозначения:
TN – true negative
TP – true positive
FN – false negative
FP – false positive
Например, ячейка TP означает, что объект действительно принадлежит классу 1, и для него предсказан класс 1. А FN – объект неправильно отнесли к классу 0, хотя он принадлежит к классу 1.
2.Отображается значение выбранной метрики.</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td></td><td>5. AUC_ROC. Для анализа качества модели применяется ROC-анализ: строится ROC-кривая, которая наиболее часто используется для представления результатов бинарной классификации. Классов два: один называется классом с положительными исходами, второй – с отрицательными исходами. ROC-кривая показывает зависимость количества верно классифицированных положительных примеров от количества неверно классифицированных отрицательных примеров.
Чем выше показатель AUC_ROC, тем качественнее классификатор, при этом значение 0,5 демонстрирует непригодность выбранного метода классификации. AUC_ROC = 0,9-1,0 означает отличное качество модели.</td><td></td></tr>
<tr><td colspan="5">Классификация</td></tr>
<tr><td>Классификация (табличные данные)</td><td>Типы решаемых задач: анализ тональности, классификация текста по категориям, распознавание речи и многое другое.
Рассмотрим функцию на примере задачи по отнесению документов к определенной категории на основании его содежимого. Процесс классификации осуществляется с помощью применения методов машинного обучения, в частности сверточных нейронных сетей. Задача</td><td></td><td><table><tr><td>Количество эпох</td><td>Параметр, который показывает сколько раз модель подвергается воздействию обучения.</td></tr><tr><td>Размер мини-батча</td><td>Количество обучающих примеров за одну итерацию. Под примерами имеются в виду наблюдения – строки в табличных данных.</td></tr><tr><td>Метрика для обучения</td><td>Метрика «Accuracy» (точность) показывает долю правильных ответов алгоритма</td></tr></table></td><td>**</td></tr>
</table>

	классификации текстов применима в решении следующих задач: борьба с массовой рассылкой рекламы, распознавание тональности текстов, сортировка документов и т.д. Задача определяется следующим образом: пусть существует конечное множество категорий, на вход алгоритма подается конечное количество документов, и есть целевая функция, которая определяет соответствие для каждой пары (документ, категория). Задача состоит в нахождении этой функции, называемой классификатором. Строится многослойная нейронная сеть, состоящая из слоев: -*входной*, на который поступают входные признаки; -*скрытый*, на котором рассчитываются промежуточные результаты; - *выходной*, на котором выводятся окончательные значения, вычисленные по гипотезе.		**Алгоритм градиентного спуска**	Метод нахождения минимального значения *функции потерь*. Алгоритмы на выбор: *SGD*, *Adam*, и др.	
			Шаг градиентного спуска	Параметр, регулирующий скорость обучения модели. Значения – 0.001 или 0.1.	
			Функция потерь	Выбирается одна из функций, в зависимости от задачи: бинарная классификация или многоклассовая.	
			Добавить слой	Задается одно из трех значений: Conv2D, Flatten, Dense (последний – полносвязный слой).	
			Перемешивать выборку перед обучением	Необходимо установить галочку в поле, чтобы случайным образом поменять местами наблюдения, соответствующие строкам в таблице	
			Порог классификации	Для бинарной классификации значение по умолчанию 0.5, для многоклассовой – параметр не заполняется	
			Флаг возврата вероятности при прогнозе	Возвращает вероятность или метки классов для дальнейшего прогноза после обучения модели. Для бинарной классификации может использоваться одно поле с метками 0 и 1, обозначающими принадлежность к тому или иному классу. Для многоклассовой классификации используется несколько полей –	

	**Сверточными* искусственные нейронные сети называются из-за специальной архитектуры, с наличием операций сверки.	**Оптимизация гиперпараметров**: Алгоритм подбирает гиперпараметры **Метрика для оптимизации**: Выбирается одна из предлагаемых метрик для оценки работы модели. **Количество фолдов для оптимизации**: –	
Классификация изображений	Алгоритм решения задачи классификации: 1.Берется *тренировочная выборка* – набор изображений с известными значениями целевого признака Y. Нейронная сеть должна восстановить зависимость между нецелевыми признаками и целевым. 2.Задаются основные параметры нейронной сети. 3.Выписываются выражения для вероятностей принадлежности наблюдения к тому или иному классу (Y = 0, 1, 2, 3, и т.д.).	**Количество эпох**: Это гиперпараметр, который определяет сколько раз *алгоритм обучения* будет обрабатывать весь *набор обучающих данных*. То есть *эпоха* – одна итерация в процессе обучения, включающая предъявление всех примеров из обучающего множества **Метрика для обучения**: Метрика «Accuracy» (точность) показывает долю правильных ответов алгоритма **Алгоритм градиентного спуска**: Метод нахождения минимального значения *функции потерь*. Минимизация любой функции означает поиск самой глубокой впадины в этой функции. Функция используется, чтобы контролировать ошибку в прогнозах	**

	4.По тренировочной выборке составляется функция потерь. 5. Функция потерь L(w) содержит вхождения весов нейронной сети. Относительно этих переменных находится *точка минимума функции L(w)*. 6.Точка минимума определяет оптимальные веса нейронной сети. 7.Весам нейронной сети присваиваются найденные оптимальные значения. Пусть изображение А не принадлежит тренировочной выборке. Объект А прогоняется через нейронную сеть и на выходе получаются вероятности – они и являются предсказаниями для объекта А (по максимальной вероятности определяется принадлежность объекта к классу).			

	модели. Поиск минимума означает получение наименьшей возможности ошибки или повышение точности модели. Точность увеличивается перебором *учебных* данных при настройке параметров модели (весов и смещений). Суть алгоритма – процесс получения наименьшего значения ошибки. Аналогично это можно рассматривать как спуск во впадину в попытке найти самое низкое значение ошибки. Можно выбрать один из следующих алгоритмов: *SGD* (Stochastic gradient descent, с англ. Стохастический градиентный спуск), *Adam*, и др. Алгоритм **Adam** является *модифицированным*, в нем также выполняется минимизация функции потерь. Рассчитываются векторы *частных* в текущей точке функции, и определяются координаты следующей точки. Частные производные вычисляются, чтобы определить, какой был вклад в ошибку по каждому весу.
Шаг градиентного спуска	Параметр, регулирующий скорость обучения модели – насколько быстро функция потерь спускается к своему минимуму (скорость спуска/поиска). Выбирается значение: 0.001, или 0.1.
Функция потерь	Функция потерь находится в центре нейронной сети. Она используется для расчета ошибки между *реальными* и

				полученными ответами. Главная цель – минимизировать эту ошибку. Или: **максимизировать вероятность принадлежности к истинному классу для каждого объекта из тренировочной выборки**. Она также может зависеть от таких переменных, как веса и смещения, где *смещения* – это веса, добавленные к скрытым слоям. Выбирается одна из функций, в зависимости от задачи: бинарная классификация или многоклассовая.	
			Добавить слой	Основой алгоритмов распознавания изображений являются сверточные нейронные сети. Для их построения используются три главных типа слоев: сверточный слой, слой подвыборки и полносвязный слой. Соответственно пользователь задает одно из трех значений: Conv2D, Flatten, Dense (последний – полносвязный). В сверточных нейронных сетях одно изображение является одним наблюдением. Таким образом, исходное изображение преобразуется, слой за слоем, от начального значения пикселя до итоговой оценки класса. Слои, идущие до полносвязного, являются средствами предобработки изображения, и используются для выделения различных признаков, которые затем подаются на вход классификатору.	

		Порог классификации	**Важно!** Параметр заполняется только для задачи бинарной классификации, значение по умолчанию 0.5. Для множественной классификации это поле остается пустым.	
Регрессия				
Регрессия (табличные данные)	Для обучения нейронной сети данные делятся на части меньшего размера, загружают их по очереди и обновляют веса нейросети в конце каждого шага, подстраивая их под данные.	**Количество эпох**	Указывается количество эпох для обучения модели. Одна эпоха – весь датасет прошел через нейронную сеть в прямом и обратном направлении только один раз. Так как одна эпоха слишком велика для компьютера, датасет делят на партии – *батчи*. С увеличением числа эпох, веса нейронной сети изменяются все большее количество раз. Кривая с каждым разом лучше подстраивается под данные, переходя последовательно из плохо обученного состояния в оптимальное. Если вовремя не остановиться, то может произойти переобучение.	**
		Размер мини-батча	Общее число тренировочных объектов, представленных в одном батче. Нельзя пропустить через нейронную сеть разом весь датасет. Поэтому делим данные на пакеты, сеты или партии.	

	Итерации – это число батчей, необходимое для завершения одной эпохи.
Метрика для обучения	Для регрессии: ['MAE', 'MAPE', 'MSE'].
Алгоритм градиентного спуска	Алгоритм итеративной оптимизации, используемой в машинном обучении для получения более точного результата. *Градиент* показывает скорость убывания или возрастания функции. *Спуск* говорит о том, что мы имеем дело с убыванием. Алгоритм **итеративный**, процедура проводится несколько раз, чтобы добиться оптимального результата. На каждом шаге результат получается лучше.
Шаг градиентного спуска	Скорость обучения модели алгоритмом градиентного спуска.
Функция потерь	Функция, которая используется для оптимизации алгоритма машинного обучения. Значение, вычисленное такой функцией, называется ‘потерей’. Потери регрессии рассчитываются путем прямого сравнения выходного и истинного значения. Самая популярная функция для регрессионных моделей – это среднеквадратическая ошибка, MSE. Функция потерь определяет, как именно выходные данные связаны с исходными.

			По сути вычисляется насколько хорошо работает модель – сравнивается то, что модель прогнозирует, с фактическим значением. Сохраняется функция потерь, которая может эффективно наказывать модель, пока та обучается на тренировочных данных.	
		Добавить слой	Из списка выбирается дополнительный слой для нейросети.	
		Перемешивать выборку перед обучением	Необходимо установить галочку в поле, чтобы случайным образом поменять местами наблюдения.	
		Оптимизация гиперпараметров	Если установить галочку в поле, то алгоритм выберет наилучшие гиперпараметры для создания модели из списка предложенных	
		Метрика для оптимизации	Выбирается одна из предлагаемых метрик для оценки работы модели.	
		Количество фолдов для оптимизации	Указывается, на сколько равных частей разбивается входной датасет при обучении модели.	
Обнаружение объектов				
YOLOv5	**** данный функционал находится в разработке, в текущей версии 2.3.3 применение функции недоступно.*** Датасет должен быть разделен на две папки: *train*	**Размер мини-батча**	Указывается количество изображений, которое одновременно подается на вход YOLO. Например, если задать размер 2,	В результате получаются изображения с обозначением детектированных объектов и значениями *confidence*. Где confidence – число от 0 до 1, характеризующее ‘уверенность’

<table>
<tr>
<td></td>
<td>(тренировочная выборка) и val (валидационная). В каждой папке лежат еще две папки: images (изображения) и labels – папка с текстовыми файлами, содержащими метки объектов на этих изображениях в формате YOLO.
Для этой функции предварительно выполняется разметка изображений на тренировочной и валидационной выборках. Пользователь с помощью ‘bounding box’ отмечает объекты на изображениях. А алгоритм по точкам объектов находит функцию их обнаружения. На валидационной выборке проверяется точность обученной модели.
После запуска функции на выходе получается обученная нейронная сеть, результаты обучения которой сохраняются в БД.</td>
<td>
<table>
<tr><td></td><td>за один подход подается два изображения.</td></tr>
<tr><td>Количество эпох</td><td>Это гиперпараметр, который определяет сколько раз алгоритм обучения будет обрабатывать весь набор обучающих данных. То есть эпоха – одна итерация в процессе обучения, включающая предъявление всех примеров из обучающего множества. Например, для выборки в десять изображений и размера мини-батча два, эпоха равна пяти прохождениям.</td></tr>
</table>
</td>
<td>модели в том, что детектирован объект или детектирован объект определенного класса. Еще один параметр conf-thres позволяет установить пороговое значение для confidence модели. Все объекты, confidence которых ниже этого значения не считаются объектами.
Также отображаются: описание модели, графики обучения модели, матрица ошибок на валидационных данных. Где описание модели содержит информацию об оптимизаторе, тренировочном и валидационном датасетах, а также параметры обучения.</td>
</tr>
<tr><td colspan="4">Отправка уведомлений</td></tr>
<tr><td>Отправка уведомлений</td><td>Функция предназначена для осуществления отправки</td><td></td><td>Оповещение в телеграм канал</td></tr>
</table>

<table>
<tr><td></td><td>уведомлений в настроенный канал телеграм или по электронной почте (не реализовано в текущей версии). Данная функция выступает в паре с блоком шлюз, где необходимо задать условия, при которых будет отправлено уведомление.</td><td><table><tr><td>Канал уведомлений</td><td>Выбирается ранее настроенный в разделе Администрирование -> Уведомления канал, на который будет осуществляться отправка сообщений.</td></tr></table></td><td></td></tr>
<tr><td colspan="4">Spark. Группа функций для фреймворка Apache Spark. Названия функций дублируются с теми, что были описаны ранее, разница заключается в использовании другого модуля машинного обучения в Apache Spark (в следующих версиях Платформы планируется сделать все функции универсальными).</td></tr>
<tr><td>Сохранение датасета Spark в CSV</td><td>Функция распределяет входные данные в несколько файлов в одну директорию. Для этого выбираем: куда сохранить данные, как их назвать, и по необходимости можем подгрузить новую порцию данных.</td><td><table><tr><td>Путь до директории для датасета</td><td>Выбирается путь до папки, в которую будут сохраняться данные.</td></tr><tr><td>Название датасета</td><td>В этом поле задается название для датасета. По умолчанию датасеты создаются с названиями формата pySpark.csv.</td></tr><tr><td>Добавить данные к датасету</td><td>Если преобразованные данные необходимо сохранять не в виде отдельного файла, а добавить к уже существующему и загруженному на платформе, необходимо установить галочку у данного признака. По умолчанию файл перезаписывается.</td></tr><tr><td>Название датасета для валидации</td><td>Указывается название, с которым будет сохранен датасет для валидации при активации параметра «Сохранить датасет для валидации»</td></tr></table></td><td>Таблица в формате csv с датасетом. Сохраняется в раздел данные</td></tr>
</table>

		Сохранить датасет для валидации	В процессе работы пайплайна, исходный вид набора данных данных теряется, поэтому его нужно передать из блока "Загрузка данных" в конец пайплайна в блок сохранени. Датасет для валидации это и есть нетронутый набор данных в первоначальном виде, к нему только добавляется столбец с результатами.	
		Загрузка датасета для валидации в БД	Позволяет загрузить датасет для вализации напрямую в базу данных ClickHouse	
Косинусное расстояние	На вход функция получает новые данные для анализа (датасет в формате csv), обученную модель, и числовой вектор. Выполняется поиск объектов, наиболее схожих с заданным вектором, и в качестве меры схожести используется косинусное расстояние - расстояние между значениями во входном векторе и значениями выбранных столбцов в наблюдениях.	–		Таблица «Косинусное расстояние»
Выбор признаков и целевых признаков	Аналогично стандартной функции.			

<table>
<tr><td>Разделение датасета на обучающую и тестовую выборки</td><td>Аналогично стандартной функции.</td><td></td><td></td></tr>
<tr><td>Валидация модели</td><td>Аналогично стандартной функции.</td><td></td><td></td></tr>
<tr><td>Прогноз модели</td><td>Аналогично стандартной функции.</td><td></td><td></td></tr>
<tr><td>Порядковое кодирование признаков</td><td>Порядковое кодирование - это метод преобразования категориальных данных в цифровой вид. Применяются, когда в датасете существуют НЕ числовые признаки, которые заданы словами и для дальнейшего анализа их нужно преобразовать в числа. Порядковое кодирование позволяет пронумеровать признаки по порядку.</td><td><table><tr><td>Выбранные признаки</td><td>Указываются признаки, над которыми необходимо провести операцию порядкового кодирования.</td></tr></table></td><td>-</td></tr>
<tr><td>Нормализация признаков</td><td>Нормализация - это приведение числовых признаков к единой шкале. Бывает что числовой признак имеет минимальное и максимального значение в очень широком диапазоне и это плохо для машинного обучения. Например, есть числовой признак, чье минимальное значение равно 0,001, а максимальное</td><td>–</td><td>–</td></tr>
</table>

<table>
<tr><td></td><td>- 100000, нормализация преобразовывает их к диапазону от 0 до 1, то есть 0.001 становится 0, а 100000 становится 1, значения между ними также преобразуются, 50 000 станет примерно равным 0.5. Данная функция позволяет оптимизировать дальнейшие вычисления.</td><td></td><td></td></tr>
<tr><td>Модель градиентного бустинга Spark для бинарной классификации</td><td>Градиентный бустинг представляет собой ансамбль деревьев решений. В основе данного алгоритма лежит итеративное обучение деревьев решений с целью минимизировать функцию потерь. Благодаря особенностям деревьев решений градиентный бустинг способен работать с категориальными признаками , справляться с нелинейностями. Бустинг – это метод преобразования слабообученных моделей в хорошообученные. В бустинге каждое новое дерево обучается на модифицированной версии исходного датасета.</td><td><table><tr><td>Количество базовых моделей</td><td>Указывается число последовательных итераций по оптимизации модели, то есть количество попыток уменьшить ошибку.</td></tr></table></td><td>–</td></tr>
</table>

19. Лист изменений

Таблица 19.1 – Лист изменений в версии 1.0.0

Наименование раздела	Содержание изменения	Обоснование
-	-	-