ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«Университет «Дубна»

ИНСТИТУТ СИСТЕМНОГО АНАЛИЗА И УПРАВЛЕНИЯ

Кафедра системного анализа и управления

КУРСОВАЯ РАБОТА

по дисциплине

«Теория принятия решений»

ТЕМА: Разработка веб-приложения для предобработки данных и построения моделей машинного обучения.

Выполнил: студент группы 2251

Белоусов Евгений Сергеевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

Руководитель:

ст. преп. Булякова Ирина Александровна

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Дубна, 2025

Содержание.

[Введение. 3](#_Toc198493355)

[Постановка задачи. 3](#_Toc198493356)

[Цель курсовой работы. 3](#_Toc198493357)

[Исходные данные. 3](#_Toc198493358)

[Модельные представления. 3](#_Toc198493359)

[Ожидаемый результат. 3](#_Toc198493360)

[Критерии оценки результата. 3](#_Toc198493361)

[Теоретическая часть. 4](#_Toc198493362)

[Предобработка данных. 4](#_Toc198493363)

[Заполнение пропусков. 4](#_Toc198493364)

[Работа с выбросами. 4](#_Toc198493365)

[Кодирование категориальных признаков. 5](#_Toc198493366)

[Нормализация. 5](#_Toc198493367)

[Используемые алгоритмы машинного обучения. 6](#_Toc198493368)

[Практическая часть. 7](#_Toc198493369)

[Формулировка задач. 7](#_Toc198493370)

[Этапы реализации. 7](#_Toc198493371)

[Результат решения. 9](#_Toc198493372)

[Анализ полученного результата. 10](#_Toc198493373)

[Заключение. 10](#_Toc198493374)

[Список литературы. 11](#_Toc198493375)

# Введение.

В современном мире накапливается всё больше информации, как следствие – данных, но работа с ними требует достаточно высокой квалификации, что не позволяет работникам не отрасли информационных технологий быстро и качественно получать необходимые выводы из накопленных сведений. Данная работа посвящена попытке решить эту проблему, а так же может служить наглядным пособием необходимости предварительной обработки данных, перед извлечением из них информации.

# Постановка задачи.

## Цель курсовой работы.

Целью моей курсовой работы было создать веб-приложение, реализующее обработку датасетов, и построение классических моделей машинного обучения.

## Исходные данные.

Возможности языка *Python*, его библиотек для анализа данных, фреймворка *Django*, собственные знания по исследуемой теме.

## Модельные представления.

Веб-приложение с интуитивно понятным интерфейсом, позволяющее пользователю загрузить датасет в формате *.csv* или *.xlsx* и обработать пропуски, выбросы, проводить нормализацию числовых признаков и кодирование категориальных переменных. Также реализована возможность на подгруженных данных построить *ML* модель классификации.

## Ожидаемый результат.

Приложение на локальном сервере, которое выполняет заявленный функционал, не имеющее различных багов.

# Критерии оценки результата.

* Реализованные возможности, отрабатывают без ошибок и корректно обрабатывают данные.
* Сайт не виснет при объёмах менее 15000 объектов в датасете и выдаёт результат менее чем за 1 секунду.

# Теоретическая часть.

## Предобработка данных.

### Заполнение пропусков.

Так как ни один алгоритм классического машинного обучения не умеет работать с пропусками (кроме предсказания деревьев), важным и обязательным пунктом перед любыми дальнейшими действиями я сделал заполнение пропусков в данных, в моей работе реализованы 5 способов:

* Удаление – потеря информации, но мы не привносим ложных значений в датасет.
* Заполнение средним значением признака – метод чувствителен к выбросам, неплохо подходит для простых случаев.
* Заполнение медианным значением признака – менее чувствителен к выбросам, для особых случаев не подходит полностью.
* Заполнение модой – подходит и для категориальных признаков, что делает его более гибким.
* Заполнение константой – метод необходим, если пользователь знает значение пропусков в признаках.

Работа с выбросами.

Объекты с аномальными значения того или иного признака могут помешать найти тенденцию или связь как на графике, так и алгоритмам. Поэтому важно перед дальнейшим исследованием обработать выбросы. В моей работе реализовано 4 варианта, как можно это сделать:

* Правило трёх сигм – значения признака, которые на лежат в диапазоне , где μ - математическое ожидание, σ - стандартное отклонение, просто удаляются. Правило плохо работает с не симметричными распределениями.
* Метод межквартильного размаха (*IQR method*) – также рассчитываются верхняя и нижняя граница и удаляется всё, что не принадлежит диапазону. Формулы выглядят следующим образом:

Итоговый диапазон равен . Плюсы этого метода, в том, что он менее чувствителен к выбросам и форме распределения, чем правило трёх сигм.

* *Winsorization* метод – тут также работает правило трёх сигм, только вместо удаления экстремальных значений мы их заменяем на граничные.
* *CAP* – это метод, который заменяет значения, выходящие за границы межквартильного размаха, на эти граничные значения.

Кодирование категориальных признаков.

Работа со строками, датами простыми методами очень трудна, поэтому на практике их часто кодируют числами, что было реализована в моей работе четырьмя способами:

* *One-Hot* кодирование – метод преобразования категориальных признаков в числовой формат, где каждая категория заменяется бинарным вектором (0 или 1).
* *Ordinal* кодирование – метод преобразования категориальных признаков в числовые значения с сохранением порядка (ранга) категорий. Подходит для данных с естественной порядковой зависимостью.
* *Target* кодирование - каждое значение категории заменяется средней величиной целевой переменной для этой категории.
* *Frequency* кодирование - каждое значение категории заменяется на частоту его встречаемости в данных (или относительную частоту).

Нормализация.

Линейные модели крайне чувствительны к размерности данных, что в последствии усложняет их интерпретацию, поэтому принято перед обучением приводить значения в одну размерность. Нормализация в моей работе была реализована путём преобразования представленных ниже.

* *Min-Max* стратегия – позволяет масштабировать данные в диапазон [0, 1]. Проста в вычислении и сохраняет распределении, но чувствительна к выбросам и не сохраняет дисперсию. Формула:

, где – минимальное значение признака, – максимальное.

* *Standard* нормализация – преобразует данные так, что среднее (μ) = 0, стандартное отклонение (σ) = 1, но не гарантирует фиксированный диапазон значений. Формула:
* *Robust* нормализация – вместо среднего и стандартного отклонения использует медиана и интерквартильный размах, что делает этот метод более устойчивым к выбросам. Формула:

*,* где – медиана, IQR – интерквартильный размах, расчёт которого приведён в пункте обработка выбросов.

* Логарифмическое преобразование – сжимает большие данные и расширяет малые, но не работает с отрицательными данными и может искажать данные, если распределение не логнормальное.

## Используемые алгоритмы машинного обучения.

После обработки данных моё приложение предоставляет возможность построить несколько классификаторов, для выбора наиболее подходящего и оценки их точности.

* Метод опорных векторов – строит разделяющую гиперплоскость, обеспечивающую максимальный зазор между классами.
* Логистическая регрессия – предсказывает вероятность принадлежности объекта к тому или иному классу, с помощью вычисления функции *f(z)*, где , z = линейной комбинации вектора признаков объекта.
* Случайный лес – из основной выборки делаются под выборки, на которых строится решающее дерево, результат принадлежности к тому или иному классу вычисляется как значение, за которое “проголосовало” большинство деревьев.

# Практическая часть.

## Формулировка задач.

1. Разработка веб-интерфейса для загрузки, обработки и предварительного просмотра датасетов.
2. Реализация модуля предобработки данных.
3. Интеграция алгоритмов машинного обучения и оценок качества их работы.

## Этапы реализации.

1. Проектирование архитектуры (см. рис. 1):
   1. Backend: *Django* (*Python*).
   2. Frontend: *HTML/CSS/JavaScript*.



Рисунок 1. Архитектура системы.

1. Реализация функционала:
   1. Загрузка и вывод данных: парсинг файлов с помощью библиотеки pandas (см. рис. 2.1.).

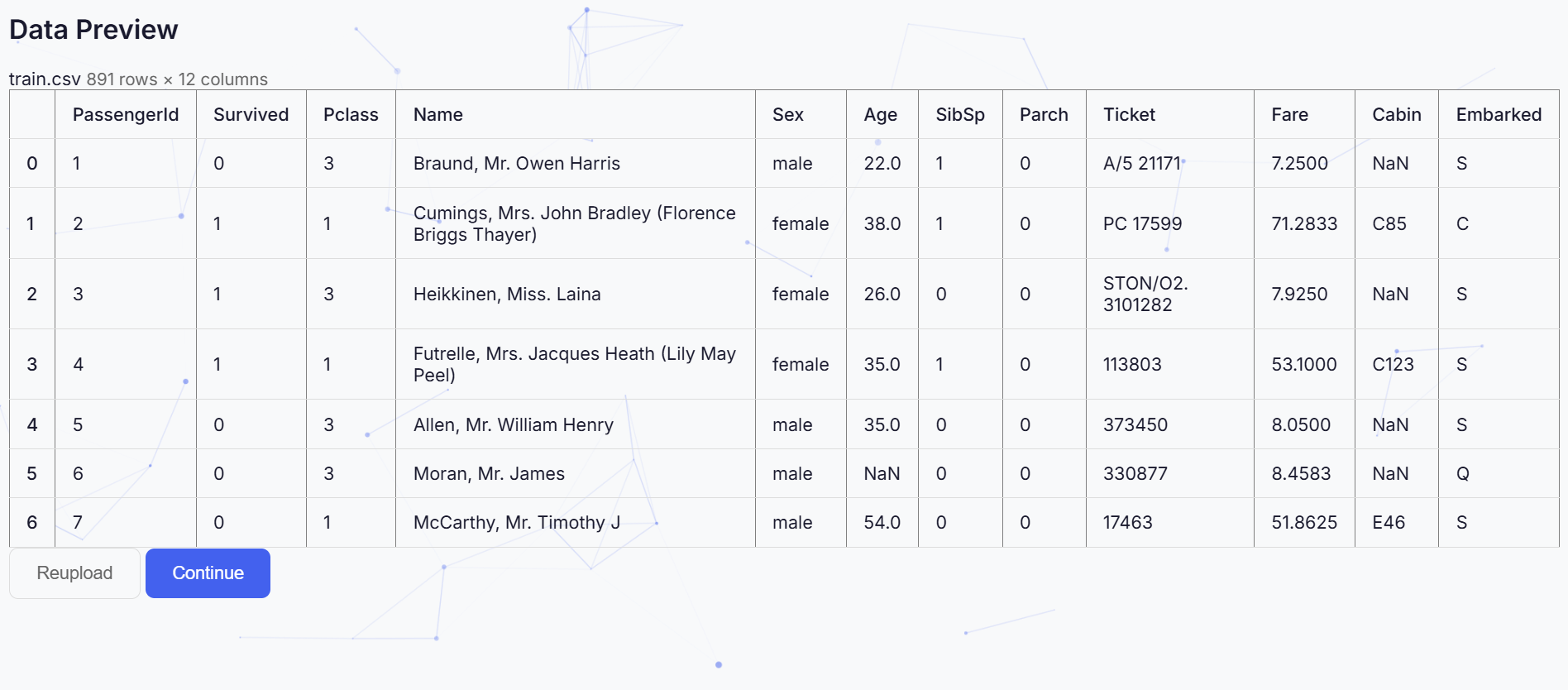
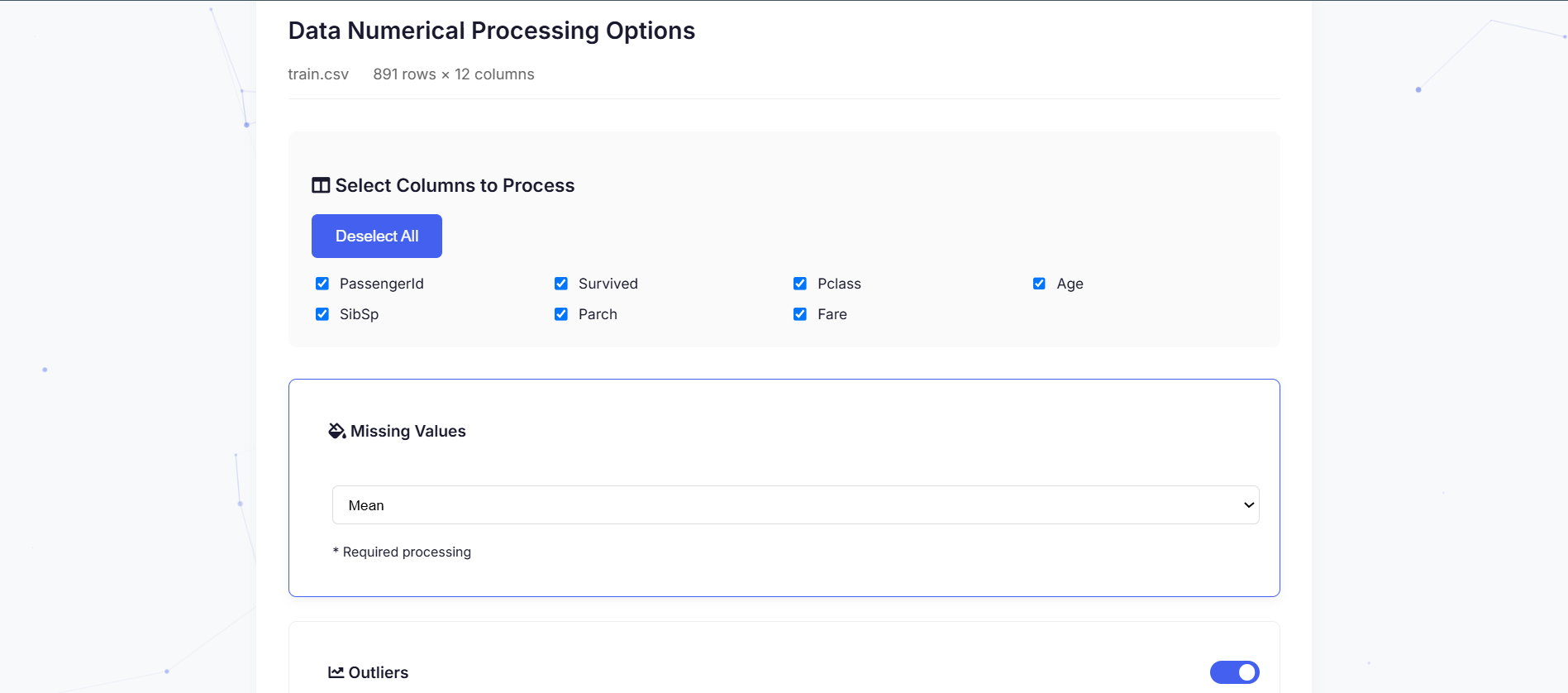


Рисунок 2.1. Вывод датасета.

* 1. Предобработка: использование библиотек *scikit-learn*, *pandas*, *scipy* (см. рис. 2.2.).

  
Рисунок 2.2. Предобработка данных.

* 1. *ML*-модели: обучение, предсказание и оценка качества через *sklearn* (см. рис. 2.3 и 2.4.).

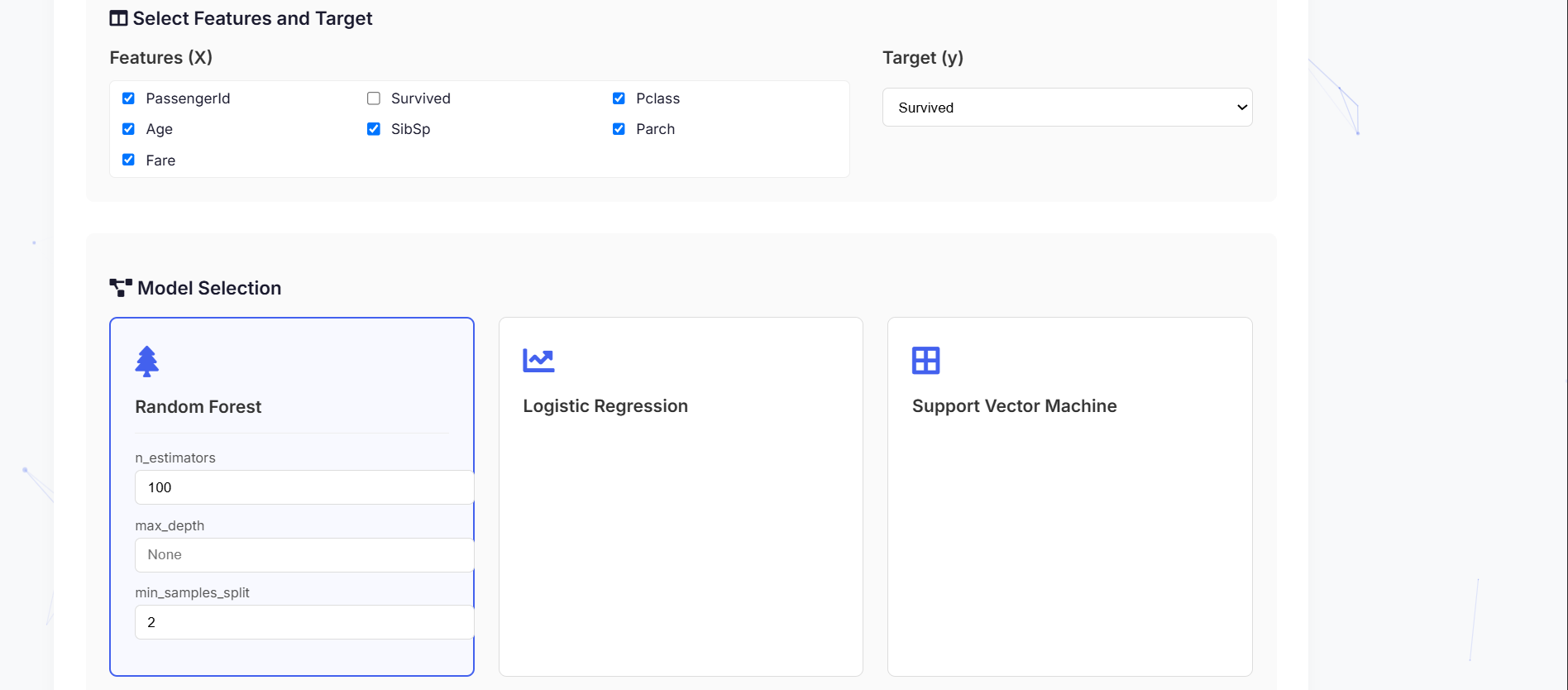


Рисунок 2.3. Выбор модели машинного обучения.

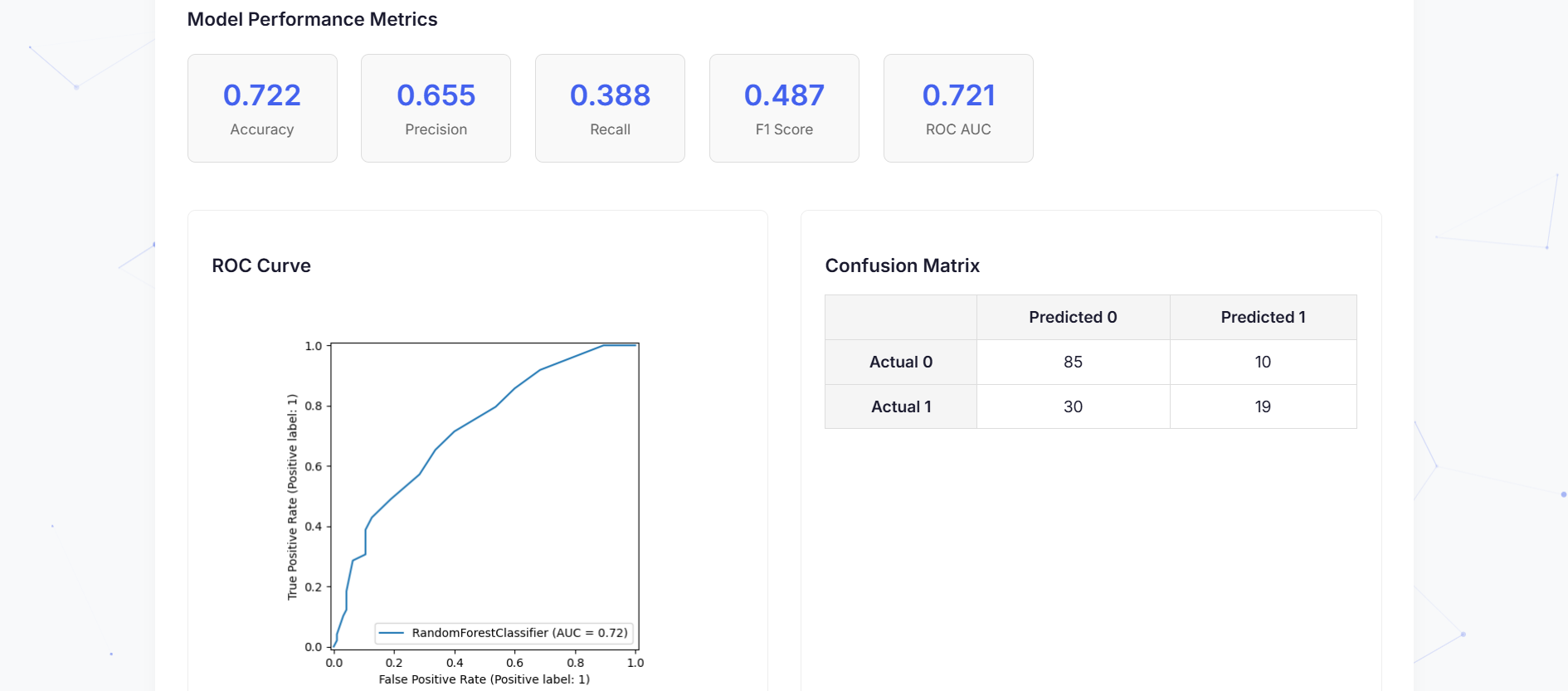


Рисунок 2.4. Оценки качества модели.

1. Тестирование:
   1. Проверки корректности обработки.
   2. Проверки на реагирование на странное поведение пользователя

## Результат решения.

1. Готовое веб-приложение с интерфейсом:

* Страница загрузки данных.
* Настройка параметров предобработки (выпадающие списки, чекбоксы).
* Визуализация изменений (до/после обработки).
* Выбор и настройка *ML*-модели.

1. Пример работы:

* Датасет Titanic: обработка пропусков в *Age* (медиана), кодирование *Sex* (*One-Hot*), нормализация *Fare* (*Standard*).
* Обучение случайного леса: *accuracy* = 0,788.

## Анализ полученного результата.

Успехи:

* Приложение корректно обрабатывает данные (заполнение пропусков, обработка выбросов, нормализация, кодирование категориальных признаков).
* Все заявленные методы предобработки работают без ошибок.
* *ML* модели обучаются и предсказывают с приемлемой точностью.
* Время обработки данных укладывается в заданные критерии (<1 сек для датасетов до 15 тыс. строк).
* Интерфейс не "подвисает" при загрузке файлов.

Проблемы и ограничения:

* При больших датасетах (>50 тыс. строк) время обработки возрастает (особенно для методов вроде *Target Encoding*).
* Нет оптимизации под распределенные вычисления (например, через *Dask*).
* Нет интерактивного пред просмотра изменений (например, графики до/после обработки).
* Не реализована возможность отмены действий (*undo*).

# Заключение.

В ходе выполнения курсовой работы было разработано веб-приложение для предобработки данных и построения моделей машинного обучения, соответствующее поставленным задачам. Приложение может использоваться как учебный инструмент для быстрого прототипирования *ML*-моделей. Для промышленного применения требуется доработка масштабируемости.

# Список литературы.

* 1. Книги (печатные издания)
     1. Кремер Н.Ш. Теория вероятностей и математическая статистика: учебник для вузов. – 3-е изд., перераб. и доп. – М.: Юрайт, 2022. – 573 с.
  2. Электронные ресурсы (документация, онлайн-курсы)
     1. Django Documentation [Электронный ресурс]. — Режим доступа: https://docs.djangoproject.com/ (дата обращения: 18.05.2025).
     2. Scikit-learn Documentation [Электронный ресурс]. — Режим доступа: https://scikit-learn.org/stable/documentation.html (дата обращения: 18.05.2025).
     3. Плейлист "Тренировки по *ML* " от Young&&Yandex [Видеозаписи]. — Режим доступа: https://www.youtube.com/watch?v=k3UJOG-DKHE&list=PLXtiZNKIobF5wGW0ExSn47db1I8uYnfIC (дата обращения: 18.05.2025).