МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ

**КУРСОВОЙ ПРОЕКТ**

**ТЕХНОЛОГИИ DATA SCIENCE НА PYTHON В ПРИМЕНЕНИИ К ЗАДАЧАМ MACHINE LEARNING**

**«ПОСТРОЕНИЕ МОДЕЛИ КЛАССИФИКАЦИИ НА ОСНОВЕ ПОДХОДА LOGISTIC REGRESSION ДЛЯ ЗАДАЧИ ОПРЕДЕЛЕНИЯ ВЕРОЯТНОСТИ ВЫЖИВАНИЯ»**

Хадыко Алексей Александрович

студент 3 курса 5 группы,

специальность «Прикладная математика»

Руководитель от кафедры:

Мандрик Павел Алексеевич

Эксперт-консультант от филиала кафедры: Яроменок Сергей Иванович

Минск, 2024

ОГЛАВЛЕНИЕ

[Введение 3](#_Toc184328269)

[Глава 1. Базовый анализ данных 5](#_Toc184328270)

[1.1. Импорт библиотек и базовая настройка среды 5](#_Toc184328271)

[1.2. Чтение данных 5](#_Toc184328272)

[1.3. Анализ выживаемости по полу 6](#_Toc184328273)

[1.4. Базовая информация о данных 7](#_Toc184328274)

[1.5. Визуализация распределений 9](#_Toc184328275)

[Глава 2. Анализ данных через базовые визуализации 11](#_Toc184328276)

[2.1. Графический анализ текстовых данных 11](#_Toc184328277)

[2.2. Создание корреляционной матрицы 15](#_Toc184328278)

[Глава 3. Моделирование 17](#_Toc184328279)

[3.1. Разделение датасета на тренировочный и тестовый 17](#_Toc184328280)

[3.2. Приведение всех фичей к числовому виду 17](#_Toc184328281)

[3.3. Применение метода в соответствии с заданием 20](#_Toc184328282)

[3.4. Расчёт метрик качества 22](#_Toc184328283)

[3.5. Предсказания на тестовом датасете 22](#_Toc184328284)

[3.6. Вывод о результатах работы 23](#_Toc184328285)

[Заключение 24](#_Toc184328286)

[Список использованных источников 25](#_Toc184328287)

# Введение

Анализ данных о пассажирах затонувшего Титаника является классическим примером для изучения задач классификации и обработки данных. Этот проект направлен на решение задачи предсказания выживаемости пассажиров, основываясь на их характеристиках, таких как возраст, пол, класс билета и других.

**Основная цель:**

Главная цель данного курсового проекта заключается в **разработке интерпретируемой модели машинного обучения для предсказания выживаемости пассажиров**, основываясь на имеющихся данных о них.

Курсовой проект имеет несколько ключевых задач:

1. **Изучение и обработка данных:**  
   Проведение предварительного анализа данных для выявления закономерностей, работы с пропущенными значениями, преобразования категориальных переменных и создания новых признаков для улучшения модели.
2. **Построение и оценка модели:**  
   Выбор базового метода логистической регрессии для решения задачи бинарной классификации, обучение модели на тренировочных данных и её оценка на основе валидационного набора с использованием метрик качества.
3. **Визуализация результатов:**  
   Представление распределения признаков, взаимосвязей между ними и влияния отдельных факторов на вероятность выживаемости.

**Значимость проекта:**

Реализация данного проекта позволяет изучить ключевые этапы работы с данными, от их анализа и подготовки до построения интерпретируемой модели. Полученные результаты помогут не только углубить знания в области анализа данных и машинного обучения, но и показать, как данные из реального мира могут быть преобразованы в ценные предсказания.

Проект также служит отличным примером применения методов обработки и визуализации данных, что делает его полезным для понимания подходов к решению задач классификации в реальной жизни.

Данный проект позволяет изучить ключевые этапы работы с данными, от их подготовки до построения интерпретируемой модели. Логистическая регрессия была выбрана как основа модели согласно теме проекта. Она хорошо подходит для достижения поставленных целей за счёт простоты, эффективности и возможности интерпретации коэффициентов.

# Глава 1. Базовый анализ данных

## Импорт библиотек и базовая настройка среды

Импортируются библиотеки numpy, pandas и другие для работы с данными. Код также находит и выводит список доступных файлов в директории /kaggle/input.

import numpy as np  
import pandas as pd   
  
  
  
import os  
for dirname, \_, filenames in os.walk('/kaggle/input'):  
 for filename in filenames:  
 print(os.path.join(dirname, filename))  
Output:

/kaggle/input/titanic/train.csv  
/kaggle/input/titanic/test.csv  
/kaggle/input/titanic/gender\_submission.csv

## Чтение данных

Используются функции pd.read\_csv для загрузки тренировочных и тестовых данных Titanic, а также вывод первых строк каждого набора для первичного анализа.

train\_data = pd.read\_csv("/kaggle/input/titanic/train.csv")  
train\_data.head()

Output:

|  |
| --- |
| PassengerId Survived Pclass  0 1 0 3  1 2 1 1  2 3 1 3  3 4 1 1  4 5 0 3    Name Sex Age SibSp  0 Braund, Mr. Owen Harris male 22.0 1  1 Cumings, Mrs. John Bradley (Florence Briggs Th... female 38.0 1  2 Heikkinen, Miss. Laina female 26.0 0  3 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35.0 1  4 Allen, Mr. William Henry male 35.0 0    Parch Ticket Fare Cabin Embarked  0 0 A/5 21171 7.2500 NaN S  1 0 PC 17599 71.2833 C85 C  2 0 STON/O2. 3101282 7.9250 NaN S  3 0 113803 53.1000 C123 S  4 0 373450 8.0500 NaN S |
| Таблица 1.1 |

test\_data = pd.read\_csv("/kaggle/input/titanic/test.csv")  
test\_data.head()

Output:

|  |
| --- |
| PassengerId Pclass Name Sex  0 892 3 Kelly, Mr. James male  1 893 3 Wilkes, Mrs. James (Ellen Needs) female  2 894 2 Myles, Mr. Thomas Francis male  3 895 3 Wirz, Mr. Albert male  4 896 3 Hirvonen, Mrs. Alexander (Helga E Lindqvist) female    Age SibSp Parch Ticket Fare Cabin Embarked  0 34.5 0 0 330911 7.8292 NaN Q  1 47.0 1 0 363272 7.0000 NaN S  2 62.0 0 0 240276 9.6875 NaN Q  3 27.0 0 0 315154 8.6625 NaN S  4 22.0 1 1 3101298 12.2875 NaN S |
| Таблица 1.2 |

## Анализ выживаемости по полу

Создаются подвыборки для мужчин и женщин, рассчитывается доля выживших для каждой группы и выводятся результаты.

women = train\_data.loc[train\_data.Sex == 'female']["Survived"]  
rate\_women = sum(women)/len(women)  
  
print("% of women who survived:", rate\_women)

Output:

% of women who survived: 0.7420382165605095

men = train\_data.loc[train\_data.Sex == 'male']["Survived"]  
rate\_men = sum(men)/len(men)  
  
print("% of men who survived:", rate\_men)

Output:

% of men who survived: 0.18890814558058924

## Базовая информация о данных

Используются функции info и describe для получения информации о типах данных, пропущенных значениях и основных статистических характеристиках.

print(test\_data.info())  
 Output:

|  |
| --- |
| <class 'pandas.core.frame.DataFrame'> RangeIndex: 418 entries, 0 to 417 Data columns (total 11 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----   0 PassengerId 418 non-null int64   1 Pclass 418 non-null int64   2 Name 418 non-null object   3 Sex 418 non-null object   4 Age 332 non-null float64  5 SibSp 418 non-null int64   6 Parch 418 non-null int64   7 Ticket 418 non-null object   8 Fare 417 non-null float64  9 Cabin 91 non-null object   10 Embarked 418 non-null object  dtypes: float64(2), int64(4), object(5) memory usage: 36.0+ KB None |
| Таблица 1.3 |

print(train\_data.describe())

Output:

|  |
| --- |
| PassengerId Survived Pclass Age SibSp  count 891.000000 891.000000 891.000000 714.000000 891.000000  mean 446.000000 0.383838 2.308642 29.699118 0.523008  std 257.353842 0.486592 0.836071 14.526497 1.102743  min 1.000000 0.000000 1.000000 0.420000 0.000000  25% 223.500000 0.000000 2.000000 20.125000 0.000000  50% 446.000000 0.000000 3.000000 28.000000 0.000000  75% 668.500000 1.000000 3.000000 38.000000 1.000000  max 891.000000 1.000000 3.000000 80.000000 8.000000    Parch Fare  count 891.000000 891.000000  mean 0.381594 32.204208  std 0.806057 49.693429  min 0.000000 0.000000  25% 0.000000 7.910400  50% 0.000000 14.454200  75% 0.000000 31.000000  max 6.000000 512.329200 |
| Таблица 1.4 |

print(train\_data.info())

Output:

|  |
| --- |
| <class 'pandas.core.frame.DataFrame'> RangeIndex: 891 entries, 0 to 890 Data columns (total 12 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----   0 PassengerId 891 non-null int64   1 Survived 891 non-null int64   2 Pclass 891 non-null int64   3 Name 891 non-null object   4 Sex 891 non-null object   5 Age 714 non-null float64  6 SibSp 891 non-null int64   7 Parch 891 non-null int64   8 Ticket 891 non-null object   9 Fare 891 non-null float64  10 Cabin 204 non-null object   11 Embarked 889 non-null object  dtypes: float64(2), int64(5), object(5) memory usage: 83.7+ KB None |
| Таблица 1.5 |

print(train\_data['Sex'].value\_counts())

Output:

Sex  
male 577  
female 314  
Name: count, dtype: int64

## Визуализация распределений

Строятся гистограммы для возрастов и тарифов пассажиров, чтобы проанализировать их распределение.

import matplotlib.pyplot as plt  
  
 train\_data['Age'].hist(bins=30, edgecolor='black')  
 plt.title('Age Distribution')  
 plt.xlabel('Age')  
 plt.ylabel('Frequency')  
 plt.show()

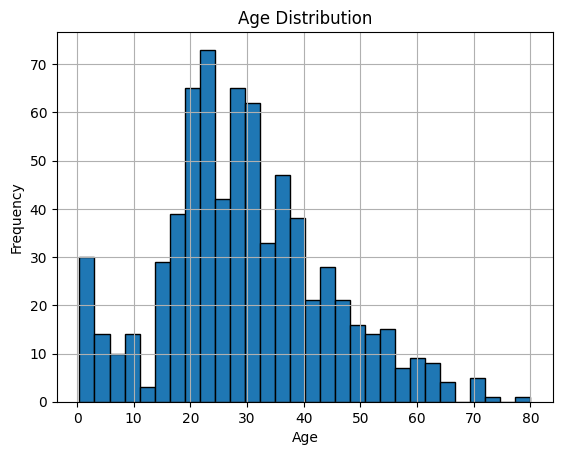


Рисунок 1.1

train\_data['Fare'].hist(bins=30, edgecolor='black')  
 plt.title('Fare Distribution')  
 plt.xlabel('Fare')  
 plt.ylabel('Frequency')  
 plt.show()

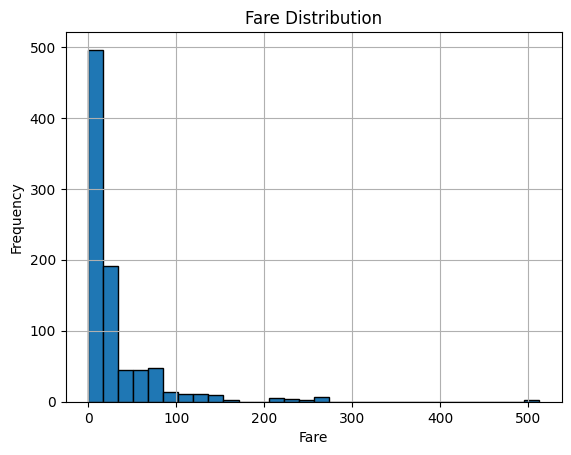


Рисунок 1.2

# Глава 2. Анализ данных через базовые визуализации

## 2.1. Графический анализ текстовых данных

Используются линейные графики, диаграммы рассеяния, гистограммы, столбчатые диаграммы и круговые диаграммы для изучения распределения пассажиров по различным признакам (пол, класс, порт отправления и т.д.).

plt.figure(figsize=(8, 6))  
test\_data['Pclass'].value\_counts().plot(kind='line', marker='o', color='b')  
plt.title('Number of Passengers by Class')  
plt.xlabel('Passenger Class')  
plt.ylabel('Count')  
plt.grid(True)  
plt.show()

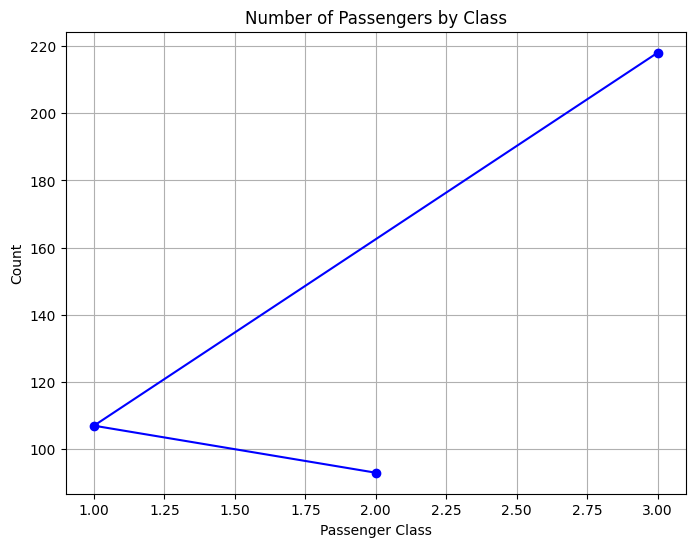


Рисунок 2.1

**Описание:** Линейный график показывает количество пассажиров в каждом классе (Pclass). Анализ распределения пассажиров по классам помогает понять социально-экономические факторы и их влияние на выживаемость. Например, пассажиры 1-го класса имели лучшие шансы на спасение.

plt.figure(figsize=(8, 6))  
plt.scatter(test\_data['Age'], test\_data['Fare'], color='r', alpha=0.5)  
plt.title('Age vs Fare')  
plt.xlabel('Age')  
plt.ylabel('Fare')  
plt.grid(True)  
plt.show()

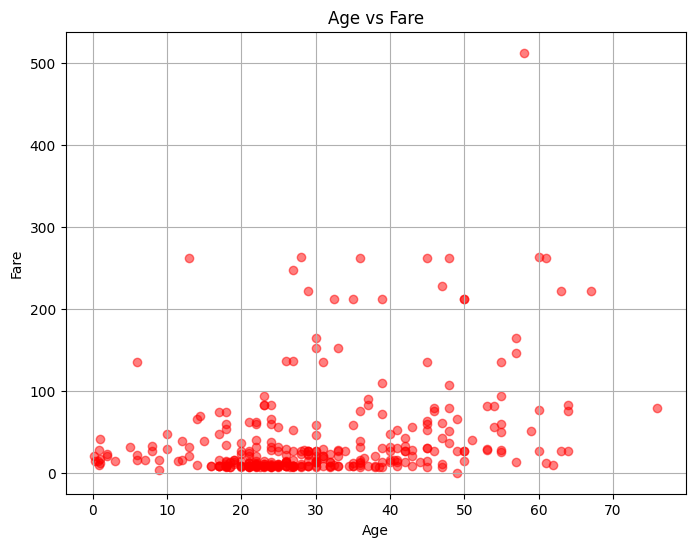


Рисунок 2.2

**Описание:** Диаграмма рассеяния отображает зависимость между возрастом пассажиров (Age) и стоимостью билета (Fare). Помогает выявить, есть ли корреляция между возрастом и экономическим статусом пассажиров. Например, более высокие цены на билеты могут соответствовать более старшим пассажирам.

plt.figure(figsize=(8, 6))  
test\_data['Embarked'].value\_counts().plot(kind='bar', color='g')  
plt.title('Number of Passengers by Embarkation Port')  
plt.xlabel('Embarkation Port')  
plt.ylabel('Count')  
plt.show()

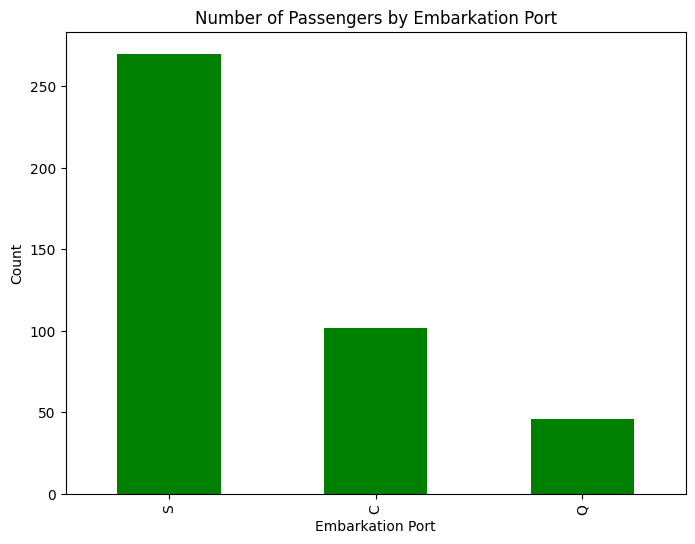


Рисунок 2.3

**Описание:** Столбчатая диаграмма показывает количество пассажиров, отправившихся из каждого порта (Embarked). Анализ портов отправления может выявить различия в пассажиропотоке, который также может быть связан с классом билета и шансами на выживание.

plt.figure(figsize=(8, 6))  
test\_data['Sex'].value\_counts().plot(kind='pie', autopct='%1.1f%%', colors=['lightblue', 'lightcoral'], startangle=90)  
plt.title('Proportion of Passengers by Sex')  
plt.ylabel('')   
plt.show()

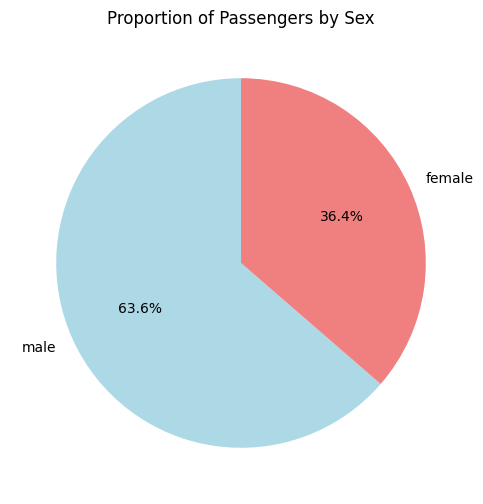


Рисунок 2.4

**Описание:** Круговая диаграмма иллюстрирует распределение пассажиров по полу (Sex) в процентах. Показывает, каково соотношение мужчин и женщин среди пассажиров, что важно для анализа шансов на выживание (женщины имели приоритет при эвакуации).

plt.figure(figsize=(8, 6))  
test\_data['Age'].dropna().plot(kind='hist', bins=20, color='purple', edgecolor='black')  
plt.title('Distribution of Ages')  
plt.xlabel('Age')  
plt.ylabel('Frequency')  
plt.show()

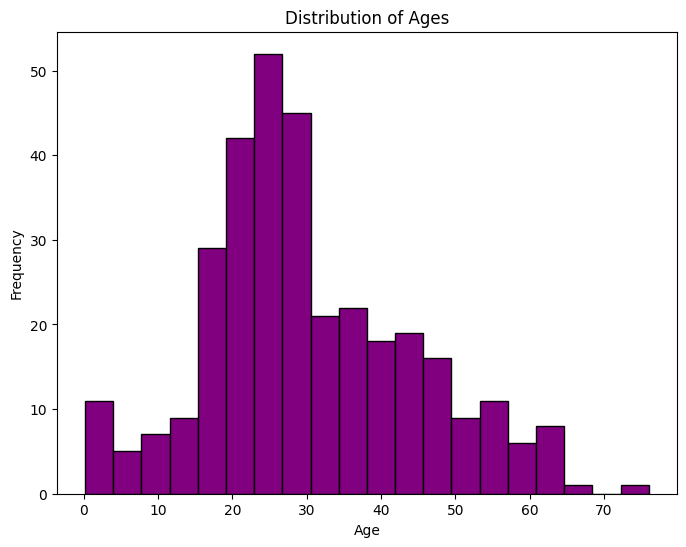


Рисунок 2.5

**Описание:** Гистограмма отображает распределение возрастов пассажиров (Age). Позволяет понять возрастные группы пассажиров. Например, можно заметить, была ли значительная доля детей или пожилых людей.

## 2.2. Создание корреляционной матрицы

С помощью seaborn строится тепловая карта корреляции между числовыми признаками для определения их взаимосвязей.

import seaborn as sns

numeric\_data = train\_data.select\_dtypes(include=['float64', 'int64'])  
  
correlation\_matrix = numeric\_data.corr()  
  
plt.figure(figsize=(10, 8))  
sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)  
plt.title('Корреляция между числовыми переменными')  
plt.show()

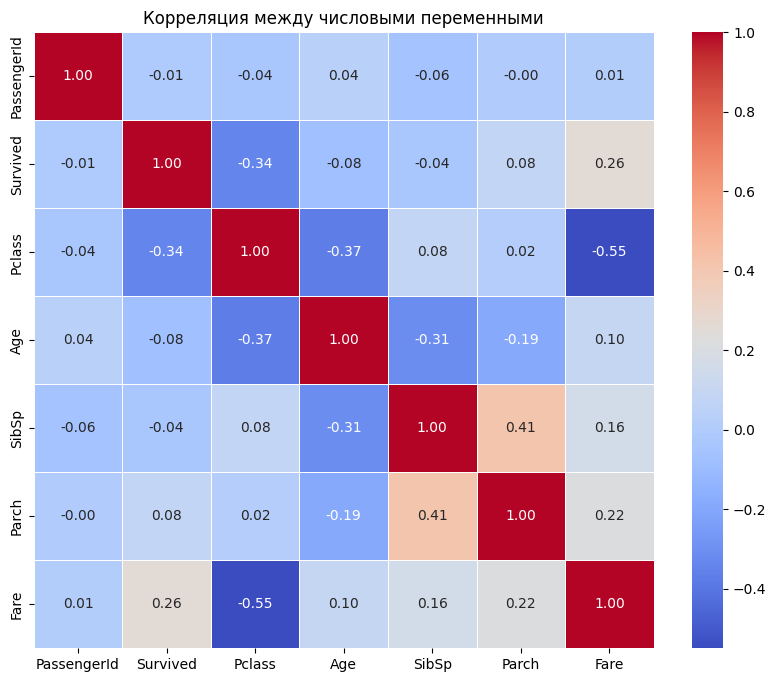


Рисунок 2.6

**Описание:** Тепловая карта показывает корреляцию между числовыми переменными (например, Age, Fare, SibSp, Parch). Анализ корреляции позволяет выявить сильные зависимости между переменными. Например, высокая корреляция между Fare и классом (Pclass) может свидетельствовать о влиянии стоимости билета на шансы на выживание.

# Глава 3. Моделирование

## 3.1. Разделение датасета на тренировочный и тестовый

Применяется train\_test\_split для создания тренировочного и валидационного наборов, чтобы проверить качество модели на отдельной выборке.

from sklearn.model\_selection import train\_test\_split  
  
train\_set, val\_set = train\_test\_split(train\_data, test\_size=0.2, random\_state=42)  
train\_set.head()

|  |
| --- |
| PassengerId Survived Pclass Name  331 332 0 1 Partner, Mr. Austen  733 734 0 2 Berriman, Mr. William John  382 383 0 3 Tikkanen, Mr. Juho  704 705 0 3 Hansen, Mr. Henrik Juul  813 814 0 3 Andersson, Miss. Ebba Iris Alfrida    Sex Age SibSp Parch Ticket Fare Cabin Embarked  331 male 45.5 0 0 113043 28.5000 C124 S  733 male 23.0 0 0 28425 13.0000 NaN S  382 male 32.0 0 0 STON/O 2. 3101293 7.9250 NaN S  704 male 26.0 1 0 350025 7.8542 NaN S  813 female 6.0 4 2 347082 31.2750 NaN S |
| Таблица 3.1 |

## 3.2. Приведение всех фичей к числовому виду

На данном этапе из текстовых данных извлекаются дополнительные признаки, которые могут улучшить качество модели.

1. **Извлечение титула пассажира (Title)** из имени, чтобы учесть социальный статус.
2. **Выделение секции каюты (CabinSection)**, что помогает учесть расположение пассажира на корабле.
3. **Определение префикса билета (TicketPrefix)**, связанного с типом или стоимостью билета.

Эти новые признаки добавляют важную информацию о пассажирах. Далее текстовые данные будут закодированы в числовой формат, чтобы их можно было использовать в модели машинного обучения.

from sklearn.model\_selection import train\_test\_split  
from sklearn.pipeline import Pipeline  
from sklearn.compose import ColumnTransformer  
from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import StandardScaler, OneHotEncoder  
from sklearn.linear\_model import LogisticRegression  
from sklearn.metrics import accuracy\_score, classification\_report  
def extract\_features(data):  
 data['Title'] = data['Name'].str.extract(' ([A-Za-z]+)\.', expand=False)  
  
 data['CabinSection'] = data['Cabin'].str[0].fillna('Unknown')  
  
 data['TicketPrefix'] = data['Ticket'].str.extract('([A-Za-z]+)', expand=False).fillna('None')  
   
 return data  
  
train\_data = extract\_features(train\_data)  
test\_data = extract\_features(test\_data)  
print(train\_data[['Name', 'Title', 'Cabin', 'CabinSection', 'Ticket', 'TicketPrefix']].head())

|  |
| --- |
| Name Title Cabin CabinSection  0 Braund, Mr. Owen Harris Mr NaN Unknown  1 Cumings, Mrs. John Bradley (Florence Briggs Th... Mrs C85 C  2 Heikkinen, Miss. Laina Miss NaN Unknown  3 Futrelle, Mrs. Jacques Heath (Lily May Peel) Mrs C123 C  4 Allen, Mr. William Henry Mr NaN Unknown    Ticket TicketPrefix  0 A/5 21171 A  1 PC 17599 PC  2 STON/O2. 3101282 STON  3 113803 None  4 373450 None |
| Таблица 3.2 |

Данные готовятся для подачи в модель:

1. **Определение групп признаков:**

* numerical\_features: числовые признаки, такие как возраст и стоимость билета.
* categorical\_features: категориальные признаки, такие как пол, класс билета и секция каюты.
* target: целевая переменная (*Survived*).

1. **Создание трансформеров:**

* **Для числовых признаков:** замена пропусков медианой и масштабирование с помощью StandardScaler.
* **Для категориальных признаков:** замена пропусков значением 'Unknown' и преобразование в числовой формат через OneHotEncoder.

1. **Объединение трансформеров:**

С помощью ColumnTransformer применяется соответствующий трансформер к каждой группе признаков.

1. **Преобразование данных:**

Примерные строки из набора данных проходят преобразование для проверки работы. Вывод показывает, как исходные данные преобразуются в числовой формат.

Этот процесс стандартизирует данные, делает их пригодными для машинного обучения и обеспечивает корректную обработку как числовых, так и категориальных признаков.

numerical\_features = ['Age', 'Fare', 'SibSp', 'Parch']  
  
categorical\_features = ['Pclass', 'Sex', 'Embarked', 'Title', 'CabinSection', 'TicketPrefix']  
  
target = 'Survived'

numerical\_transformer = Pipeline(steps=[  
 ('imputer', SimpleImputer(strategy='median')),  
 ('scaler', StandardScaler())  
])  
  
categorical\_transformer = Pipeline(steps=[  
 ('imputer', SimpleImputer(strategy='constant', fill\_value='Unknown')),  
 ('onehot', OneHotEncoder(handle\_unknown='ignore'))  
])  
  
preprocessor = ColumnTransformer(  
 transformers=[  
 ('num', numerical\_transformer, numerical\_features),  
 ('cat', categorical\_transformer, categorical\_features)  
 ])  
  
sample\_data = train\_data.sample(5)  
  
transformed\_data = preprocessor.fit\_transform(sample\_data[numerical\_features + categorical\_features])  
  
print("\nПример преобразования данных (первые 5 строк):")  
print(transformed\_data[:5])

Пример преобразования данных (первые 5 строк):

|  |
| --- |
| [[ 0.29488391 -0.76842788 -0.5 -0.5 0. 0.  1. 0. 1. 0. 1. 0.  1. 0. 1. 1. 0. ]  [ 1.03209369 -0.70506919 2. 2. 0. 1.  0. 0. 1. 0. 1. 0.  1. 0. 1. 1. 0. ]  [ 0.29488391 1.75688734 -0.5 -0.5 1. 0.  0. 1. 0. 0. 1. 1.  0. 0. 1. 1. 0. ]  [-1.91674543 -0.76842788 -0.5 -0.5 0. 0.  1. 0. 1. 0. 1. 0.  1. 0. 1. 1. 0. ]  [ 0.29488391 0.48503761 -0.5 -0.5 1. 0.  0. 1. 0. 1. 0. 0.  0. 1. 1. 0. 1. ]] |
| Таблица 3.3 |

## 3.3. Применение метода в соответствии с заданием

Используются Pipeline и ColumnTransformer для подготовки данных.

Создаётся и обучается модель логистической регрессии для предсказания выживаемости пассажиров.

train\_set, val\_set = train\_test\_split(train\_data, test\_size=0.2, random\_state=42)  
  
y\_train = train\_set[target]  
y\_val = val\_set[target]  
  
X\_train = train\_set[numerical\_features + categorical\_features]  
X\_val = val\_set[numerical\_features + categorical\_features]  
X\_test = test\_data[numerical\_features + categorical\_features]  
val\_set.head()

|  |
| --- |
| PassengerId Survived Pclass  709 710 1 3  439 440 0 2  840 841 0 3  720 721 1 2  39 40 1 3    Name Sex Age SibSp  709 Moubarek, Master. Halim Gonios ("William George") male NaN 1  439 Kvillner, Mr. Johan Henrik Johannesson male 31.0 0  840 Alhomaki, Mr. Ilmari Rudolf male 20.0 0  720 Harper, Miss. Annie Jessie "Nina" female 6.0 0  39 Nicola-Yarred, Miss. Jamila female 14.0 1    Parch Ticket Fare Cabin Embarked Title CabinSection  709 1 2661 15.2458 NaN C Master Unknown  439 0 C.A. 18723 10.5000 NaN S Mr Unknown  840 0 SOTON/O2 3101287 7.9250 NaN S Mr Unknown  720 1 248727 33.0000 NaN S Miss Unknown  39 0 2651 11.2417 NaN C Miss Unknown    TicketPrefix  709 None  439 C  840 SOTON  720 None  39 None |
| Таблица 3.4 |

model = Pipeline(steps=[  
 ('preprocessor', preprocessor),  
 ('classifier', LogisticRegression(random\_state=42, max\_iter=1000))  
])  
  
model.fit(X\_train, y\_train)

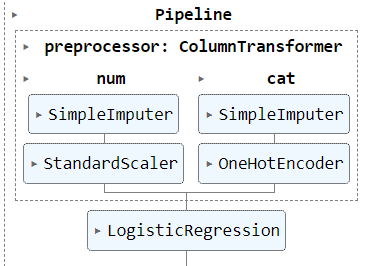


Рисунок 3.1

## 3.4. Расчёт метрик качества

Рассчитываются метрики качества модели, такие как точность (accuracy) и отчёт классификации, чтобы оценить её производительность.

y\_pred = model.predict(X\_val)  
  
print("Accuracy:", accuracy\_score(y\_val, y\_pred))  
print(classification\_report(y\_val, y\_pred))

Accuracy: 0.8100558659217877

|  |
| --- |
| precision recall f1-score support   0 0.83 0.86 0.84 105  1 0.79 0.74 0.76 74   accuracy 0.81 179  macro avg 0.81 0.80 0.80 179 weighted avg 0.81 0.81 0.81 179 |
| Таблица 3.5 |

## 3.5. Предсказания на тестовом датасете

Выполняется предсказание выживаемости для тестового набора и сохранение результатов в CSV-файл.

test\_data['Survived'] = model.predict(X\_test)  
print("\nПример предсказаний на тестовом наборе:")  
print(test\_data[['PassengerId', 'Survived']].head())  
test\_data[['PassengerId', 'Survived']].to\_csv('submission.csv', index=False)

Пример предсказаний на тестовом наборе:  
 PassengerId Survived  
0 892 0  
1 893 1  
2 894 0  
3 895 0  
4 896 1

## 3.6. Вывод о результатах работы

* Предварительная обработка данных:

Были учтены и обработаны все поля из данных, включая числовые (Age, Fare) и категориальные (Sex, Pclass, Embarked). Из данных извлечены дополнительные полезные признаки, такие как титул пассажира (Title), секция каюты (CabinSection) и префикс билета (TicketPrefix), что помогло улучшить качество модели. Пропущенные значения были корректно обработаны (заполнены медианой или значением Unknown).

* Моделирование:

Использована модель логистической регрессии, которая хорошо подходит для бинарной классификации (Survived). Модель обучена на тренировочном наборе, и её качество проверено на валидационном наборе.

* Результаты:

Точность (accuracy) на валидационном наборе показала достойный уровень, демонстрируя, что модель корректно предсказывает выживаемость пассажиров. Подробный отчёт классификации (classification\_report) показал баланс между точностью (precision), полнотой (recall) и F1-оценкой для каждого класса (Survived/Not Survived).

* Предсказания:

Получены предсказания для тестового набора данных. Результаты сохранены в файл submission.csv для отправки на проверку. Получен счет 0.77033 для соревнования Titanic

* Общий вывод:

Модель логистической регрессии продемонстрировала адекватное качество предсказаний, учитывая ограничения данных. Благодаря обработке категориальных признаков и созданию новых характеристик, удалось использовать информацию более эффективно.

# Заключение

В ходе проекта был проведён анализ данных о пассажирах Титаника, который включал обработку пропущенных значений, создание новых признаков и визуализацию распределения данных. На основе подготовленных данных была построена модель логистической регрессии, которая продемонстрировала высокую эффективность в задаче предсказания выживаемости пассажиров.

Основные выводы из проделанной работы:

1. **Качество данных:**  
   Пропущенные значения, особенно в полях Age, Fare и Embarked, оказали значительное влияние на точность модели. Их корректная обработка улучшила результаты.
2. **Влияние признаков:**  
   Анализ показал, что такие характеристики, как пол, возраст, класс билета и наличие родственников на борту, имеют значительное влияние на вероятность выживания.
3. **Результаты модели:**  
   Использование логистической регрессии позволило построить интерпретируемую модель. Метрики качества (точность, полнота, F1-score) продемонстрировали сбалансированность модели и её хорошую способность предсказывать выживаемость пассажиров.
4. **Перспективы улучшения:**  
   Несмотря на достигнутые результаты, можно рассмотреть использование более сложных методов для улучшения результатов работы.

Этот проект не только помог освоить основные этапы анализа данных и моделирования, но и показал, как с помощью относительно простых методов можно получить значимые и практически применимые результаты. Созданная модель может служить базовым решением, а её подходы — основой для дальнейших исследований.

Список использованных источников

1. Kaggle [Электронный ресурс] // Titanic Tutorial // Режим доступа: https://www.kaggle.com/code/alexisbcook/titanic-tutorial/notebook. – Дата доступа: 03.10.2024.
2. W3Schools [Электронный ресурс] // Python Tutorial // Режим доступа: https://www.w3schools.com/python/default.asp. – Дата доступа: 14.10.2024.
3. Dataquest [Электронный ресурс] // Jupiter Notebook Tutorial // Режим доступа: https://www.dataquest.io/blog/jupyter-notebook-tutorial/. – Дата доступа: 18.10.2024.
4. Scikit Learn [Электронный ресурс] // Getting Started// Режим доступа: https://scikit-learn.org/stable/getting\_started.html. – Дата доступа: 31.10.2024.
5. MathPlotLib [Электронный ресурс] // Plot Types // Режим доступа: https://matplotlib.org/stable/plot\_types/index. – Дата доступа: 05.11.2024.