МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и информационной безопасности

Кафедра автоматизированных и вычислительных систем

ОТЧЕТ

по лабораторной работе № 1

по дисциплине «Интеллектуальные системы»

Тема «Введение в Python для анализа данных»

Выполнил

студент гр. мУПИ-251 Н.К. Клёсов

Подпись, дата Инициалы, фамилия

Руководитель С.А. Олейникова

Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата

ВОРОНЕЖ 2025

**Цель:** Овладеть базовыми навыками работы с библиотеками numpy, pandas и matplotlib/seaborn для предварительной обработки данных и визуализации.

**Задание к работе**

1. Загрузить датасет titanic.scv.
2. Провести первичный анализ данных (определить размер данных, проверить наличие пропусков и обработать их, вывести основные статистики).
3. Построить не менее трех графиков для анализа данных и сделать выводы о результатах.
4. Исследовать выбросы и обработать их.
5. Сделать выводы.

**Краткие теоретические сведения**

1. **Библиотеки NumPy и Pandas**

Numpy - это открытая бесплатная Python-библиотека для работы с многомерными массивами, которую применяют для математических вычислений: начиная с базовых функций и заканчивая линейной алгеброй.

Pandas – это библиотека, которая предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временными рядами. Работа pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня.

К основным структурам pandas относят DataFrame и Series.

Pandas Series (серия)— это одномерный массив. Визуально он похож на пронумерованный список: слева в колонке находятся индексы элементов, а справа — сами элементы.

### ****Визуализация данных****

Pandas DataFrame — это двумерный массив, похожий на таблицу/лист Excel. В нем можно проводить такие же манипуляции с данными: объединять в группы, сортировать по определенному признаку, производить вычисления. Как любая таблица, датафрейм состоит из столбцов и строк, причем столбцами будут уже известные объекты — Series.

**Визуализация является важным этапом работы с данными.** Визуализация помогает выявлять связи между объектами, фактами и сведениями, переводит сложные, масштабные или числовые данные в визуальное представление, которое легче воспринимать. Для визуализации в python используются библиотеки Matplotlib и Seaborn. **Matplotlib** — это фундаментальная библиотека Python для создания любых видов статических, интерактивных и анимированных визуализаций, предоставляющая полный контроль над графиками. Seaborn — это мощная и гибкая библиотека визуализации данных для Python, строящая свою работу поверх библиотеки Matplotlib.

Ряд инструментов для визуализации данных:

* Гистограмма.
* Диаграмма рассеяния.
* Столбчатые диаграммы.
* Тепловая карта.

### ****Основные этапы предобработки данных****

Предобработка данных (Data Preprocessing) — это комплекс методов и действий по преобразованию сырых, необработанных данных в чистый, структурированный формат, пригодный для обучения моделей машинного обучения. Качество данных напрямую влияет на качество прогнозов модели, что часто описывается принципом GIGO ("garbage in, garbage out" — мусор на входе, мусор на выходе).

**3.1. Загрузка данных**

Данные для загрузки могут быть в \*.csv – файлах, в таблицах Excel или храниться в базе данных. Для работы с данными в python предусмотрены библиотеки numpy и pandas (название от panel data).

* 1. **Обработка данных. Выбросы**

Обработка данных включает в себя обработку пропущенных значений и обработку выбросов

Выбросы (outliers) - наблюдения, значительно отклоняющиеся от общего распределения данных. Они могут быть вызваны:

* ошибками измерения (например, сенсорные сбои);
* экспериментальными ошибками;
* реальными, но редкими событиями (мошеннические транзакции).

### Методы обнаружения выбросов

Графические методы — это простой способ визуализации данных, который может помочь обнаружить выбросы. Примеры графических методов включают:

* Boxplot (ящик с усами): Этот график показывает медиану, нижний и верхний квартили, а также выбросы в данных. Выбросы определяются как значения, находящиеся за пределами усов, которые представляют 1,5 межквартильного размаха (IQR).
* Scatter plot (диаграмма рассеяния): Эта диаграмма показывает взаимосвязь между двумя переменными, на которой можно увидеть выбросы, которые отклоняются от общей тенденции.
  + 1. **Методы обработки выбросов**

Детекция выбросов необходима для последующей обработки соответствующих значений. Рассмотрим основные методы обработки выбросов.

Первый и наиболее простой способ – удаление всех выбросов. Для больших выборок данная операция, как правило, не будет иметь никаких негативных последствий (особенно, при небольшом числе выбросов).

#### **Другим вариантом обработки выбросов является их замена.**

Наиболее широко используемыми вариантами для замены выбросов являются:

* медиана,
* среднее значение,
* граничное значение, выбранное экспертом,
* предиктивными методами.
  1. **Обработка пропущенных значений**

Для обнаружения пропущенных значений в DataFrame можно использовать два аналогичных метода isna() или isnull().Метод isna() в Pandas создает булев (логический) DataFrame того же размера, что и исходный DataFrame, где True указывает на пропущенные значения (NaN или None), а False – на обычные данные.

**Ход работы**

1. **Название и структура датасета**

Для начала необходимо загрузить данные из датасета «titanic.csv»

Содержимое датасета состоит из: Survived, Pclass, Name, Sex, Age, Siblings/Spouses Aboard, Parents/Children Aboard,Fare.

1. **Анализ данных**
   1. **Размерность и структура данных**

print(f"Размер датасета: {df.shape[0]} строк, {df.shape[1]} столбцов")

print(f"\nНазвания столбцов: {list(df.columns)}")

print(f"\nТипы данных:")

print(df.dtypes)

Результат (Рис. 1):

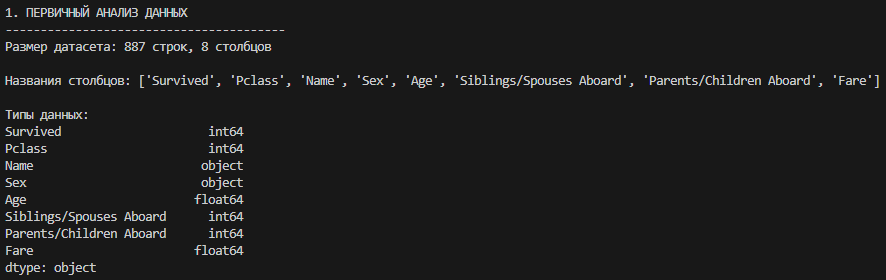


Рисунок 1 – Первичный анализ данных

**2.2.** **Статистический анализ числовых признаков**

print("Основные статистики (числовые признаки):")

print(df\_processed.describe())

**Ключевые статистики:**

* **Survived**: среднее = 0.386 (38.6% выживших)
* **Pclass**: среднее = 2.31 (преобладает 3-й класс)
* **Age:** среднее = 29.47 лет, медиана = 28 лет
* **Fare**: среднее = 32.31£, медиана = 14.45£ (большой разброс)

### ****2.3. Анализ категориальных признаков****

print("\nКатегориальные признаки:")

print(f"Уникальные значения в 'Sex': {df\_processed['Sex'].unique()}")

print(f"Уникальные значения в 'Pclass': {sorted(df\_processed['Pclass'].unique())}")

### Результат (Рис. 2):



Рисунок 2 – Категориальные признаки

### ****2.4. Распределение целевой переменной (Survived)****

print("\nАНАЛИЗ ВЫЖИВАЕМОСТИ:")

total\_survived = df\_processed['Survived'].sum()

total\_passengers = len(df\_processed)

survival\_rate = df\_processed['Survived'].mean()

print(f"Общая выживаемость: {survival\_rate:.2%} ({total\_survived}/{total\_passengers})")

### Результат (Рис. 3):



Рисунок 3 – Распределение целевой переменной

### ****2.5. Детальный анализ по группам****

print("\nВыживаемость по классам:")

survival\_by\_class = df\_processed.groupby('Pclass')['Survived'].agg(['mean', 'count'])

survival\_by\_class['mean\_pct'] = survival\_by\_class['mean'].apply(lambda x: f"{x:.2%}")

print(survival\_by\_class[['mean\_pct', 'count']].rename(columns={'mean\_pct': 'Выживаемость', 'count': 'Количество'}))

### Результат (Рис.4):

### 

### Рисунок 4 – Выживаемость по классам

print("\nВыживаемость по полу:")

survival\_by\_sex = df\_processed.groupby('Sex')['Survived'].agg(['mean', 'count'])

survival\_by\_sex['mean\_pct'] = survival\_by\_sex['mean'].apply(lambda x: f"{x:.2%}")

print(survival\_by\_sex[['mean\_pct', 'count']].rename(columns={'mean\_pct': 'Выживаемость', 'count': 'Количество'}))

### Результат (Рис. 5):

### 

### Рисунок 5 – Выживаемость по полу

### Предобработка

## **3.1.** **Анализ пропущенных и нулевых значений**

* # Проверка NaN
* print("\n1. Стандартные NaN значения:")
* missing\_data = df.isnull().sum()
* missing\_percent = (df.isnull().sum() / len(df)) \* 100
* missing\_info = pd.DataFrame({
* 'Пропущено': missing\_data,
* 'Процент': missing\_percent
* })
* print(missing\_info)
* # Проверка пустых строк
* print("\n2. Проверка пустых строк в текстовых полях:")
* text\_columns = df.select\_dtypes(include=['object']).columns
* empty\_found = False
* for col in text\_columns:
* empty\_count = (df[col] == '').sum()
* if empty\_count > 0:
* print(f"  {col}: {empty\_count} пустых значений ({empty\_count/len(df)\*100:.1f}%)")
* empty\_found = True
* if not empty\_found:
* print("  Пустых строк не обнаружено")
* # ДЕТАЛЬНАЯ ПРОВЕРКА НУЛЕВЫХ ЗНАЧЕНИЙ ВО ВСЕХ ПОЛЯХ
* print("\n3. ДЕТАЛЬНАЯ ПРОВЕРКА НУЛЕВЫХ ЗНАЧЕНИЙ ВО ВСЕХ ПОЛЯХ:")
* # Для числовых полей
* print("\n3.1. Числовые поля:")
* numeric\_columns = df.select\_dtypes(include=[np.number]).columns
* numeric\_zero\_summary = []
* for col in numeric\_columns:
* zero\_count = (df[col] == 0).sum()
* zero\_percent = zero\_count / len(df) \* 100
* numeric\_zero\_summary.append({
* 'Поле': col,
* 'Нулевых значений': zero\_count,
* 'Процент': zero\_percent,
* 'Интерпретация': ''
* })

Начальный этап предобработки включал в себя комплексный аудит данных для выявления и каталогизации пропущенных значений. Были применены следующие методы:

1. Детекция стандартных пропусков (NaN): Для количественной оценки отсутствующих данных в каждом признаке был использован метод df.isnull().sum(). Рассчитан абсолютный и относительный (в процентах) объем пропусков для каждого столбца, что позволило идентифицировать ключевые признаки, требующие импутации (Age, Fare).
2. Анализ строковых данных**:** Проведена верификация категориальных признаков на наличие пустых строк ('') и строк, состоящих исключительно из пробельных символов. По результатам проверки аномальные строковые значения данного типа не были обнаружены.
3. Семантический анализ нулевых значений: Выполнен анализ числовых признаков на предмет содержания нулевых значений (0). Для каждого случая была проведена семантическая оценка валидности:
   * Валидные нули: Survived, Siblings/Spouses Aboard, Parents/Children Aboard (интерпретируются как "не выжил" и "отсутствие родственников" соответственно).
   * Условно-валидные нули: Fare (может означать бесплатный проезд или ошибку данных, что требует дополнительного анализа).
   * Невалидные нули: Pclass, Age (значение 0 не является допустимым в контексте данных признаков).

### 3.2. Обработка пропущенных значений

df\_processed = df.copy()

# Обработка пропусков в Age

age\_before = df\_processed['Age'].isnull().sum()

if age\_before > 0:

    df\_processed['Age'] = df\_processed.groupby(['Sex', 'Pclass'])['Age'].transform(

        lambda x: x.fillna(x.median())

    )

    age\_after = df\_processed['Age'].isnull().sum()

    print(f"✓ Заполнено пропусков в Age: {age\_before - age\_after}")

# Обработка пропусков в Fare

fare\_before = df\_processed['Fare'].isnull().sum()

if fare\_before > 0:

    df\_processed['Fare'] = df\_processed.groupby('Pclass')['Fare'].transform(

        lambda x: x.fillna(x.median())

    )

    fare\_after = df\_processed['Fare'].isnull().sum()

    print(f"✓ Заполнено пропусков в Fare: {fare\_before - fare\_after}")

# Обработка текстовых пропусков

for col in text\_columns:

    if df\_processed[col].isnull().sum() > 0:

        df\_processed[col] = df\_processed[col].fillna('Unknown')

        print(f"✓ Заполнены пропуски в {col}")

print("✅ Все пропущенные значения обработаны")

Для устранения пропусков и обеспечения целостности данных были применены методы групповой импутации, позволяющие сохранить статистические свойства исходного распределения.

1. Импутация признака Age: Пропущенные значения были заполнены медианным значением, рассчитанным стратифицированно по группам, определенным комбинацией признаков Sex и Pclass. Данный подход (df.groupby(['Sex', 'Pclass'])['Age'].transform('median')) обусловлен сильной корреляцией возраста с социально-демографическими характеристиками пассажиров.
2. Импутация признака Fare: Аналогично, пропуски в стоимости билета были заменены медианным значением, сгруппированным по признаку Pclass, так как класс обслуживания является основным фактором, определяющим стоимость проезда.
3. Обработка категориальных пропусков: Все пропущенные значения в текстовых полях были заменены на константу 'Unknown', что позволяет алгоритмам машинного обучения обрабатывать их как отдельную категорию.
   1. **Анализ и обработка выбросов**

def analyze\_outliers(column\_name, data, russian\_name):

    """Анализ выбросов для указанного столбца"""

    print(f"\nАнализ выбросов для '{russian\_name}':")

    Q1 = data[column\_name].quantile(0.25)

    Q3 = data[column\_name].quantile(0.75)

    IQR = Q3 - Q1

    lower\_bound = Q1 - 1.5 \* IQR

    upper\_bound = Q3 + 1.5 \* IQR

    outliers = data[(data[column\_name] < lower\_bound) | (data[column\_name] > upper\_bound)]

    print(f"  Q1: {Q1:.2f}, Q3: {Q3:.2f}, IQR: {IQR:.2f}")

    print(f"  Границы: [{lower\_bound:.2f}, {upper\_bound:.2f}]")

    print(f"  Количество выбросов: {len(outliers)} ({len(outliers)/len(data)\*100:.1f}%)")

    return outliers, (lower\_bound, upper\_bound)

# Обработка выбросов в возрасте

age\_outliers\_before = len(age\_outliers)

df\_processed['Age\_processed'] = np.where(

    df\_processed['Age'] > age\_bounds[1],

    age\_bounds[1],

    np.where(

        df\_processed['Age'] < age\_bounds[0],

        age\_bounds[0],

        df\_processed['Age']

    )

)

# Обработка выбросов в стоимости билета

fare\_outliers\_before = len(fare\_outliers)

df\_processed['Fare\_log'] = np.log1p(df\_processed['Fare'])

df\_processed['Fare\_processed'] = np.where(

    df\_processed['Fare'] > fare\_bounds[1],

    fare\_bounds[1],

    df\_processed['Fare']

)

Для повышения робастности моделей был проведен анализ и последующая обработка аномальных значений (выбросов).

* 1. Метод детекции: Для идентификации выбросов использовался метод межквартильного размаха (IQR). Значения, выходящие за пределы диапазона [Q1 - 1.5 \* IQR, Q3 + 1.5 \* IQR], были классифицированы как выбросы.
  2. Обработка выбросов в Age: Применен метод "обрезки" (capping/winsorizing), где все значения, выходящие за вычисленные границы, были заменены на значения этих границ. Это позволило сохранить данные, ограничив при этом их влияние на модель.
  3. Обработка выбросов в Fare: Использован комбинированный подход. Сначала было применено логарифмическое преобразование (np.log1p) для сглаживания асимметрии в распределении и уменьшения влияния экстремально высоких значений. Затем, к трансформированным данным был применен метод "обрезки" по границам IQR.
  4. **Визуализация процесса предобработки**

# Визуализация выбросов ДО и ПОСЛЕ обработки

fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2, figsize=(15, 10))

# Возраст до обработки

sns.boxplot(y=df['Age'], ax=ax1)

ax1.set\_title('Возраст ДО обработки', fontsize=14, fontweight='bold')

ax1.set\_ylabel('Возраст', fontsize=12)

# Возраст после обработки

sns.boxplot(y=df\_processed['Age\_processed'], ax=ax2)

ax2.set\_title('Возраст ПОСЛЕ обработки', fontsize=14, fontweight='bold')

ax2.set\_ylabel('Возраст', fontsize=12)

# Стоимость до обработки

sns.boxplot(y=df['Fare'], ax=ax3)

ax3.set\_title('Стоимость ДО обработки', fontsize=14, fontweight='bold')

ax3.set\_ylabel('Стоимость билета (£)', fontsize=12)

# Стоимость после обработки

sns.boxplot(y=df\_processed['Fare\_processed'], ax=ax4)

ax4.set\_title('Стоимость ПОСЛЕ обработки', fontsize=14, fontweight='bold')

ax4.set\_ylabel('Стоимость билета (£)', fontsize=12)

Эффективность процедур обработки выбросов была визуально валидирована с помощью диаграмм размаха (boxplot).

* Цель визуализации: Сравнительный анализ распределений признаков Age и Fare до и после применения методов обработки выбросов.
* Результат: Графики наглядно демонстрируют успешное устранение выбросов, что выражается в значительном сокращении "усов" диаграмм и отсутствии точек за их пределами после обработки. Это подтверждает корректность выполненных преобразований и готовность данных для дальнейшего моделирования.

1. **Визуализация**

#### **4.1. Зависимость выживаемости от класса и пола**

На сгруппированной столбчатой диаграмме представлена доля выживших пассажиров в зависимости от пола (Рис. 6).

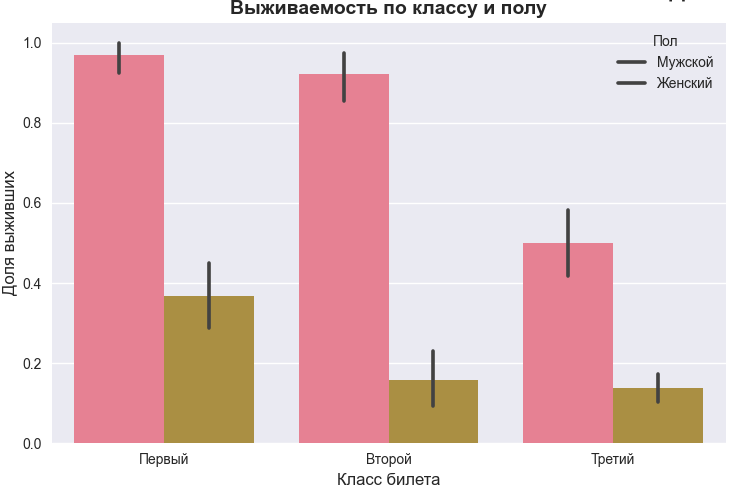


Рисунок 6 – Доля выживших в зависимости от пола

* Наблюдение 1: Гендерный фактор. Во всех трех классах наблюдается значительное превосходство выживаемости среди женщин (розовые столбцы) по сравнению с мужчинами (коричневые столбцы). Доля выживших женщин в первом классе приближается к 95-97%, в то время как для мужчин в третьем классе этот показатель едва достигает 15%. Это подтверждает гипотезу о приоритетной эвакуации женщин ("женщины и дети первыми").
* Наблюдение 2: Социально-экономический фактор. Прослеживается четкая отрицательная корреляция между номером класса и вероятностью выживания. Пассажиры первого класса имели наивысшие шансы на спасение, в то время как пассажиры третьего класса — наименьшие. Это справедливо для обоих полов.

График наглядно демонстрирует, что пол и класс билета являются двумя наиболее сильными предикторами выживаемости. Комбинация этих факторов (например, женщина в 1-м классе) давала максимальные шансы на выживание.

#### **4.2. Распределение возраста в контексте выживаемости**

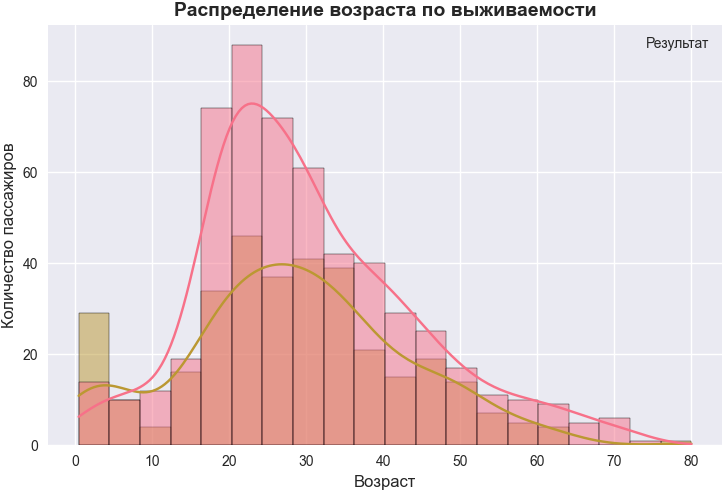
На гистограмме показано распределение возраста пассажиров, разделенное по исходу (выжил/не выжил), с наложенными кривыми ядерной оценки плотности (KDE).

Рисунок 7 – выживаемость в зависимости от пола

* Наблюдение 1: Выживаемость детей. Наблюдается отчетливый пик выживаемости среди детей в возрасте до 10 лет. В этой возрастной группе количество выживших (желтоватая гистограмма) заметно превышает количество погибших, что является еще одним подтверждением правила "женщины и дети первыми".
* Наблюдение 2: Уязвимость молодых взрослых. Основную массу пассажиров составляли люди в возрасте от 20 до 40 лет. Именно в этой группе зафиксировано наибольшее количество жертв (розовая гистограмма), что указывает на низкий уровень выживаемости среди трудоспособного населения.
* Наблюдение 3: Пожилые пассажиры. Количество пассажиров старше 65 лет было невелико, и их шансы на выживание также были низкими.

Возраст является нелинейным фактором, влияющим на выживаемость. Дети имели значительно более высокие шансы на спасение, в то время как для большинства взрослых пассажиров возраст не играл решающей роли, уступая по значимости полу и классу.

**4.3. Корреляционный анализ числовых признаков**

Для количественной оценки линейных взаимосвязей между числовыми признаками была построена и визуализирована матрица корреляций Пирсона в виде тепловой карты (Рис. 8).

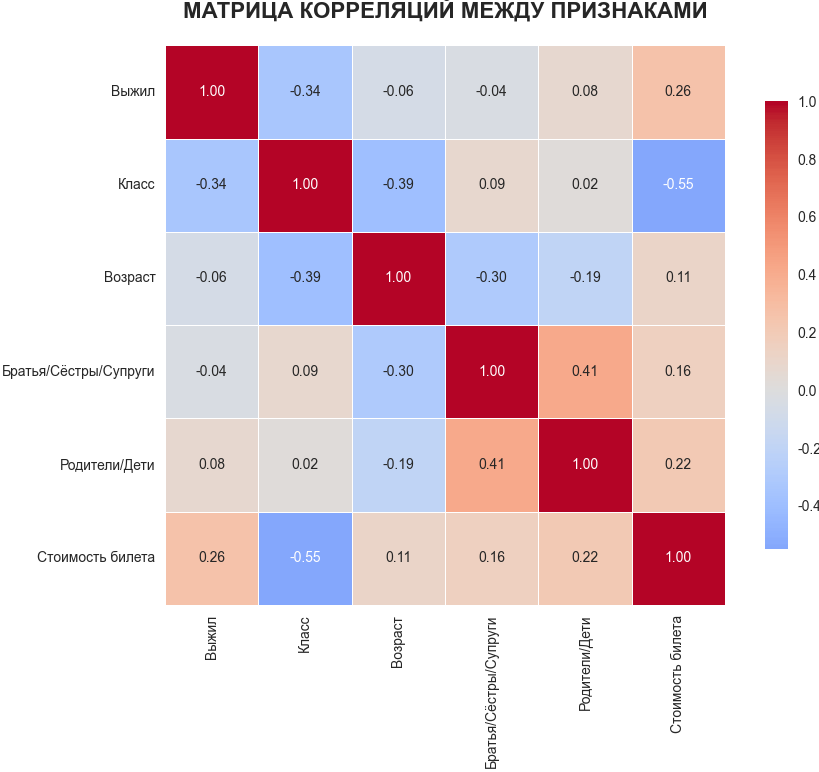


Рисунок 8 – Матрица корреляций между признаками

* Интерпретация: Цветовая шкала отражает силу и направление корреляции:
  + Насыщенный красный цвет (близко к +1.0) указывает на сильную положительную линейную связь (при росте одного признака другой также имеет тенденцию к росту).
  + Насыщенный синий цвет (близко к -1.0) указывает на сильную отрицательную линейную связь (при росте одного признака другой имеет тенденцию к убыванию).
  + Нейтральные цвета (близко к 0.0) свидетельствуют об отсутствии линейной зависимости.

#### Ключевые выводы из анализа:

1. Корреляция с целевой переменной ("Выжил"):
   * **"**Класс" и "Выжил" (-0.34): Наблюдается умеренная отрицательная корреляция. Это означает, что с увеличением номера класса (т.е. с понижением социального статуса с 1-го до 3-го) вероятность выживания снижается. Это наиболее сильный линейный предиктор среди представленных.
   * **"**Стоимость билета" и "Выжил" (0.26): Умеренная положительная корреляция. Более дорогие билеты ассоциируются с более высокими шансами на выживание, что логически связано с классом каюты.
2. Взаимосвязи между предикторами (мультиколлинеарность**):**
   * **"**Класс" и "Стоимость билета" (-0.55): Сильная отрицательная корреляция, что является ожидаемым. Билеты более высокого класса (например, 1-го) стоят дороже, а билеты низкого класса (3-го) — дешевле.
   * "Братья/Сёстры/Супруги" и "Родители/Дети" (0.41): Умеренная положительная корреляция. Это указывает на то, что пассажиры часто путешествовали семьями. Наличие родственников одной категории (например, супруга) повышает вероятность наличия родственников другой категории (например, детей).
   * **"**Возраст" и "Класс" (-0.39): Умеренная отрицательная связь. Пассажиры более высокого класса в среднем были старше.

Тепловая карта подтверждает, что социально-экономический статус (Класс, Стоимость билета) является ключевым фактором, линейно связанным с выживаемостью. Также она выявляет внутренние зависимости в данных, такие как связь между семейным положением и возрастом, что важно учитывать при построении моделей машинного обучения во избежание проблемы мультиколлинеарности.

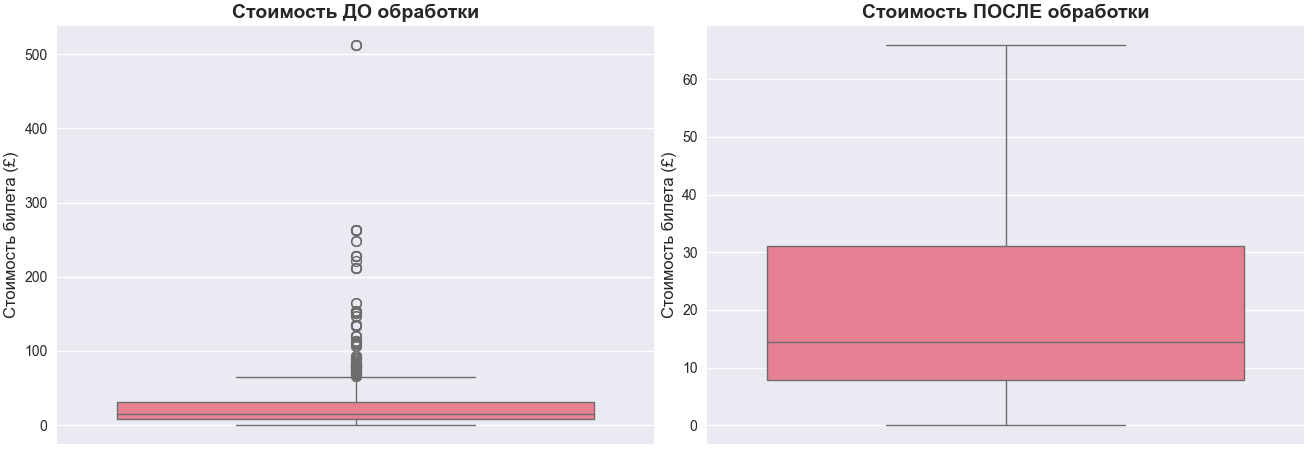
**4.4. Анализ признака "Возраст"**

### 

### Рисунок 9 - Анализ признака "Возраст"

* График "ДО обработки" (слева вверху): Исходное распределение возраста демонстрирует наличие ряда выбросов в верхней части — значения, превышающие примерно 65 лет, которые выходят за пределы верхнего "уса" диаграммы (Q3 + 1.5 \* IQR).
* График "ПОСЛЕ обработки" (справа вверху): После применения метода "обрезки" (capping), все точки-выбросы были устранены. Верхний предел распределения теперь ограничен максимальным значением, установленным в ходе обработки. Важно отметить, что медиана и межквартильный размах (тело "ящика") практически не изменились, что свидетельствует о сохранении основной структуры данных при удалении аномалий.

#### **4.5. Анализ признака "Стоимость билета"**



#### Рисунок 9 - Анализ признака "Стоимость билета"

* График "ДО обработки" (слева внизу): Исходное распределение стоимости билета характеризуется крайне высокой асимметрией и наличием большого числа экстремальных выбросов (значения доходят до 500+). Из-за этого основная часть распределения (IQR) сжата в нижней части графика, что затрудняет его интерпретацию.
* График "ПОСЛЕ обработки" (справа внизу): После применения логарифмического преобразования и последующей "обрезки", распределение стало значительно более сбалансированным. Все экстремальные выбросы были успешно обработаны. Масштаб оси Y кардинально изменился, что позволяет более детально рассмотреть распределение для основной массы пассажиров.

Общий вывод: Визуальное сравнение диаграмм подтверждает высокую эффективность проведенной предобработки. Процедура "обрезки" успешно устранила выбросы в признаках Возраст и Стоимость билета, сделав их распределения более робастными и пригодными для использования в моделях машинного обучения, чувствительных к аномальным значениям и масштабу данных.

### Заключение

В ходе выполнения данной лабораторной работы были успешно достигнуты поставленные цели: освоены базовые навыки работы с ключевыми библиотеками Python для анализа данных, такими как pandas, numpy и seaborn, а также применены основные методики предварительной обработки и визуализации данных на примере датасета «Титаник».

В процессе анализа были выявлены и решены следующие ключевые проблемы качества данных:

1. Пропущенные значения: Основной проблемой было наличие пропусков в столбце Age. Эта задача была решена с помощью метода стратифицированной импутации: пропущенные значения заполнялись медианой, рассчитанной для каждой группы пассажиров в зависимости от их пола и класса обслуживания. Такой подход является более точным, чем заполнение общей медианой по всему датасету.
2. Аномальные значения (выбросы): Было обнаружено значительное количество выбросов в числовых признаках Age (пожилые пассажиры) и Fare (чрезвычайно дорогие билеты). Выбросы могут искажать результаты статистического анализа и негативно влиять на качество моделей машинного обучения. Для их обработки был применен метод «обрезки» (capping) на основе межквартильного размаха (IQR). Для признака Fare, который имел сильно асимметричное распределение, был использован комбинированный подход: сначала логарифмическое преобразование для сглаживания, а затем «обрезка». Эффективность обработки была наглядно продемонстрирована с помощью сравнительных диаграмм размаха.

Важно отметить, что реализованный программный код обладает гибкостью. Алгоритмы для обработки пропусков в числовых (Fare) и текстовых полях являются универсальными. Это означает, что если бы в исходном .csv файле отсутствовали данные в других столбцах, разработанный скрипт автоматически обнаружил и обработал бы их согласно заложенной логике, что обеспечивает его устойчивость и масштабируемость.

В результате проведенного анализа и многоэтапной предобработки исходный набор данных был успешно очищен, структурирован и приведен в формат, пригодный для дальнейшего построения и обучения моделей машинного обучения.