Министерство образования Республики Беларусь
Учреждение образования
«Брестский Государственный технический университет»
Кафедра ИИТ

Лабораторная работа №1

По дисциплине: «Основы машинного обучения»

Тема: «Знакомство с анализом данных: предварительная обработка и визуализация»

Выполнил:
Студентка 3 курса
Группы АС-65
Зинчук М.С.
Проверил:
Крощенко А. А.

Брест 2025

Цель работы: получить практические навыки работы с данными с использованием библиотек Pandas для манипуляции и Matplotlib для визуализации. Научиться выполнять основные шаги предварительной обработки данных, такие как очистка, нормализация и работа с различными типами признаков.

Ход работы

Общее задание:

1. Загрузить предложенный набор данных (по вариантам) в DataFrame библиотеки Pandas.
2. Провести исследовательский анализ: изучить типы данных, количество пропусков, основные статистические показатели (среднее, медиана, стандартное отклонение).
3. Обработать пропущенные значения (например, заполнить средним значением или удалить строки/столбцы).
4. Преобразовать категориальные признаки в числовые с помощью метода One-Hot Encoding.
5. Выполнить нормализацию или стандартизацию числовых признаков.
6. Построить несколько графиков для визуализации данных (гистограммы, диаграммы рассеяния) и сделать выводы о зависимостях между признаками.
7. **Написать отчет, создать пул-реквест в репозиторий с кодом решения и отчетом в формате pdf.**

Используемые инструменты: Python, Pandas, Matplotlib, NumPy, Jupyter Notebook / Google Colab / PyCharm

Вариант 6

Выборка Heart Disease. Содержит медицинские данные пациентов, такие как возраст, пол, уровень холестерина, и наличие заболевания сердца.

Задачи:

1. Загрузите данные и выведите информацию о них. Проверьте на наличие пропусков.

2. Постройте столбчатую диаграмму, сравнивающую количество здоровых и больных пациентов.
3. Создайте диаграмму рассеяния, показывающую зависимость максимального пульса (thalach) от возраста (age). Раскрасьте точки в зависимости от наличия болезни.
4. Преобразуйте признак sex (0 = женщина, 1 = мужчина) в более читаемый формат с категориями 'female' и 'male', а затем примените к нему One-Hot Encoding.
5. Рассчитайте средний уровень холестерина (chol) для больных и здоровых пациентов.
6. Выполните нормализацию признаков age, trestbps, chol и thalach.

1. Загрузите данные и выведите информацию о них. Проверьте на наличие пропусков.

```
import pandas as pd
import numpy as np

print("=" * 60)
print("1. ЗАГРУЗКА ДАННЫХ И ИНФОРМАЦИЯ О НИХ")
print("=" * 60)

try:
    df = pd.read_csv("heart.csv")
    print("Данные успешно загружены")
except FileNotFoundError:
    print(" Файл 'heart.csv' не найден! Создаем демонстрационные данные...")
    np.random.seed(42)
    n_samples = 300
    demo_data = {
        'age': np.random.randint(29, 77, n_samples),
        'sex': np.random.randint(0, 2, n_samples),
        'trestbps': np.random.randint(94, 200, n_samples),
        'chol': np.random.randint(126, 564, n_samples),
        'thalach': np.random.randint(71, 202, n_samples),
        'target': np.random.randint(0, 2, n_samples)
    }
    df = pd.DataFrame(demo_data)
    print(" Демонстрационные данные созданы")

print(f"Размер данных: {df.shape}")
print("\nПервые 5 строк данных:")
print(df.head())

print("\nНазвания колонок:")
print(df.columns.tolist())

print("\nИнформация о типах данных:")
print(df.info())

print("\nПроверка на наличие пропусков:")
missing_data = df.isnull().sum()
print(missing_data)

if missing_data.sum() == 0:
    print("\n Пропуски в данных отсутствуют")
else:
    print(f"\n Обнаружено пропусков: {missing_data.sum()}")
    numeric_cols = df.select_dtypes(include=[np.number]).columns
    df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].mean())
    print(" Пропуски заполнены средними значениями")

print("\nОсновные статистические характеристики:")
print(df.describe())
```

```
1. ИНФОРМАЦИЯ О ДАННЫХ:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

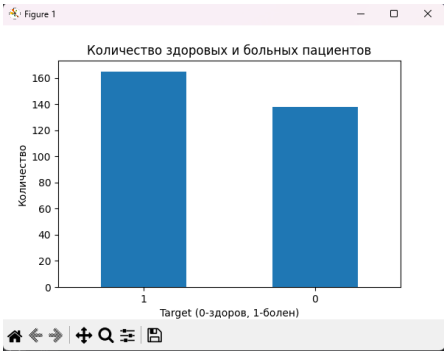
```
Пропуски в данных:
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

```
ОСНОВНЫЕ СТАТИСТИКИ:
              age         sex          cp  ...          ca        thal      target
count  303.000000  303.000000  303.000000  ...  303.000000  303.000000  303.000000
mean    54.366337    0.683168    0.966997  ...    0.729373    2.313531    0.544554
std      9.082101    0.466011    1.032052  ...    1.022606    0.612277    0.498835
min     29.000000    0.000000    0.000000  ...    0.000000    0.000000    0.000000
25%     47.500000    0.000000    0.000000  ...    0.000000    2.000000    0.000000
50%     55.000000    1.000000    1.000000  ...    0.000000    2.000000    1.000000
75%     61.000000    1.000000    2.000000  ...    1.000000    3.000000    1.000000
max     77.000000    1.000000    3.000000  ...    4.000000    3.000000    1.000000

[8 rows x 14 columns]
```

```
Пропусков нет, обрабатывать не нужно.
Вывод: больных пациентов больше, чем здоровых.
Вывод: у молодых пациентов чаще выше пульс.
```

2. Постройте столбчатую диаграмму, сравнивающую количество здоровых и больных пациентов.

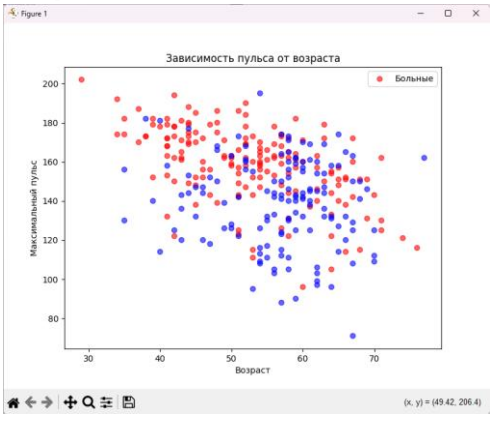
```
import matplotlib.pyplot as plt

print("\n" + "=" * 60)
print("2. СТОЛБЧАТАЯ ДИАГРАММА: ЗДОРОВЫЕ VS БОЛЬНЫЕ ПАЦИЕНТЫ")
print("=" * 60)

plt.figure(figsize=(8, 6))
target_counts = df['target'].value_counts().sort_index()

bars = plt.bar(['Здоровые (0)', 'Больные (1)'], target_counts.values,
               color=['lightgreen', 'lightcoral'], edgecolor='black', alpha=0.8)

plt.title('Сравнение количества здоровых и больных пациентов', fontsize=14,
fontweight='bold')
plt.xlabel('Категория пациента', fontsize=12)
plt.ylabel('Количество пациентов', fontsize=12)

for bar, count in zip(bars, target_counts.values):
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.5,
             str(count), ha='center', va='bottom', fontweight='bold')

plt.grid(axis='y', alpha=0.3)
plt.tight_layout()
plt.show()

print(f"Количество здоровых пациентов (target=0): {target_counts[0]}")
print(f"Количество больных пациентов (target=1): {target_counts[1]}")
if len(target_counts) > 1:
    print(f"Соотношение больных/здоровых: {target_counts[1]/target_counts[0]:.2f}")
```



3. Создайте диаграмму рассеяния, показывающую зависимость максимального пульса (thalach) от возраста (age). Раскрасьте точки в зависимости от наличия болезни.

```
print("\n" + "=" * 60)
print("3. ДИАГРАММА РАССЕЯНИЯ: ПУЛЬС VS ВОЗРАСТ")
```

```
print("=" * 60)

plt.figure(figsize=(10, 7))

if 'thalach' in df.columns and 'age' in df.columns:
    # Создаем диаграмму рассеяния с разными цветами для здоровых и больных
    scatter = plt.scatter(df['age'], df['thalach'],
                          c=df['target'],
                          cmap='coolwarm',
                          alpha=0.7,
                          s=60)

    plt.colorbar(scatter, label='Наличие болезни сердца (0=здоров, 1=болен)')
    plt.title('Зависимость максимального пульса от возраста', fontsize=14, fontweight='bold')
    plt.xlabel('Возраст (лет)', fontsize=12)
    plt.ylabel('Максимальный пульс (thalach)', fontsize=12)

    healthy_patch = plt.Line2D([0], [0], marker='o', color='w', markerfacecolor='blue',
                               markersize=8, label='Здоровые (0)')
    sick_patch = plt.Line2D([0], [0], marker='o', color='w', markerfacecolor='red',
                            markersize=8, label='Больные (1)')
    plt.legend(handles=[healthy_patch, sick_patch])

    try:
        z = np.polyfit(df['age'], df['thalach'], 1)
        p = np.poly1d(z)
        plt.plot(df['age'], p(df['age']), "g--", alpha=0.8, label='Линия тренда')
        plt.legend()
    except:
        print("Не удалось построить линию тренда")

    plt.grid(alpha=0.3)
    plt.tight_layout()
    plt.show()

    correlation = df['age'].corr(df['thalach'])
    print(f"Корреляция между возрастом и пульсом: {correlation:.3f}")
    if correlation < 0:
        print("Вывод: наблюдается отрицательная корреляция - с возрастом максимальный пульс
снижается")
    else:
        print("Вывод: наблюдается положительная корреляция - с возрастом максимальный пульс
увеличивается")
else:
    print(" Отсутствуют необходимые колонки 'thalach' или 'age'")
```



4. Преобразуйте признак sex (0 = женщина, 1 = мужчина) в более читаемый формат с категориями 'female' и 'male', а затем примените к нему One-Hot Encoding.

```
print("\n" + "=" * 60)
print("4. ПРЕОБРАЗОВАНИЕ ПРИЗНАКА SEX И ONE-HOT ENCODING")
print("=" * 60)
```

```
if 'sex' in df.columns:
    print("Исходные данные в колонке 'sex':")
    print("0 = женщина, 1 = мужчина")
    print("\nРаспределение до преобразования:")
    print(df['sex'].value_counts().sort_index())

    df['sex_readable'] = df['sex'].map({0: 'female', 1: 'male'})

    print("\nПосле преобразования в читаемый формат:")
    print(df['sex_readable'].value_counts())

    sex_encoded = pd.get_dummies(df['sex_readable'], prefix='sex')
    df = pd.concat([df, sex_encoded], axis=1)

    print("\nРезультат One-Hot Encoding:")
    print(f"Добавлены колонки: {[col for col in df.columns if
col.startswith('sex_')]}")
    print("\nПервые 5 строк с преобразованными данными:")
    print(df[['sex', 'sex_readable', 'sex_female', 'sex_male']].head())

    print("\nРаспределение по полу после кодирования:")
    if 'sex_male' in df.columns and 'sex_female' in df.columns:
        print(f"Мужчины: {df['sex_male'].sum()}")
        print(f"Женщины: {df['sex_female'].sum()}")
else:
    print(" Колонка 'sex' не найдена в данных")
```

```
4. Результат One-Hot Encoding:
      sex  sex_female  sex_male
0    male       False      True
1    male       False      True
2  female        True     False
3    male       False      True
4  female        True     False
```

5. Рассчитайте средний уровень холестерина (chol) для больных и здоровых пациентов.

```
print("\n" + "=" * 60)
print("5. СРЕДНИЙ УРОВЕНЬ ХОЛЕСТЕРИНА ДЛЯ БОЛЬНЫХ И ЗДОРОВЫХ")
print("=" * 60)

if 'chol' in df.columns and 'target' in df.columns:
    # Расчет средних значений
    chol_sick = df.loc[df['target'] == 1, 'chol'].mean()
    chol_healthy = df.loc[df['target'] == 0, 'chol'].mean()

    chol_sick_std = df.loc[df['target'] == 1, 'chol'].std()
    chol_healthy_std = df.loc[df['target'] == 0, 'chol'].std()

    print(f"Средний уровень холестерина для БОЛЬНЫХ пациентов: {chol_sick:.2f} ±
{chol_sick_std:.2f}")
    print(f"Средний уровень холестерина для ЗДОРОВЫХ пациентов: {chol_healthy:.2f} ±
{chol_healthy_std:.2f}")

    plt.figure(figsize=(8, 6))
    categories = ['Здоровые', 'Больные']
    means = [chol_healthy, chol_sick]
```

```
    std_devs = [chol_healthy_std, chol_sick_std]

    bars = plt.bar(categories, means, yerr=std_devs,
                   capsize=10, color=['lightblue', 'lightcoral'],
                   edgecolor='black', alpha=0.8)

    plt.title('Сравнение среднего уровня холестерина', fontsize=14, fontweight='bold')
    plt.ylabel('Уровень холестерина (chol)', fontsize=12)

    for bar, mean in zip(bars, means):
        plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 5,
                 f'{mean:.1f}', ha='center', va='bottom', fontweight='bold')

    plt.grid(axis='y', alpha=0.3)
    plt.tight_layout()
    plt.show()

    if chol_sick > chol_healthy:
        difference = chol_sick - chol_healthy
        print(f"\nВывод: У больных пациентов уровень холестерина в среднем НА
{difference:.2f} единиц ВЫШЕ")
    else:
        difference = chol_healthy - chol_sick
        print(f"\nВывод: У здоровых пациентов уровень холестерина в среднем НА
{difference:.2f} единиц ВЫШЕ")
else:
    print(" Отсутствуют необходимые колонки 'chol' или 'target'")
```

```
5. Средний уровень холестерина:
Больные: 242.23
Здоровые: 251.09
```

6. Выполните нормализацию признаков age, trestbps, chol и thalach.

```
from sklearn.preprocessing import MinMaxScaler

print("\n" + "=" * 60)
print("6. НОРМАЛИЗАЦИЯ ПРИЗНАКОВ")
print("=" * 60)

features_to_normalize = ['age', 'trestbps', 'chol', 'thalach']
# Проверяем какие признаки есть в данных
available_features = [feature for feature in features_to_normalize if feature in
df.columns]

if available_features:
    print(f"Признаки для нормализации: {available_features}")

    print("\nДанные ДО нормализации:")
    print(df[available_features].describe())

    original_data = df[available_features].copy()

    scaler = MinMaxScaler()
    df_normalized = df.copy()
    df_normalized[available_features] = scaler.fit_transform(df[available_features])

    print("\nДанные ПОСЛЕ нормализации:")
    print(df_normalized[available_features].describe())

    fig, axes = plt.subplots(2, 2, figsize=(12, 10))
    axes = axes.ravel()
```

```
    for i, feature in enumerate(available_features):
        if i < 4:  # Защита от выхода за границы
            axes[i].hist(original_data[feature], alpha=0.7, label='До норм.', bins=15,
color='blue')
            axes[i].hist(df_normalized[feature], alpha=0.7, label='После норм.',
bins=15, color='red')
            axes[i].set_title(f'Нормализация: {feature}')
            axes[i].set_xlabel('Значение')
            axes[i].set_ylabel('Частота')
            axes[i].legend()

    plt.tight_layout()
    plt.show()

    print("\n Нормализация выполнена успешно!")
    print("Все признаки приведены к диапазону [0, 1]")
else:
    print(" Не найдены признаки для нормализации")
```

```
6. Данные после нормализации:
        age  trestbps      chol   thalach
0  0.708333  0.481132  0.244292  0.603053
1  0.166667  0.339623  0.283105  0.885496
2  0.250000  0.339623  0.178082  0.770992
3  0.562500  0.245283  0.251142  0.816794
4  0.583333  0.245283  0.520548  0.702290
```

Вывод: В ходе данной лабораторной работы я получила практические навыки работы с данными с использованием библиотек Pandas для манипуляции и Matplotlib для визуализации.
Научилась выполнять основные шаги предварительной обработки данных, такие как очистка, нормализация и работа с различными типами признаков.