

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

**Лабораторная работа №2**  
**По дисциплине : “ОМО”**  
**Тема: “Линейные модели**  
**для задач регрессии и классификации”**

**Выполнил:**  
Студент 3 курса  
Группы АС-66  
Ярома А. И .  
**Проверил:**  
Крощенко А.А

**Брест 2025**

**Цель:** Изучить применение линейной и логистической регрессии для решения практических задач. Научиться обучать модели, оценивать их качество с помощью соответствующих метрик и интерпретировать результаты.

## Вариант 2

**Регрессия** (Прогнозирование медицинских расходов)

1. Medical Cost Personal Datasets

2. Предсказать страховые выплаты (charges)

3. Задания:

Загрузите и обработайте категориальные признаки (например, sex, smoker);

Обучите модель линейной регрессии для предсказания charges

Рассчитайте MAE (Mean Absolute Error) и R2;

Визуализируйте зависимость charges от bmi (индекс визуализируйте массы тела) с помощью диаграммы рассеяния и линии регрессии.

**Классификация** (Диагностика заболеваний сердца)

1. Heart Disease UCI

2. Предсказать наличие у пациента болезни сердца (target)

3. Задания:

Загрузите данные и разделите их на обучающую и тестовую выборки;

Обучите модель логистической регрессии;

Оцените модель помощью Accuracy, Precision, Recall и F1-score; C

Остройте матрицу ошибок.

## Ход работы

**Регрессия**

```
import os
```

```
import sys
```

```
import numpy as np
```

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.linear_model import LinearRegression
```

```
MED_CSV = "medical_cost_personal_dataset.csv"
```

```
OUT_PNG = "charges_vs_bmi.png"
```

```
RANDOM_STATE = 42
```

```
def ensure_exists(path):
```

```
    if not os.path.exists(path):
```

```
        raise FileNotFoundError(f"Файл не найден: {path}")
```

```
def save_figure(fig, path):
```

```
    fig.savefig(path, bbox_inches="tight", dpi=200)
```

```
    plt.close(fig)
```

```
    print(f"PNG сохранён: {path}")
```

```

def main():
    try:
        ensure_exists(MED_CSV)
    except FileNotFoundError as e:
        print(e)
        sys.exit(1)

    df = pd.read_csv(MED_CSV)
    print("Загружен файл:", MED_CSV, "| shape:", df.shape)

    # Проверка обязательных столбцов
    if 'charges' not in df.columns:
        print("В датасете отсутствует столбец charges")
        sys.exit(0)

    if 'bmi' not in df.columns:
        print("В датасете отсутствует столбец bmi")
        sys.exit(0)

    plot_df = df[['bmi', 'charges']].dropna()
    if plot_df.shape[0] < 2:
        print("Недостаточно данных с обеими колонками 'bmi' и 'charges' для построения графика — ничего не делаю.")
        sys.exit(0)

    X_bmi = plot_df[['bmi']].values
    y_charges = plot_df['charges'].values

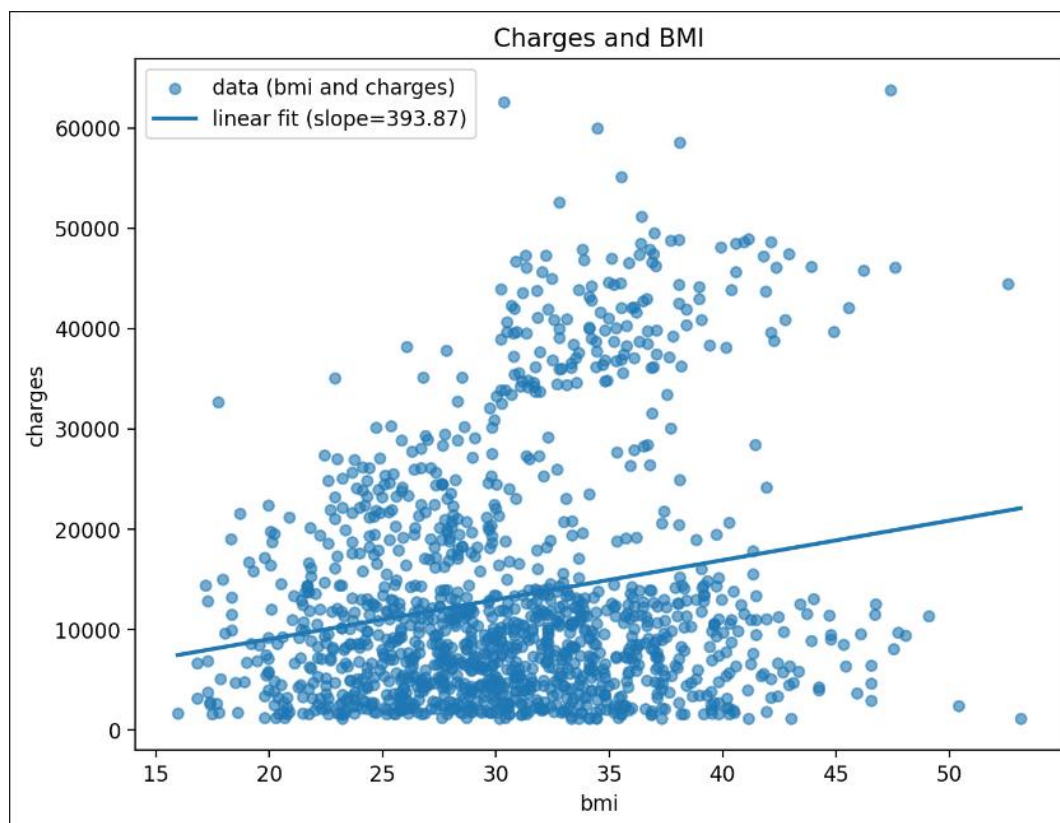
    # Простая линейная регрессия
    lr = LinearRegression()
    lr.fit(X_bmi, y_charges)
    xs = np.linspace(X_bmi.min(), X_bmi.max(), 200).reshape(-1, 1)
    ys = lr.predict(xs)

    fig, ax = plt.subplots(figsize=(8, 6))
    ax.scatter(X_bmi, y_charges, alpha=0.6, s=25, label="data (bmi and charges)")
    ax.plot(xs, ys, linewidth=2, label=f"linear fit (slope={lr.coef_[0]:.2f})")
    ax.set_xlabel("bmi")
    ax.set_ylabel("charges")
    ax.set_title("Charges and BMI")
    ax.legend()

    save_figure(fig, OUT_PNG)
    print("График построен и сохранён. Завершение.")

if __name__ == "__main__":
    main()

```



## Классификация

```
import os
import sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

CSV_PATH = "heart_disease_uci.csv"
OUT_PNG = "confusion_matrix.png"
TEST_SIZE = 0.2
RANDOM_STATE = 42

def main():
    if not os.path.exists(CSV_PATH):
        print(f"Файл не найден: {CSV_PATH}")
        sys.exit(1)

    df = pd.read_csv(CSV_PATH)
    print("Загружен датасет:", df.shape)
    print("Колонки:", list(df.columns))

    # Целевой столбец
    if 'num' not in df.columns:
        print("Столбец 'num' (target) не найден — проверь CSV.")
```

```

sys.exit(1)

# Цель: болезнь сердца
y = (df['num'] > 0).astype(int)
X = df.drop(columns=['num'])

# Кодирование категориальных признаков
cat_cols = X.select_dtypes(include=['object']).columns
if len(cat_cols) > 0:
    print("Кодируем категориальные признаки:", list(cat_cols))
    X = pd.get_dummies(X, columns=cat_cols, drop_first=True)

imputer = SimpleImputer(strategy='mean')
X = pd.DataFrame(imputer.fit_transform(X), columns=X.columns)

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Разделение выборки
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=TEST_SIZE, random_state=RANDOM_STATE, stratify=y
)

# Обучение логистической регрессии
clf = LogisticRegression(max_iter=1000, solver='liblinear', random_state=RANDOM_STATE)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

# Метрики
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred, zero_division=0)
rec = recall_score(y_test, y_pred, zero_division=0)
f1 = f1_score(y_test, y_pred, zero_division=0)

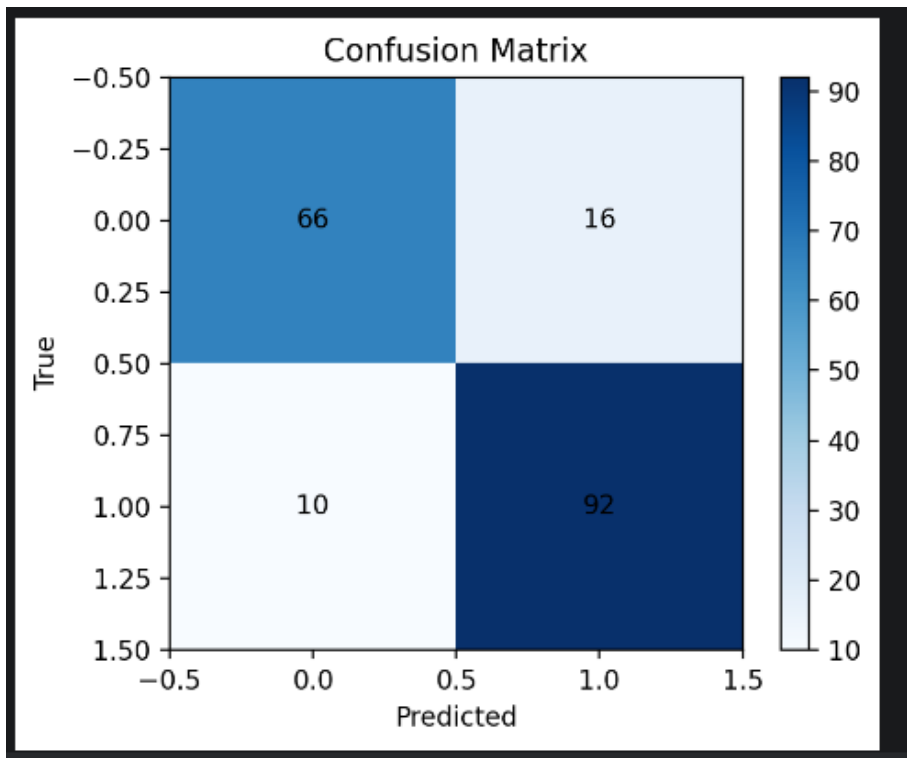
print("\n=== Метрики ===")
print(f"Accuracy = {acc:.4f}")
print(f"Precision = {prec:.4f}")
print(f"Recall = {rec:.4f}")
print(f"F1-score = {f1:.4f}")

# Матрица ошибок
cm = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(5, 4))
im = ax.imshow(cm, cmap=plt.cm.Blues)
ax.set_title("Confusion Matrix")
ax.set_xlabel("Predicted")
ax.set_ylabel("True")
for (i, j), val in np.ndenumerate(cm):
    ax.text(j, i, int(val), ha='center', va='center', color='black')
plt.colorbar(im)
plt.tight_layout()
fig.savefig(OUT_PNG, dpi=200, bbox_inches='tight')
plt.close(fig)
print(f"\nМатрица ошибок сохранена в: {OUT_PNG}")

```

```
if __name__ == "__main__":  
    main()
```

```
=== Метрики ===  
Accuracy  = 0.8587  
Precision = 0.8519  
Recall    = 0.9020  
F1-score  = 0.8762  
  
Матрица ошибок сохранена в: confusion_matrix.png
```



**Вывод:** Я изучил применение линейной и логистической регрессии для решения практических задач. Научиться обучать модели, оценивать их качество с помощью соответствующих метрик и интерпретировать результаты.