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-го курса
Группы АС-66
Цеван К.А.
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

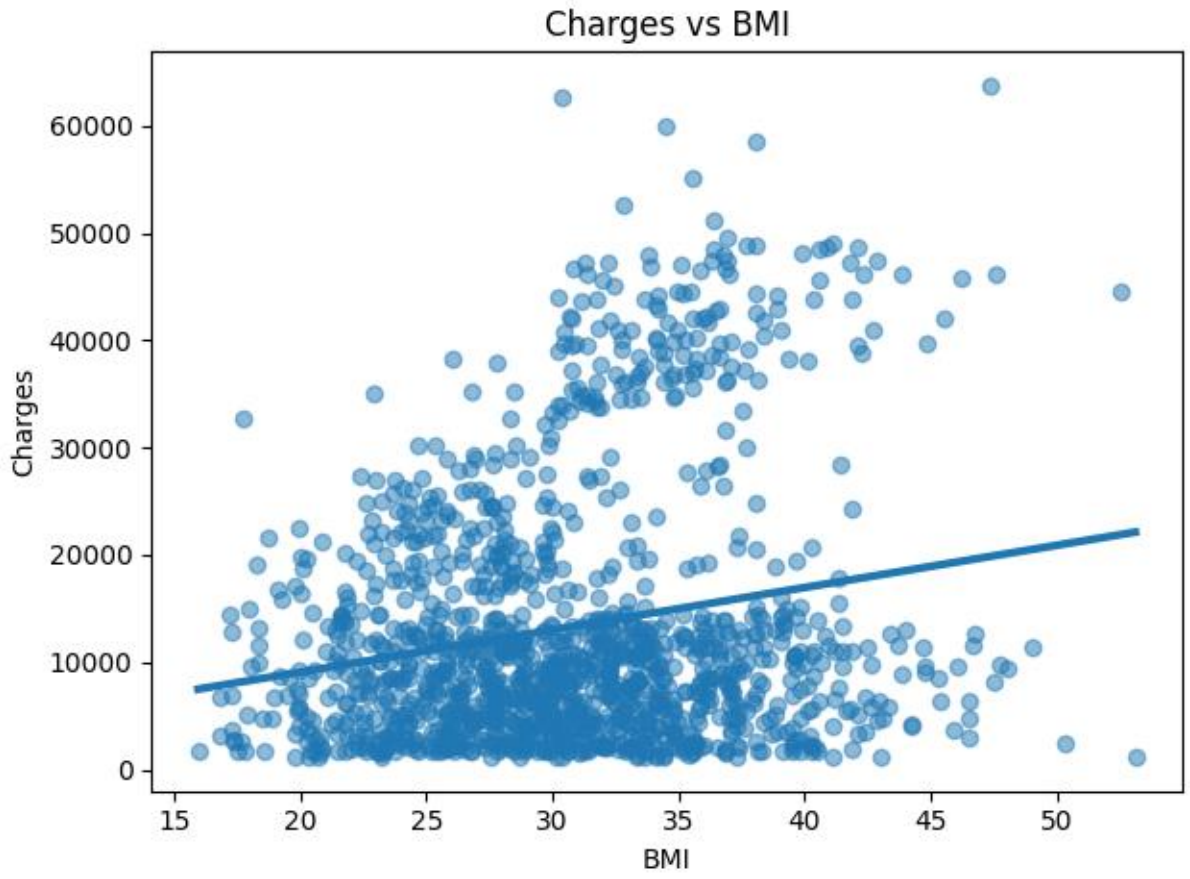
• Регрессия (Прогнозирование медицинских расходов)
1. Medical Cost Personal Datasets
2. Предсказать страховые выплаты (charges)
3. Задания:
§ загрузите и обработайте категориальные признаки (например, sex, smoker);
§ обучите модель линейной регрессии для предсказания charges;
§ рассчитайте MAE (Mean Absolute Error) и R2;
§ визуализируйте зависимость charges от bmi (индекс 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 с помощью Accuracy, Precision, Recall и F1-score;
§ постройте матрицу ошибок.

charges:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score

df = pd.read_csv("medical_cost_personal_dataset.csv")
df.columns = [c.lower() for c in df.columns]

y = df["charges"]
feature_cols = ["age", "sex", "bmi", "children", "smoker", "region"]
X = df[feature_cols]
```

```
cat_cols = ["sex", "smoker", "region"]
num_cols = [c for c in feature_cols if c not in cat_cols]

preprocess = ColumnTransformer(
    transformers=[
        ("cat", OneHotEncoder(handle_unknown="ignore"), cat_cols),
        ("num", "passthrough", num_cols),
    ]
)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

model = LinearRegression()

X_train_tr = preprocess.fit_transform(X_train)
X_test_tr = preprocess.transform(X_test)

model.fit(X_train_tr, y_train)
y_pred = model.predict(X_test_tr)

print(f"MAE: {mean_absolute_error(y_test, y_pred):.2f}")
print(f"R²: {r2_score(y_test, y_pred):.3f}")

bmi_only = df[["bmi"]]
charges = df["charges"]
model_bmi = LinearRegression()
model_bmi.fit(bmi_only, charges)

plot_df = df[["bmi", "charges"]].copy().sort_values("bmi")
line_pred = model_bmi.predict(plot_df[["bmi"]])

plt.figure()
plt.scatter(df["bmi"], df["charges"], alpha=0.5)
plt.plot(plot_df["bmi"], line_pred, linewidth=3)
plt.xlabel("BMI")
plt.ylabel("Charges")
plt.title("Charges vs BMI")
plt.tight_layout()
plt.savefig("charges_vs_bmi.png")
```



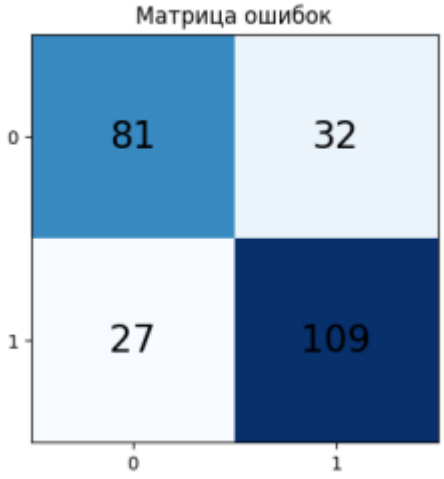

```
MAE: 4181.19
R²: 0.784
```

2.

```
from google.colab import files
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

files.upload()
data = pd.read_csv('heart_disease_uci.csv')

data['target'] = (data['num'] > 0).astype(int)
cols = ['age', 'trestbps', 'chol', 'thalch', 'oldpeak']
data = data[cols + ['target']].dropna()

X = data[cols]
y = data['target']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)

model = LogisticRegression(max_iter=1000)
```

```
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

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

cm = confusion_matrix(y_test, y_pred)
print("\nМатрица ошибок:")
print(cm)

plt.figure(figsize=(5, 4))
plt.imshow(cm, cmap='Blues')
for i in range(2):
    for j in range(2):
        plt.text(j, i, cm[i, j], ha='center', va='center', fontsize=18)

plt.xticks([0, 1])
plt.yticks([0, 1])
plt.title('Матрица ошибок')
plt.show()
```

```
Accuracy: 0.7631
Precision: 0.7730
Recall: 0.8015
F1-score: 0.7870

Матрица ошибок:
[[ 81  32]
 [ 27 109]]

 [ [TN, FP]
 [FN, TP] ]
```



Вывод: Изучили применение линейной и логистической регрессии для решения практических задач.