

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

Лабораторная работа №2  
По дисциплине «ОМО»

Выполнил:  
Студент 3-го курса  
Группы АС-65  
Грущинский Д.Д.  
Проверил:  
Крощенко А.А.

Брест 2025

**Цель работы:** 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

### Ход работы

Общее задание: выполнить задания по варианту (регрессия и классификация), построить все требуемые визуализации и рассчитать метрики, написать отчет, создать пул-реквест в репозиторий с кодом решения и отчетом в формате pdf.

- 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

### Код для задания прогнозирования медицинских расходов:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

```

from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score

data = pd.read_csv('medical_cost_personal_dataset.csv')

data_encoded = pd.get_dummies(data, drop_first=True)

X = data_encoded.drop('charges', axis=1)
y = data_encoded['charges']

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

model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"MAE: {mae}, R2: {r2}")

plt.figure(figsize=(10, 6))
sns.scatterplot(x=data['bmi'], y=data['charges'])
sns.lineplot(x=data['bmi'],
y=model.predict(pd.get_dummies(data.drop('charges', axis=1),
drop_first=True)), color='red')
plt.title('Зависимость Charges от BMI с линией регрессии')
plt.xlabel('BMI')
plt.ylabel('Charges')
plt.show()

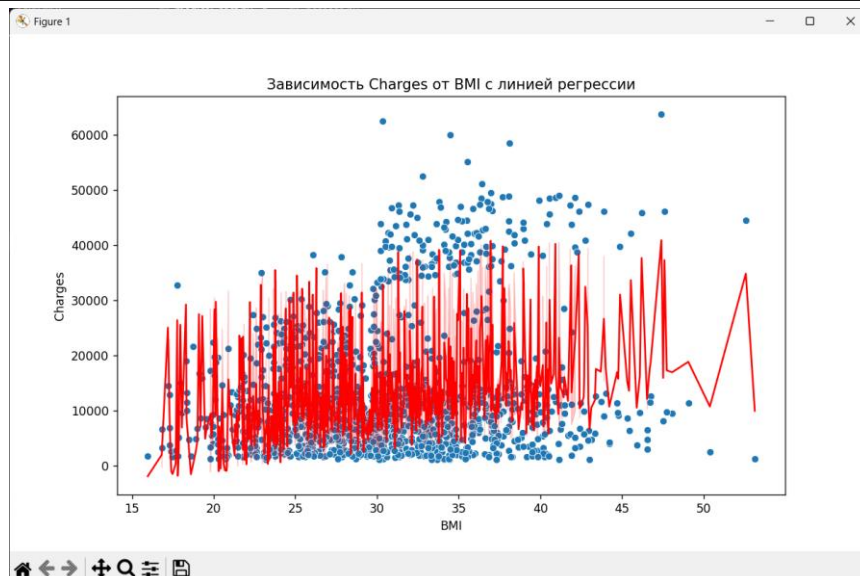
```

Вывод данного задания:

```

C:\Users\user\Documents>python 10_10_2022/ma_1000
MAE: 4181.194473753654, R2: 0.7835929767120723

```



### Код для задания диагностики заболеваний сердца:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, confusion_matrix, ConfusionMatrixDisplay

heart_data = pd.read_csv('heart_disease_uci.csv').dropna()

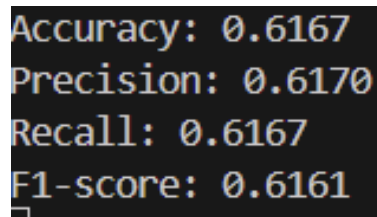
X = heart_data.drop('num', axis=1)
y = heart_data['num']

X_encoded = pd.get_dummies(X, drop_first=True)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_encoded)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)
logreg = LogisticRegression(max_iter=5000)
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)

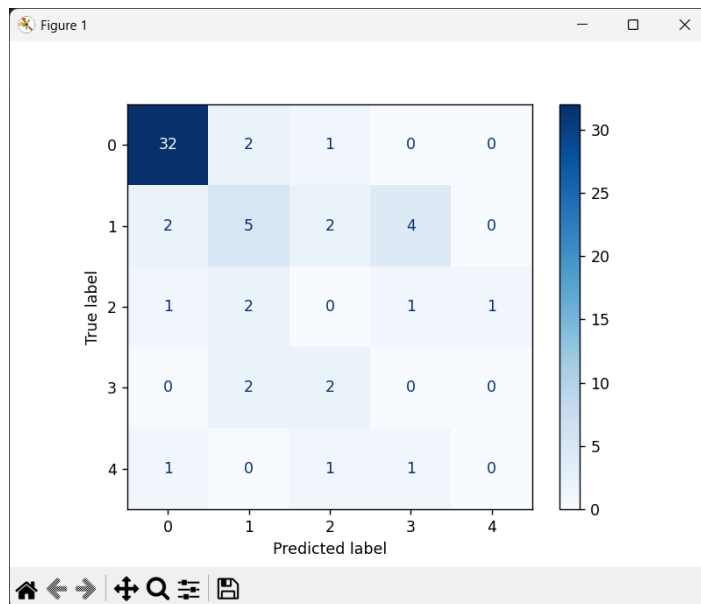
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
conf_matrix = confusion_matrix(y_test, y_pred)
ConfusionMatrixDisplay(conf_matrix).plot(cmap='Blues')
plt.show()
```

### Вывод данного задания:



```
Accuracy: 0.6167
Precision: 0.6170
Recall: 0.6167
F1-score: 0.6161
```



**Вывод:** в процессе выполнения данной лабораторной работы был получен практический опыт обучения и реализации регрессивной модели, а также усовершенствованы знания в области анализа исходных данных и их последующей интерпретации.