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-65
Гуща И.В.
Вариант 3
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.

Ход работы

Вариант 3

• Регрессия (Прогнозирование расхода топлива)

1. Auto MPG
2. Предсказать расход топлива (mpg)
3. Задания:

загрузите данные, обработайте пропуски и категориальные признаки;

обучите модель линейной регрессии, используя в качестве признаков cylinders, horsepower, weight;

рассчитайте MSE и R^2;

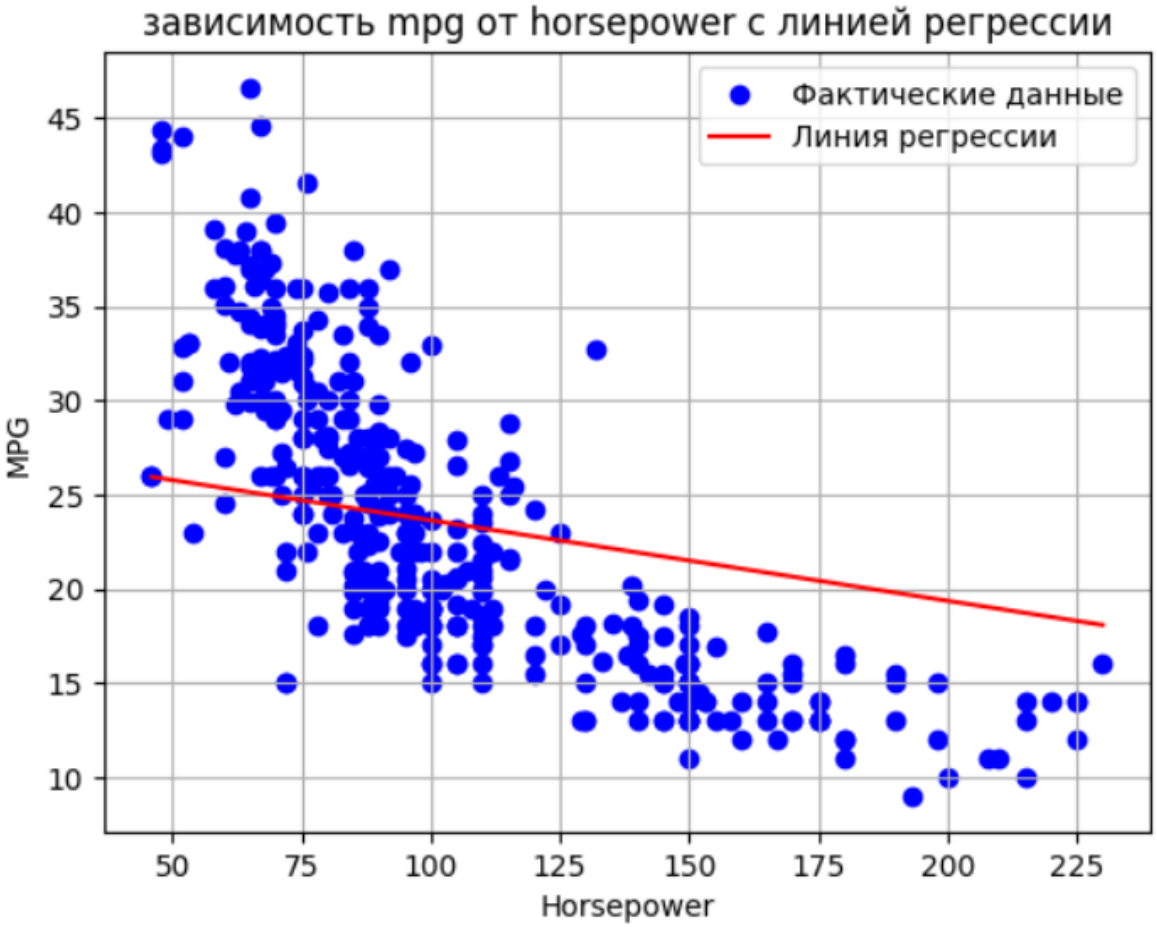
визуализируйте зависимость mpg от horsepower с линией регрессии.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# 1. Загрузка данных
df = pd.read_csv('auto-mpg.csv')  # путь к твоему файлу .csv
#2. обработка нулей
df['horsepower'] = pd.to_numeric(df['horsepower'], errors='coerce')
df = df.dropna(subset=['cylinders', 'horsepower', 'weight', 'mpg'])
#3. обучение
X = df[['cylinders', 'horsepower', 'weight']]
y = df['mpg']
model=LinearRegression().fit(X,y)
y_m = model.predict(X)
#4. признаки
mse=mean_squared_error(y,y_m)
print(mse)
r2=r2_score(y,y_m)
print(r2)
```

```
17.763871479977333
0.7076518943128077
```

```
#5. визуализация
plt.scatter(df['horsepower'], df['mpg'], color='blue',  label='Фактические данные')
mean_cylinders = np.mean(df['cylinders'])
mean_weight = np.mean(df['weight'])
sorted_hp = np.sort(df['horsepower'])
#значения
hp_line = pd.DataFrame({
    'cylinders': mean_cylinders,
    'horsepower': sorted_hp,
    'weight': mean_weight
```

```
})
plt.plot(sorted_hp, model.predict(hp_line), color='red', label='Линия ре-
грессии')
plt.title('зависимость mpg от horsepower с линией регрессии')
plt.xlabel('Horsepower')
plt.ylabel('MPG')
plt.legend()
plt.grid(True)
plt.show()
```



• Классификация (Диагностика диабета)

1. Pima Indians Diabetes
2. Предсказать наличие диабета (Outcome)
3. Задания:

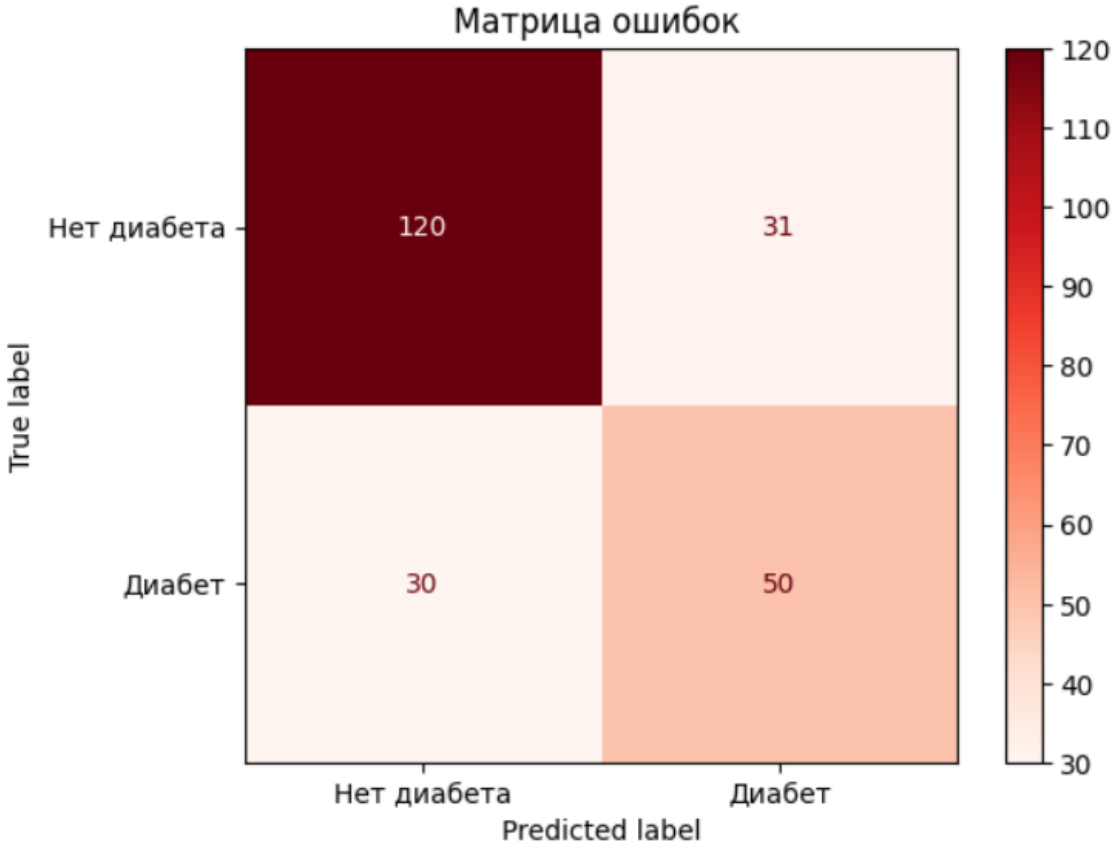
загрузите данные, выполните стандартизацию признаков;
обучите модель логистической регрессии;
рассчитайте Accuracy, Precision и Recall;
постройте матрицу ошибок и сделайте выводы о количестве ложноположительных и ложноотрицательных срабатываний.

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

#1. Загрузка
df = pd.read_csv("pima-indians-diabetes.csv", skiprows=9, header=None)
df.columns = [
    'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
```

```
    'Insulin', 'BMI', 'DiabetesPedigree', 'Age', 'Outcome'
]
print(df.head())
print("Размер данных:", df.shape)

#2. разделение признаков и целевой
X = df.drop('Outcome', axis=1)
y = df['Outcome']

#3. Стандартизация признаков
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)
print(X_scaled)

#4. разделение на обучающую и тестовую выборку и обучение
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3,
random_state=42)
model = LogisticRegression()
model.fit(X_train, y_train)

#5. Предсказание и рассчет
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
```

```
Accuracy: 0.7359
Precision: 0.6173
Recall: 0.6250
```

```
#6.матрица
matrix = confusion_matrix(y_test, y_pred)
print(matrix)
disp = ConfusionMatrixDisplay(confusion_matrix=matrix, display_labels=['Нет
диабета', 'Диабет'])
disp.plot(cmap='Reds')
plt.title("Матрица ошибок ")
plt.show()
print(f"Ложноположительные: {matrix[0][1]}")
print(f"Ложноотрицательные: {matrix[1][0]}")
```




```
Ложноположительные: 31
Ложноотрицательные: 30
```

Вывод: линейная регрессия применяется в случае числовых значений (расход топлива), а логистическая регрессия – когда нужна классификация по принципу да/нет (наличие или отсутствие диабета).