

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  
Янчук А.Ю.  
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 13

### 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 R2;

§ визуализируйте зависимость mpg от horsepower с линией регрессии.

- 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
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (
    mean_squared_error, r2_score,
    accuracy_score, precision_score, recall_score, confusion_matrix
)

print("\n=== РЕГРЕССИЯ: Auto MPG ===")

df_auto = pd.read_csv("auto-mpg.csv")
df_auto['horsepower'] = pd.to_numeric(df_auto['horsepower'], errors='coerce')
df_auto['horsepower'] = df_auto['horsepower'].fillna(df_auto['horsepower'].mean())

X_auto = df_auto[['cylinders', 'horsepower', 'weight']]
y_auto = df_auto['mpg']
```

```

X_train, X_test, y_train, y_test = train_test_split(
    X_auto, y_auto, test_size=0.2, random_state=42
)

# === Обучение модели ===
reg_model = LinearRegression()
reg_model.fit(X_train, y_train)
y_pred = reg_model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("MSE:", round(mse, 3))
print("R²:", round(r2, 3))

plt.figure(figsize=(8,6))
sns.scatterplot(x=df_auto['horsepower'], y=df_auto['mpg'], alpha=0.6,
label="Данные")

hp_range = np.linspace(df_auto['horsepower'].min(),
df_auto['horsepower'].max(), 100)
cyl_mean = df_auto['cylinders'].mean()
weight_mean = df_auto['weight'].mean()

X_line = pd.DataFrame({
    'cylinders': [cyl_mean]*len(hp_range),
    'horsepower': hp_range,
    'weight': [weight_mean]*len(hp_range)
})

y_line = reg_model.predict(X_line)

plt.plot(hp_range, y_line, color='red', label="Модель (LinearRegression)")
plt.xlabel("Horsepower")
plt.ylabel("MPG")
plt.title("Зависимость расхода топлива от мощности двигателя (по обученной модели)")
plt.legend()
plt.show()

print("\n=== КЛАССИФИКАЦИЯ: Pima Indians Diabetes ===")

columns = [
    "Pregnancies", "Glucose", "BloodPressure", "SkinThickness",
    "Insulin", "BMI", "DiabetesPedigreeFunction", "Age", "Outcome"
]

df_pima = pd.read_csv(
    "pima-indians-diabetes.csv",
    comment="#", header=None, names=columns
)

X_pima = df_pima.drop("Outcome", axis=1)
y_pima = df_pima["Outcome"]

scaler = StandardScaler()
X_pima_scaled = scaler.fit_transform(X_pima)

X_train, X_test, y_train, y_test = train_test_split(
    X_pima_scaled, y_pima, test_size=0.2, random_state=42
)

clf_model = LogisticRegression(max_iter=1000)
clf_model.fit(X_train, y_train)
y_pred = clf_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("Accuracy:", round(accuracy, 3))
print("Precision:", round(precision, 3))
print("Recall:", round(recall, 3))

cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=["Нет диабета", "Диабет"],
            yticklabels=["Нет диабета", "Диабет"])
plt.xlabel("Предсказание")
plt.ylabel("Истина")
plt.title("Матрица ошибок")
plt.show()

tn, fp, fn, tp = cm.ravel()
print("\nРазбор матрицы ошибок:")
print(f"Истинно отрицательные (TN): {tn}")
print(f"Ложно положительные (FP): {fp}")
print(f"Ложно отрицательные (FN): {fn}")
print(f"Истинно положительные (TP): {tp}")

print("\nВыводы:")
print(f"- FP ({fp}) → модель ошибочно предсказала диабет у здоровых.")
print(f"- FN ({fn}) → модель пропустила случаи диабета (это критичнее в медицине).")

```

График регрессия:

Зависимость расхода топлива от мощности двигателя (по обученной модели)

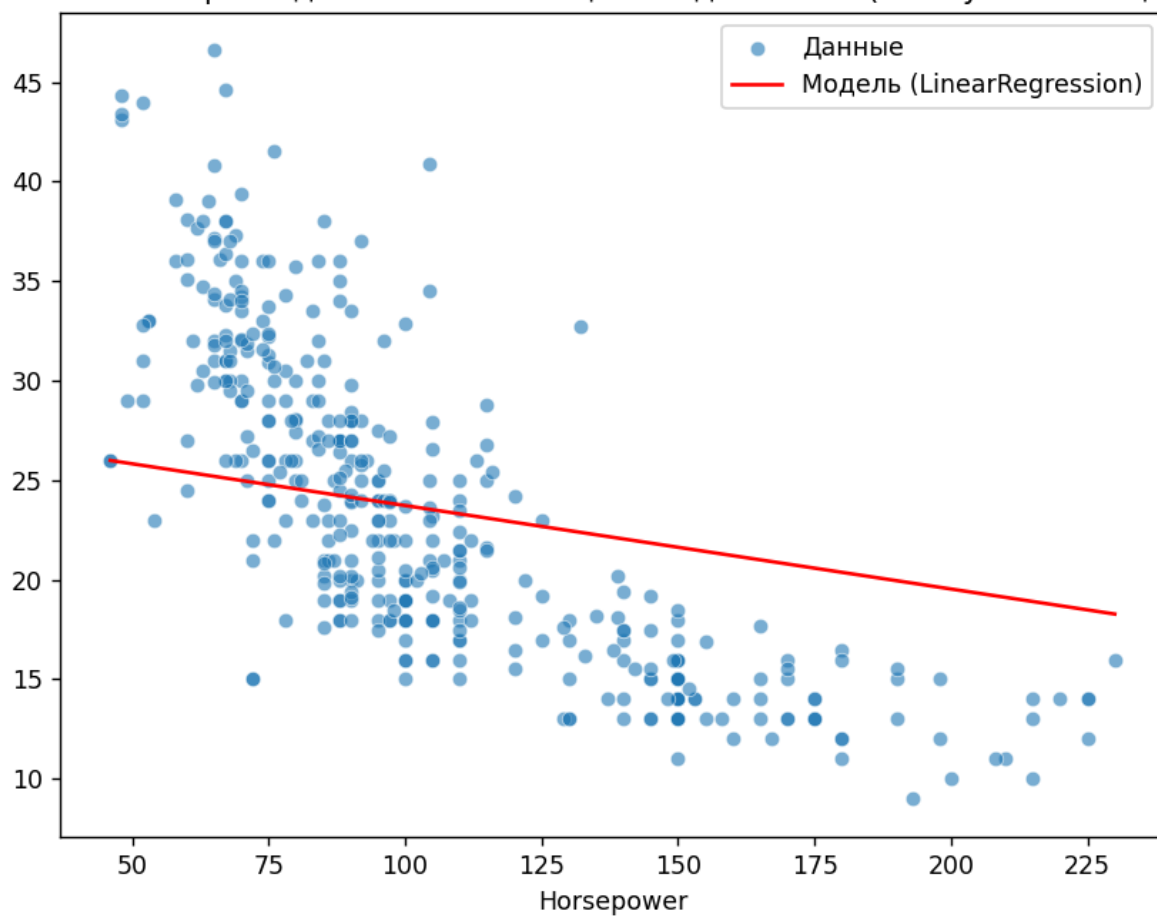
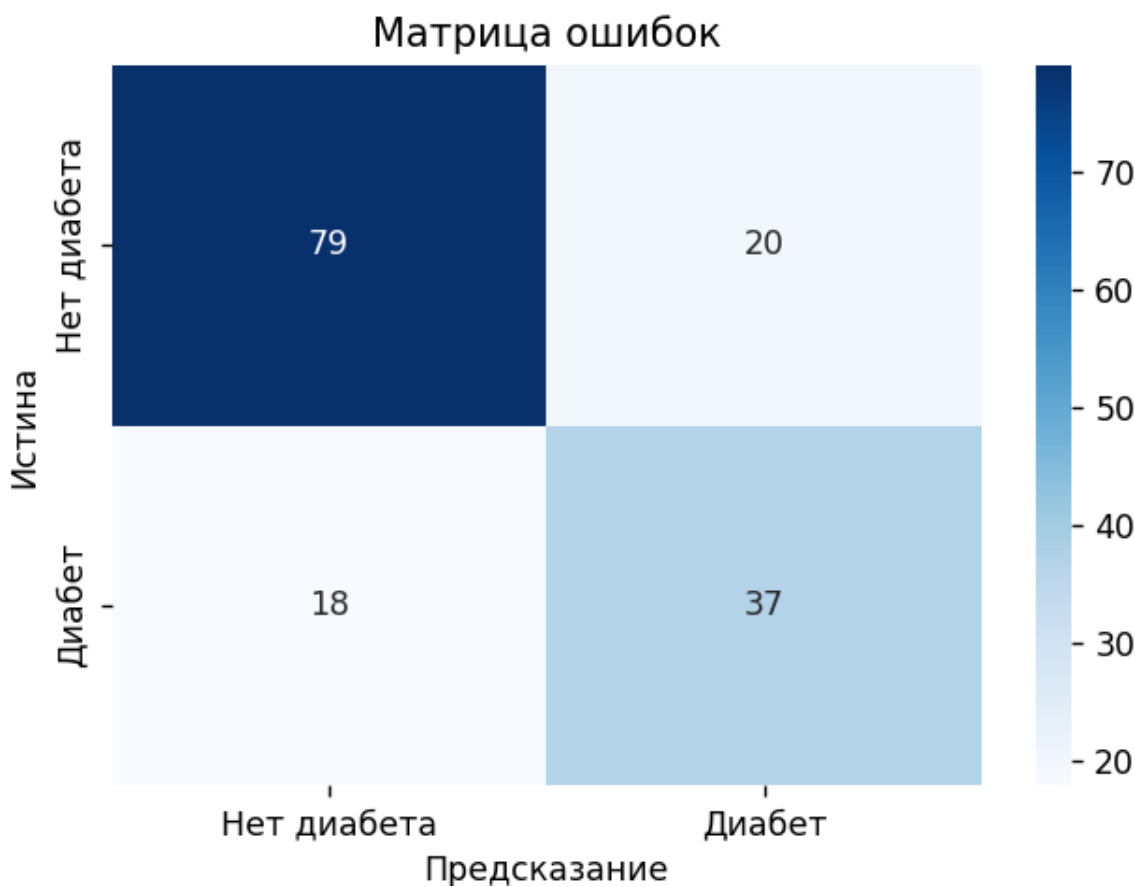


График классификация:



## Результаты:

=== РЕГРЕССИЯ: Auto MPG ===

MSE: 14.497

$R^2$ : 0.73

=== КЛАССИФИКАЦИЯ: Pima Indians Diabetes ===

Accuracy: 0.753

Precision: 0.649

Recall: 0.673

Разбор матрицы ошибок:

Истинно отрицательные (TN): 79

Ложно положительные (FP): 20

Ложно отрицательные (FN): 18

Истинно положительные (TP): 37

Выводы:

- FP (20) → модель ошибочно предсказала диабет у здоровых.
- FN (18) → модель пропустила случаи диабета (это критичнее в медицине).

**Вывод:** в результате выполнения данной лабораторной работы изучили применение линейной и логистической регрессии для решения практических задач. Научились обучать модели, оценивать их качество с помощью соответствующих метрик и интерпретировать результаты.