

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

Лабораторная работа №2
По дисциплине «Основы машинного обучения»
Тема: «Линейные модели
для задач регрессии и классификации»

Выполнил:
Студент 3 курса
Группы АС-66
Езепчук А.С.
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 3 Ход работы

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.

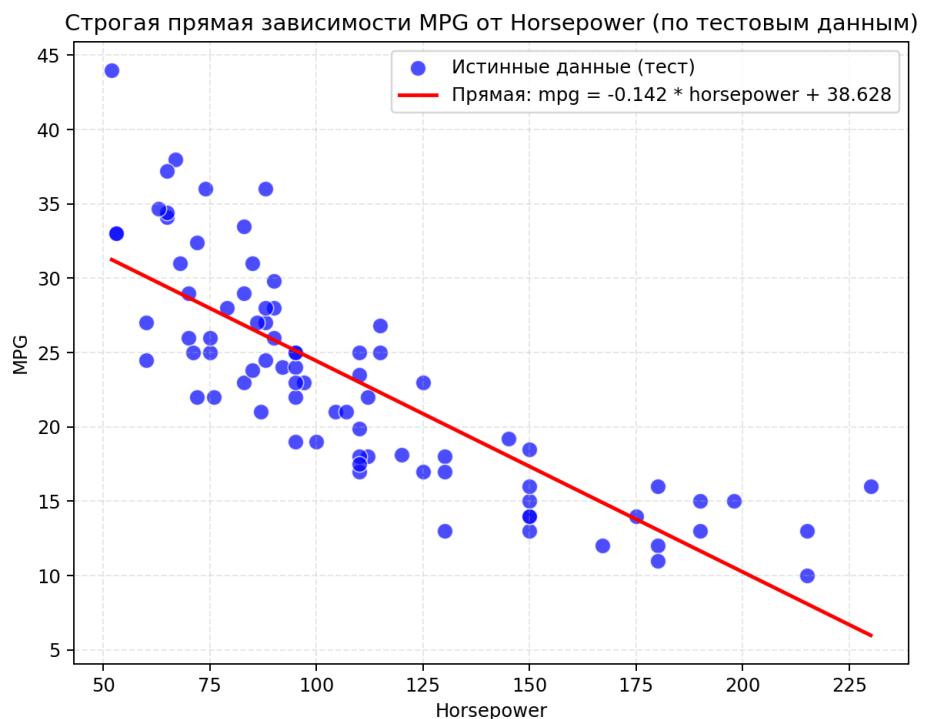
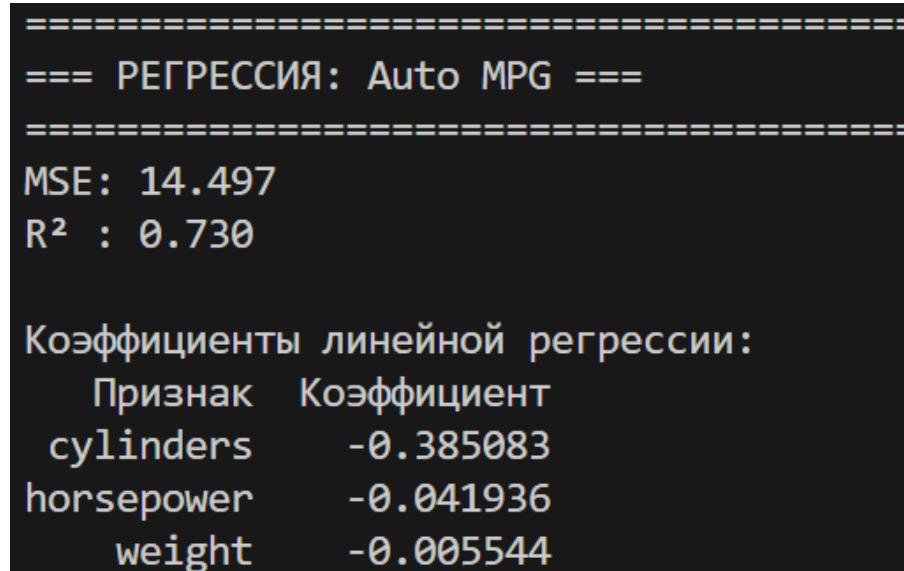
Регрессия

```
print("\n" + "="*40)
print("==== РЕГРЕССИЯ: Auto MPG ===")
print("="*40)
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
)
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(f"MSE: {mse:.3f}")
print(f"R2 : {r2:.3f}")
coef_df = pd.DataFrame({
```

```

"Признак": X_auto.columns,
"Коэффициент": reg_model.coef_
})
print("\nКоэффициенты линейной регрессии:")
print(coef_df.to_string(index=False))
plt.figure(figsize=(8, 6))
sns.scatterplot(
x=X_test['horsepower'], y=y_test,
alpha=0.7, label="Истинные данные (тест)", color="blue", s=70
)
k, b = np.polyfit(X_test['horsepower'].values, y_test.values, 1)
x_min, x_max = X_test['horsepower'].min(), X_test['horsepower'].max()
x_line = np.array([x_min, x_max])
y_line = k * x_line + b
plt.plot(x_line, y_line, color="red", linewidth=2,
label=f"Прямая: mpg = {k:.3f} * horsepower + {b:.3f}")
plt.xlabel("Horsepower")
plt.ylabel("MPG")
plt.title("Строгая прямая зависимости MPG от Horsepower (по тестовым данным)")
plt.grid(alpha=0.3, linestyle="--")
plt.legend()
plt.show()

```



Классификация

```
print("\n" + "="*40)
print("==== КЛАССИФИКАЦИЯ: Pima Indians Diabetes ===")
print("="*40)
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, class_weight="balanced")
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)
f1 = f1_score(y_test, y_pred)
print(f"Accuracy : {accuracy:.3f}")
print(f"Precision: {precision:.3f}")
print(f"Recall : {recall:.3f}")
print(f"F1-score : {f1:.3f}")
cm = confusion_matrix(y_test, y_pred)
cm_norm = cm.astype("float") / cm.sum(axis=1)[:, np.newaxis]
plt.figure(figsize=(6, 4))
sns.heatmap(
    cm_norm, annot=cm, fmt="d", cmap="Blues",
    xticklabels=["Нет диабета", "Диабет"],
    yticklabels=["Нет диабета", "Диабет"],
    annot_kws={"size": 14}
)
plt.xlabel("Предсказание")
plt.ylabel("Истина")
plt.title("Матрица ошибок (с нормировкой)")
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("\nВыходы:")
print(f"- FP ({fp}) → модель ошибочно предсказала диабет у здоровых.")
print(f"- FN ({fn}) → модель пропустила случаи диабета (это критичнее в медицине).")
```

```

=====
== КЛАССИФИКАЦИЯ: Pima Indians Diabetes ==
=====

Accuracy : 0.695
Precision: 0.557
Recall    : 0.709
F1-score  : 0.624

Разбор матрицы ошибок:
Истинно отрицательные (TN): 68
Ложно положительные (FP): 31
Ложно отрицательные (FN): 16
Истинно положительные (TP): 39

Выводы:
- FP (31) → модель ошибочно предсказала диабет у здоровых.
- FN (16) → модель пропустила случаи диабета (это критичнее в медицине).

```



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

