Московский государственный технический университет им. Н. Э. Баумана

Курс «Технологии машинного обучения»

Отчёт по рубежному контролю №2

«Технологии разведочного анализа и обработки данных.»

Вариант № 12

Выполнил:
Серов Савелий
Группа ИУ5-61Б

Дата: 13.05.25

Подпись:

Проверил:
Гапанюк Ю.Е.

Дата:

Подпись:

2025 г.

РК2 по дисциплине Технологии машинного обучения

Задание (вариант 12):

Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2 (по варианту для Вашей группы). Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

1. Датасет: https://www.kaggle.com/fivethirtyeight/fivethirtyeight-comic-characters-dataset (файл dc-wikia-data.csv)

РК2

Серов Савелий ИУ5-61Б

Вариант 12

Загрузка данных и импорт необходимых библиотек

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix
```

[23] ✓ 0.0s Python

```
# Загрузка данных
data = pd.read_csv('dc-wikia-data.csv')
data.head()
```

[24] ✓ 0.0s Python

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	FIRST APPEARANCE	YEAR
0	1422	Batman (Bruce Wayne)	\/wiki\/Batman_(Bruce_Wayne)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	3093.0	1939, May	1939.0
1	23387	Superman (Clark Kent)	\/wiki\/Superman_(Clark_Kent)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	2496.0	1986, October	1986.0
2	1458	Green Lantern (Hal Jordan)	\/wiki\/Green_Lantern_(Hal_Jordan)	Secret Identity	Good Characters	Brown Eyes	Brown Hair	Male Characters	NaN	Living Characters	1565.0	1959, October	1959.0
3	1659	James Gordon (New Earth)	\/wiki\/James_Gordon_(New_Earth)	Public Identity	Good Characters	Brown Eyes	White Hair	Male Characters	NaN	Living Characters	1316.0	1987, February	1987.0
4	1576	Richard Grayson (New Earth)	\/wiki\/Richard_Grayson_(New_Earth)	Secret Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	1237.0	1940, April	1940.0

Предварительный анализ и отбор целевой переменной

Целевой переменной в данной задаче выберем `SEX` — пол персонажа. Ограничим данные только строками, где явно указан пол (мужской или женский).

```
# Удалим строки с неопределенным полом
data = data[data['SEX'].isin(['Male Characters', 'Female Characters'])]
```

[25] ✓ 0.0s Python

Выбор признаков

Выбор признаков

Выберем признаки, влияющие на пол персонажа (цвет глаз, цвет волос, хорошесть)

```
# Упрощение и отбор признаков
features = ['EYE', 'HAIR', 'ALIGN']
X = data[features]
y = data['SEX']
```

[26] ✓ 0.0s Python

Обработка пропусков

Пропущенные значения будут заполнены наиболее частыми (модой) для соответствующего признака.

```
# Обработка пропусков
imputer = SimpleImputer(strategy='most_frequent')
X_imputed = pd.DataFrame(imputer.fit_transform(X), columns=X.columns)
```

[27] ✓ 0.0s Python

Кодирование признаков

```
# Кодирование категориальных признаков
X_encoded = pd.get_dummies(X_imputed)
```

[28] ✓ 0.0s Python

Разделение данных на обучающую и тестовую выборки

Тестовая выборка составит 20% от всего набора. Обучение будет проводиться на 80%.

```
# Кодирование целевой переменной
y_encoded = LabelEncoder().fit_transform(y)
# Разделение данных
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y_encoded, test_size=0.2, random_state=42)
```

[29] ✓ 0.0s Python

Generate + Code + Markdown

Обучение моделей

Мы обучим две модели:

- Логистическая регрессия
- Случайный лес

```
lr_model = LogisticRegression(max_iter=1000)
lr_model.fit(X_train, y_train)
lr_preds = lr_model.predict(X_test)
```

[30] ✓ 0.0s Python

```
# Случайный лес
```

```python
# Случайный лес
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train, y_train)
rf_preds = rf_model.predict(X_test)
```

[31] ✓ 0.1s Python

Оценка качества моделей

Оценим модели с использованием метрик:

- **Accuracy** — доля верных предсказаний
- **F1 Score** — гармоническое среднее точности и полноты, особенно важно при дисбалансе классов

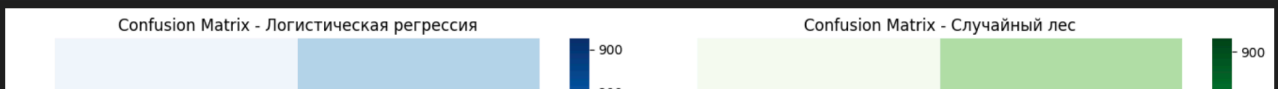
```python
print("--- Линейная/логистическая регрессия ---")
print("Accuracy:", accuracy_score(y_test, lr_preds))
print("F1 Score:", f1_score(y_test, lr_preds))
cm_lr = confusion_matrix(y_test, lr_preds)

print("\n--- Случайный лес ---")
print("Accuracy:", accuracy_score(y_test, rf_preds))
print("F1 Score:", f1_score(y_test, rf_preds))
cm_rf = confusion_matrix(y_test, rf_preds)
```

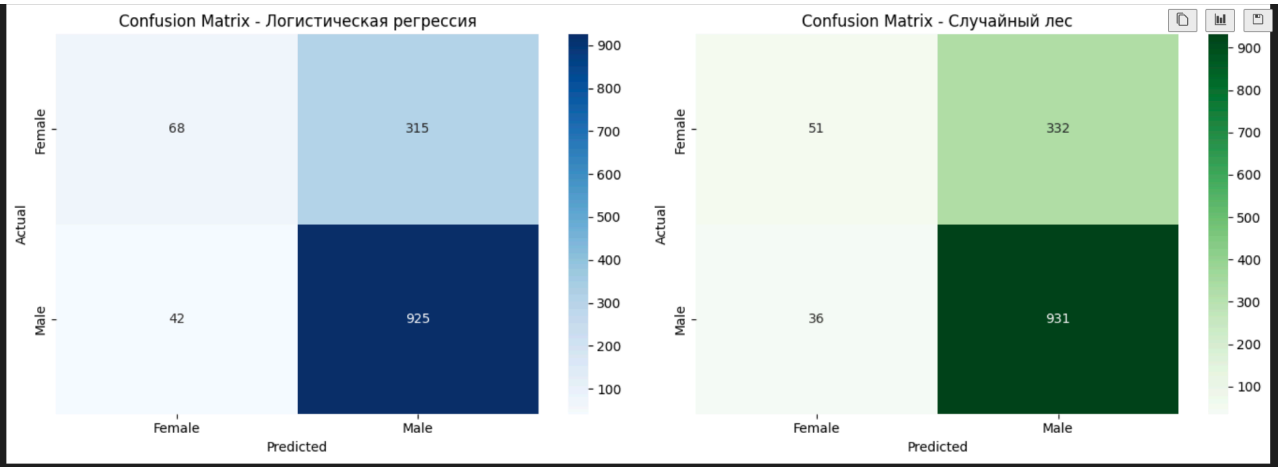
[32] ✓ 0.0s Python

```
--- Линейная/логистическая регрессия ---
Accuracy: 0.7355555555555555
F1 Score: 0.8382419574082465

--- Случайный лес ---
Accuracy: 0.7274074074074074
F1 Score: 0.8349775784753363
```

```python
# Визуализация матриц ошибок
fig, axs = plt.subplots(1, 2, figsize=(14, 5))

# Линейная/логистическая регрессия
sns.heatmap(cm_lr, annot=True, fmt='d', cmap='Blues', ax=axs[0])
axs[0].set_title('Confusion Matrix – Логистическая регрессия')
axs[0].set_xlabel('Predicted')
axs[0].set_ylabel('Actual')
axs[0].set_xticklabels(['Female', 'Male'])
axs[0].set_yticklabels(['Female', 'Male'])

# Случайный лес
sns.heatmap(cm_rf, annot=True, fmt='d', cmap='Greens', ax=axs[1])
axs[1].set_title('Confusion Matrix – Случайный лес')
axs[1].set_xlabel('Predicted')
axs[1].set_ylabel('Actual')
axs[1].set_xticklabels(['Female', 'Male'])
axs[1].set_yticklabels(['Female', 'Male'])

plt.tight_layout()
plt.show()
```

[33] ✓ 0.1s Python





Вывод:

- **Линейная/логистическая регрессия** демонстрирует базовое качество классификации, отличаясь простотой в реализации и интерпретации. Это хороший стартовый алгоритм для понимания линейных связей в данных.

- **Случайный лес** предлагает более высокую точность и F1-метрику, особенно эффективно справляясь со сложными, нелинейными зависимостями между признаками. Его ансамблевая природа часто обеспечивает лучшую производительность и устойчивость.

- В условиях дисбаланса классов (когда один класс, например, мужские персонажи, значительно преобладает), F1 Score остаётся ключевой метрикой для объективной оценки производительности модели. Она помогает получить более полную картину, учитывая как точность, так и полноту предсказаний.