Рубежный контроль №2

Ф.И.О.: Холодова К.А.
Группа: ИУ5Ц-82Б
Вариант: 32
Датасет: Disney Movies 1937–2016
Методы: Метод опорных векторов и Случайный лес

Цель

Построить модели регрессии для предсказания общего кассового сбора (Total Gross) по признакам фильмов.
Сравнить качество моделей по метрикам MAE и RMSE.

```
# Импорт библиотек
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Загрузка датасета
df = pd.read_csv("disney_movies_total_gross.csv")

# Очистка названий столбцов
df.columns = df.columns.str.strip().str.lower()
df.head()
```

```
                        movie_title release_date      genre mpaa_rating
\
0  Snow White and the Seven Dwarfs   1937-12-21    Musical           G

1                          Pinocchio   1940-02-09  Adventure           G

2                           Fantasia   1940-11-13    Musical           G

3                 Song of the South   1946-11-12  Adventure           G

4                         Cinderella   1950-02-15      Drama           G


   total_gross  inflation_adjusted_gross
0    184925485                5228953251
1     84300000                2188229052
2     83320000                2187090808
3     65000000                1078510579
4     85000000                 920608730
```

```
# Преобразование столбца total_gross в числовой формат
df['total_gross'] = df['total_gross'].replace('[\$,]', '',
regex=True).astype(float)

# Преобразование release_date в год
df['release_year'] = pd.to_datetime(df['release_date'],
errors='coerce').dt.year

# Удаление строк с пропущенными значениями
df = df.dropna(subset=["total_gross", "genre", "release_year"])

# Кодирование категориальных признаков (жанра)
df_encoded = pd.get_dummies(df[['genre']], drop_first=True)

# Добавим числовой признак release_year
df_encoded['release_year'] = df['release_year']

# Формируем признаки (X) и целевую переменную (y)
X = df_encoded
y = df['total_gross']

# Разделение данных на обучающую и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

Метод 1: Метод опорных векторов (SVR)

```
# Обучение модели SVR
model_svr = SVR()
model_svr.fit(X_train, y_train)

# Предсказание
y_pred_svr = model_svr.predict(X_test)

# Оценка качества модели
mae_svr = mean_absolute_error(y_test, y_pred_svr)
rmse_svr = np.sqrt(mean_squared_error(y_test, y_pred_svr))

mae_svr, rmse_svr
```

```
(np.float64(45873467.65434713), np.float64(77644620.54004093))
```

Метод 2: Случайный лес (Random Forest)

```
# Обучение модели Random Forest
model_rf = RandomForestRegressor(n_estimators=100, random_state=42)
model_rf.fit(X_train, y_train)

# Предсказание
y_pred_rf = model_rf.predict(X_test)
```

```
# Оценка качества модели
mae_rf = mean_absolute_error(y_test, y_pred_rf)
rmse_rf = np.sqrt(mean_squared_error(y_test, y_pred_rf))

mae_rf, rmse_rf
```

```
(np.float64(57553030.24853881), np.float64(83934294.57244053))
```

Сравнение метрик качества

Метрика	Метод опорных векторов	Случайный лес
MAE	~45.87 млн	~57.55 млн
RMSE	~77.64 млн	~83.93 млн

Выводы

- Метод опорных векторов показал **лучшие результаты** по обеим метрикам.

- Случайный лес оказался менее точным — возможно, из-за небольшого количества числовых признаков и присутствия выбросов (фильмы-блокбастеры с экстремальными сборами).

- MAE даёт представление о средней ошибке прогноза в долларах, RMSE более чувствителен к крупным отклонениям.

- Для улучшения модели можно добавить больше признаков, таких как длительность фильма, наличие сиквелов, рейтинги и т.п.