**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана (национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Информатика и системы управления»**

**Кафедра «Системы обработки информации и управления»**

**Курс «Технологии машинного обучения»**

Отчёт по лабораторной работе №3

«Подготовка обучающей и тестовой выборки, кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей.»

Выполнил:

студент группы ИУ5-63Б

Рассказов Никита

Проверил:

к.т.н., доц., Ю. Е. Гапанюк

2024 г.

**import** pandas **as** pd

data = pd.read\_csv(**'Walmart\_sales.csv'**) data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Store | Date | Weekly\_Sales | Holiday\_Flag | Temperature | Fuel\_Price | CPI | Unemploy |  |
| 0 | 1 | 05-  02-  2010 | 1643690.90 | 0 | 42.31 | 2.572 | 211.096358 | 8.106 |
| 1 | 1 | 12-  02-  2010 | 1641957.44 | 1 | 38.51 | 2.548 | 211.242170 | 8.106 |
| 2 | 1 | 19-  02-  2010 | 1611968.17 | 0 | 39.93 | 2.514 | 211.289143 | 8.106 |
| 3 | 1 | 26-  02-  2010 | 1409727.59 | 0 | 46.63 | 2.561 | 211.319643 | 8.106 |
| 4 | 1 | 05-  03-  2010 | 1554806.68 | 0 | 46.50 | 2.625 | 211.350143 | 8.106 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 6430 | 45 | 28-  09-  2012 | 713173.95 | 0 | 64.88 | 3.997 | 192.013558 | 8.684 |
| 6431 | 45 | 05-  10-  2012 | 733455.07 | 0 | 64.89 | 3.985 | 192.170412 | 8.667 |
| 6432 | 45 | 12-  10-  2012 | 734464.36 | 0 | 54.47 | 4.000 | 192.327265 | 8.667 |
| 6433 | 45 | 19-  10-  2012 | 718125.53 | 0 | 56.47 | 3.969 | 192.330854 | 8.667 |
| 6434 | 45 | 26-  10-  2012 | 760281.43 | 0 | 58.85 | 3.882 | 192.308899 | 8.667 |

6435 rows × 8 columns

print(data.isnull().sum()) data.drop([**"Date"**], axis=1, inplace=**True**)

Store 0

Date 0

Weekly\_Sales 0

Holiday\_Flag 0

Temperature 0

Fuel\_Price 0

CPI 0

Unemployment 0

dtype: int64

*# разделение на признаки и целевую переменную*

X = data.drop([**"Weekly\_Sales"**], axis=1) y = data[**"Weekly\_Sales"**]

**from** sklearn.model\_selection **import** train\_test\_split

*# разделение на обучающую и тестовую выборки*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0

*# проверка размеров выборок* print(**"X\_train shape:"**, X\_train.shape) print(**"X\_test shape:"**, X\_test.shape) print(**"y\_train shape:"**, y\_train.shape) print(**"y\_test shape:"**, y\_test.shape)

X\_train shape: (5148, 6)

X\_test shape: (1287, 6)

y\_train shape: (5148,)

y\_test shape: (1287,)

**from** sklearn.neighbors **import** KNeighborsRegressor

**from** sklearn.metrics **import** mean\_absolute\_error, mean\_squared\_error,

*# создание экземпляра класса KNeighborsRegressor с K=5*

knn = KNeighborsRegressor(n\_neighbors=5)

*# обучение модели*

knn.fit(X\_train, y\_train)

*# прогнозирование на тестовой выборке*

y\_pred = knn.predict(X\_test)

*# MAE - средняя абсолютная ошибка*

mae = mean\_absolute\_error(y\_test, y\_pred)

*# RMSE - среднеквадратичная ошибка (более чувствителен к наблюдением*

rmse = mean\_squared\_error(y\_test, y\_pred, squared=**False**)

*# MAPE - средняя абсолютная ошибка в процентах*

mape = mean\_absolute\_percentage\_error(y\_test, y\_pred)

print(**"MAE:"**, mae)

print(**"RMSE:"**, rmse)

print(**"MAPE:"**, mape)

MAE: 191597.05669308468

RMSE: 301749.621612451

MAPE: 0.24296767380093

**from** sklearn.model\_selection **import** GridSearchCV, RandomizedSearchCV,

**from** sklearn.metrics **import** mean\_absolute\_error, make\_scorer knn = KNeighborsRegressor()

*# задание сетки гиперпараметров для GridSearchCV*

param\_grid = {**'n\_neighbors'**: [5, 10, 15, 20]}

*# задание сетки гиперпараметров для RandomizedSearchCV*

param\_dist = {**'n\_neighbors'**: [5, 10, 15, 20]}

*# задание стратегий кросс-валидации*

cv\_kfold = KFold(n\_splits=5, shuffle=**True**, random\_state=42) cv\_shuffle = ShuffleSplit(n\_splits=5, test\_size=0.2, random\_state=42)

*# подбор гиперпараметра K с помощью GridSearchCV и оценка качества с* grid\_search = GridSearchCV(knn, param\_grid, scoring=make\_scorer(mean\_ grid\_search.fit(X\_train, y\_train)

print(**"Best parameters (GridSearchCV):"**, grid\_search.best\_params\_) print(**"MAPE (GridSearchCV):"**, grid\_search.best\_score\_)

*# подбор гиперпараметра K с помощью RandomizedSearchCV и оценка качес* random\_search = RandomizedSearchCV(knn, param\_dist, scoring=make\_scor random\_search.fit(X\_train, y\_train)

print(**"Best parameters (RandomizedSearchCV):"**, random\_search.best\_par print(**"MAPE (RandomizedSearchCV):"**, random\_search.best\_score\_)

Best parameters (GridSearchCV): {'n\_neighbors': 20} MAPE (GridSearchCV): 0.40404523695799066

Best parameters (RandomizedSearchCV): {'n\_neighbors': 20} MAPE (RandomizedSearchCV): 0.4079354283992271

/opt/python/envs/default/lib/python3.8/site-packages/sklearn/model\_ warnings.warn(