Московский государственный технический университет им. Н. Э. Баумана

Курс «Технологии машинного обучения» Отчёт по лабораторной работе №3

Дувакин А.В. группа ИУ5-63Б	Гапанюк Ю.Е.
Дата: 14.03.25	Дата:
Подпись:	Подпись:

Проверил:

Москва, 2025 г.

Цель лабораторной работы: изучение способов подготовки выборки и подбора гиперпараметров на примере метода ближайших соседей.

Задание:

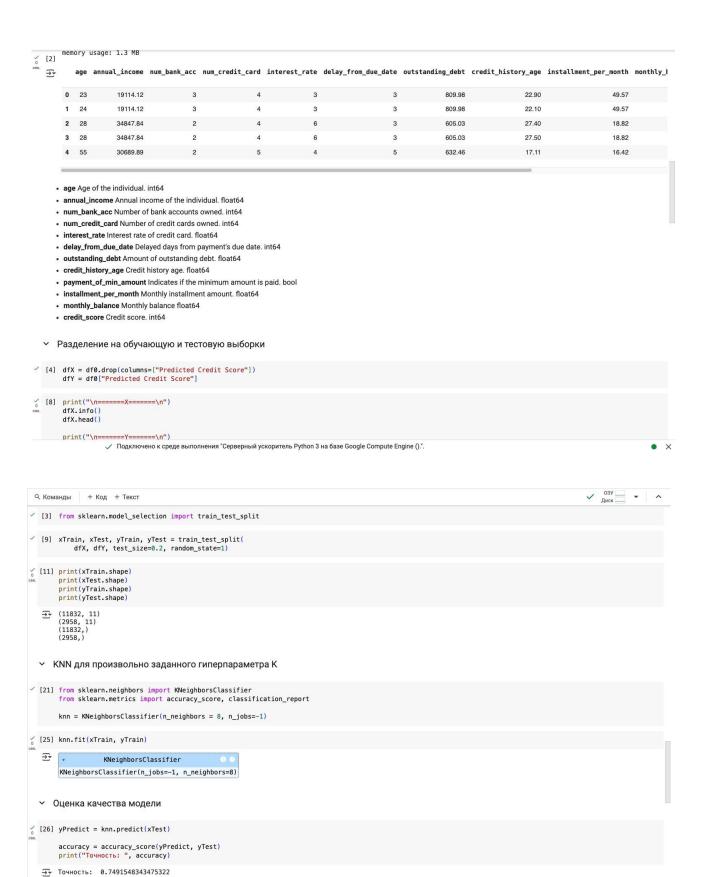
Выполнил:

1. Выберите набор данных (датасет) для решения задачи классификации или регрессии.

- 2. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
- 3. С использованием метода train_test_split разделите выборку на обучающую и тестовую.
- 4. Обучите модель ближайших соседей для произвольно заданного гиперпараметра К. Оцените качество модели с помощью подходящих для задачи метрик.
- 5. Произведите подбор гиперпараметра К с использованием GridSearchCV и RandomizedSearchCV и кросс-валидации, оцените качество оптимальной модели. Используйте не менее двух стратегий кроссвалидации.
- 6. Сравните метрики качества исходной и оптимальной моделей.

Ход выполнения:





```
У Подбор гиперпараметра К через GridSearchCV и RandomizedSearchCV
                                                                                                                                                                        ↑ ↓ ♦ © ■ $ 1 1 1 1
| from sklearn.model_selection import GridSearchCV from sklearn.neighbors import KNeighborsClassifier from sklearn.metrics import make_scorer, accuracy_score
         knn = KNeighborsClassifier()
         param_grid = {'n_neighbors': list(range(3, 8))}
         grid_search = GridSearchCV(knn, param_grid, cv=3, scoring='accuracy', n_jobs=-1)
         grid_search.fit(np.array(dfX), np.array(dfY))
         print(f"Лучшие параметры: {grid_search.best_params_}")
print(f"Лучшая точность: {grid_search.best_score_}")
    [30] from sklearn.model_selection import RandomizedSearchCV from sklearn.neighbors import KNeighborsClassifier from scipy.stats import randint
         knn = KNeighborsClassifier()
         param_dist = {'n_neighbors': randint(3, 8)}
         \label{eq:continuous} random\_search = RandomizedSearchCV(knn, param\_distributions=param\_dist, \\ n\_iter=2, cv=3, scoring='accuracy', \\ n\_jobs=-1, random\_state=42) \\
         random\_search.fit(np.array(dfX), np.array(dfY))
         print(f"Лучшие параметры: {random_search.best_params_}")
print(f"Лучшая точность: {random_search.best_score_}")
```