МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра систем автоматизации управления

Знакомство с библиотекой sklearn

Отчет по лабораторной работе №1
по дисциплине

«Теория информации, данные, знания»

Выполнил студент гр. ИТб-4302-02-00 _____ /Вершинин П.А./
(Подпись)

Руководитель к.т.н., доцент _____ /Чистяков Г.А./
(Подпись)

Работа защищена с оценкой «_____» «___» _____ 2023 г.

Киров 2023

1 Описание лабораторной работы

Тема: «Знакомство с библиотекой sklearn».

Цель: получение студентами навыков работы с классическими методами машинного обучения библиотекой sklearn на языке программирования Python.

В ходе работы необходимо выполнить следующее задание:

- импорт библиотек и загрузка данных;
- подготовка данных;
- выбор модели;
- обучение модели;
- предсказания и создание файла для оценки результата;

2 Задание

Задача бинарной классификации, оценивается по точности классификации (проценту меток, которые правильно предсказывают). Обучающий набор содержит 1000 образцов, а тестовый - 9000. Прогноз должен представлять собой вектор размером 9000 x 1, состоящий из единиц или нулей. Также нужен столбец Id (от 1 до 9000), который должен содержать заголовок. Формат выглядит следующим образом на рисунке 1.

```
Id,Solution
1,0
2,1
3,1
...
9000,0
```

Рисунок 1 – Формат данных в конечном виде

3 Импорт библиотек и загрузка данных

Для начала работы необходимо импортировать следующие библиотеки: Pandas, scikit-learn (sklearn). На рисунке 2 отображен импорт данных библиотек.

«Pandas» используется для работы с данными, включая чтение и обработку файлов CSV.

«MLPClassifier» – это классификатор на основе многослойного персептрона из библиотеки scikit-learn (sklearn), который мы будем использовать для обучения модели.

```
import pandas as pd
from sklearn.neural_network import MLPClassifier as MLPC
```

Рисунок 2 – Импорт библиотек

Для загрузки данных из файлов «train.csv», «trainLabels.csv» и «test.csv», как показано на рисунке 3.

```
train_data = pd.read_csv( filepath_or_buffer: "train.csv", header=None)
train_labels = pd.read_csv( filepath_or_buffer: "trainLabels.csv", header=None)
test_data = pd.read_csv( filepath_or_buffer: "test.csv", header=None)
```

Рисунок 3 – Загрузка данных из файлов

«train_data.csv» содержит обучающие данные, представленные в виде матрицы признаков.

«train_labels» содержит метки классов для обучающих данных.

«test_data» содержит данные, для которых нужно выполнить предсказания.

4 Подготовка данных

При необходимости, вы можете разделить обучающий набор данных на обучающий и валидационный наборы. Это может помочь в оценке производительности модели и настройке параметров модели. Пример разделения данных представлен на рисунке 4.

```
from sklearn.model_selection import train_test_split

# Разделение обучающего набора данных на обучающий и валидационный
X_train, X_val, y_train, y_val = train_test_split( *arrays: train_data, train_labels, test_size=0.2, random_state=42)
```

Рисунок 4 – Разделение обучающего набора данных

Разделением в конечном виде не используем. Обучающий набор данных обладает значительным размером, что обеспечивает модели достаточное количество примеров для обучения. Это позволяет модели рассмотреть больше разнообразных ситуаций и обучиться более надежно.

Нормализация признаков может быть полезной в некоторых случаях для улучшения сходимости алгоритмов машинного обучения. Однако, в некоторых случаях она может не приносить выгоды, как это случилось в данной работе. На рисунке 5 представлены результаты оценки обученных моделей на тестовых данных.

submission (8).csv Complete · 3d ago · without Normalize data	0.88932	0.89605
submission (7).csv Complete · 3d ago · scaler = MinMaxScaler(feature_range=(-3, 3))	0.83961	0.8383
submission (6).csv Complete · 3d ago · mlpc with scaler = MinMaxScaler(feature_range=(0,1))	0.80668	0.81892

Рисунок 5 – Результаты оценок моделей с и без использования нормализации

В «submission (6).csv» используется нормализация данных с помощью метода «MinMaxScaler(feature_range=(0, 1)) » из библиотеки sklearn. «MinMaxScaler» не уменьшает влияние выбросов, но линейно масштабирует их до фиксированного диапазона, где наибольшая встречающаяся точка данных соответствует максимальному значению, а наименьшая – минимальному значению. С данным методом оценка является наименьшей.

В «submission (7).csv» используется нормализация данных с помощью метода MinMaxScaler(feature_range=(-3, 3)) из библиотеки sklearn.

В «submission (8).csv» не используется нормализация данных.

Таким образом, нормализация данных отрицательно влияет на оценку выходных результатов.

5 Выбор модели и обучение

Для решения задачи классификации, мы выбрали использовать классификатор Multi-Layer Perceptron (MLP) из библиотеки scikit-learn. MLP – это многослойный персептрон, представляющий собой тип нейронной сети, который позволяет моделировать сложные нелинейные зависимости в данных.

«MLPClassifier» обучается итеративно, поскольку на каждом временном шаге вычисляются частные производные функции потерь относительно параметров модели для обновления параметров. К функции потерь также может быть добавлен термин регуляризации, который сокращает параметры модели, чтобы предотвратить переобучение.

Данный классификатор имеет наиболее высокую оценку результатов из следующих классификаторов, которые использовались: «GradientBoostingClassifier», «KNeighborsClassifier», «DecisionTreeClassifier», «RandomForestClassifier», «LogisticRegression».

Обучение модели выполняется с использованием обучающего набора данных. Используем следующий код для создания и обучения модели (рис. 6).

```
model = MLPC(random_state=63_716_841)

# .values.ravel() для преобразования меток в одномерный массив
model.fit(train_data, train_labels.values.ravel())
```

Рисунок 6 – Обучение модели

Модель обучается на обучающих данных, где пытается выучить зависимости между признаками и метками классов. Используем метод «values.ravel()» для преобразования меток в одномерный массив.

6 Предсказание и создание файла для оценки результата

После успешного обучения модели на обучающем наборе данных, можно перейти к предсказаниям классов на тестовых данных. В данной работе используем обученную модель «MLPClassifier» для этой цели (рис. 7).

```
# Предсказание классов для тестового набора данных
test_predictions = model.predict(test_data)

submission_df = pd.DataFrame({'Id': range(1, len(test_predictions) + 1), 'Solution': test_predictions})

submission_df.to_csv( path_or_buf: "submission.csv", index=False)
```

Рисунок 6 – Предсказание классов для тестового набора данных и запись в заданный формат

Данный код выполняет предсказания для всех объектов в тестовом наборе данных и сохраняет результаты в переменной «test_predictions». Для оценки результатов и подготовки их к отправке или анализу, создается CSV-файл, который содержит идентификаторы объектов (Id) и предсказанные моделью метки классов

(Solution). В результате выполнения данного кода будет создан файл "submission.csv", который содержит предсказания для каждого объекта в тестовом наборе данных. Этот файл можно использовать для отправки результатов задачи или для дальнейшего анализа. На рисунке 7 представлен результат анализа данных.

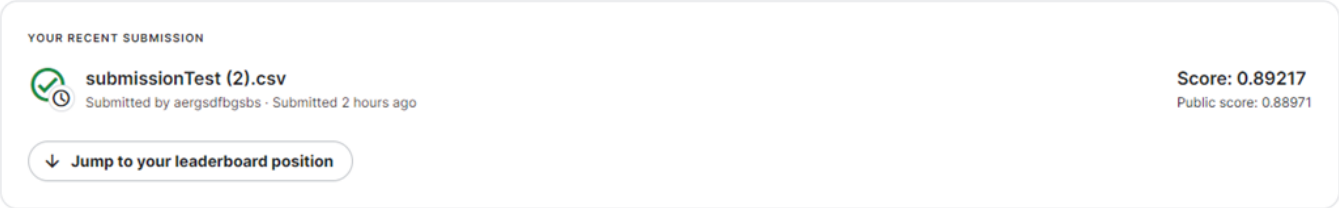

Рисунок 7 – Результат оценки обученной модели без использования нормализации данных

При оценке модели, использовались 2 анализа: публичный и приватный.

Публичный рассчитывается примерно на основе 30% тестовых данных. Окончательные результаты будут основаны на остальных 70%, поэтому итоговый зачет может отличаться.

Приватный рассчитывается примерно на основе 70% тестовых данных.

В итоге получаем оценку в 0.89217 приватный и 0.88971 публичный.

Вывод

В ходе выполнения работы были освоены основы работы с библиотекой scikit-learn для решения задач машинного обучения. Это включает в себя загрузку данных, предварительную обработку, выбор модели, обучение, оценку и предсказания. Важно также учитывать нормализацию данных и подбор параметров модели для достижения лучших результатов. Полученные навыки могут быть полезными для решения разнообразных задач, связанных с анализом данных и машинным обучением.