МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Информационные технологии»

Тема: Введение в анализ данных

Студент гр. 3343 _____ Пименов П.В.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2024

Цель работы

Изучить общие понятиями об анализе данных, библиотеку scikit-learn. Создать программу, анализирующую ассортимент магазина вин.

Задание

Вы работаете в магазине элитных вин и собираетесь провести анализ существующего ассортимента, проверив возможности инструмента классификации данных для выделения различных классов вин. Для этого необходимо использовать библиотеку sklearn и встроенный в него набор данных о вине.

- Загрузка данных

Реализуйте функцию load_data(), принимающей на вход аргумент train_size (размер обучающей выборки, по умолчанию равен 0.8), которая загружает набор данных о вине из библиотеки sklearn в переменную wine. Разбейте данные для обучения и тестирования в соответствии со значением train_size, следующим образом: из данного набора запишите train_size данных из data, взяв при этом только 2 столбца в переменную X_train и train_size данных поля target в y_train. В переменную X_test положите оставшуюся часть данных из data, взяв при этом только 2 столбца, а в y_test — оставшиеся данные поля target, в этом вам поможет функция train_test_split модуля sklearn.model_selection (в качестве состояния рандомизатора функции train_test_split необходимо указать 42.). В качестве результата верните X_train, X_test, y_train, y_test. Пояснение: X_train, X_test - двумерный массив, y_train, y_test. — одномерный массив.

- Обучение модели. Классификация методом k-ближайших соседей

Реализуйте функцию train_model(), принимающую обучающую выборку (два аргумента - X_train и y_train) и аргументы n_neighbors и weights (значения по умолчанию 15 и 'uniform' соответственно), которая создает экземпляр классификатора KNeighborsClassifier и загружает в него данные

X_train, y_train с параметрами n_neighbors и weights.В качестве результата верните экземпляр классификатора.

- Применение модели. Классификация данных

Реализуйте функцию predict(), принимающую обученную модель классификатора и тренировочный набор данных (X_test), которая выполняет классификацию данных из X_test. В качестве результата верните предсказанные данные.

- Оценка качества полученных результатов классификации

Реализуйте функцию estimate(), принимающую результаты классификации и истинные метки тестовых данных (y_test), которая считает отношение предсказанных результатов, совпавших с «правильными» в y_test к общему количеству результатов. (или другими словами, ответить на вопрос «На сколько качественно отработала модель в процентах»). В качестве результата верните полученное отношение, округленное до 0,001. В отчёте приведите объяснение полученных результатов. Пояснение: так как это вероятность, то ответ должен находиться в диапазоне [0, 1].

- Забытая предобработка:

После окончания рабочего дня перед сном вы вспоминаете лекции по предобработке данных и понимаете, что вы её не сделали… Реализуйте функцию scale(), принимающую аргумент, содержащий данные, и аргумент mode - тип скейлера (допустимые значения: 'standard', 'minmax', 'maxabs', для других значений необходимо вернуть None в качестве результата выполнения функции, значение по умолчанию - 'standard'), которая обрабатывает данные соответствующим скейлером. В качестве результата верните полученные после обработки данные.

Выполнение работы

Все требуемые в задании функции успешно реализованы согласно условиям задания. Функция load_data загружает данные о вине, разделяет их

на обучающую и тестовую выборки. Функция train_model выполняет создание классификатора, загружает в него данные. Функция predict на основе обученной модели делает предсказание над тестовыми данными. Функция estimate выполняет оценку точности предсказания результатов моделью. Функция scale выполняет предобработку данных на основе выбранного типа скейлера.

Тестирование программы:

1. Исследование работы классификатора, обученного на данных разного размера

   Код для тестирования:

```
train_size = [?]
X_train, X_test, y_train, y_test = load_data(train_size)
clf = train_model(X_train, y_train)
pred = predict(clf, X_test)
print(estimate(pred, y_test))
```

   Оценка точности классификатора:

train_size	Точность
0.1	0.379
0.3	0.8
0.5	0.843
0.7	0.815
0.9	0.722

Пояснение: По мимо «качества» входных данных на точность предсказания моделью влияет и размер тренировочной выборки: при малом размере выборки данных для обучения недостаточно (отчего точность получается низкой), при сильно больших размерах – происходит переобучение модели, она начинает хорошо работать только с данными из обучающей выборки, неточно классифицируя тестовые данные (отчего точность снижается).

2. Исследование работы классификатора, обученного с различными значениями n_neighbors

   Код для тестирования:

```
n_neighbors = [?]
X_train, X_test, y_train, y_test = load_data()
clf = train_model(X_train, y_train, n_neighbors)
pred = predict(clf, X_test)
print(estimate(pred, y_test))
```

   Оценка точности классификатора:

n_neighbors	Точность
3	0.861
5	0.833
9	0.861
15	0.861
25	0.833

Пояснение: Видно, что точность предсказания во всех случаях относительно близка. Тем не менее, при больших значениях количества соседей точность будет снижаться, поскольку, как правило, более крупное количество соседей подавляет влияние выбросов, но делает границы классификации менее четкими.

3. Исследование работы классификатора с предобработанными данными
   Код для тестирования:

```
mode = [?]
X_train, X_test, y_train, y_test = load_data()
X_train = scale(X_train, mode)
X_test = scale(X_test, mode)
clf = train_model(X_train, y_train)
pred = predict(clf, X_test)
print(estimate(pred, y_test))
```

   Оценка точности классификатора:

Тип скейлера	Точность

standard	0.889
minmax	0.806
maxabs	0.75

Пояснение: Видно, что точность предсказания наибольшая при использовании StandardScaler и наименьшая при использовании MaxAbsScaler. Подобные результаты можно объяснить тем, что MinMaxScaler и MaxAbsScaler очень чувствительны к выбросам.

Разработанный программный код см. в приложении А.

Выводы

Были изучены общие понятия об анализе данных, классификации, обучении моделей, предобработке данных, библиотеке scikit-learn. Создана программа, анализирующая ассортимент магазина вин.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from sklearn.datasets import load_wine
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler, MinMaxScaler,
MaxAbsScaler


def load_data(train_size=0.8):
    wine = load_wine()
    X_train, X_test, y_train, y_test = train_test_split(
                    wine.data, wine.target, train_size=train_size,
random_state=42
    )
    return X_train[:, :2], X_test[:, :2], y_train, y_test


def train_model(X_train, y_train, n_neighbors=15,
weights="uniform"):
              return KNeighborsClassifier(n_neighbors=n_neighbors,
weights=weights).fit(
          X_train, y_train
    )


def predict(clf, X_test):
    return clf.predict(X_test)


def estimate(res, y_test):
    return round(accuracy_score(y_test, res), 3)


def scale(data, mode="standard"):
    scalers = {
        "standard": StandardScaler(),
        "minmax": MinMaxScaler(),
        "maxabs": MaxAbsScaler(),
    }
    if mode not in scalers.keys():
        return None
    return scalers[mode].fit_transform(data)
```