**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

ОТЧЕТ

**по лабораторной работе №3**

**по дисциплине «Информатика»**

Тема: **Введение в анализ данных**

|  |  |  |
| --- | --- | --- |
| **Студент гр. 3344** |  | **Сьомак Д.А.** |
| **Преподаватель** |  | **Иванов** **Д.В.** |

**Санкт-Петербург**

**2023**

**Цель работы**

Получение навыков работы с библиотеками, содержащими базовые инструменты для анализа данных на языке программирования python.

**Задание**

Вариант 1.

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.

1) 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.

2) 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 c параметрами n\_neighbors и weights.

В качестве результата верните экземпляр классификатора.

3) 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.

4) 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].

5) 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**

1. Описание реализации 5и требуемых функций:

load\_data – функция загружает набор данных о вине из библиотеки *sklearn* в переменную wine, разбивает его на обучающую и тестовую выборки. В конце возвращает четыре массива: *X\_train, X\_test, y\_train, y\_test*.

train\_model – создаёт и обучает модель классификатора K-ближайших соседей на обучающей выборке, после этого он возвращается.

predict – использует обученный классификатор и тестовую выборку для прогнозирования меток тестовых данных и возвращает массив предсказанных меток.

estimate – принимает массив предсказанных меток и массив истинных меток, вычисляет точность прогнозов через сравнение предсказанных и заданными метками, возвращает точность.

scale – принимает массив данных, режим масштабирования и возвращает масштабированный массив данных.

2. Исследование работы классификатора, обученного на данных разного размера:

Таблица 1 – Результаты работы классификатора

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Размер набора | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
| Точность | 0.379 | 0.8 | 0.843 | 0.815 | 0.722 |

Видно, что точность классификации зависит от размера выборки. Слишком большая или наоборот маленькая выборка может быть неэффективна для классификации, так как может приводить к переобучению модели и увеличению времени этого самого обучения.

3. Исследование работы классификатора, обученного с различными значениями *n\_neighbors:*

Таблица 2 – результаты работы классификатора

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Количество соседей | 3 | 5 | 9 | 15 | 25 |
| Точность | 0.861 | 0.833 | 0.861 | 0.861 | 0.833 |

Видно, что точность работы классификаторов с разными значениями n\_neighbors почти не различаются. Для данного набора данных наиболее эффективными значениями n\_neighbors является 3, 9, 15, однако разница в точности c другими значениями незначительна.

4. Исследование работы классификатора с предобработанными данными:

Таблица 3 – результаты работы классификатора

|  |  |  |  |
| --- | --- | --- | --- |
| Скейлер | standart | minmax | maxabs |
| Точность | 0.417 | 0.417 | 0.278 |

Видно, что точность классификации для различных способов масштабирования данных различается. Выбор способа масштабирования данных может влиять на точность классификации, таким образом MinMaxScaler и StandardScaler в данном случае обладают лучшими результатами.

Исходный код см. в приложении A

## **Выводы**

Были получены практические навыки работы с библиотеками, содержащими базовые инструменты для анализа данных. Был получен опыт анализа данных при написании программы на язык python.

# **ПРИЛОЖЕНИЕ А** **ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: Somak\_Demid\_lb3.py

from sklearn import datasets

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

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score

from sklearn.preprocessing import StandardScaler, MinMaxScaler, MaxAbsScaler

def load\_data(train\_size=0.8):

wine = datasets.load\_wine()

X,y = wine.data, wine.target

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X[:, :2], y, train\_size=train\_size, random\_state=42)

return X\_train, X\_test, y\_train, y\_test

def train\_model(X\_train, y\_train, n\_neighbors=15, weights='uniform'):

clf = KNeighborsClassifier(n\_neighbors=n\_neighbors, weights=weights).fit(X\_train, y\_train)

return clf

def predict(clf, X\_test):

return clf.predict(X\_test)

def estimate(res, y\_test):

return round(accuracy\_score(y\_true=y\_test, y\_pred=res), 3)

def scale(X, mode='standard'):

if mode not in ['standard', 'minmax', 'maxabs']:

return None

scaler = StandardScaler()

if mode == 'minmax':

scaler = MinMaxScaler()

elif mode == 'maxabs':

scaler = MaxAbsScaler()

scaler = scaler.fit(X)

x\_scaled = scaler.transform(X)

return x\_scaled

def scale(args, mode='standard'):

if mode == 'standard':

scaler = StandardScaler()

elif mode == 'minmax':

scaler = MinMaxScaler()

elif mode == 'maxabs':

scaler = MaxAbsScaler()

else:

return None

x\_scaled = scaler.fit\_transform(args)

return x\_scaled