**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. 2384 |  | Кузьминых Е.М |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

**Цель работы.**

Научиться использовать библиотеку *sklearn*, создать программу, которая будет загружать информацию, на основе ее обучать модель и анализировать полученные данные.

**Задачи.**

Вариант №3

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) В функции *loadData()* загрузить набор данных о вине из библиотеки *sklearn* в переменную *wine*. Разбейте данные для обучения и тестирования в соотношении 80% к 20% соответственно, следующим образом: из данного набора запишите 80% данных из *data*, взяв при этом только 2 столбца в переменную *X\_train* и 80% данных поля *target* в *y\_train*. В переменную *X\_test*- положите оставшиеся 20% данных из *data*, взяв при этом только 2 столбца, а в *y\_tes*t — оставшиеся 20% данных поля *target*. В качестве результата работы функции верните *X\_train*, *y\_train*, *X\_test*, *y\_test*. В качестве состояния рандомизатора необходимо указать 42.

Пояснение: *X\_train*, *X\_test* - двумерный массив, *y\_train*, *y\_test*. — одномерный массив.

2) Обучение модели. Выполнение классификации:

В функции классификации однородных весов *distance()* cоздайте экземпляр классификатора соседей *KNeighborsClassifier* и загрузите в него данные *X\_train*, *y\_train* из пункта выше c *n\_neighbors* = 15 и *weights* = *di*stance.

В качестве результата верните экземпляр классификатора соседей.

3) Выполнение анализа данных. Реализуйте функцию *predict()*. Выполните предсказание данных, используя данные *X\_test* — (те 20% данных из *data*). Верните предсказанные данные.

4) Оцените качество полученных результатов классификации.

В функции *estimate()* сравните *y\_test* с результатом, полученным Вами в пункте 3. Посчитайте и верните отношение предсказанных результатов, совпавших с «правильными» в *y\_test* к количеству результатов. (Или другими словами, ответить на вопрос «На сколько качественно отработала модель в процентах»). Объясните полученные результаты.

Пояснение: так как это вероятность, то ответ должен находиться в диапазоне [0, 1].

**Выполнение работы.**

В начале программы импортируются необходимые библиотеки и классификатор *KNN* из *Scikit-Learn*.

Функция *loadData()*: Эта функция загружает набор данных о вине, используя функцию *load\_wine()*. Затем функция разделяет данные на обучающую и тестовую выборки, используя функцию *train\_test\_split()*. Данные разбиваются на *X\_train, X\_test, y\_train* и *y\_test*. Для этого берутся только первые два столбца из wine.data и поле target. Функция возвращает *X\_train*, *X\_test*, *y\_train* и *y\_test*.

Функция *distance(X\_train, y\_train)*: Эта функция создает объект классификатора *KNN* с параметрами *n\_neighbors=15* и *weights='distance'*. Затем она обучает классификатор на обучающей выборке, используя метод fit(), и возвращает обученный классификатор.

Функция *predict(clf, X\_test)*: Эта функция использует обученный классификатор, чтобы сделать предсказания на тестовой выборке, используя метод predict().

Функция *estimate(y\_pred, y\_test)*: Эта функция использует функцию mean()*,* чтобы вычислить точность классификации. Она сравнивает предсказанные метки *y\_pred* с истинными метками *y\_test* с помощью выражения *(y\_pred==y\_test).mean()* и возвращает точность классификации.

**Тестирование.**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Входные данные | Выходные данные | Комментарий |
| 1 | X\_train, X\_test, y\_train, y\_test = loadData()  clf = distance(X\_train, y\_train)  res = predict(clf, X\_test)  est = estimate(res, y\_test)  print(y\_train) | [2 2 1 2 0 1 1 1 2 0 1 1 2 0 1 0 0 2 2 1 1 0 1 0 2 1 1 2 0 0 0 2 0 0 1 2 1  0 2 1 0 2 1 1 0 1 0 0 1 0 0 2 1 1 1 0 1 1 1 2 2 0 1 2 2 1 1 0 1 2 2 1 2 1  1 1 0 0 2 0 2 0 0 1 1 0 0 0 1 0 1 2 1 1 1 2 2 1 0 0 1 2 2 0 1 2 2 2 2 1 0  1 0 2 0 0 1 0 0 2 1 0 2 2 0 0 2 2 2 1 1 1 1 1 1 2 0 1 1 0 1 1] | Ответ верный |
| 2 | X\_train, X\_test, y\_train, y\_test = loadData()  clf = distance(X\_train, y\_train)  res = predict(clf, X\_test)  est = estimate(res, y\_test)  print(y\_test) | [0 0 2 0 1 0 1 2 1 2 0 2 0 1 0 1 1 1 0 1 0 1 1 2 2 2 1 1 1 0 0 1 2 0 0 0] | Ответ верный |

**Вывод.**

В ходе выполнения лабораторной работы была написана программа на языке программирования *Python* с использованием библиотеки *scikit-learn*. Программа загружает данные о вине, разбивает их на обучающую и тестовую выборки, обучает модель классификации методом k-ближайших соседей, делает предсказания для тестовой выборки и оценивает точность модели.

**Приложение А.**

**Исходный код программы.**

from sklearn.datasets import load\_wine

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

from sklearn.neighbors import KNeighborsClassifier

def loadData():

wine = load\_wine()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

wine.data[:, :2], wine.target, test\_size=0.2, random\_state=42)

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

def distance(X\_train, y\_train):

clf = KNeighborsClassifier(n\_neighbors=15, weights='distance')

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

return clf

def predict(clf, X\_test):

return clf.predict(X\_test)

def estimate(y\_pred, y\_test):

return (y\_pred == y\_test).mean()