Федеральное агентство связи (Россвязь)

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

090301 Информатика и вычислительная техника

№ кода и наименование направления подготовки

Лабораторная работа №2

по дисциплине «Методы машинного обучения»

Выполнил:
студент гр. ИП-513_____/Сахарчук Д.П./

подпись

Проверил:
преподаватель
кафедры ПМиК

_____/Бочкарев Б.В./

ОЦЕНКА, подпись

Новосибирск 2018

Содержание

1. ПОСТАНОВКА ЗАДАЧИ

1.1. ЗАДАНИЕ

К заданию на лабораторную работу прилагается файл, в котором представлен набор данных из 10000 объектов. Каждый объект описывается двумя признаками ($f_j(x) \in R$) и соответствующим ему классом ($y \in \{0,1\}$).

Суть лабораторной работы заключается в написании классификатора на основе метода k ближайших соседей. Данные из файла необходимо разбить на две выборки, обучающую и тестовую, согласно общепринятым правилам разбиения. На основе этих данных необходимо обучить разработанный классификатор и протестировать его на обеих выборках. Разбиение выборки необходимо выполнять программно, случайным образом, при этом, не нарушая информативности обучающей выборки. Кроме того, обучающая выборка должна быть сгенерирована таким образом, чтобы минимизировать разницу между количеством представленных в ней объектов разных классов.

Тип классификатора для реализации — метод парзеновского окна с относительным размером окна, а функция ядра для классификатора — треугольная. Количество соседей (k) подбирается путем исключения объектов по одному (Leave-One-Out).

1.2. ВАРИАНТ

1. Метод парзеновского окна с относительным размером окна .
2. *Q –квартическое* $K(x) = (1 - r^2)^2[r \leq 1]$.
3. Файл № 5.

2. ВЫПОЛНЕНИЕ ЗАДАНИЯ

2.1. ИСХОДНЫЙ КОД

```
import random
import math
import csv

distances = []

def get_distances_train(train_data):
    distances.clear()
    length = len(train_data)
    point_distance = []
    for i in range(length):
        for j in range(length):
            point_distance.append([dist(train_data[i], train_data[j]), train_data[j][2]])
        distances.append(point_distance.copy())
        point_distance.clear()
        distances[i].sort(key=lambda e: e[0])

def get_distances_test(train_data, test_data):
    distances.clear()
    point_distance = []
    test_length = len(test_data)
    train_length = len(train_data)
    for i in range(test_length):
        for j in range(train_length):
            point_distance.append([dist(test_data[i], train_data[j]), train_data[j][2]])
        distances.append(point_distance.copy())
        point_distance.clear()
        distances[i].sort(key=lambda e: e[0])

def dist(a, b):
    return math.sqrt((a[0] - b[0]) * (a[0] - b[0]) + (a[1] - b[1]) * (a[1] - b[1]))

def get_k(train_data):
    minimum = 100000
    answer = 0
    for k in range(3, 50, 2):
        neigh = LOO(train_data, k)
        if neigh < minimum:
            minimum = neigh
            answer = k
    return answer

def LOO(train_data, k):
    summa = 0
    length = len(train_data)
    for i in range(length):
        summa += int(parsen(i, k) != train_data[i][2])
    return summa

def kernel(y):
```

```
    return ((1 - abs(y) * abs(y)) * (1 - abs(y) * abs(y))) * int(abs(y) <= 1)

def parsen(index_u, k):
    maximum = 0
    answer = 0
    for y in range(2):
        neigh_k = distances[index_u][k + 1]
        summa = 0
        for i in range(1, k + 1):
            neigh_i = distances[index_u][i]
            summa += int(neigh_i[1] == y) * kernel(neigh_i[0] / neigh_k[0])
        if summa > maximum:
            maximum = summa
            answer = y
    return answer

def split_on_test_and_train(data, percent_test):
    train_data = []
    test_data = []
    for i in range(len(data)):
        random_number = random.randint(0, len(data) - 1)
        if random.random() < percent_test:
            test_data.append(data[random_number])
            data.remove(data[random_number])
        else:
            train_data.append(data[random_number])
            data.remove(data[random_number])
    return train_data, test_data

def read_file():
    data = []
    csv_file = open('E:/Programs/Python/data5.csv')
    csv_reader = csv.reader(csv_file)
    line_count = 0
    for row in csv_reader:
        if line_count == 0:
            line_count += 1
            continue
        else:
            data.append([int(row[0]), int(row[1]), int(row[2])])
    return data

def count_of_objects(list, object_class):
    count = 0
    for i in range(len(list)):
        if list[i][2] == object_class:
            count += 1
    return count

def main():
    data = read_file()
    count = count_of_objects(data, 0)
    print('Количество объектов класса 0 в исходной выборке:', count, '(',
          (count * 100) / len(data), '% )')
```

```
    count = count_of_objects(data, 1)
    print('Количество объектов класса 1 в исходной выборке:', count, '(',
          (count * 100) / len(data), '% )\n')
    for i in range(10):
        data = read_file()
        print('Разбиение', i + 1, '\n')
        train, test = split_on_test_and_train(data, 0.4)
        count = count_of_objects(train, 0)
        print('Количество объектов класса 0 в обучающей выборке:', count, '(',
              (count * 100) / len(train), '% )')
        count = count_of_objects(train, 1)
        print('Количество объектов класса 1 в обучающей выборке:', count, '(',
              (count * 100) / len(train), '% )')
        count = count_of_objects(test, 0)
        print('Количество объектов класса 0 в тестовой выборке:', count, '(',
              (count * 100) / len(test), '% )')
        count = count_of_objects(test, 1)
        print('Количество объектов класса 1 в тестовой выборке:', count, '(',
              (count * 100) / len(test), '% )')
        get_distances_train(train)
        k = get_k(train)
        print('\nКоличество соседей k =', k)
        get_distances_test(train, test)
        mistakes = 0
        for val in test:
            answer = parsen(test.index(val), k)
            mistakes += int(answer != val[2])
        print('Количество ошибок на тестовой выборке:', mistakes, '(', (mistakes * 100) / len(test), '%
)\n')

if __name__ == "__main__":
    main()
```

2.2 РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ

Ниже представлен вывод программы на 10 разбиениях:

Количество объектов класса 0 в исходной выборке: 1468 (14.68 %)
Количество объектов класса 1 в исходной выборке: 8532 (85.32 %)

Разбиение 1

Количество объектов класса 0 в обучающей выборке: 871 (14.663299663299663 %)
Количество объектов класса 1 в обучающей выборке: 5069 (85.33670033670033 %)
Количество объектов класса 0 в тестовой выборке: 597 (14.704433497536947 %)
Количество объектов класса 1 в тестовой выборке: 3463 (85.29556650246306 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 286 (7.044334975369458 %)

Разбиение 2

Количество объектов класса 0 в обучающей выборке: 911 (15.218843969261611 %)
Количество объектов класса 1 в обучающей выборке: 5075 (84.78115603073839 %)
Количество объектов класса 0 в тестовой выборке: 557 (13.876432486297958 %)
Количество объектов класса 1 в тестовой выборке: 3457 (86.12356751370204 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 264 (6.576980568011958 %)

Разбиение 3

Количество объектов класса 0 в обучающей выборке: 879 (14.574697396783286 %)
Количество объектов класса 1 в обучающей выборке: 5152 (85.4253026032167 %)
Количество объектов класса 0 в тестовой выборке: 589 (14.840010078105315 %)
Количество объектов класса 1 в тестовой выборке: 3380 (85.15998992189468 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 253 (6.37440161249685 %)

Разбиение 4

Количество объектов класса 0 в обучающей выборке: 905 (15.266531713900134

%)
Количество объектов класса 1 в обучающей выборке: 5023 (84.73346828609986 %)
Количество объектов класса 0 в тестовой выборке: 563 (13.826129666011788 %)
Количество объектов класса 1 в тестовой выборке: 3509 (86.17387033398822 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 238 (5.844793713163065 %)

Разбиение 5

Количество объектов класса 0 в обучающей выборке: 866 (14.371058745436443 %)
Количество объектов класса 1 в обучающей выборке: 5160 (85.62894125456356 %)
Количество объектов класса 0 в тестовой выборке: 602 (15.148465022647207 %)
Количество объектов класса 1 в тестовой выборке: 3372 (84.8515349773528 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 285 (7.171615500754907 %)

Разбиение 6

Количество объектов класса 0 в обучающей выборке: 902 (15.055917209147054 %)
Количество объектов класса 1 в обучающей выборке: 5089 (84.94408279085295 %)
Количество объектов класса 0 в тестовой выборке: 566 (14.11823397355949 %)
Количество объектов класса 1 в тестовой выборке: 3443 (85.8817660264405 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 257 (6.41057620354203 %)

Разбиение 7

Количество объектов класса 0 в обучающей выборке: 858 (14.174789360647614 %)
Количество объектов класса 1 в обучающей выборке: 5195 (85.82521063935239 %)
Количество объектов класса 0 в тестовой выборке: 610 (15.454775779072714 %)
Количество объектов класса 1 в тестовой выборке: 3337 (84.54522422092728 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 252 (6.384595895616925 %)

Разбиение 8

Количество объектов класса 0 в обучающей выборке: 880 (14.765100671140939 %)
Количество объектов класса 1 в обучающей выборке: 5080 (85.23489932885906 %)
Количество объектов класса 0 в тестовой выборке: 588 (14.554455445544555 %)
Количество объектов класса 1 в тестовой выборке: 3452 (85.44554455445545 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 270 (6.683168316831683 %)

Разбиение 9

Количество объектов класса 0 в обучающей выборке: 882 (14.773869346733669 %)
Количество объектов класса 1 в обучающей выборке: 5088 (85.22613065326634 %)
Количество объектов класса 0 в тестовой выборке: 586 (14.540942928039703 %)
Количество объектов класса 1 в тестовой выборке: 3444 (85.4590570719603 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 254 (6.302729528535981 %)

Разбиение 10

Количество объектов класса 0 в обучающей выборке: 901 (14.981709344861988 %)
Количество объектов класса 1 в обучающей выборке: 5113 (85.01829065513802 %)
Количество объектов класса 0 в тестовой выборке: 567 (14.224786753637732 %)
Количество объектов класса 1 в тестовой выборке: 3419 (85.77521324636227 %)

Количество соседей k = 3
Количество ошибок на тестовой выборке: 263 (6.5980933266432515 %)