Задача 2 (Классификация в непрерывном пространстве)

Стельмах Татьяна

Январь 2021

Содержание

1 Введение

Задача: Необходимо для заданного местоположения квартиры человека найти ближайшую школу (не перебором) и расстояние до неё за $O(\log n)$, где n - кол-во школ, при этом двигаться можно лишь по дорогам на плоскости (дороги в данном случае - перпендикулярная сетка на графике, Рисунок 1);

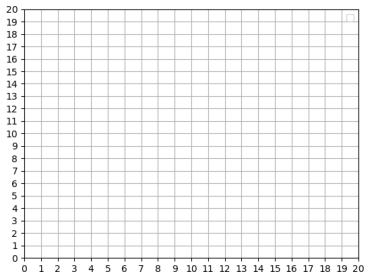

Рисунок 1. Линии на графике - дороги

Выходы на улицу находятся в правом верхнем углу каждого квадрата; квартира находится в центре квадрата; количество школ подаётся на вход программе, поле можно расширять (увеличить значения на осях графика). Необходимо визуализировать ситуацию и полученные результаты, сделать краткие выводы.

2 Описание метода

При решении задачи мы будем пользоваться библиотекой `Sklearn.neighbors`, а именно `KDTree` для нахождения ближайшей школы за $O(\log n)$, используя в параметрах метрику *Манхетеннского расстояния* (как раз для того, чтобы двигаться по дорогам).

3 Описание данных

Местоположение школ и квартиры мы генерируем рандомно в теле программы, а количество школ мы получаем с клавиатуры. В итоге на вход имеем 2 массива: один состоит из координат местоположений школ, а второй из координат местоположения квартиры человека.

4 Описание постановки эксперимента

Необходимые библиотеки:

```
import matplotlib.pyplot as plt
import numpy as np
from random import randint
from sklearn.neighbors import KDTree
from math import sqrt
```

Генерация данных и подготовка их для модели:

```
men = [randint(1,18) + 0.5, randint(1,18) + 0.5] # местоположение человека
n_schools = int(input())
schools = []
for i in range(n_schools):
    schools.append([randint(1, 18) + 0.5, randint(1, 18) + 0.5])
free_men = [[men[0] + 0.5, men[1] + 0.5]]   # пришли к выходу
exit_schools = []
for i in range(n_schools):
    exit_schools.append([schools[i][0] + 0.5, schools[i][1] + 0.5]) # местоположение выходов
exit_schools = np.array(exit_schools)
free_men = np.array(free_men)
```

Создание модели и её использование для получения ответа:

```
kdt = KDTree(exit_schools, leaf_size=30, metric='manhattan')
res = kdt.query(free_men, k=1, return_distance=False)       # индекс нужной школы
exit_schools = exit_schools.tolist()
free_men = free_men.tolist()
dist = abs(exit_schools[res[0][0]][0] - free_men[0][0]) + abs(exit_schools[res[0][0]][1] - free_men[0][1]) + sqrt(2)
# найденное расстояние
```

5 Результаты

В итоге наша программа корректно определяет расстояние и нужную школу в зависимости от количества школ:

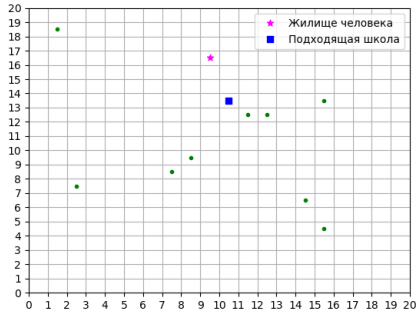

Рисунок 2. 10 школ

Рисунок 3. 50 школ

Рисунок 4. 100 школ

6 Краткие выводы

Была написана программа, которая довольно-таки быстро (благодаря использованию библиотеки) отвечает на вопрос о местонахождении ближайшей школы к квартире с некоторыми допущениями, которые были указаны выше.