Introduction to CS for Data Science

HW5 – Final Assignment 2020

Open Questions (35 points)

The first question can be answered in a word or pdf file (unimodal.pdf). The second question can be submitted as a python file (midval.py). Submit these files zipped with the other files of your submission.

1. UniModal List (20 points)

   A unimodal list is a list where the first part of the list is in ascending order and the second part of the list is in descending order. These parts are not necessarily equal-sized.

   Examples: [5, 9, 20, 90, 70, 18, 6, 4, 3, 2, 1]

   [10, 20, 25, 91, 18]

   1. Write an **efficient** algorithm for finding the min value in a unimodal list and argue its running time. (10 points)
   2. Write an efficient algorithm for sorting a unimodal list and argue its running time. (10 points)

2. Linked List (15 points)

```
class LinkedList:
   def __init__(self):
       self.head = None
       self.tail = None
   def get_head(self):
       return self.head
   def get_tail(self):
       return self.tail
   def insert_tail(self):
       pass
   def mid_val(self):
       pass
       # Implement this method



class Node:
   def __init__(self, val=None):
       self._val=val
       self._next=None
       self._prev=None
   def get_data(self):
       return self._val
   def get_next(self):
       return self._next
   def set_next(self, node):
       self._next = node
   def get_prev(self):
       return self._prev
   def set_prev(self, node):
       self._prev = node
```

In class, we saw the linked list data structure. Above is the code for a doubly linked list where every element has a pointer to the next element

as well as to the previous element.
Certain irrelevant implementation details of the LinkedList class were omitted (the *insert_tail* method).
Notice this linked list has a pointer to the first element in the list (head), and a pointer to the last element in the list (tail).

1. Implement the *mid_val* method which returns the middle value of the list if the list has an odd length, and the 2 middle values if the list is even. Your implementation should traverse the list once (if you traverse the list twice you will lose 10 points) (15 points)
   Examples: The list 1 <-> 5 <-> 4 <-> 7 <-> 10
   returns 4
   The list 1 <-> 5 <-> 4 <-> 7 <-> 10 <-> 3
   returns (4, 7)

Vector Class - (35 points)

You are not allowed to import any libraries in this section.

Implement a Vector class in the supplied **vector.py** file.
The Vector class represents an n-dimensional vector.

Each clause below is worth 5 points except the first 2 which are worth 5 points together.

This class should have the following capabilities:

1. You can initialize a vector object with the call:
   v = Vector(list)
   | Examples : *v =Vector([1, 2])*
   v =Vector([5, 2, 9, 11])
   v =Vector([100, 5, 99, 1, 1000, 234])

2. You can query the vector dimension with the call:
   len(v)
   | Examples: *v =Vector([1, 2]), len(v) = 2*
   v =Vector([5, 2, 9, 11]), len(v) = 4

3. When you print the Vector object it prints 'Vector: [coordinates]'.
   Examples: *v =Vector([1, 2]), print(v) -> Vector: [1, 2]*
   v =Vector([5, 2, 9, 11]), print(v) -> Vector: [5, 2, 9, 11]

4. You can access the vector coordinates using the syntax:

   v[i]

   Examples: *v =Vector([1, 2]), v[0] = 1, v[1] = 2*

   v =Vector([5, 2, 9, 11]), v[0] = 5, v[1] = 2, v[2] = 9, v[3] = 11

5. You can add 2 vector objects using the + operator.

   v1 + v2

   a. The operation should only work if v1 and v2 have the same dimension, otherwise return None.
   b. The operation should not affect v1 nor v2, meaning their coordinates should remain the same.
   c. The operation should return a new vector, v, where v[i] = v1[i] + v2[i]

      Examples of *v1 + v2* :

      v1 =Vector([1, 2]), v2 =Vector ([5, 9])

      v1+v2 = Vector([6, 11])

      v2 =Vector([5, 2, 9]), v2 = Vector([6, 2, 8])

      v1+v2 = Vector([11, 4, 17])

6. You can get the dot product of 2 vectors using the * operator.

   a. The operation should only work if v1 and v2 have the same dimension, otherwise return None.
   b. The operation should not affect v1 nor v2, meaning their coordinates should remain the same.

      Examples of *v1 * v2* :

      v1 =Vector([1, 2]), v2 = Vector([5, 9])

      *v1*v2 = 23*

      v2 =Vector([5, 2, 9]), v2 = Vector([6, 2, 8])

      *v1*v2 = 106*

7. You can get the norm of the vector using the call:

   abs(v)

   Examples: *v =Vector([1, 2]), abs(v) = 2.23606*
   v =Vector([5, 2, 9, 11]), abs(v) = 15.19868

8. You can check if two vectors are equal using the call:

   v1 == v2

   The operation should be evaluated to True is the vectors have the same length and the same value in each coordinate, otherwise it is False.

   Examples: *v1 =Vector([1, 2]), v2 = Vector([3, 4]) v1 == v2 -> False*
   v1 =Vector([3, 4]), v2 = Vector([3, 4]) v1 == v2 -> True
   v =Vector([5, 2, 9, 11]), Vector([5, 2]) v1 == v2 -> False

Streamlit App- (30 points)

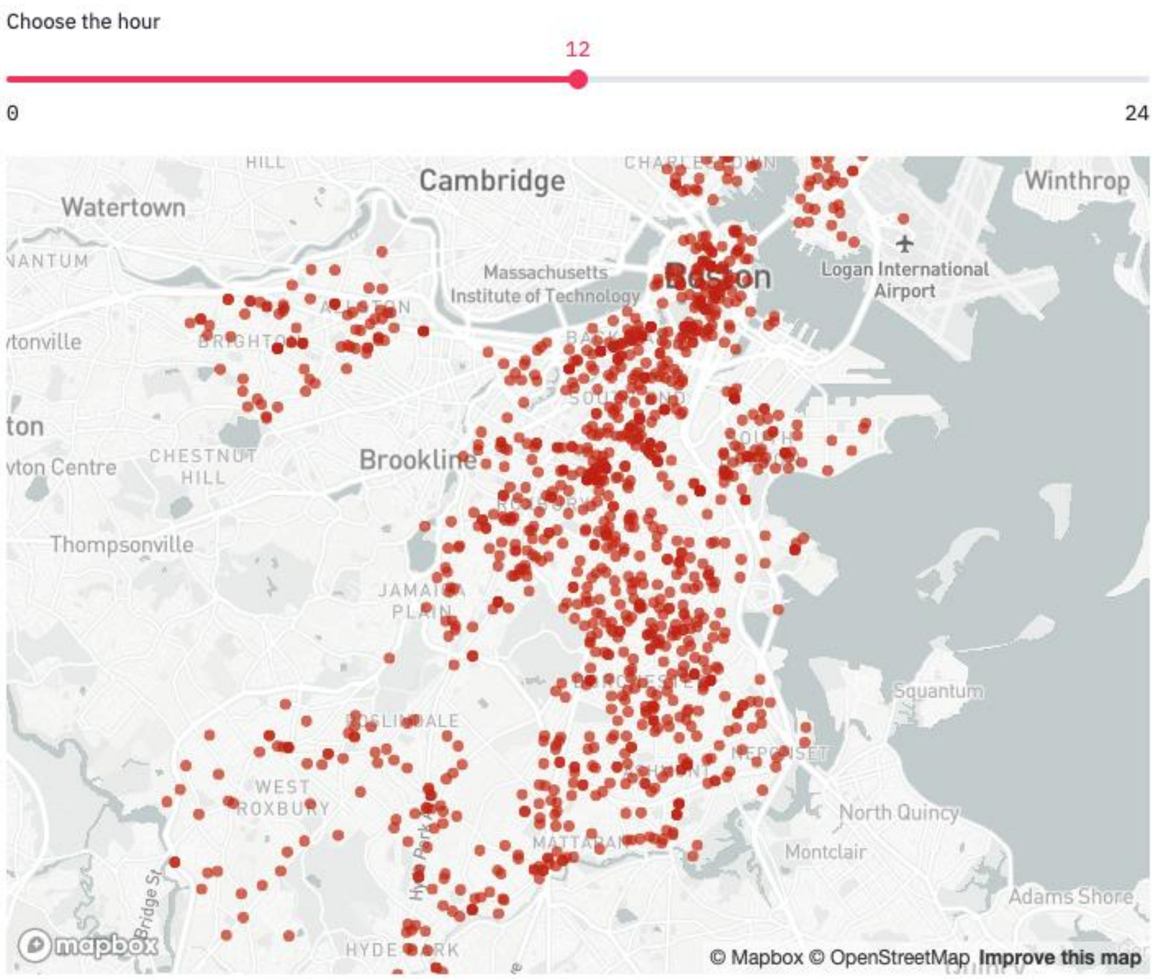
Each clause is worth 10 points

Continuing our dark theme, download the **crimes in Boston** dataset from here.

You already have a function that processes the data in a way that will make it easier to work with. You may use it ☺

Implement the following behavior in the supplied app.py file:

1. Give the option to choose an hour and then plot on a map all the locations where crimes were committed in the city:



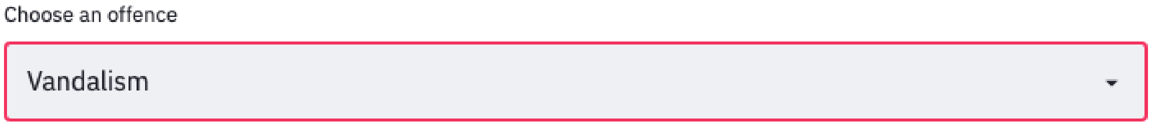
Choose the hour

12

0 24

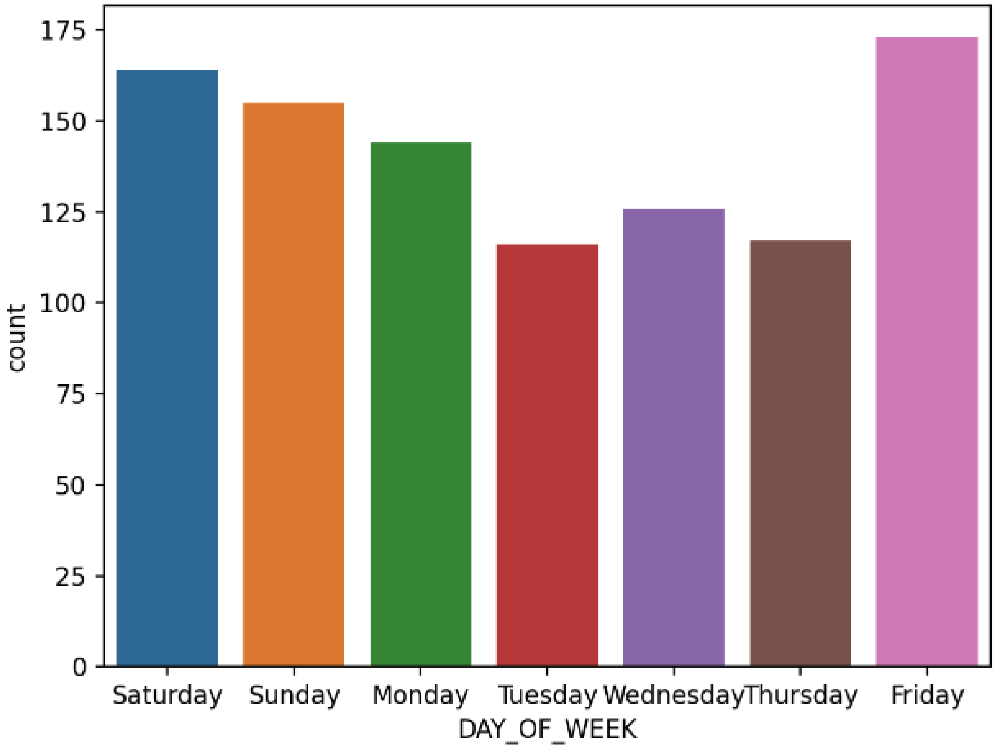
2. Give the option to choose from the top 10 offenses (The offenses that happen most often, each row counts as one offense) and then count the

number of times this offense happened per day of the week.

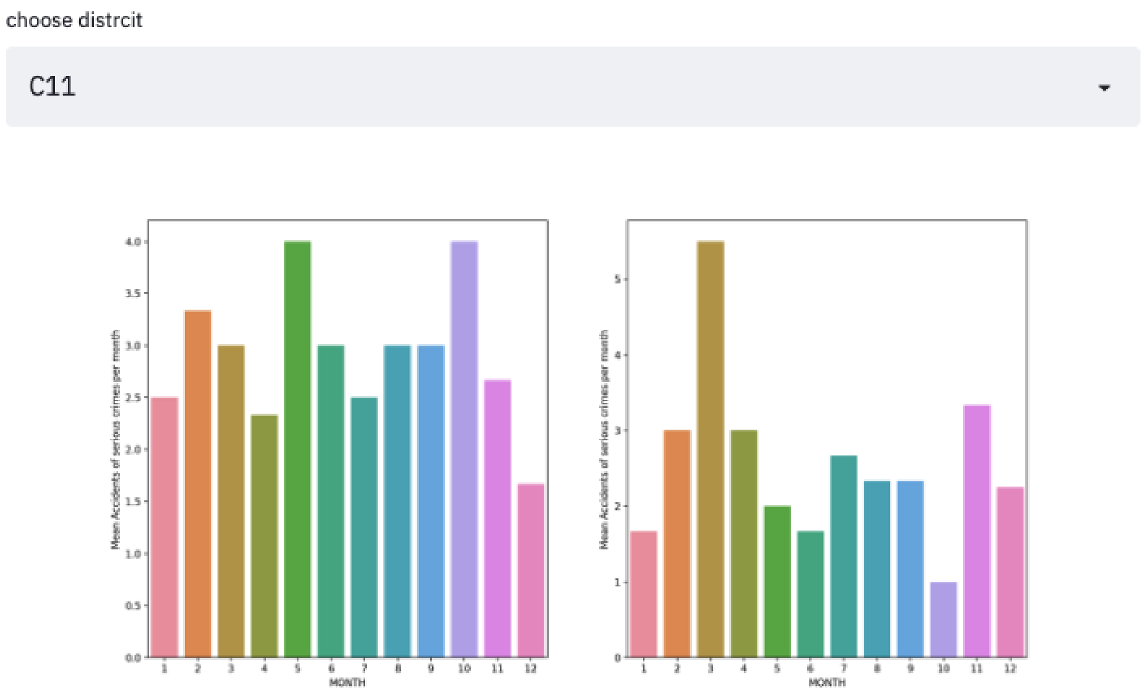


Choose an offence

Vandalism



3. Give the option to choose one of the possible districts in the dataset and then plot 2 bar plots (one for 2016 and one for 2017) of the average amount of serious crime in each month in that district.
    a. Serious crimes are [Larceny, Robbery, Drug Violation, Auto Theft]
    b. Example: If in a district in January you had 10 Larceny incidents, 15 robberies, 20 drug violation and 3 auto theft, then your average serious crime rate is 12.



choose distrcit

C11

Mean Accidents of serious crimes per month
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
1
2
3
4
5
6
7
8
9
10
11
12
MONTH
Mean Accidents of serious crimes per month
0
1
2
3
4
5
1
2
3
4
5
6
7
8
9
10
11
12
MONTH

Submission is individual. Please ask questions in piazza (you may use the private / anonymous modes). Zip your files and name the zipped file your ID number (123456789.zip), and submit to Moodle.

Good luck.