SHENKAR
ENGINEERING. DESIGN. ART.

Final Project
Scientific computing & Algorithms in Python

Project report

Lecturer: Or Perets
Author: Tal Levi – 308516285

1. Abstract.

The goal of this project was to implement the different subjects that were learned in the course and independently, these subjects vary from Data Structures and Algorithms to Data Analysis, Machine Learning and Graphical User Interface.

In this project there was use in data science libraries such as: numpy, matplotlib, pandas, sklearn and tkinter.

The first part of the project consist implementation of two Data Structures:

- Single way Linked list that represents a Stack.
- Hash table with single way linked list.

In The second part of the project it was required to preform Data Analysis over two large sets of data.

The first data set included data of apps from GooglePlay store, and the second data set included user reviews regarding the apps from the first data set.

There are a lot of insights that can be drown out the data, a few examples are:

- apps that were classified as “Paid”, got more positive reviews than apps that were classified as “Free”.
- apps that got the highest ratings were: “comics- creativity”, and “bord- pretend play”.

More of the insights will be shown in the next sections of the report.

2. Methods.

2.1 Data Structures and Algorithms.

The first section of this project is composed of two parts:

a. Representing a generic Stack that keeps its values sorted, the values that Stack was intended to contain were student details. The stack needed to sort from top to bottom each value by the student average.

The Stack was implemented as a single way linked list, out of the idea that there is a need to keep time and memory complexity balanced. Linked list and a stack as the same time complexity, but a linked list at a better memory complexity than a stack.

In order to achieve that, it was required to construct three different classes:

- Student Class: the idea behind this class was, that every student as the same attributes, therefore every student is an object within the class. When creating a new student within the class the are some static methods that are used to validate the student details.
- Linked List Class: this class is a typical single way linked list class, the method that was add to it is "Sorted_Node", this method was added to sort the Stack according to student average parameter.
- Sorted Stack Class: this class contains all the basic methods of a stack, but every method inside this class refers to other linked list methods or attributes.

b. Implementing a graph class that represent undirected and weighted Graph, in order to represent a Graph while keeping time and memory complexity balanced, the chosen data structure was an hash table with a single way linked list, in other wards the graph is represented as an Adjacency List.

An undirected and weighted graph has a set of vertices and a set of edges, each edge includes the source vertex, destination vertex and weight.

In order to implement an adjacency list, it was required to use a linked list. Every node within the linked list represents an edge and contains a list that has a vertex and weight.

The Graph class constructor is a parameterized constructor, the parameter is dictionary and during initialization the contractor iterate the dictionary keys and values, the keys are inserted to the hash table as keys and they represent a vertex, the values represents the vertex edge and weight and its implemented as linked list. There are two methods that needs some more explanation:

- Add Vertex: in order to add a vertex to graph the parameter needs to be of a dictionary type.
- BFS: this method finds all of the possible paths, for each path it calculating the sum of weights, therefore the function returns a list that contains all the paths, each path has the source and end vertex and the sum of weights that represents the distance between the two vertices.

In order to save a graph object there is method that serializes the object from an adjacency list to a dictionary, the reason for that it's because the best way to save a graph is via a JSON file, that is represented a dictionary. In order to load a graph object there is a need to deserialized a JSON file and transformed it to a dictionary so that the class constructor could make it into a graph object.

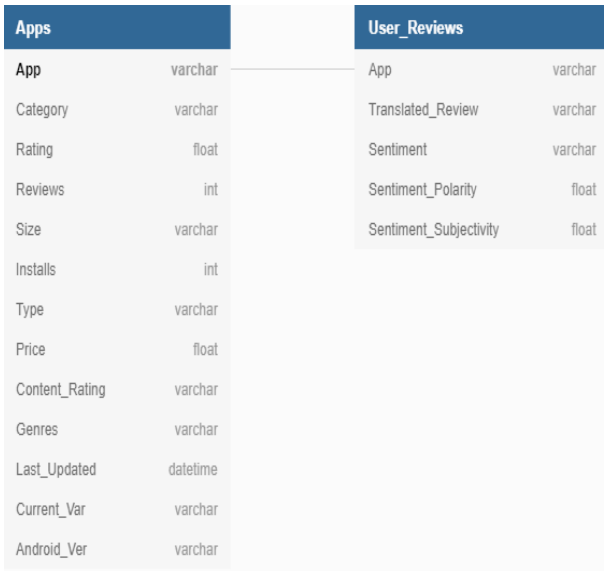
2.2 Data Analysis and Machine Learning.

As before this section was also composed out of two parts:

a. Analyzing and visualizing the data sets:

The data sets that were given in this section are: "Apps" and "User Reviews", the apps data set consist data of apps in GooglePlay store, and the user reviews data set refers to the apps within the first data set. When first approaching to analyzing the data there is a great importance of understanding the scale of the data sets and the types that the data is consist of. in order to load the data, manipulate and analyzed it, there was use in panda's library. Some alteration was made on two columns types in apps data set: "Installs" and "Last Updated" were changed from data frame object type to an integer and datetime accordingly. After understanding the basics of each data set, it was needed to understand to connection between the two.

In order to visualize each data set, it's types and the connection between the two data sets, an ERD was created, as shown.



The "App" in apps table is the primary key, for its data set, "App" in user reviews is a foreign key that links the two tables. User reviews table as no primary key, the reason for that is because primary key can't contain null values and it needs to be injective, because none of the user reviews attributes meets the requirements that described above that data set has no primary key.

After understating the data sets that needed to be analyzed, the next step was use panda's methods to make queries that will allow to extract insights out of the data. Using panda's methods of query and matplotlib methods had enabled to visualize the data and therefore preform visual analysis on the data. Some functions were created in order to analyze the data in a more efficient way.

Note: the functions that were created are specific to the given data sets and are based on panda's library.

b. Building K-Nearest-Neighbors Classifier:

In this part of the project it was required to implement Knn Machin Learning algorithm, in order the implement the algorithm the first step was to preprocess the data sets, the preprocess phase included the following steps:

i. Copy the relevant attributes for each data set, changing some of the attribute's types from data frame object to numeric types, merging the two data sets and finally split the data to "Data" that is symbolized as "X", and to labels that is symbolized as "Y".

ii. Completing missing numeric values in X with "Sklearn" simple imputer method (the missing values were completed by using mean strategy). And completing missing labels in "Y" also with the same method (the strategy here was to use "most frequent" values).

iii. Converting categorical values into indicators for "X" and "Y" by using panda's 'get dummies' method.

iv. The final step in the preparation phase was to split "X" and "Y" into two groups, each one of them was split into train and test group, after splitting to groups there were four group train and test to "X" and the same for "Y". in order to split the data into groups, there was use in " Sklearn" train test split method, and the test size was calibrated to a third of the data.

After preprocessing the data, the X test and train groups were scaled in order to standardize the data and arrange it in standard normal distribution. After scaling the data, it was needed to train the model by using "Sklearn" KNeighborsClassifier method. finally predict the X test group values in order to calculate the accuracy over the test group and find the best model.

In order to use Knn algorithm and find the best K values, a function was created that to iterate a range of K's and for each K to calculate is accuracy score.

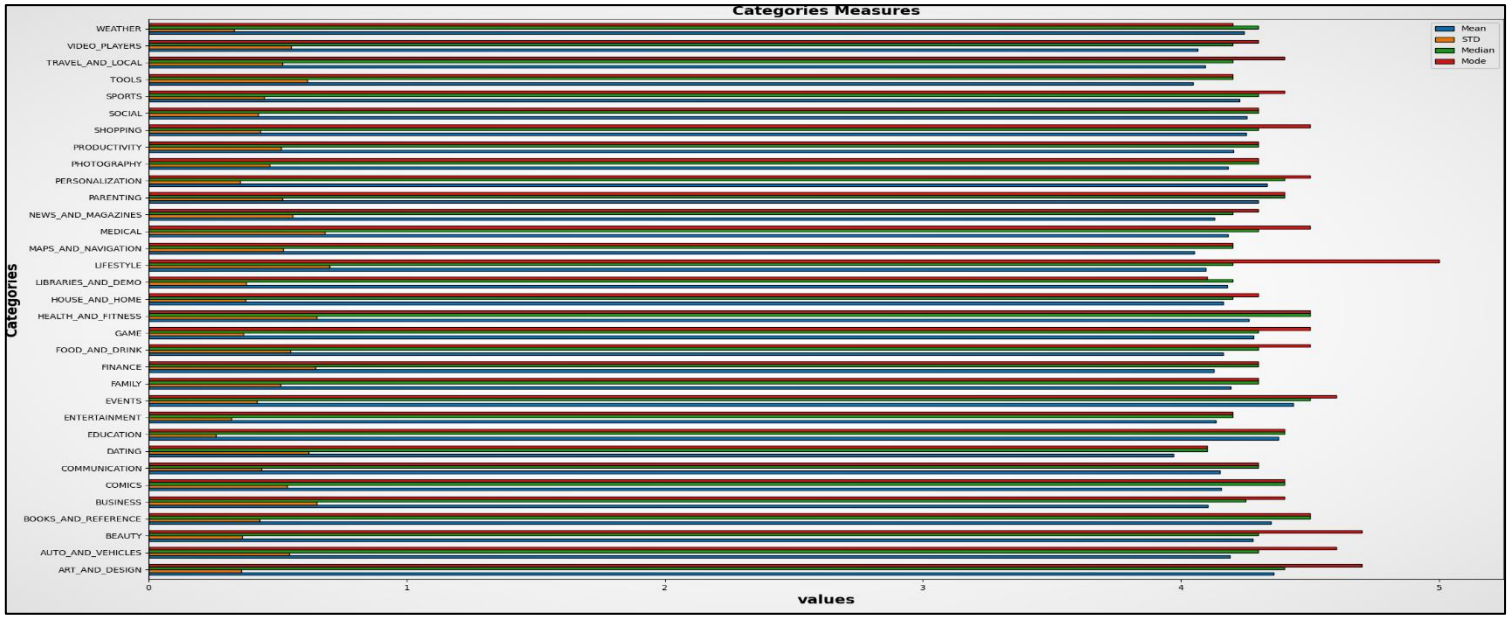
The reason for not wanting to use K that is equal to the length of that data is because Knn algorithm assumes that similar things exist in proximity.

Note: in the different functions that were created for finding the best K accuracy score, the K and It accuracy score were save into a dictionary in each iteration and finally were save into a JSON file, the reason for that is because the Knn algorithm was running on a local computer, and because of that it takes a tremendous amount of time to iterate a wide range of K's. so, for using the results of the Knn algorithm without running it every single time that results were saved into an external file.

3. Conclusions.

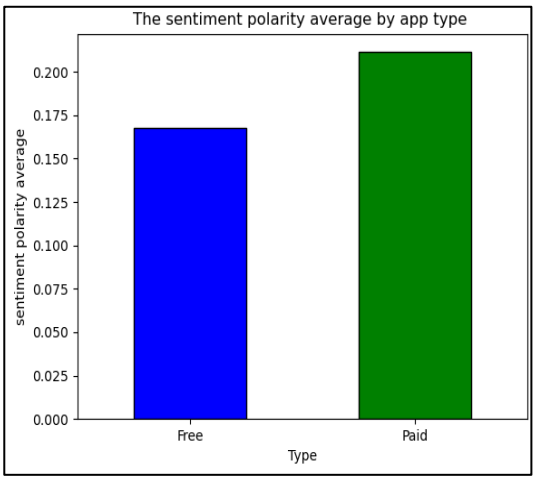
The conclusions that can be derived from analyzing the data in the second part of the project are as following:

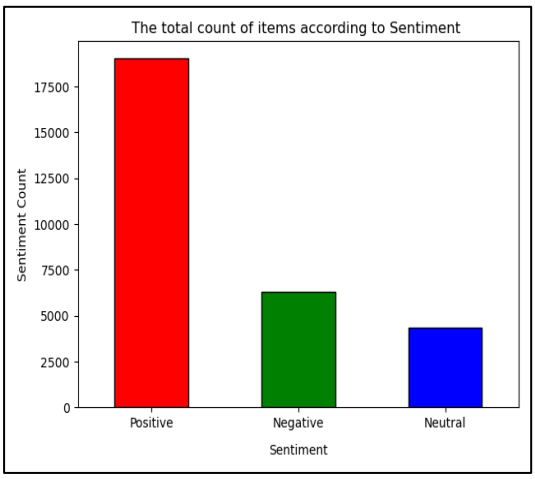
- After calculating for each app category different statistical measures according to its rating, the most stable category is the one that has the lowest value of std. The reason for that is because a low standard deviation indicates that the values tend to be close to the mean. Therefore, the most stable app category is: **Education.**
- the category that is values don't distribute normally is**:** **lifestyle**, the reason for that is because one of the attributes of normal distribution is that the mode needs to equal to the mean and the median, even



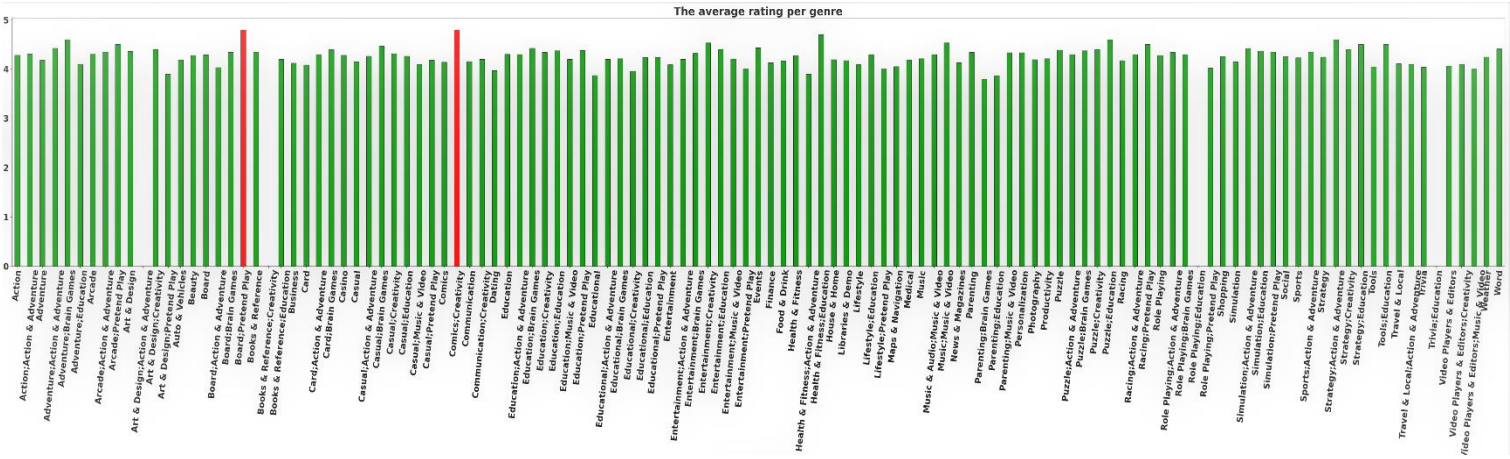
when taking into to consideration an error range of: $0 < \varepsilon < 0.5$ lifestyle category doesn't meet the required condition. More than that as shown in the bar diagram below its easy to see that the standard deviation of lifestyle category is quiet high which indicates that most of its values are scattered largely.

- Apps that the user needed to pay in order to use them got a larger amount of positive reviews, the explanation for that is that sentiment polarity varies in range of: $-1 < \text{polarity} < 1$.
  The average sentiment polarity for paid apps is higher than average of free apps as shown in the bar diagram. The difference between the two is: 0.008.
- Most user reviews are mostly positive, but the difference between negative and neutral sentiment are quite close to one another.





- The genres that got the highest average rating are comics-creativity and bored- pretend play, the reason for that might because this genre allows the users to 'escape' out of their daily routine.



- In order to see the results of all of K's and their accuracy's, run the **"Graphical user interface" script**.