Final Project

Tutors

Omri Ashkenazi

Gal Zohar

Abstract

In our final project we focused on three subjects:

1.Data structure and Algorithms

2.Data Analysis

3.Machine Learning

Subject 1 was divided to two parts, in the first one we build a data structure for students data using Stack. The students data structure supports all function needed for stack data structure.

In the second part we build class that can convert json file to weighted and undirected graph. The graph is using linked-list for each vertex.

Subject 2, in this part we needed to analyze two data bases that are connected by the App name. To analyze properly the data with minimal errors we used another py file called "Data_prepare" where the data is being edited to a proper way. After fixing the data we answer each Queue mostly using Pandas and Numpy libraries.

Subject 3, after fixing data and merging the data bases using only the columns needed, we implemented the KNN algorithm and checked the best K and accuracy.

Method

Subject 1

Sorted Stack

First we created a class for students details . The class supports average and get details functions.

The second class is for the sorted stack , it has pop, top, Is_empty , size and push functions. The push function is using outer functions that validate the student data entered the class. The sort od the stack by average is implanted in the end of the push function using lambda.

Weighted Graph

First we created Node and Linked List classes to be used in graph class. Node class has the values : data- vertex data, weight- the weight between vertexes, next- the next vertex it point on. Linked List class has the functions: *add_new_head add a new node to a vertex linked list .

*linked_list_to_list that transform only the vertexes.

*linked_list_to_list_with_weight that transform a linked list to list including the vertex weight.

*delete_node- the function delete the vertex linked list and the vertex from the other vertexes linked lists.

Class graph- we created a class graph that support the json data structure and creates a linkeslist to each vertex. Then we wrote the functions:

*get_vertices that returns all the vertexes exsit in the class.

*get_edges- returns a list of all edges and their weights.

*add_vertex – this function adds a new vertex and its edges,wegith to the class , the vertex is added to all linkes lists of its edges and open a new linked list for imself. The conction list of dicts that our code supports is : [{'tel-aviv":5.344}, {"Haifa":6.44}].

*deledte_vertex – this function delets vertex from all the class his self linkedlist and its edges linked list.

*bfs – gets a source and goal and returns a list of all the paths between both.

*shortest_path- the function takes all paths from bfs function and cheaks the weights between all vertexes in the path. The function will return the path with the lowest weight from all possible function.

*load_data- the function get a json file , load it and use deserialize to convert it .

*save_data-- the function gets a graph and file name and save the graph to a json with file name mentioned using serialize function.

We also used function out of the classes: *list of edges- get a list of connection and return list of edges,*list of dict to list of list ,*find weight.

Subject 2

Data Prepare

The purpose of the following py file is to edit the data bases so we could analyze it properly and with compatible to the data.

*change_coll_only_num- the function get column and return it as int and not str for example-(10k=10000). We used 're' library for this function to switch M ,K ,and other str to numbers. The function is used later to clean 'Size' column.

*clean_install_coll-using str.replace to clean'+' and ','. The function is used later to clean 'Installs' column.

*change_time- using datetime library it changes the column date from str to date. The function is used later to clean 'Last Updated' column.

*change_app_coll_nan-using isnull to switch nan values with string. The function is used later to clean 'App' column.

*change_app_coll_nan-using isnull to switch nan values with 0. The function is used later to clean 'Rating' column.

*clean_nan_from_sentiment_subj-using isnull to switch nan values with average. The function is used later to clean 'Sentiment_Subjectivity' column.

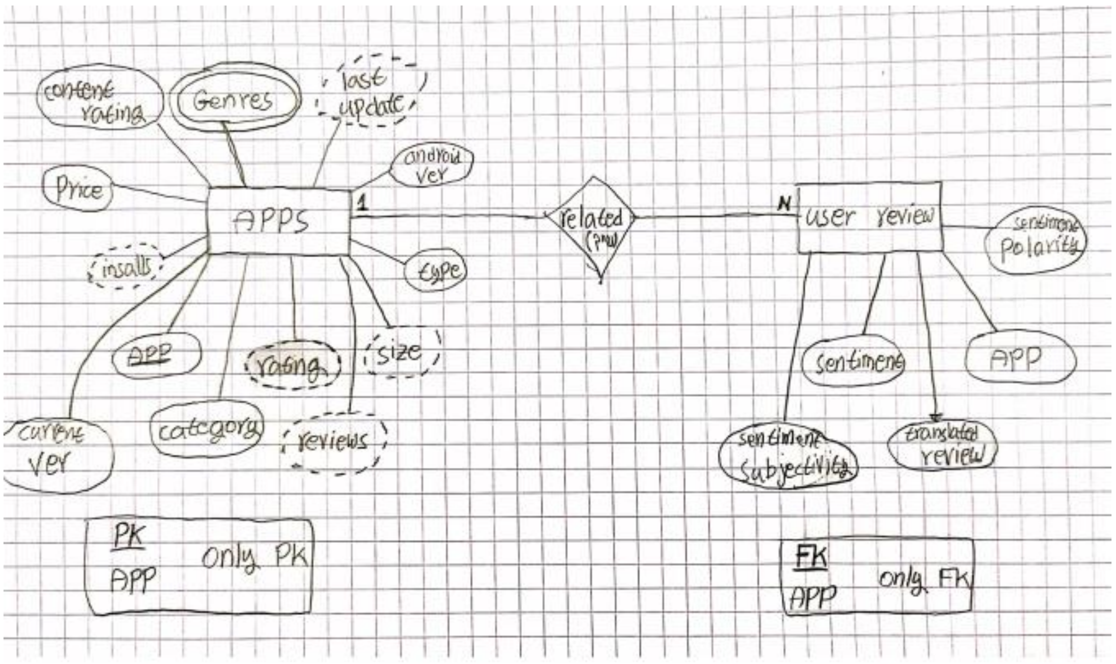
*clean_nan_from_sentiment_pol-using isnull to switch nan values with average. The function is used later to clean 'Sentiment_Polarity' column.

*insert_most_frequent_to_nan_sentiment- using isnull to switch nan values with most common value. The function is used later to clean 'Sentiment' column.

Data Prepare

Q1

We printed the num of rows using shape and column names using head(0).



Q2

-We printed the total amount of categories using format() for the print.

-After cleaning 'Size' getting the app with the biggest size using Pandas.

-After cleaning 'Installs' getting the app with the most installs using Pandas.

- After cleaning 'Last_updated' getting the app with the most latest update date.

- *split_genres- the function get 'Genres' column and splitting the duplicates values using str.split and combing them using concat. The function return the genre with its rating.

Q3

-We printed the number of apps using value_counts()

- We printed a list of the free apps using list() on the app with type='Free'.

-To check which type of apps more used, the code check the sum of installs to free apps and paid apps ,compare between them and print the text related to the outcome.

-*get_app_details_by_letter- the function returns all apps that starts with the letter given using str.startwith

Q4

-We checked the number of apps that classified as positive, negative and neutral with Pandas.

-After cleaning the "Rating' column we merged the maximum rating apps with user reviews and printed their sentiment.

-to print the average polarity of free apps we merged the free apps with user reviews and printed their average sentiment polarity using mean().

-*get_average_polarity-if the app name exist thr function calculate it average sentiment polarity.

-*get_sentiment- the function get app name and return it sentiment using get_average_polarity.

Q5

-In the first three parts of the question is explained in the conclusions.

-In this part we checked for each category statistics information about its rating. First we created a new data frame of the relevant columns the we grouped the data frame according to the category and then took out the statistic information. From the statistic information we got it seems like the most stable category is entertament with the lowest std.

-In this part we checked if the rating for each category distribute normally. For this test we used the library scipy. From the test that exist in the library we get back the p value of the rate data and then we cheek if the p value is lower or bigger then the significance level according to instruction (0<x<0.05) .then each category is printed with its name and if its distribute normally or not or there is not enough data.

Subject 3

Knn_classifier

For using knn algoritem we first needed to replace all the nan values in our data. all the changes fuction are in the data_preper file. We replaced each numeric column to the average of all the other cells and replaced in the string column to the most frequent cell. Then we changed all the data to be numeric. We used one hot vector for showing genres, app name got a number from a dict we created that each name gets a value, the sentiments we changes from positive, negative, neutral to -1,0,1.

2. We merged between both data bases with the key 'app' and got al the data we needed for the knn alogritem. We devided the new data we got to X (all data) and Y(only sentiment).

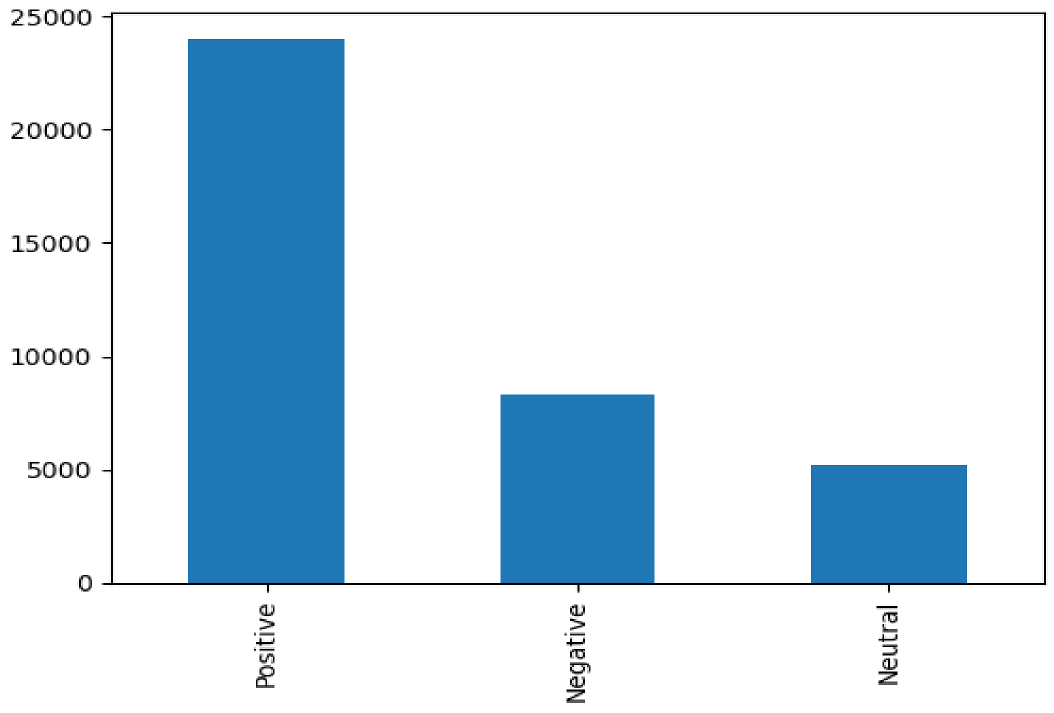
3. We trained and splited our data giving test 20% weight. And after finding x-test and x-train we normalized them.

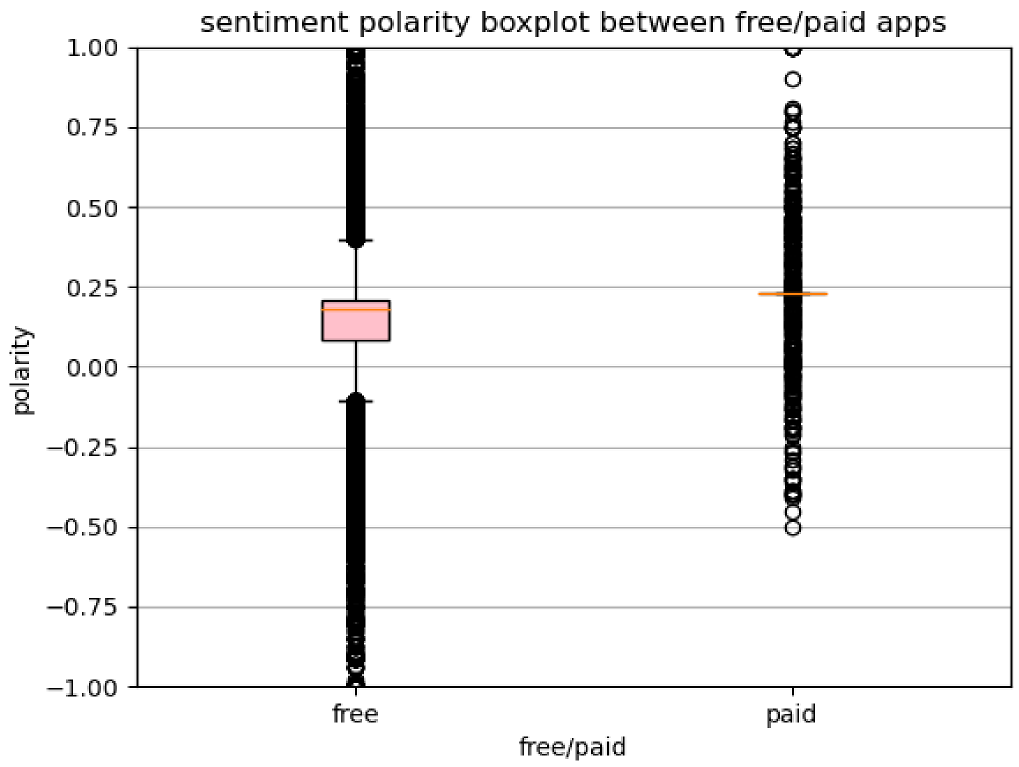
4. Created a classifier of kneighborsclassifier using the 'Euclidean' matric. We run on this algorithm 20 time checking each time a different k neighbors. at the end we returned the highest accuracy and the best k adding to it a graph that shows each k and his accuracy.

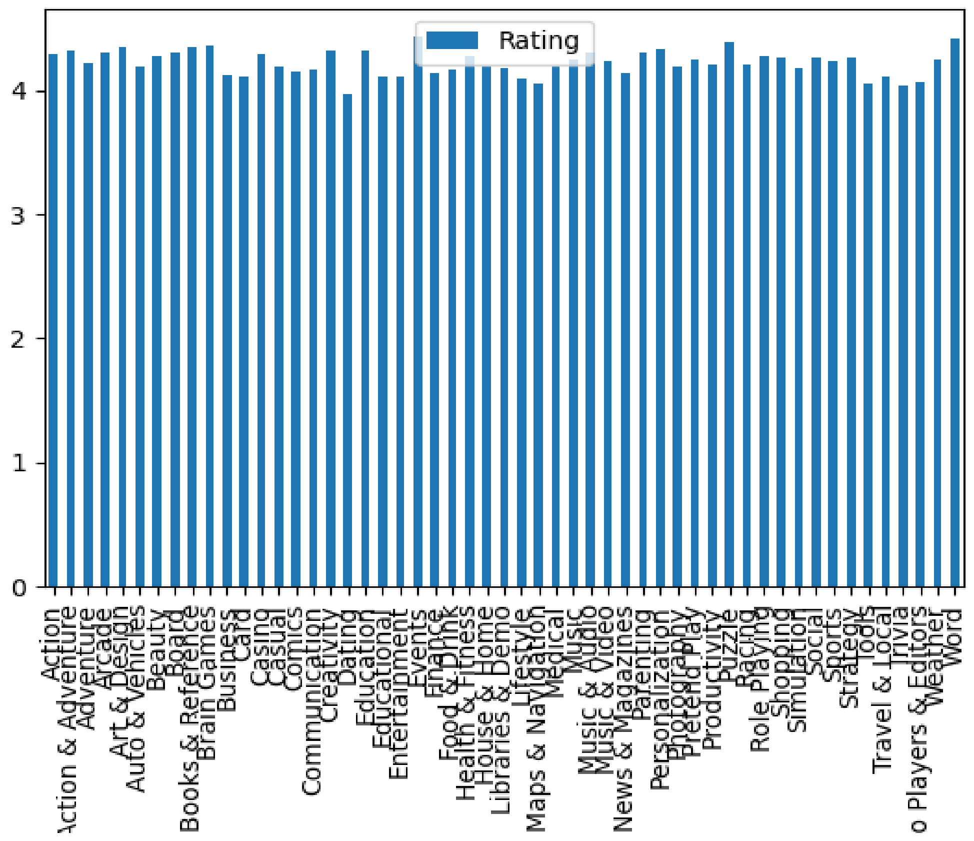
Why we do not want to use K=data-length? In this case our k will be a big number on the one hand high value of k causes reduce of the effect of noise on classification. On the other hand the cause the limit between departments to be less pronounced this why we will not want to use k=data-length.

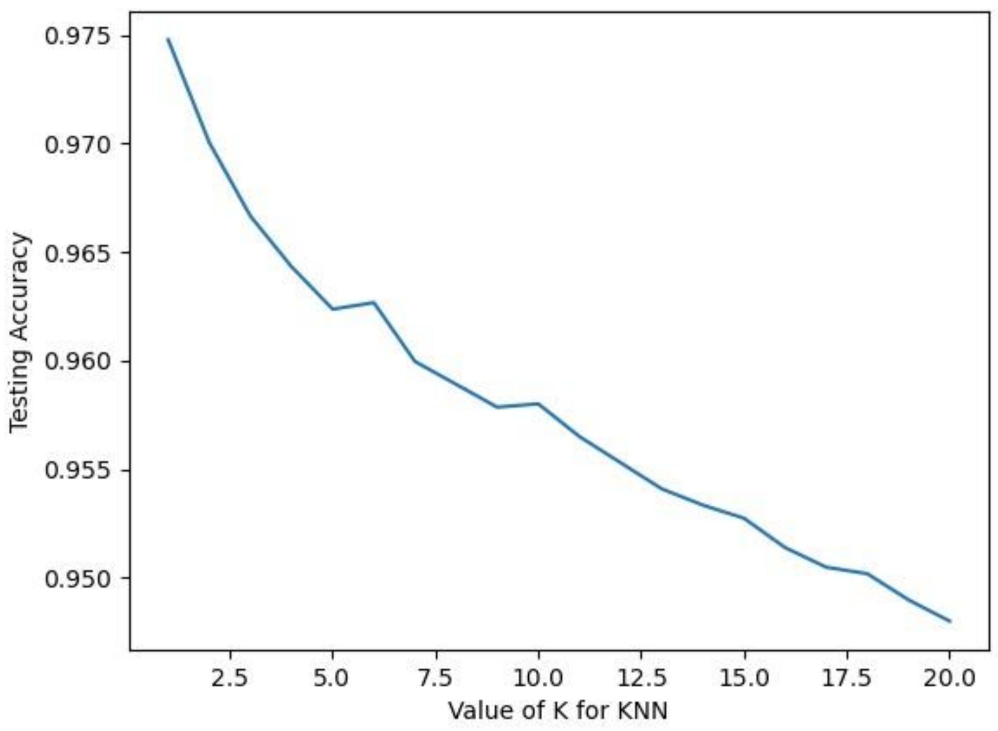
Conclusions

All conclusions from part B and C:




sentiment polarity boxplot between free/paid apps
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
polarity
free
paid
free/paid


Rating
4
3
2
1
0
Action
ction & Adventure
Adventure
Arcade
Art & Design
Auto & Vehicles
Beauty
Board
Books & Reference
Brain Games
Business
Card
Casino
Casual
Comics
Communication
Creativity
Dating
Education
Educational
Entertainment
Events
Finance
Food & Drink
Health & Fitness
House & Home
Libraries & Demo
Lifestyle
Maps & Navigation
Medical
Music
Music & Audio
Music & Video
News & Magazines
Parenting
Personalization
Photography
Pretend Play
Productivity
Puzzle
Racing
Role Playing
Shopping
Simulation
Social
Sports
Strategy
Tools
Travel & Local
Trivia
o Players & Editors
Weather
Word



1. the bar shows the amount of apps that their sentiment is positive, negative and neutral. Ass we can see the most common sentiment value is positive (0<x<1).

   2. This bar shows the average rating for each genrer.as we can see in the bar the genres with the highest average rating is events and words.

   3. This box plot show us the differences in the sentiment polarity between free apps and paid apps. First we can learn from the black circles that there are many more free apps then paid apps. Else we can see that the sentiment polarity in free apps is more common to be between -0.12 – 0.37 and in paid app its more evenly spread. Else we can see the polarity average in paid apps are higher then polarity in free apps.

   4. This linear graph shows us what is the accuracy of each k between 1-20. Ass we can see the best accuracy is when k=1. Else we can see the the graph downward trend. As we take a bigger k more neighbors our accuracy is going down.