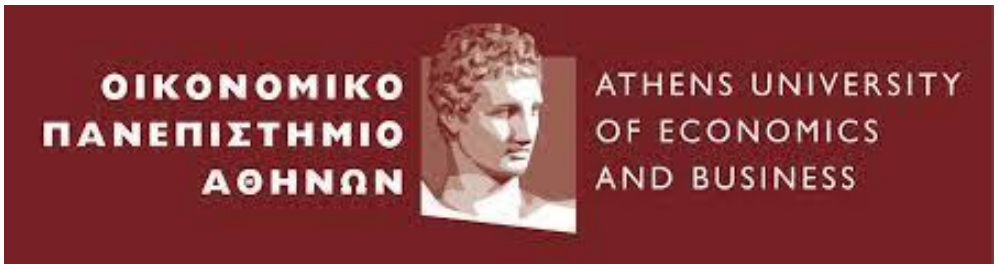

ATHENS UNIVERSITY OF ECONOMICS & BUSINESS

DEPARTMENT OF MANAGEMENT, SCIENCE & TECHNOLOGY

MSc BUSINESS ANALYTICS

"Redis - MongoDB Assignment"

Full Name: STAMATIOS SIDERIS

Register Number: f2822113

ATHENS, 2022

Scenario

You are a data analyst at a consulting firm and you have access to a dataset of ~30K classified ads from the used motorcycles market. You also have access to some seller related actions that have been tracked in the previous months. You are asked to create a number of programs/queries for the tasks listed in the "TASKS" section.

Introduction to Task 1

In this task you are going to use the "recorded actions" dataset to generate some analytics with REDIS. The dataset includes 2 csv files, the emails_sent.csv and the modified_listings.csv.

At the end of each month, the classifieds provider sends a personalized e-mail to some of the sellers with a number of suggestions on how they could improve their listings. Some e-mails may have been sent two or three times in the same month due to a technical issue. Not all users open these e-mails. However, we keep track of the e-mails that have been read by their recipients in emails_sent.csv. Apart from that you are also given access to a dataset containing all the user ids along with a flag on whether they performed at least one modification on their listing for each month in modified_listings.csv.

Preparation

The months included in the datasets are January, February and March. We create 3 subsets, one for each month, of the emails_sent.csv dataset and 3 subsets, one for each month, of the modified_listings.csv dataset.

Questions

1.1 How many users modified their listing in January?

Using the January_listings subset, we create a BIT where "1" values show the users that modified their listing in January. The total number is equal to **9969**.

1.2 How many users did NOT modify their listing in January?

By reversing the BIT of Question 1.1 we create a new BIT where "1" values show the users that did not modified their listing in January. The total number is equal to **10031**. Although the total of January listings is equal to 19999, the sum of totals of Questions 1.1 and 1.2 is equal to 20000. The difference of 1 occurs because BITTOP performs in bits while BITCOUNT performs in bytes. As a byte is equal to 8 bits, 19999 observations do not create a round number of bits and so 1 more bit is added to solve this problem.

1.3 How many users received at least one e-mail per month (at least one e-mail in January and at least one e-mail in February and at least one e-mail in March)?

By iterating, per user ID, through the subsets we created for each month on dataset emails_sent, we create a BIT for each month where value "1" shows the users thar received at least one email on that month. By using BITOP and "AND" we create a BIT where value "1" shows the users that received at least one email on all the 3 months. The total of users is **2668**.

1.4 How many users received an e-mail in January and March but NOT in February?

We use the previously created BITs EmailsJanuary and EmailsMarch with BITOP and "AND" in order to create a BIT where value "1" shows the users that received email on both months. Afterwards, we use subset EmailsFebruary and we create an inverse BIT using BITOP and "NOT". Finally, we create a BIT using BITOP, "AND" and the 2 BITS we created where value "1" shows the users that received an email in January and March but not in February. The total is **2417**.

1.5 How many users received an e-mail in January that they did not open but they updated their listing anyway?

We create a subset of emails sent in January including the columns UserID, MonthID and EmailOpened. We aggregate the subset by using sum on column EmailOpened where the UserID is the same as, if sum is greater than "0", it will mean that the user has opened at least one of the emails he received in January. We change the value of each row on column x to "1" if its value is greater than "0". We iterate through our created dataset and create a BIT where value "1" shows the users who opened their emails in January. Moreover, we create a BIT to inverse the previous BIT using BITOP and "NOT" where value "1" will show users that did not open their emails in January. Finally, we create a BIT of the Union of the BIT for users that did not open their emails in January and the BIT created in Question 1.1 for the users that modified their listings in January. The total is **7157**.

1.6 How many users received an e-mail on January that they did not open but they updated their listing anyway on January OR they received an e-mail on February that they did not open but they updated their listing anyway on February OR they received an e-mail on March that they did not open but they updated their listing anyway on March?

We perform the same steps as in question 1.5 for months February and March. Using BITOP and "OR" we create a BIT where value "1" shows the users received an e-mail on January, February or March that they did not open but they updated their listing anyway. The total is **12575.**

1.7 Does it make any sense to keep sending e-mails with recommendations to sellers? Does this strategy really work? How would you describe this in terms a business person would understand?

We calculate a BIT of emails opened followed by a modification on listing for each month and then created a BIT using BITOP and "OR" where value "1" shows all the users that received and opened an email and modified their listing afterwards in January, February or March. The total is **9817.** The number of users that did not open their emails but proceed with a modification is 12575 (as answered in question 1.6) and so higher than 9817. So, it seems that the strategy of sending emails is irrational.

Introduction to Task 2

In this task you are going to use the “bikes” dataset to generate some analytics with MongoDB. Dataset “bikes” is consisted of 29701 json files which refer to bike ads scrapped from the web.

Questions

2.1 Add your data to MongoDB.

We use library mongolite in R to import the data in MongoDB. First we create a new collection called “mycol” and a new database called “mydb” as a localhost. Then, we create an indexing txt file which includes the paths of all the json files included in the dataset “bikes”. By iterating through the list, we transform json files to data frames and pass them through a cleaning function in order to clean them. In the cleaning procedure we check for blanks and NAs, we substitute words using regex and change the type of variables and finally create a new column “Negotiable” which shows when an ad is negotiable about its price or not. Finally, we retransform data frames to json files and import them to mongoDB.

2.2 How many bikes are there for sale?

By using function count we count **29701** bikes for sale.

2.3 What is the average price of a motorcycle (give a number)? What is the number of listings that were used in order to calculate this average (give a number as well)? Is the number of listings used the same as the answer in 2.2? Why?

As 1395 ads with price lower or equal to 200 exist, we calculate the average excluding them as they are not realistic prices for bikes and might refer to parts. The total of the bikes left is 28306 (1395 ads difference from the initial dataset) and the average price for the bikes left is **3049.28**.

2.4 What is the maximum and minimum price of a motorcycle currently available in the market?

The maximum logical price is **89000** and the minimum logical price is **210** (as we excluded prices equal or lower to 200).

2.5 How many listings have a price that is identified as negotiable?

By counting the TRUE values of column Negotiable, we count **1348** negotiable listings.

2.7 What is the motorcycle brand with the highest average price?

The brand with the highest average price is Semog with an average price of 15600.

2.9 How many bikes have “ABS” as an extra?

The total number of bikes with ABS as an extra is **4008.**