CS838 PROJECT REPORT - STAGE 5

DATA ANALYSIS

(05/07/2017)

Gaurav Mishra
gmishra2@wisc.edu

Om Jadhav
ojadhav@wisc.edu

Pallavi M Kakunje
kakunje@wisc.edu

1. Overview:

The goal of this project stage is to perform data analysis on integrated cleaned table obtained in stage 4 in order to gain some insights. In particular, we are trying to classify developers to different classes based on the number of games released by them and the average rating of those games.

2. Schema

In this stage, we used the integrated cleaned table obtained by merging the tuples of the common games from Google Play Store (google table) and Apple Store (apple table) in stage 4. Each tuple represents a mobile game application. The schema of the table is of the form:

("ID", "name", "category", "developer", "rating")

There are total 490 tuples in the table.

Few tuples of the table are as follows:

ID	name	category	developer	rating
1	Candy Crush Saga	Casual	King	4.554035
2	Super Mario Run	Action	Nintendo Co., Ltd.	3.33734
3	Jetpack Joyride	Arcade	Halfbrick Studios	4.421695
4	Temple Run	Arcade	Imangi Studios	4.170195
5	Subway Surfers	Arcade	Kiloo	4.496675
6	Hill Climb Racing	Racing	Fingetsoft	4.423215
7	Sonic Dash	Arcade	SEGA	4.49338
8	Super Stickman Golf 2	Sports	Noodlecake Studios Inc	4.41765

3. What was the data analysis task that you wanted to do?
We wanted to classify developers to various classes based on the number of games released by them and the average rating of those games.

4. For that task, describe in detail the data analysis process that you went through.
We created a feature vector describing the number of games released by the developer. We made few groups - less than 2 games, between 2 to 5 games, between 5 to 10 games and more than 10 games. We added one feature for each of them. We also added features for average rating - less than 3.8, between 3.8 to 4.1, between 4.1 to 4.5 and more than 4.5. Again, we added 1 feature for each of them.

We labeled our training data based on certain rules. Basically we assigned lower classes to those developers who have released less games. Their class is high only if the average rating is really good. On the other hand, we assigned higher classes to developers who have released more games even if their average rating is not very good. The reason for this assignment is that developers with less games are rated by less people and so their ratings does not represent the opinion of large sample space. Also, based on the rating of one or two games we cannot infer that develop is actually good or not. Similarly, for developers with more games. The exact rules are as follows (class 1 is lowest and class 5 is highest):

Games less than 2 and rating less than 3.8 ---> 1 (class)
Games less than 2 and rating 3.8 to 4.1 ---> 2 (class)
Games less than 2 and rating 4.1 to 4.5 ---> 3 (class)
Games less than 2 rating more than 4.5 ---> 4 (class)

Game 2 to 5 and rating less than 3.8 ---> 2 (class)
Game 2 to 5 and rating 3.8 to 4.1 ---> 2 (class)
Game 2 to 5 and rating 4.1 to 4.5 ---> 3 (class)
Game 2 to 5 and rating more than 4.5 ---> 4 (class)

Game 5 to 10 and rating less than 3.8 ---> 2 (class)
Game 5 to 10 and rating 3.8 to 4.1 ---> 3 (class)
Game 5 to 10 and rating 4.1 to 4.5 ---> 4 (class)
Game 5 to 10 and rating more than 4.5 ---> 5 (class)

Game more than 10 and rating less than 3.8 ---> 3 (class)
Game more than 10 and rating 3.8 to 4.1 ---> 4 (class)
Game more than 10 and rating 4.1 to 4.5 ---> 5 (class)
Game more than 10 and rating more than 4.5 ---> 5 (class)

Then we divided our data into training sample and test sample. We learned several models and performed 5 fold cross validation. We chose decision tree as our classifier as it had highest precision and recall among all the classifier. The precision and recall obtained by all the classifiers is given below:

Classifier	Precision (%)	Recall (%)	F1 (%)
Decision Tree	98.28	97.81	98.04
Random Forest	97.84	98.16	97.99
KNN	95.90	95.40	95.65
SVM	90.32	91.46	90.88
Logistic Regression	92.66	93.14	92.89

Finally, in this step we use our learned classifier to predict output for the test set.
The reading looks as follows:
Precision : 97.64 %
Recall : 96.12 %
F1 : 96.87

5. What did you learn/conclude from your data analysis? Were there any problems with the analysis process and with the data?
We observed that most of the developers got classes in the range 3 to 5. Based on these results, we now have an idea of the reputation of the developer. Just like an average rating of games, developer's rating can also be used in Playstore or Appstore so that people have an idea before downloading the application. This can also be used by investment companies to decide which developer looks more promising and whether they should invest on them or not.

We had to spend some time to figure out how to assign classes developers and to create feature vector for them. Thanks to previous stages of the project, our table was in pretty good shape with no missing values. So rest of the process was smooth.

6. If you have more time, what would you propose you can do next?
1. Have more data to train the classifier.
2. Visualize our results using graphs and charts.
3. We can also perform some other data analysis tasks like:
    - Group games according to category and see which category has higher ratings
    - Cluster different versions of games according to name as there are a lot of games with slight variation in their names (eg “temple run” and “temple run 2”)