**Team 12 SA4108 CA submission**

**Supervised Techniques Report**

Group members:​

​

Li Zhengyi​

Liou Cheng Xiang​

Sivakumar Jayachelvi Nivvya​

Stephanie Cheng​

Thant Htet Myet​

Wang Wang Wang

**Classification Technique**

**Problem statement**

For our classification technique analysis, we have chosen the 17K Mobile Strategy Games dataset on Kaggle. It contains 16 variables, including the average user rating of the app on the Apple App Store. Generally, we can perceive the average user rating as a testament to how successful an app is. Our group wants to explore whether we can predict the average user rating of an app by applying Machine Learning algorithm on the other variables in the dataset. In other words, we want to be able to predict how successful an app will be, based on its features.

**Data Dictionary**

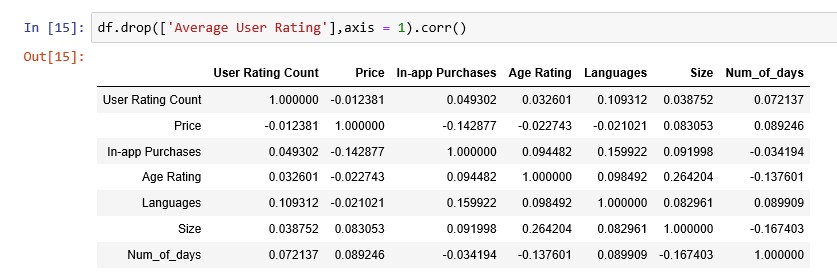
|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Data Type | Field Size | Description |
| Average User Rating | float | 1.1 | Rounded to nearest .5, requires at least 5 ratings |
| User Rating Count | integer | 4 | Number of ratings internationally, null means it is below 5 |
| Price | float | 3.2 | Price in USD |
| In-app Purchases | float | 3.2 | Prices of available in-app purchases |
| Age Rating | string | 3 | Either 4+, 9+, 12+ or 17+ |
| Languages | string | 30 | ISO2A language codes |
| Size | integer | 12 | Size of the app in bytes |
| Num\_of\_days | integer | 4 | Number of days the app has been in the app store |

**Data Analysis Processes**

Basic feature engineering

Our group did the following feature engineering before training the dataset with the various Machine Learning model.

1. The ‘in app purchases’ column shows the prices of available in-app purchases in the app. To make the data more meaningful, we encoded the column to a value of 0 or 1 to indicate if an in-app purchase is available for the app.
2. We drop all the rows will null values.
3. The 'Language' column shows the list of languages the app supports. We will instead do a count of the number of languages in the list and assign the value to this column.
4. The dataset contains an “Original Release Date” feature. We can derive the number of days the app is available on the Apple App store by subtracting the original release date from the data collection date. We believe this feature is more meaningful for machine learning.
5. The dataset contains an “Age Rating” feature, which has four categorical values. We encoded it to 4 values (1,2,3,4).
6. The “Average User Rating” feature in the dataset has 11 possible scores, from 0.0 to 5.0, in the increment of 0.5. We apply binning on this feature to generate three groups. Ratings that are 2.5 and below will be 1, 3.0 to 4.0 will be 2, and 4.5 & 5.0 will be 3.
7. Some of the features in the dataset contains text, and some are meaningless features like URL, ID, etc. Our group decided not to use these features for analysis.
8. With the remaining features, we tabulate the pairwise correlation score, as shown below. We observe no strong correlation between the features. Thus, we will not drop any of them.

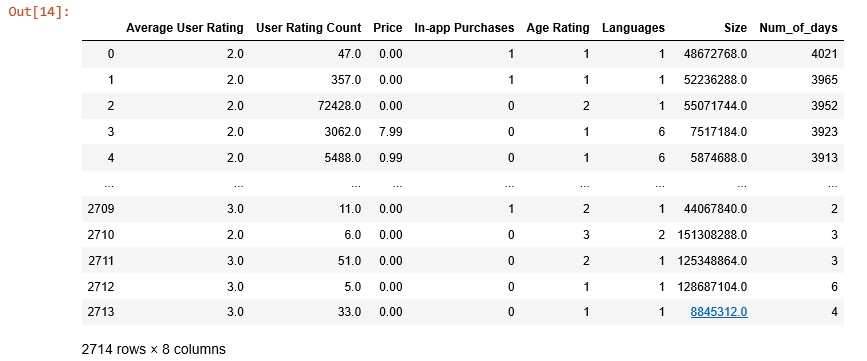


Training classification models

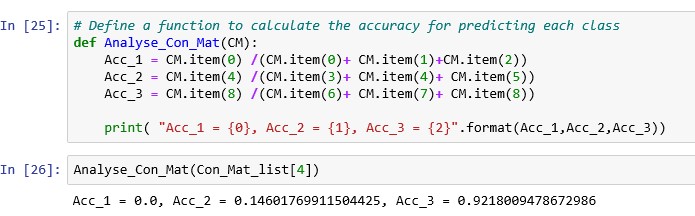
With the data ready, we proceed to train the three machine learning models and evaluate the results. The dependent variable and the independent variables are shown below. As mentioned before, we bin the dependent variable into 3 values, 1, 2 and 3.

Dependent variable

Independent variables



To evaluate the model, we will make use of the accuracy\_score method from sklearn. We will also check the confusion matrix to see the accuracy for each class of the independent variable. To do that, we define a function to evaluate the confusion matrix.

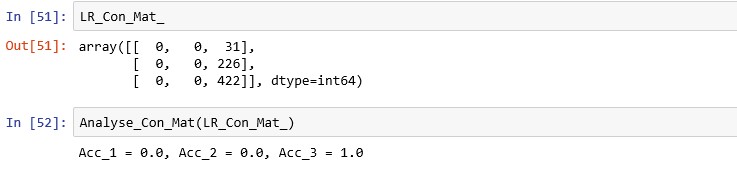


In addition to accuracy score, we are also interested in the training duration for each model.

The scores of the three models are tabulated below.

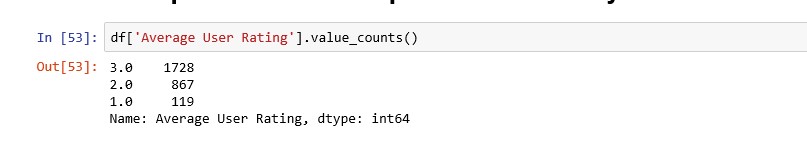
|  |  |  |  |
| --- | --- | --- | --- |
|  | KNN (K = 19) | Decision Tree | Logistic Regression |
| Accuracy (%) | 62.2 | 57 | 62.2 |
| Acc\_1 (%) | 0 | 22.6 | 0 |
| Acc\_2 (%) | 14.6 | 39 | 0 |
| Acc\_3 (%) | 92.2 | 68 | 100 |
| Train duration (ms) | 2 | 17 | 40 |

Looking at the accuracy score, we can see that it ranges around 57 to 62 % for the three ML models. However, a closer look at the confusion matrix reveals that most of the correct prediction comes from the class “3” prediction, as shown in the example below.



Looking at the results, we can see that the decision tree model has the best accuracy at this stage.

A closer study of our dataset shows that it is imbalanced, will most of the samples having ‘Average User Rating’ equal to 3. This imbalance is the reason for the failure of our models.

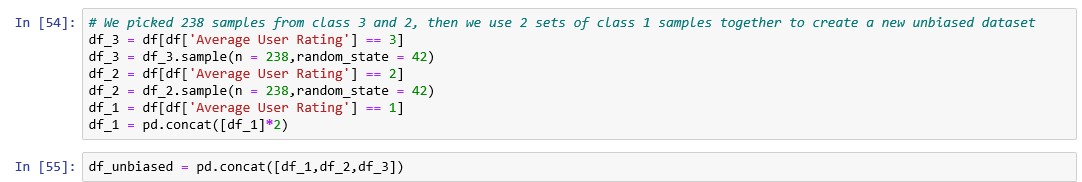


An observation we made is that the decision tree model has the best performance with imbalanced data, compared to KNN and logistic regression model.

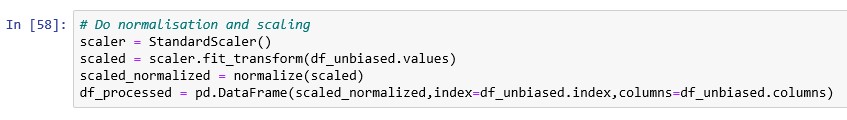
Thus, we need to perform data balancing in the next step to see if we can improve the results. We will also perform scaling and normalization of the features to enhance the performance further.

**Data balancing, scaling and normalization**

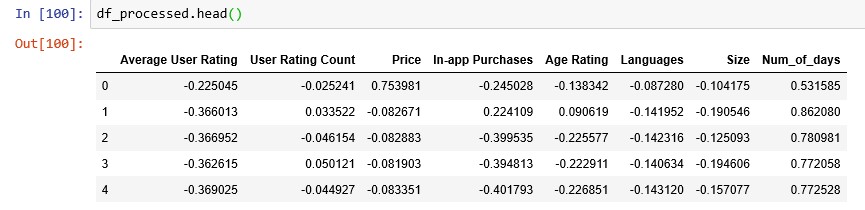
We perform data balancing by under-sample the majority groups and multiplying the minority group.



We then make use of the sklearn preprocessing library to scale and normalize our data.



With the new unbiased dataset, we proceed to train the three machine learning models and evaluate the results. The dependent variable and the independent variables are shown below.



The scores are tabulated below.

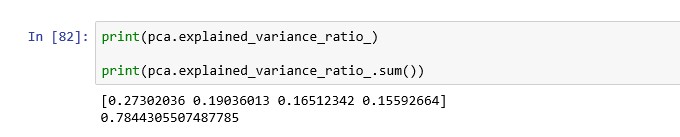
|  |  |  |  |
| --- | --- | --- | --- |
|  | KNN (K = 3) | Decision Tree | Logistic Regression |
| Accuracy (%) | 66 | 77.7 | 39.1 |
| Acc\_1 (%) | 73.1 | 91 | 46.3 |
| Acc\_2 (%) | 67.2 | 72.1 | 13.1 |
| Acc\_3 (%) | 55 | 67 | 60.8 |
| Train duration (ms) | 3 | 7 | 19 |

From the result, we can see that the training duration is reduced for decision tree and logistic regression model, but the duration increased slightly for the KNN model.

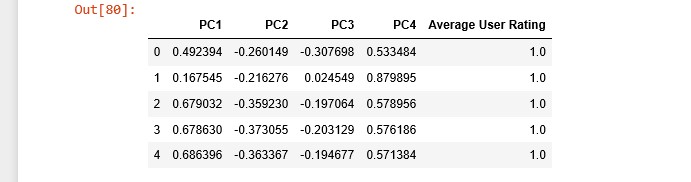
From the accuracy score, we can see that all three models has better performance when they are predicting samples with ‘Average User Rating’ equals to 1 and 2. This is an improvement compared to the last set of models. As such, we have improved the machine learning models with data balancing, scaling, and normalization. For our next step, we will perform Principal Component Analysis to see if we can improve the model further.

**Principal Components Analysis**

The motive of doing Principal Components Analysis is to see if we can shorten the training time as the new dataset will has fewer features. We make use of the PCA module from sklearn.decompositon library and generate a new set of features for our analysis. We chose the number of principal components to be 4 as it captures about 78 percent of the variance in the dataset.



With the new unbiased dataset, we proceed to train the three machine learning models and evaluate the results. The dependent variable and the independent variables are shown below.



The scores are tabulated below.

|  |  |  |  |
| --- | --- | --- | --- |
|  | KNN (K = 3) | Decision Tree | Logistic Regression |
| Accuracy (%) | 99 | 98.9 | 100 |
| Acc\_1 (%) | 100 | 100 | 100 |
| Acc\_2 (%) | 98.3 | 96.7 | 100 |
| Acc\_3 (%) | 100 | 100 | 100 |
| Train duration (ms) | 0.9 | 3.9 | 19 |

From the result, we can see that all three models can predict test data with almost 100 percent accuracies after Principal Components Analysis. Looking at the confusion matrix, we can see that the accuracy for predicting each class is close to 100 percent. We believe the increase in accuracy can be attributed to the fact that as we are now using fewer features, the model has less tendency to overfit. In addition, with fewer dimensions, it is easier for the Machine Learning algorithm to find a solution.

**Conclusion**

Throughout the process of performing classification on the dataset, our group can improve the accuracy score of the model with data engineering and principal components analysis. From our result, we can see that the biggest improvement come after we apply principal components analysis, after which all 3 models achieved almost 100 percent accuracy. Looking at the final score, we conclude that the logistic regression algorithm performs slightly better than the rest in term of accuracy. If we are looking at the model training duration, the K nearest neighbor algorithm is the most efficient model out of the 3.

**Linear Regression**

For our linear regression dataset, we have chosen the resale flat prices dataset on data.gov.sg. As the original dataset which contains information from 2017 to 2019 has 58155 rows, we have extracted out only 2019’s data to analyze. The dataset contains 11 features.

**Problem Statement**

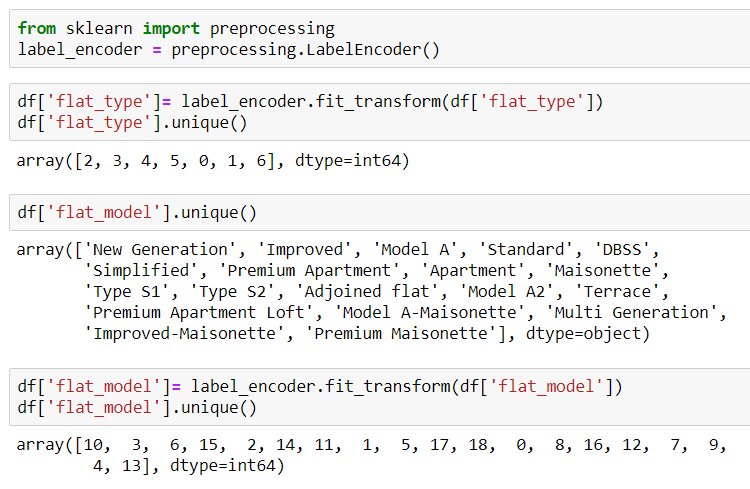
We would like to be able to estimate what would the resale price of the flat be, when given the required inputs. First off, we will need to know which inputs are the ones that have a correlation with the resale price so that we are able to use these inputs to predict with a better accuracy what the resale price will be, before we can proceed to try to predict the resale price.

**Data Dictionary**

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Data Type | Field Size | Description |
| floor\_area\_sqm | float | 2.1 | Floor area of the flat in square meters |
| lease\_commence\_date | integer | 4 | Year the lease of the flat commenced. |
| flat\_type | string | 20 | Type of flat |
| flat\_models | string | 25 | Model of the flat |

**Data Preparation**

After importing the data into a pandas dataframe, we remove the columns that we assume are less relevant in determining the resale price. After that, we use a label encoder to encode the categorical data (i.e. flat\_type and flat\_model).

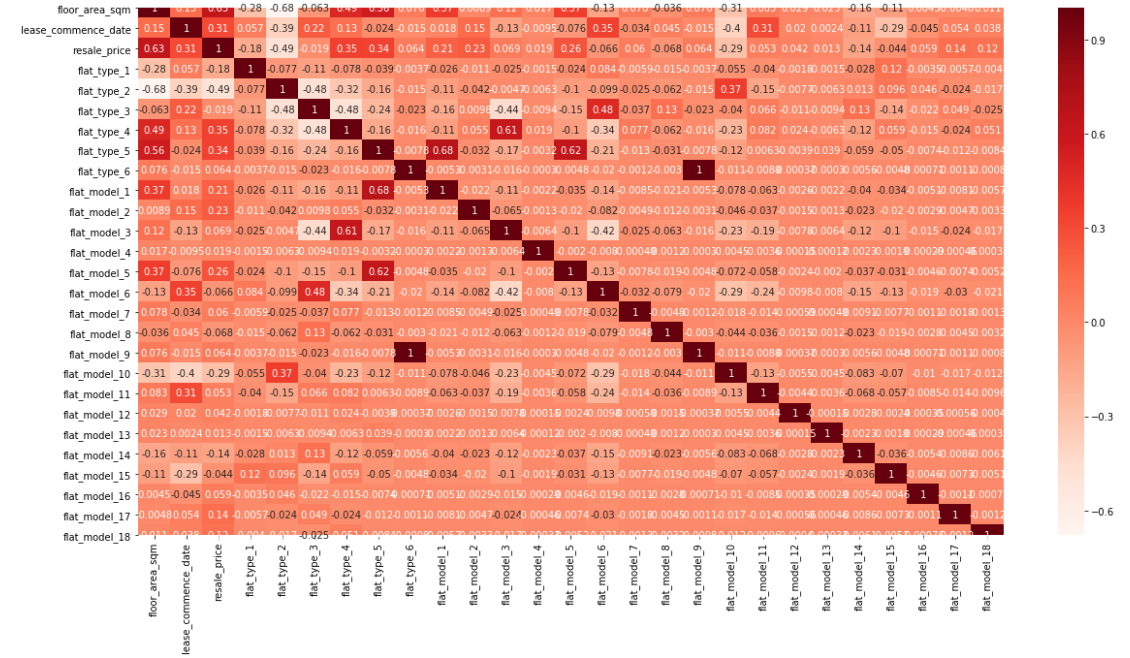


In order to ensure that the model does not think that the bigger numbers are better, we have also used one hot encoding to encode the data. Normalization was attempted, but there was no difference in the data.

**Feature Engineering**

We tried to do feature selection by using the Pearson Correlation. We only highlighted those that had a correlation of more than 0.5 to show which features have a stronger correlation to the resale price and only floor\_area\_sqm had a correlation of more than 0.5. We also realized that the Pearson Correlation is not a good gauge of correlation for categorical data unless the categories can be separated into 0 or 1 (i.e. 2 categories only). In this case, we chose not to group the different types and models of the flat into 2 categories only as we felt that this would distort the data.

As you can see from the heatmap below, it does not provide any useful information for the categorical data.

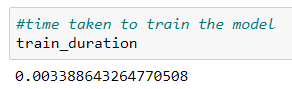


We then plotted the data (using floor\_area\_sqm and resale\_price), and even though there is a lot of data points, there seem to be a liner relationship between the two.

**Training the data**

We proceeded to train the data and adjusted the test size to see how the accuracy would be affected. In this case, a smaller test size (we used 0.002) resulted in higher accuracy. While plotting the test data, we noticed that there were several outliers. It could be due to the data having a substantial amount of noise that resulted in a lower accuracy as more data is trained.

In this case, it does not take long to train the data.



Since we are unable to determine previously if flat\_type and flat\_model are correlated to the resale price, we tried to test the accuracy by adding in these two features to see if the accuracy would change. Based on the accuracy derived, we assume that flat\_type probably has a higher correlation to the resale price than flat\_model.

|  |  |
| --- | --- |
|  | Accuracy (%) |
| floor\_area\_sqm | 53.5 |
| floor\_area\_sqm and flat\_type | 55.4 |
| floor\_area\_sqm and flat\_model | 51.3 |
| floor\_area\_sqm and flat\_type and flat\_model | 49.2 |
| floor\_area\_sqm and flat\_type and flat\_model  (using label encoding only, no one hot encoding) | 56.9 |

We also tried to see what the accuracy would be like if we did not do the one hot encoding and just proceed to train the data, and it resulted in the highest accuracy of 56.9%. Our group realized that no matter which method performed, the accuracy remains low (in the range of 40+ to 50+ percent).

**Conclusion**

In conclusion, we think that the resale price might take a lot more factors to be calculated (like the location of the property) but in this case without better domain knowledge, we are unable to give a number to these values for linear regression, and thus why the accuracy is not as high. It is possible that better accuracy could be achieved if we had used PCA as well. Overall, even though the data seemed suited for linear regression, we learned that it might not necessarily be so.