Capstone Project
Data Scientist Nanodegree

Fatma Mohamed Hassanein
28 March, 2021

Correlation between Mobile Handsets prices and features with their popularity

Definition

Project Overview

The evolution of mobile phones has been very swift in the past few years. The main aim of this project is to try to correlate the mobile phones technical features and prices with their popularity in the online markets then try to predict the popularity of new mobile phones.

To achieve this, three datasets were used; the first one is **GSMArena** as a reference for all mobile phones' commercial names and their features. The second and third datasets are **Amazon Cell Phones Reviews** and **Amazon Reviews: Unlocked Mobile Phones** which are used as reviews datasets to reflect from people reviews the popularity (average rating) of the released mobile phones.

The project mainly serves the telecommunications field and new handsets enhancements, pricing and marketing.

Problem Statement

There is a lot of challenges in every step in this project; the first one was to correlate the reviews that were web scrapped from Amazon with the actual mobile model and its features since the information in the datasets were not straight forward to correlate.

The second challenge was to extract the features that are not mostly missing, useful and can be correlated with the mobile phone popularity.

The third and main challenge was to build a machine learning model that can choose the most important features from the processed ones then predict the popularity of mobile phones from the resulting datasets. This part was very challenging since the final dataset used for the model was not very big so it can be subjected to over-fitting easily and also the features were numerous so choosing the right features without over-fitting was also challenging.

A lot of machine learning regression models were tried out and compared to get the most fitting model for the dataset and predict with the best accuracy.

Scoring

The model used in this project was the K-Nearest Neighbors Regressor and it out-performed the decision trees algorithms and the XGBoost due to the data size limitations producing a root of mean squared error of 0.542 after all performed refinements. The most important features that helped in the rating prediction was the price, the battery life and the weight of the mobile body.

Analysis

Data Exploration and Visualization

To make this project, three datasets were used as reference datasets for the information required. The first dataset was **GSMArena**; it was used to reference the original mobile name and extract its features.

The other two datasets which are **Amazon Cell Phones Reviews** and **Amazon Reviews: Unlocked Mobile Phones** were used to get the prices and average rating for each mobile phone after linking it to the original commercial name in GSMArena through the title of the review and the mentioned brand of the mobile phone.

After finishing the pre-processing of the three datasets and producing one final dataset with all the information required, analysis was done on various aspects to know more information about the key features and their correlation to the mobile popularity.

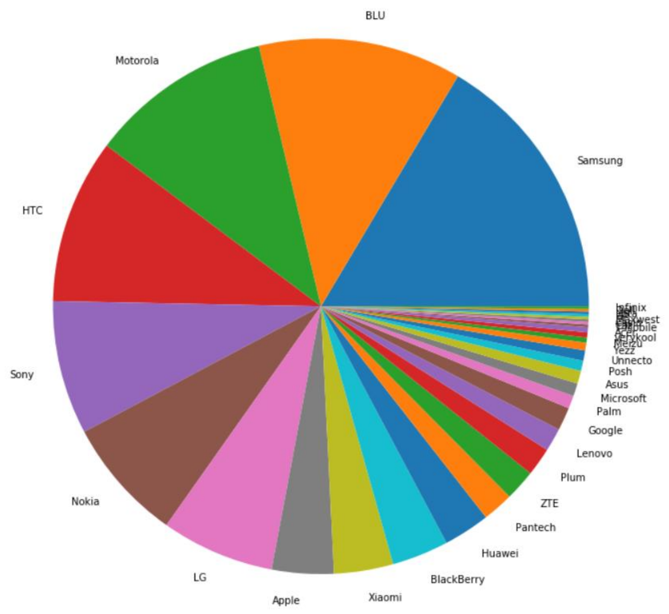
The following Questions were proposed to clearly understand the data:

- What is the percentage of each brand in the dataset?
- How old is the mobiles in the dataset?
- What is the average price of mobiles per brand?
- Which brand is the most popular?
- What are the top 25 popular mobile phones?
- What is the Operating System of mobile phones in dataset?
- What is the supported networks in the dataset mobile phones?
- How many of the mobile phones support dual sim?
- Is the mobile rating correlated to the mobile features and its price?

Data Visualization

Q1: What is the percentage of each brand in the dataset?

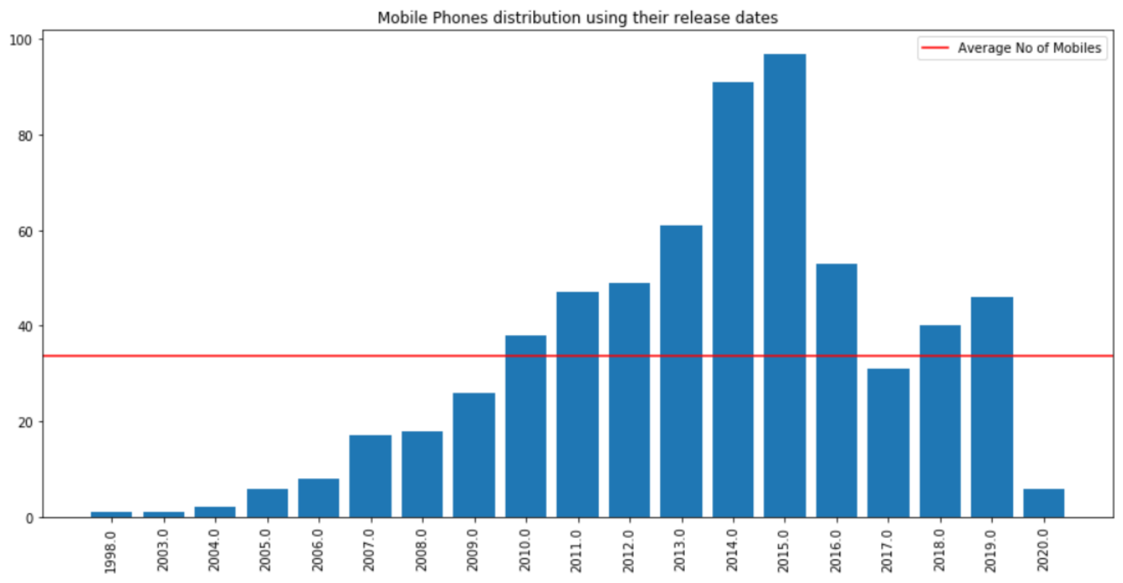
The first aspect to look at after data-preprocessing was the brands distribution in the dataset to understand the diversity of the dataset and whether it is biased to a specific brand.



As shown in the figure, there are some brands that were having the most records but the highest percentage was 16% which indicates an acceptable diversity in the mobile brands in the dataset.

Q2: How old is the mobiles in the dataset?

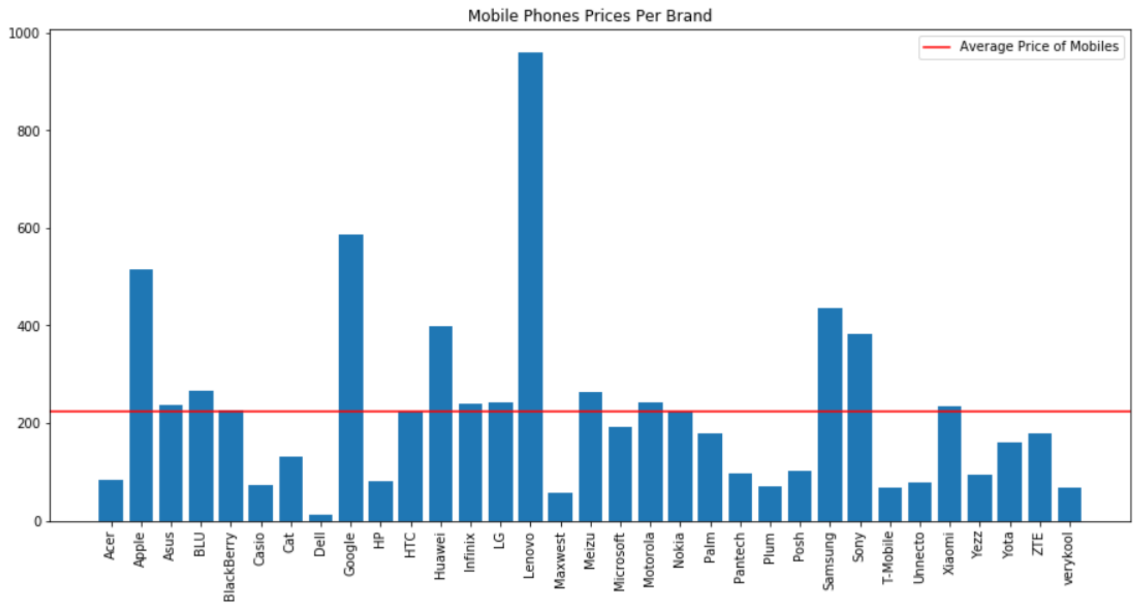
Another aspect to look at the release dates of the mobile phones of the datasets in order to know the expected available features during this time and how much the model will be correlation to the current market.



As shown in the figure, linking between the two reviews datasets made a wide range of mobile phones with various release dates but most of them are released during 2014 and 2015.

Q3: What is the average price of mobiles per brand?

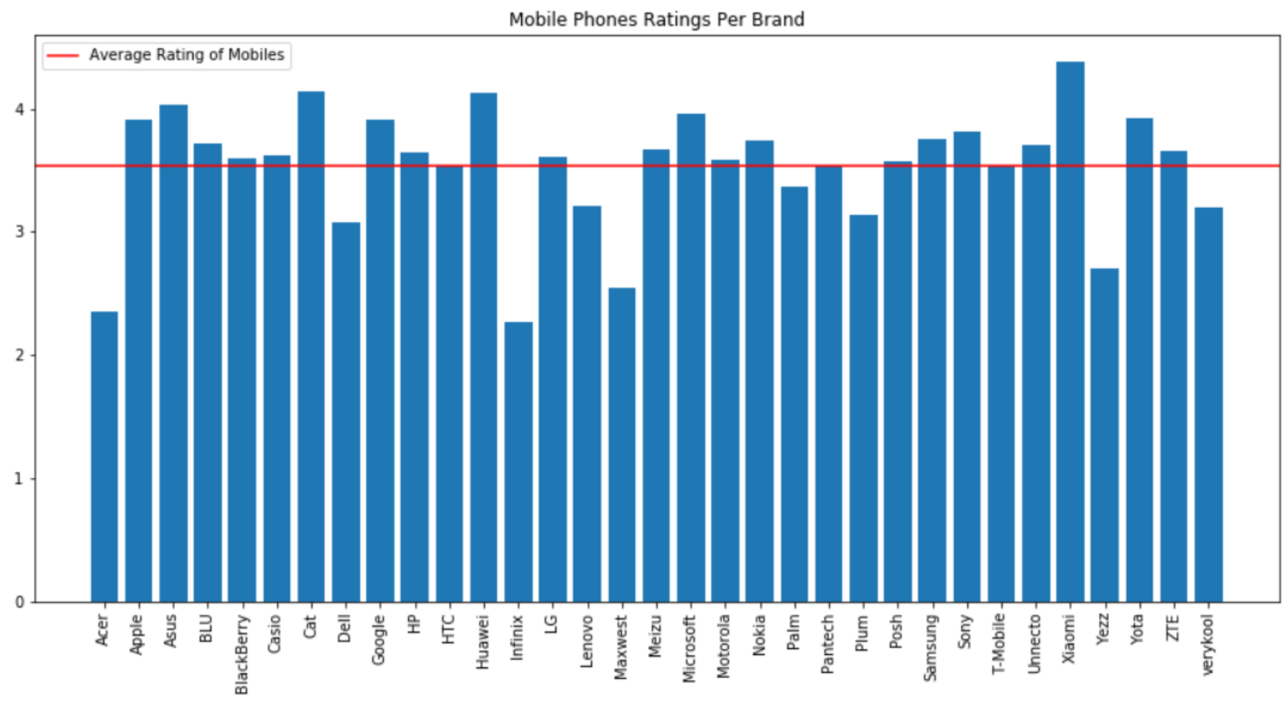
A possible aspect that is expected to affect the popularity was the price of the mobile phone so the average price per brand was an indication of the price ranges in the dataset.



The prices showed a huge diversity in terms of range which can be somehow logical since every brand targets different user tiers and sometimes, the same brand have different types of handsets varying between normal and high-end handsets.

Q4: Which brand is the most popular?

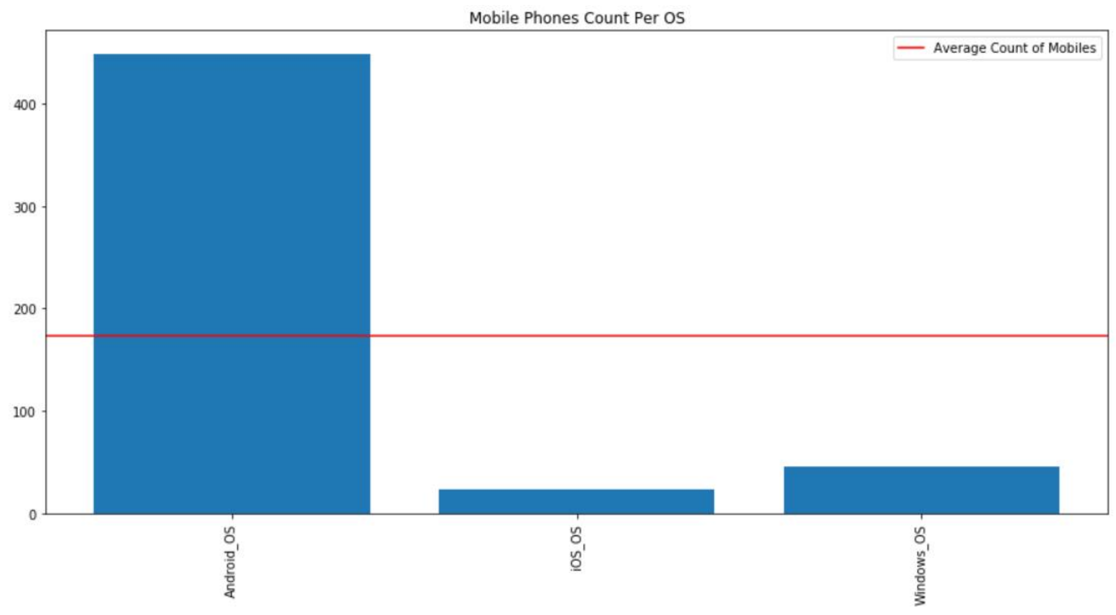
A very important aspect was to know whether the ratings are biased to a certain brand or not in order to know the effect of the brand as a feature in the model.



As shown, the brand has a very good effect on the rating as there are brands that have average rating a lot higher than other ones. That is why adding the brand as a feature in the model might be useful in the prediction process.

Another questions were asked to explore the categorical features included in the dataset and how diverse they are. The results of this visualizations are indicative whether these features can be useful to the model or not.

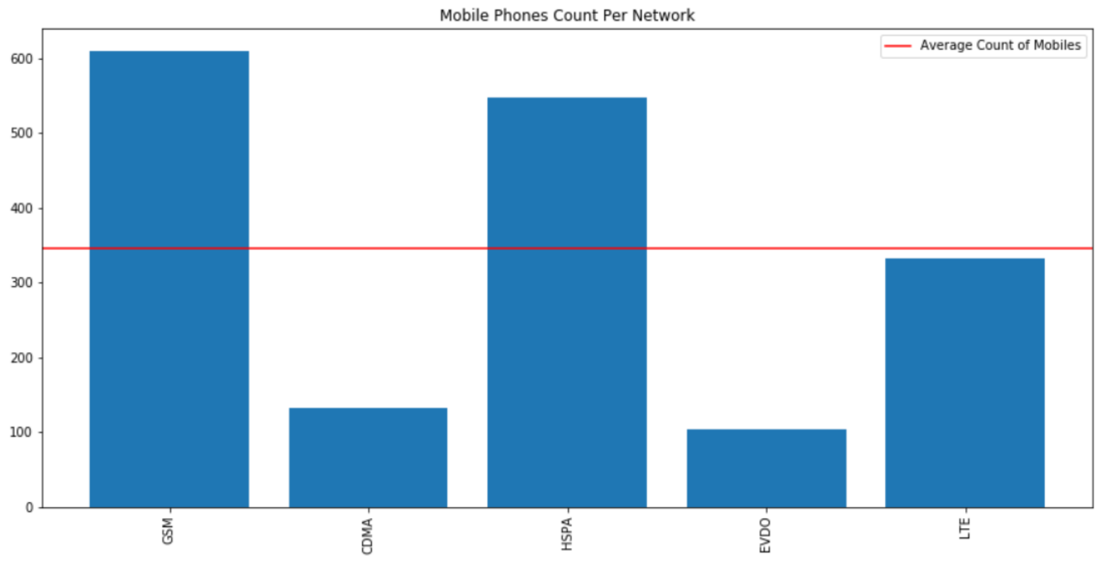
Q6: What is the Operating System of mobile phones in dataset?



Most of the dataset turned out to have Android OS so most probably, the OS features are not diverse enough to be used.

Q7: What is the supported networks in the dataset mobile phones?

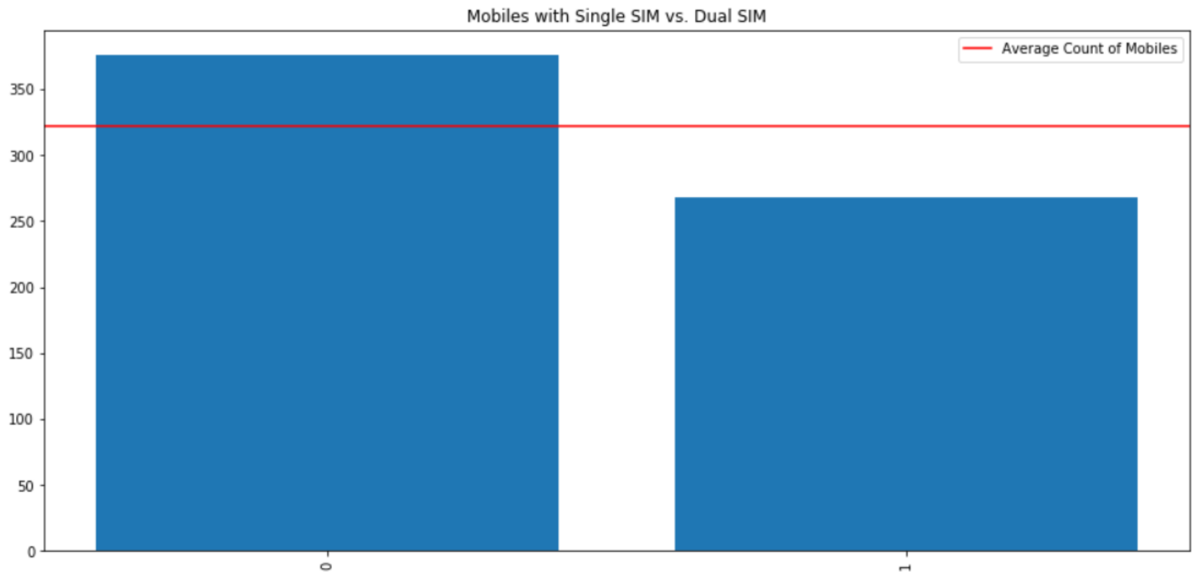
A main feature in the mobile phone is the types of mobile Networks it supports; this mainly affect the internet service in the phone the quality of voice calls.



After taking a look at the distribution of the network features, the LTE only can be useful since it is near the mean of the dataset while the rest are either available in most phones or not available in most phones.

Q8: How many of the mobile phones support dual sim?

The dual sim feature is the ability of the mobile phone to have 2 SIM cards instead of one; this feature is sometimes main to have for users.



Looking at the distribution, it indicates that the feature might be useful to lead the model in case it exists in highly rated handsets.

Methodology

Data Pre-processing

This part is considered a very important milestone in this project. This is because the reviews data only contain titles with long description about the products. To be able to link the reviews and prices in the reviews dataset, the brand was firstly used the first clue to the mobile phone name. However, the brand column had some spelling mistakes and differences from the GSM namings. As a result, firstly, some strings were updated and spelling mistakes were corrected. Then, find the closest brand to the one in the dataset in order to unify their wording as in the below block of code. Finally, remove the unfound brands and merge the two reviews datasets.

```
# Solve some data entry problems
mobiles_revs_2_df['brand'] = mobiles_revs_2_df['brand'].str.replace('Samsybg','Samsung')
mobiles_revs_2_df['brand'] =
mobiles_revs_2_df['brand'].str.replace('Amazon.com,','Apple')

# Get all possible brands from features df
ref_brands = list(set(mobiles_features['oem']))

cleaned_brands = []
for brand in mobiles_revs_2_df['brand']:
    added = False
    for ref_brand in ref_brands:
        # Check if one of the reference brands in the current brand column value
        if(str(ref_brand).lower() in str(brand).lower()):
            # Add it to the new column list
            cleaned_brands.append(ref_brand)
            added = True
            break
    if(added == False):
        # Find the closest reference brands to the given column value
        closest_brand_match = difflib.get_close_matches(str(brand), ref_brands)
        if(len(closest_brand_match) != 0):
            # Add the closest one to the column list
            cleaned_brands.append(closest_brand_match[0])
        else:
            # Add the original given value
            cleaned_brands.append(brand)
# Adjust the brand column with the created list
mobiles_revs_2_df['brand'] = cleaned_brands
```

The next problem is to process the titles to indicate the mobile phone actual name. To do this, the GSMArena “oem” column is used to search inside the title for the longest and closest match to the mobile names available taking the brand into account to increase the accuracy.

This logic is applied in the two methods `create_closest_strings_dict` and `get_nearest_matching_names` in the 01_Data_ETL_Preparation_NB notebook with very detailed documentation in every step. The method `add_new_mobile_name` is used to add the new mobile name to the rows.

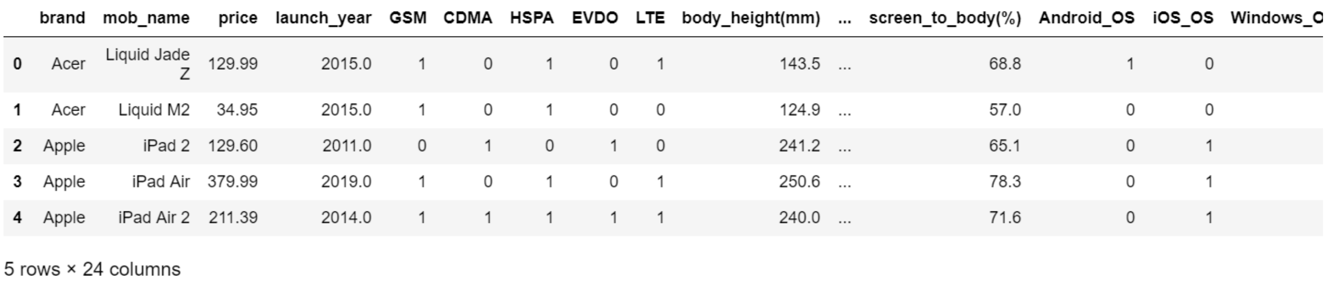
After this step, the features df and the reviews df are easily linked. However, one last step to finish data pre-processing was to process the features columns and convert the useful ones to either categorical columns or numerical columns in order to be able to use in the machine learning model directly.

The chosen features to extract and process were the following features:

- price
- launch_announced
- network_technology
- body_dimensions
- body_weight
- body_sim
- display_size
- platform_os

- memory_internal
- selfie_camera_single
- battery

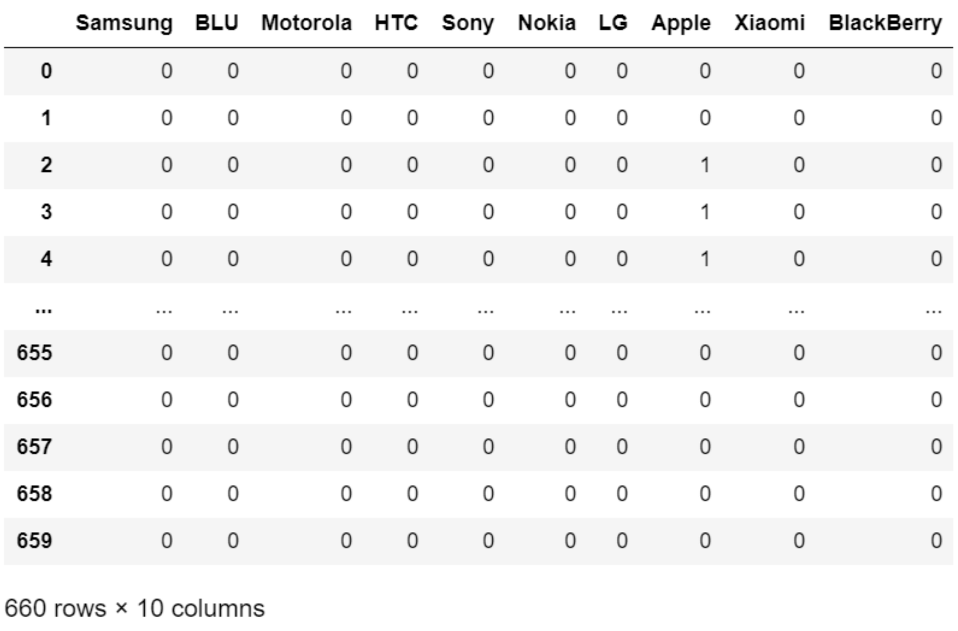
After processing, the final df contains 20 numeric and binary feature columns in addition to the average rating of each mobile phone with its brand as shown below



	brand	mob_name	price	launch_year	GSM	CDMA	HSPA	EVDO	LTE	body_height(mm)	...	screen_to_body(%)	Android_OS	iOS_OS	Windows_O
0	Acer	Liquid Jade Z	129.99	2015.0	1	0	1	0	1	143.5	...	68.8	1	0	
1	Acer	Liquid M2	34.95	2015.0	1	0	1	0	0	124.9	...	57.0	0	0	
2	Apple	iPad 2	129.60	2011.0	0	1	0	1	0	241.2	...	65.1	0	1	
3	Apple	iPad Air	379.99	2019.0	1	0	1	0	1	250.6	...	78.3	0	1	
4	Apple	iPad Air 2	211.39	2014.0	1	1	1	1	1	240.0	...	71.6	0	1	

5 rows × 24 columns

Then finally, the top 10 dominant brands in df are converted using get_dummies function to categorical columns and added to the dataframe.



	Samsung	BLU	Motorola	HTC	Sony	Nokia	LG	Apple	Xiaomi	BlackBerry
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...
655	0	0	0	0	0	0	0	0	0	0
656	0	0	0	0	0	0	0	0	0	0
657	0	0	0	0	0	0	0	0	0	0
658	0	0	0	0	0	0	0	0	0	0
659	0	0	0	0	0	0	0	0	0	0

660 rows × 10 columns

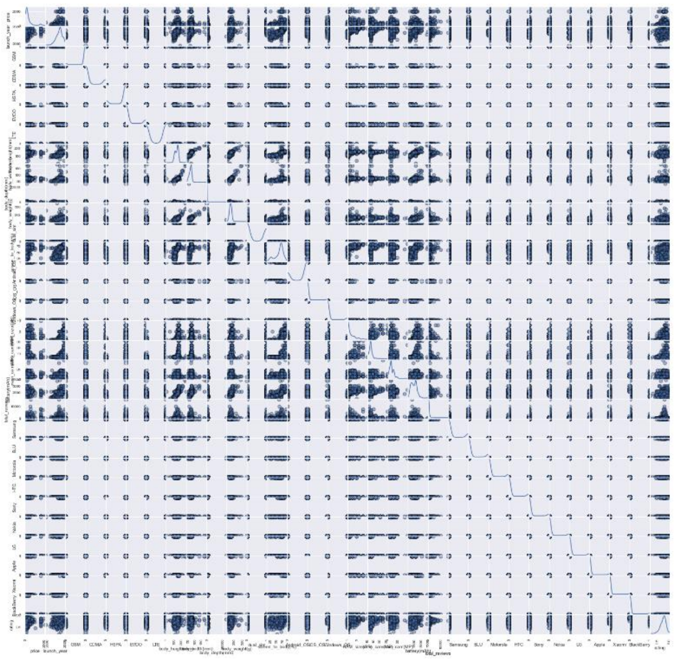
Implementation

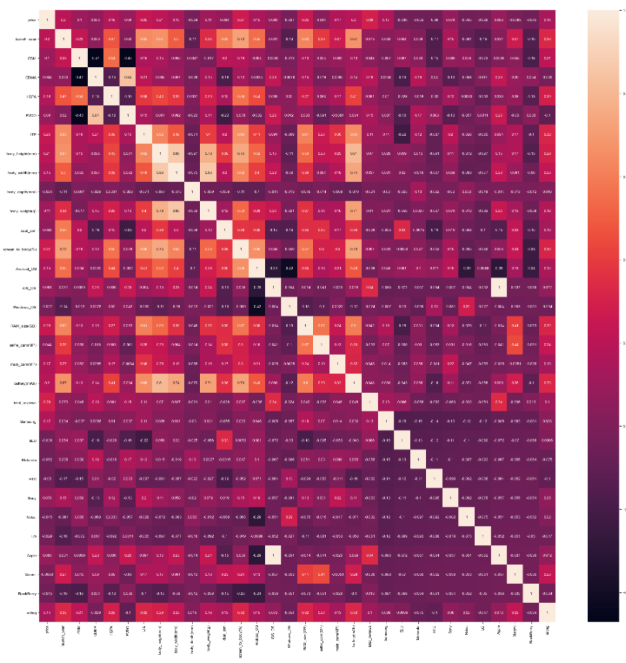
In this project, the problem needs a regression machine learning algorithm to predict the ratings of the mobile phone using its features. Consequently, to be able to know the suitable machine learning model, the correlation between the rating and the available features should be studied very well.

The first method used was a heatmap using seaborn showing the linear correlation coefficient between the features as shown below in the figure below. In addition, draw the graphs between the features to discover any linear or parabolic correlation between the features.

Finally, compute the correlation distance between 1-D array of each feature and the ratings to find the nearest in terms of distance as shown in the below code.

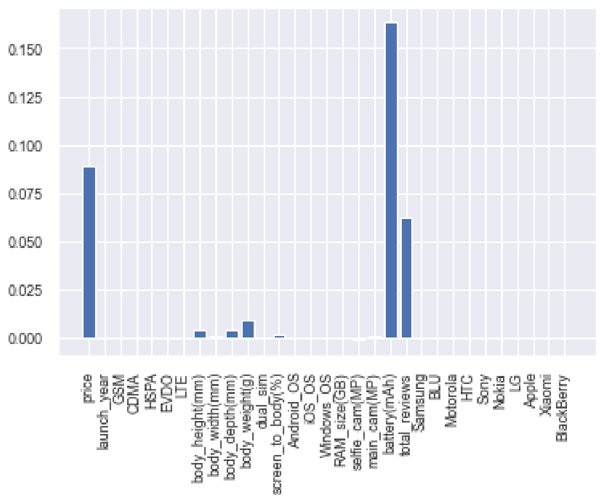
```
features_dict = {}
features_cols= list(df_with_brands.columns)

for col in features_cols:
    # Compute the distance
    features_dict[col] = dc(df_with_brands[col], df_with_brands['rating'])
```





The figures and the distances showed no strong and direct correlation between any of the features and the rating column. That is why the linear and logistic regression were excluded and non-linear algorithms were suggested for this dataset.

The dataset is then splitted to 80% training and 20% testing datasets; the X of the model was defined to be all features available in the df and Y was defined to be the rating column.



Random Forest and XGBoost were tried out but also excluded due to the problems of over-fitting and the fact that the dataset is very small for a decision tree algorithm. Finally, the K-Nearest Neighbors Regression algorithm was used. To be able to define the most important features, *permutation_importance* was used to identify the importance mean for each feature resulting in the graph on the right.

After various permutations the highest score was achieved using the features *'price', 'body_weight(g)' and 'battery(mAh)'* with a root mean squared error of 0.4459 for the training dataset and 0.54897 for the test dataset.

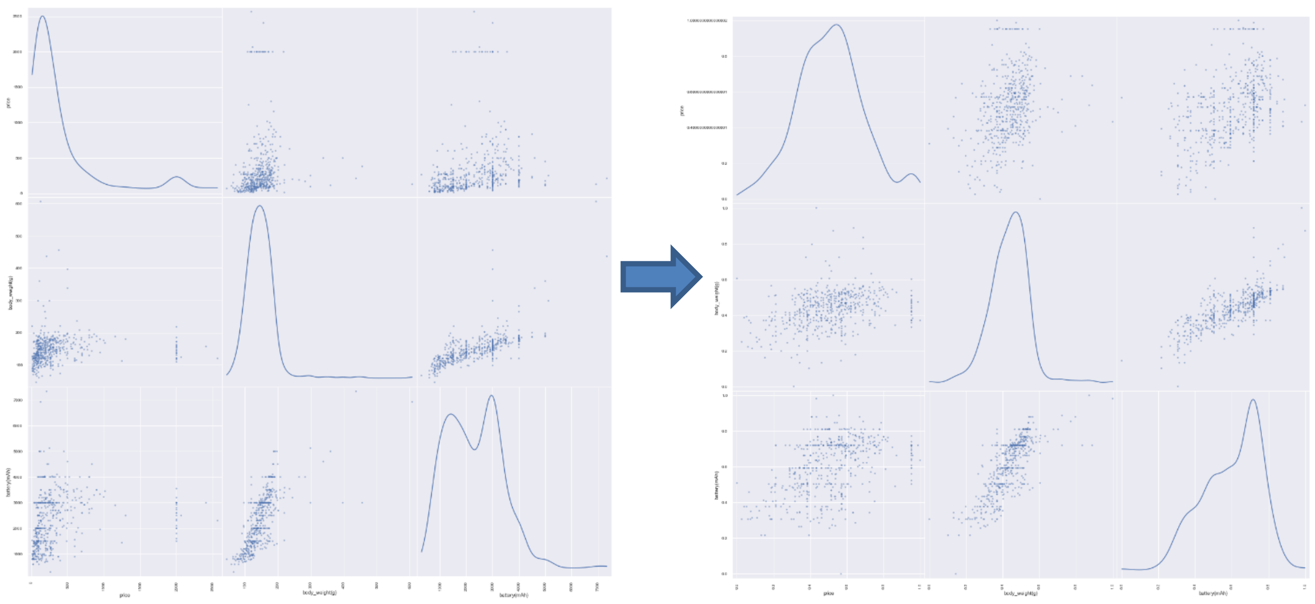
Refinement

To be able to optimize the model used, three techniques will be applied to achieve the least error:

1. Re-scale the features using logarithmic transformation and MinMaxScaler
2. Apply GridSearch to find the best parameters for the model
3. Push KNN to its maximum performance using bagging (BaggingRegressor)

The re-scaling and logarithmic transformation resulted in the transformation of the features to more normally-distributed plots as shown in the below figures that highly enhanced the training of the model.

As a next step, GridSearch was used to find the best parameters values for the chosen algorithms which is the KNN Regressor. The chosen parameters were *n_neighbors* and *weights*. The resulting parameter values after fitting the model was *{'n_neighbors': 38, 'weights': 'distance'}*.



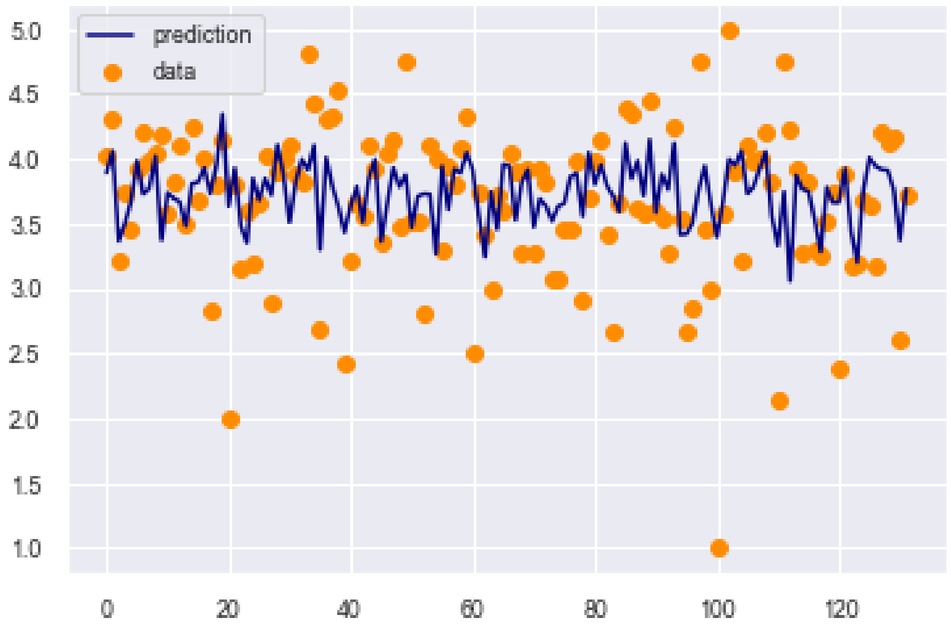
Finally, the Bagging Regressor was used to enhance the performance of the KNN Regressor with 100 estimators and combining all the past enhancements as shown in the below code.

```
# Apply the best parameters from gridsearch
best_k = gridsearch.best_params_["n_neighbors"]
best_weights = gridsearch.best_params_["weights"]
bagged_knn = KNeighborsRegressor(n_neighbors=best_k, weights=best_weights)

# Initialize the BaggingRegressor and find the root of the mean_squared_error
bagging_model = BaggingRegressor(bagged_knn, n_estimators=100)
bagging_model.fit(x_train_fitted, y_train_2)
predict_and_find_rmse(bagging_model,x_train_fitted, y_train_2)
predict_and_find_rmse(bagging_model,x_test_fitted, y_test_2)
```

Results

Model Evaluation and Validation

The final training root of mean squared error was 0.19 while the testing root of mean squared error of 0.542.



Plotting the resulting predictions vs. the actual rating values, the results are not very far from the actual ratings. However, the graph showed a lot of outliers that could be due to the low number of samples in some mobile phones or user-specific issues.

Justification

The nature of the data enforced some limitations in choosing the appropriate algorithm for the problem. The final processed dataset was small (nearly 700 rows) which is highly subjective to over-fitting in some algorithms and under-fitting in other algorithms. It showed as well no linear correlations with the rating column which also resulted in the exclusion of all linear regression algorithms.

The final algorithm which is the KNN Regressor performed well specially after refinements, however, could not solve the problems of biased user reviews due to other problems or the low samples that do not maybe cover all cases.

Conclusion

Reflection

The initial challenge was the data pre-processing since the accuracy of the data was not that good specially when trying to link it to the reference commercial name of the brands and mobile phones. It required a lot of processing and refinements producing very good and appropriate dataset for our problem. However, the datasets did not contain many recent data and also was small relative to any other dataset required for machine learning model training.

The other challenge was to find the most important features that can predict the mobiles rating without over-fitting or under-fitting the model. Choosing the appropriate regression model as well was very hard due to the data amount limitation; a lot of algorithms were tried out but with very low accuracy due to the fitting problems. The KNN Regressor proved to be the best one relative to the decision tree algorithms and the XGBoost algorithm.

As expected, the price, the battery life and the mobile body weight proved to have a great impact on the mobile phones popularity and the model proved a good accuracy predicting another phones' popularity.

Improvement

In this project, a great improvement will be a larger and more recent dataset from the current online platforms with various ratings from different users per mobile phone. This will help us generalize the model more and make sure of its correlations results. Another improvement could also be a much easier and more accurate reference for the current mobile phones features that can enhance our study to be able to make use of the results in the mobile manufacturing industries and their marketing.