Predicted Future Sales of a Brazilian E-Commerce Company, Olist

Hirunya Hirunsirisombut
Sojeong Yang
Ying Kam Chiu

A. Executive Summary

Olist is the one of the largest department stores in Brazil. It helps small businesses connect to customers using its online store and lets them ship directly ship to customers. There are many features that influence Olist's sales at different levels. In this case, Olist's revenue management and sales planning are important for our goal.

This project is concentrated on Olist data from kaggle.com, which were made at multiple marketplaces in Brazil, from 2016 to 2018. The three goals of this project are: 1) the relationship between location and delivery 2) discover which product category is the best-selling and 3) explore how sales management affects sales performance. We tested and trained the different types of parameters to improve the accuracy. In order to reach our goal, we applied the variety techniques to the dataset such as unsupervised methods (e.g., PCA), supervised methods (e.g., classification and regression), ensemble methods (e.g., Random Forest Classifier and AdaBoost Classifier), and several other auxiliary analytical techniques.

According to the results of those analyses, we discovered the following:

First, we built multiple models to explore the relationship order concentration and quantity by location and quantity by location. With all the available data, we selected 12-Nearest Neighbors Classifier to be our best model. The model has the highest accuracy of 63.61%. If the needed location variables and delivery variables are available, there will be a 63.61% chance that the model would classify the "delivery_grade" accurately. While the accuracy is not that high, we think that if we have more delivery case data that we could train the model to produce greater accuracy results.

Our second line of analysis was to find the best-selling product. Out of all the models that we built; we conclude that the Random Forest Classifier is the best model with an accuracy of 81%. In other words, the model would be able to classify 'product_category_name' fairly accurately.

Lastly, we selected AdaBoost Classifier as our final model to explain the relationship between sales management and sales range. The accuracy is 53%, which is fairly low. The probability of the model to predict sales range classes accurately is just barely above 53%. The accuracy is generally low in all the models that we have built. It is probably not because of the inappropriate techniques that we used. But the variables are not significant/ relevant enough to answer the question.

B. Main Report

1. Introduction

Olist is one of the largest department stores in Brazil. Olist connects small businesses from all over Brazil to channels without hassle and with a single contract. Those merchants are able to sell their products through the Olist Store and ship them directly to the customers using Olist logistics partners. The Brazilian E-commerce Public Dataset was sourced from Kaggle website. (https://www.kaggle.com/olistbr/brazilian-ecommerce?select=olist_customers_dataset.csv) The dataset has information of 100k orders from 2016 to 2018 made at multiple marketplaces in Brazil. Its features allow viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers.

This project is focused on Olist data from 2016 to 2018 made at multiple marketplaces in Brazil, drawn from kaggle.com, and contains 100k orders. The purpose of this project was threefold: 1) the relationship between location and delivery 2) discover which product category name is the best-selling and 3) explore how sales management affects sales performance. We wanted to focus on exploring the relationship between delivery performance and sales. However, there is not much information available and then we decided to change the direction a little.

While building models, we tested and tried a variety of parameters to improve the accuracy. To achieve this goal, we applied multiple techniques to this dataset, including basic exploratory analysis of the variables, unsupervised methods (e.g., PCA), supervised methods (e.g., classification and regression), ensemble methods (e.g., Random Forest Classifier and AdaBoost Classifier), and several other auxiliary analytical techniques.

In order to answer these questions, we have preprocessed the data by dropping missing values and merged different files for usual attributes. For 1), we merged customers, orders and items, for 2) we merged items, product, translation and review and for 3) we merged customers, orders, items and reviews. To explore how sales management affects sales performance, there is no usual attribute to answer the questions directly. We had to tweak the data, such as grouping by seller Next, we convert the date attributes to get the average delivery days (avg_del_days), average estimated delivery days (avg_est_del_days) and average response days (avg_resp_days) and take the average of these date attributes and unit per transaction, freight values, review score. These steps are taken on excel spreadsheet before importing to jupyter notebook.

2. Exploratory Analysis

2.1 Order concentration and quantity by location

Assuming we are the CEO of Olist Store, when we look at the geolocation and customer-datasets, we would think about what questions we would ask and some of the possible the answers might look to get. E-commerce is happening all over Brazil, but we thought that you would want to know about which regions where e-commerce is most concentrated and the frequency of usage using e-commerce by region. Then you will be able to make different marketing plans for each region accordingly. In order to analyze the sales volume by region, we analyzed the data combined with customer-data, geolocation-data, and item-data. After we merged the three datasets, we mapped the distribution based on 'order-id' on a map (*Figure 2.1.1*). After that, with the combined data, we identified the top 10 cities with the highest population of customer between 2016 and 2018 (*Figure 2.1.2*). Also, we visualized the order counts by location (cities) between 2016 and 2018 (*Figure 2.1.3*, *Figure 2.1.4*)

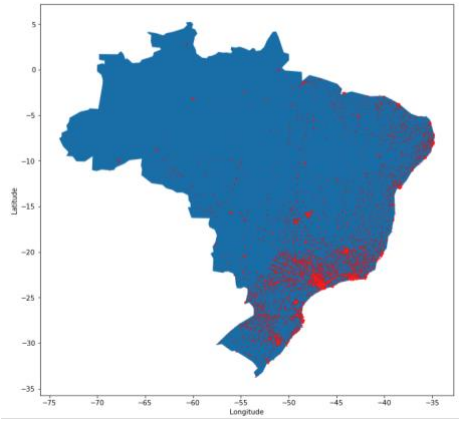

Figure 2.1.1 Order Distribution

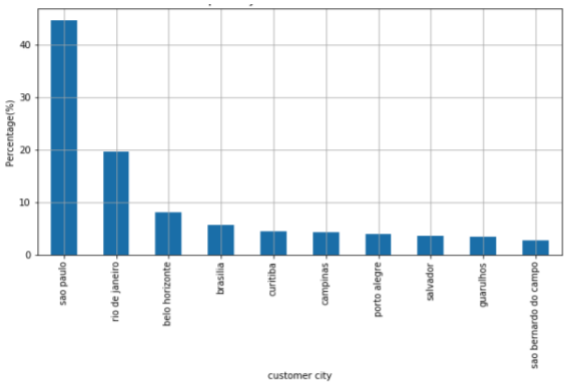

Figure 2.1.2 Top10 City Distribution of Customers

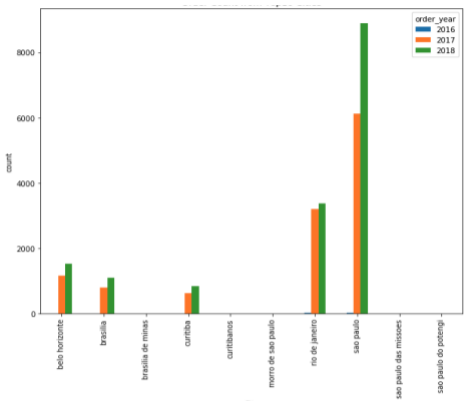

Figure 2.1.3 Order Count between 2016 and 2018

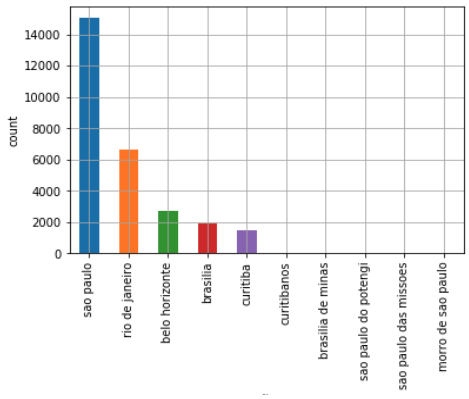

Figure 2.1.4 Order Distribution of Cities between 2016 and 2018

2.2 The best-selling of product categories year by year

The main idea is finding which product category name has got the most purchase. After the process of data cleaning, count the value of the category and find the max of the value. The result shows that “bed_bath_table” is the best-selling product category. In addition, from the analysis process shows (*Figure 2.2.1)* that in 2026 the product category name ‘furniture_decor’ is the best-selling, in 2017 the product category name (*Table 2.2.2*) ‘bed_bath_table’ is the best-selling, and in 2018 the product category name ‘health_beauty is the best-selling.

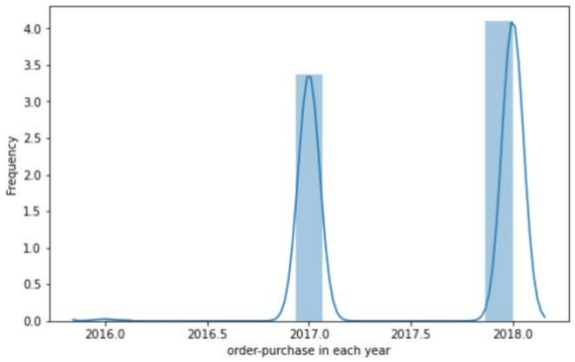

Figure 2.2.1 Order Purchase in each year

Purchase Year	Product Category Name
2016	furniture_decor
2017	bed_bath_table
2018	health_beauty

Table 2.2.2 Summary of Best-Selling Product Category Name

2.3 Sales Range Distribution

After segmenting sales by value ranges, we can see from the bar chart in figure2.3.1 and the histogram in figure 2.3.2 that class 0 has the highest percentage/ frequency, followed by 3, 4, 2 and 1.

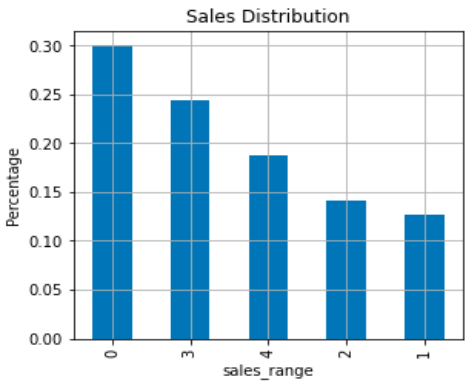

Figure 2.3.1 Sales Range Distribution Bar Chart

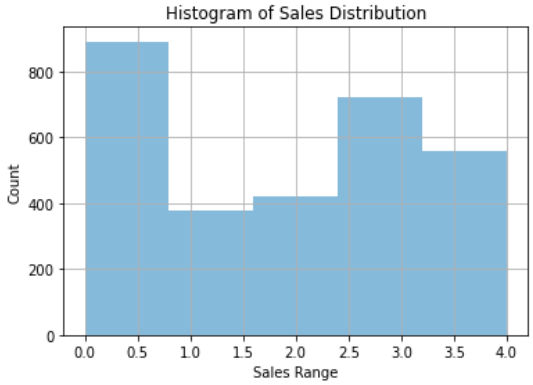

Figure 2.3.2 Sales Range Distribution Histogram

In figure 2.3.3, it shows the basic correlation analysis matrix among all the variables. In general, we do not see any collinearity among the independent variables. Notably, 'avg_est_del_days' and 'avg_del_days' have a fairly strong and positive relationship while 'avg_del_days' and 'review_score' have a fairly weak and negative relationship.

Basic Correlation Analysis Matrix:

	avg_del_days	avg_est_del_days	upt	freight_value	review_score	avg_resp_days	sales_range
avg_del_days	1.000000	0.456719	-0.019380	0.183198	-0.377852	0.060785	0.084747
avg_est_del_days	0.456719	1.000000	-0.021489	0.163968	-0.097259	0.067550	0.063704
upt	-0.019380	-0.021489	1.000000	-0.071512	-0.153831	-0.018975	0.045628
freight_value	0.183198	0.163968	-0.071512	1.000000	-0.016073	-0.022546	0.191796
review_score	-0.377852	-0.097259	-0.153831	-0.016073	1.000000	-0.016369	-0.042841
avg_resp_days	0.060785	0.067550	-0.018975	-0.022546	-0.016369	1.000000	-0.016089
sales_range	0.084747	0.063704	0.045628	0.191796	-0.042841	-0.016089	1.000000

Figure 2.3.3 Ridge Regression (Basic Correlation Analysis)

In exploring the relationship between sales ranges and individual variables, we can use boxplots in figure 2.3.4-8. In general, we see that the value ranges of the middle 50%(boxes) are small and many outliers are beyond the whiskers.

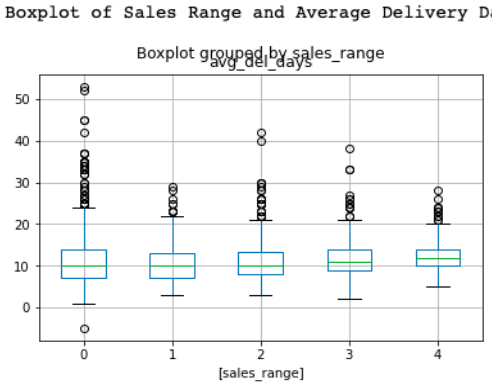

Figure 2.3.4 Boxplot of Sales Range and Average Delivery Days

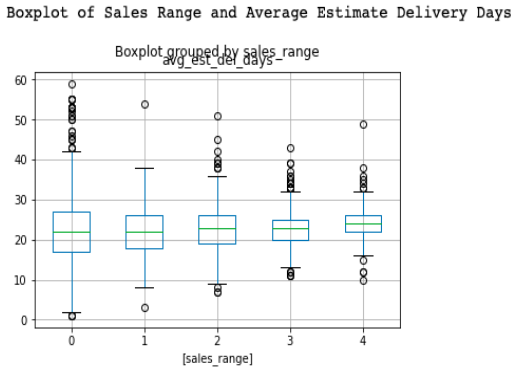

Figure 2.3.5 Boxplot of Sales Range and Average Estimated Delivery Days

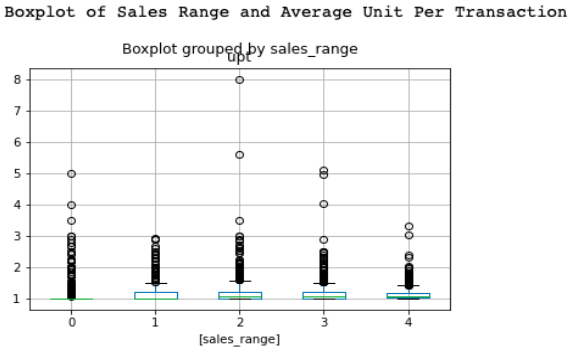

Figure 2.3.6 Boxplot of Sales Range and Average Unit Per Transaction

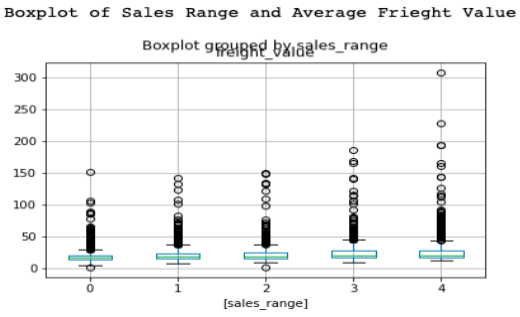

Figure 2.3.7 Boxplot of Sales Range and Average Freight Values

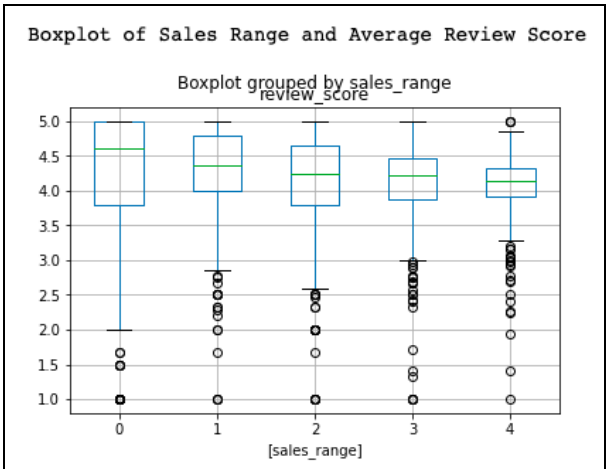

Figure 2.3.8 Boxplot of Sales Range and Average Review Scores

3. Relationship Between Location and Delivery

After basic statistical analysis, to determine the relationship between delivery and regions, we used various data-analysis tasks. For unsupervised approaches, we used PCA and clustering to explore different delivery times by region. For supervised approaches, we used K-nearest neighbor classification and regression. Also, we used ensemble method such as Random Forest. The modified data set after cleaning the data was grouped into each customers' city (customer_city). Each row consists of region, delivery time, latitude, longitude, zip code and total value of customer orders. We created a target attribute delivery_grade' based on delivery days (delivery_days), and it consisted of between 1 and 5 grades. The number of 'delivery_grade' is 1 means that delivery days between 1day and 7 days after order the items and the number of 'delivery_grade' is 2 means that the items delivered between 8 days and 15 days and so on. We create an 80% randomized training set 20% randomized testing sets.

3.1.1 Application of KNN classification

The KNN algorithm is a very intuitive algorithm. You can think of it as considering K nearest neighbors in applying a classification or regression problem. To define close neighbors, you first need to define a distance scale. The most commonly used distance measure is 'Euclidean distance', which is the linear distance between two observations. There are other ways to measure the distance, for example, 'Cosine similarity' and 'Manhattan distance'. Here, we used the tool from 'scikit-learn' machine learning package for python. This tool uses 'Minkowski distance' to measure the distance and variable 'weight' has 'uniform' and 'distance' options (default is 'uniform'). We tried to find the best value of k in the range k=1 to k=20 in increments of 1 with/without weighting.

3.1.2 Diagnostics (KNN classification)

Using the KNN Classifier tool from ‘scikit-learn’ with weighting, the best k was 12 and we tried this number to predict. We got the accuracy rate of the test, 63.61% and the accuracy rate of training, 100%. Without distance weighting, we checked the test score, and we found the best test score at k=13. We tried this number to predict and we got the accuracy rate of the test, 62.18% and the accuracy rate of training, 63.61%.

3.1.3 Analysis of the result and discussion (KNN classification)

According to experiment, with weighting showed the high accuracy rate of training but it didn’t show the similar rate on the test. It seems that hyperparameters affect our result. When we ran it with distance weighting, we got the train score: 1.0 and the test score: 0.637. It gets 100% accuracy on the train data, but it gets worse accuracy on the test data. We can say that this is 'overfitting' in this case. On the other hand, when we ran it without distance weighting, we got the train score: 0.622 and the test score: 0.636. The gap of the training and the test accuracy is less than with distance weighting. Therefore, we should experiment with different hyperparameters to improve our accuracy and check the 'overfitting' and 'underfitting'.

3.2.1 Application of Standard Linear Regression, Ridge Regression and Lasso Regression

The main purpose of machine learning is to create a model based on real data and predict the output that will occur if other input values are entered. At this point, the most intuitive and simple model we can find is the line. So, the method of analyzing data and finding a line that best describes it is called linear regression analysis. To reduce the overfitting and improve the linear model, we used regularization, Ridge and Lasso.

3.2.2 Diagnostics (Standard Linear Regression, Ridge Regression and Lasso Regression)

In the experiment, we used the ‘featureSelection ()’ function to find the optimal percentile of feature. We got the optimal percentile of feature at 51and optimal percentile was 0.58. We apply the optimal percentile feature for feature selection and train the model. We got the value of the mean absolute error, 0.537 on the test. After that, we discovered the best value of alpha for regularization. We got the best value of alpha for both Ridge and Lasso at 0.01. Using these two alphas, we built our model. We got the value of the mean absolute error, 0.5362 for Ridge and 0.5363 for Lasso on the test.

3.2.3 Analysis of the result and discussion (Standard Linear Regression, Ridge and Lasso)

The value of mean absolute error reduced from the result of Standard Linear Regression, 0.537 to 0.536 after Ridge and Lasso regularization. In theory, Ridge leads to low variance and low bias, but the Ridge regression decreases the model's complexity but doesn't reduce the variables' number because it only leads to minimized coefficient. If we increase the value of alpha, the coefficient decreases where the values reach to zero but not to zero. Compared to Ridge, Lasso tends to make the coefficient to absolute zero by increasing the value of alpha and it might lead to loss of information resulting in higher error in our model. In the graph below of Lasso (Figure 3.2.3.2), the lines of training and CV show very similar patterns. We can see that the value of alpha, 0.01 we got the minimum error, 0.058308 and it sticks with a similar result after 0.01. Also, Lasso reduces the degree of overfitting.

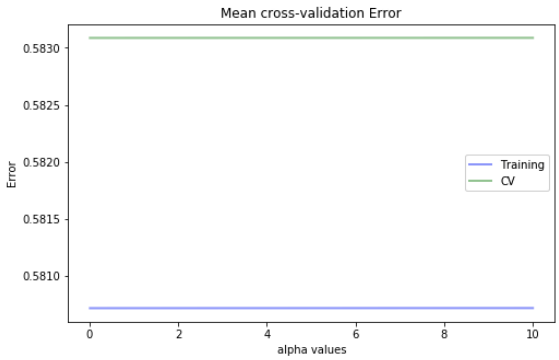

Figure 3.2.3.1 Ridge

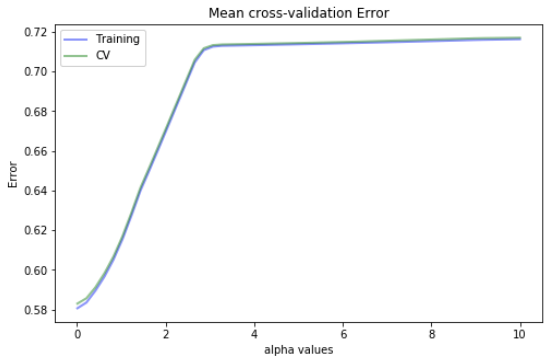

Figure 3.2.3.2 Lasso

3.3.1 Application of K-means Clustering and Principal Component Analysis (PCA)

As a type of unsupervised learning, similar data are collected and classified into clustered classes only by the characteristics of the data. The K-means algorithm is an algorithm that uses the distance between data to group data in close distance into one class. PCA is a technique that reduces high-dimensional data to low-dimensional data. The PCA technique can reduce noise. You can reduce the uncertainty by reducing the noise. By reducing the dimensions, you can save the amount of memory you use.

3.3.2 Diagnostics (K-means Clustering and Principal Component Analysis (PCA))

In this experiment, we used a tool called ‘KMeans’ from ‘scikit-learn’. Initially, we randomly assigned objects to create 5 partitions and then, we computed the centroid of the clusters of the 5 partitions. We got the value of completeness, 0.088 and the value of homogeneity, 0.111. To visualize our clusters efficiently, we used ‘silhouette_samples’ from ‘scikit-learn’ and drew the

graph (Figure 3.3.3.1). Next, we performed PCA on the normalized dataset. We calculated eigenvectors and eigenvalues of the covariance matrix. After we clustered, we got the value of completeness, 0.087 and the value of homogeneity, 0.111.

3.3.3 Analysis of the result and discussion (K-means Clustering and PCA)

According to the results from the experiments, using the KMeans clustering, we got the value of silhouettes mean, 0.296. Using the PCA, we got the value of silhouettes mean, 0.313. We can check the graph below (Figure 3.3.3.1 and Figure 3.3.3.2), it showed similar volume between them. According to the result in above, Completeness and Homogeneity from the full data and reduced Data shows similar and didn't see the big differences. Therefore, if we analyze big amounts of data, it's better to use the reduced data to improve the computational speed.

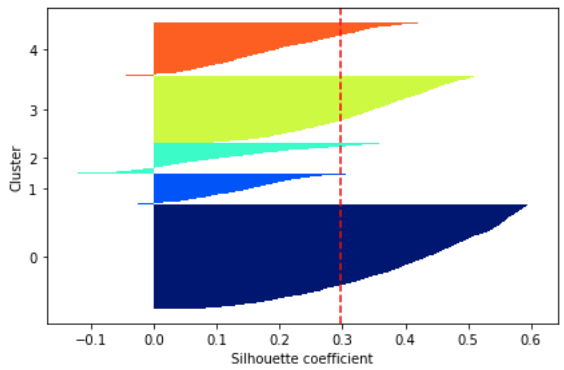

Figure 3.3.3.1 K-means Clustering

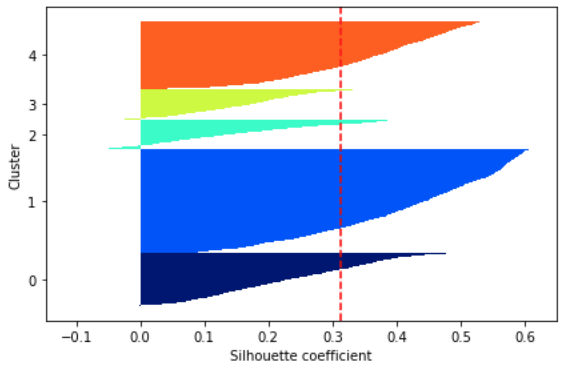

Figure 3.3.3.2 PCA

3.4.1 Application of Random Forest, Decision Tree

The Decision tree is a popular tool for classification and prediction, and it structured like a tree. Each internal node denotes a test on an attribute, each branch represents results of the test, and each leaf node holds a label. Random Forest uses an ensemble algorithm based on a decision tree that finds the optimal result by making several classifiers with the same algorithm and different training data. Features are learned by finding the best feature among feature candidates already randomly selected rather than finding the best feature when dividing the nodes of the tree. It is quite efficient on large data sets because it can be parallelized. Also, Random Forest is designed to solve the limitation of decision tree. We explored to find the optimal value for important parameters, ‘n_estimators’, ‘min_samples_leaf’ and ‘max_depth’.

3.4.2 Diagnostics (Random Forest, Decision Tree)

In our experiment, we used a tool 'DecisionTreeClassifier ()' from 'scikit-learn' and the accuracy value of Decision Tree was 0.491. On the other hand, the value of accuracy using random forest was 0.603. We found the best value 20 for 'min_samples_leaf' parameter in range 1 to 20. The optimal value for 'max_depth', we got the value 2 in range 1 to 10. Lastly, we got the best value for 'n_estimators' at 70 in range 5 to 100. We used these 3 parameters for the model to train and we got the value of accuracy, 0.61 on test. We used the default value for 'max_features' parameter for our model.

3.4.3 Analysis of the result and discussion (Random Forest, Decision Tree)

Random Forest has the benefit of increasing performance when applied to decision trees that are prone to overfitting. In our experiment, what we expected, we got the value of accuracy, 0.610 and it is higher than the accuracy value of Decision Tree (accuracy 0.491). However, Random Forest had a lot of parameters to adjust like 'min_samples_leaf', 'n_estimators' and 'max_depth'. The initial model of Random Forest, we got the value of accuracy, 0.603 but we got the value of accuracy 0.610 after the hyperparameter tuning.

4. Discover which product category name is the Best-selling

According to the data we would like to find which product category name has the best-selling. In this case, we will focus on the product data, purchase timestamp, and item data. In order to analyze the data to perform cross tabulation, and in order to perform regression, classification, PCA for reduced Dimensionality in clustering and random forest. The data should be merged and for the purchase timestamp, we separated it in terms of year, day, month for more detail between purchase and product category. Therefore, the data contains 111023 rows that have details of product and purchase time includes specific years, which are 2016, 2017 and 2018. In the process, the data was split with 80% of training and 20% for testing.

4.1.1 Application for Regression

To perform the standard linear regression using the closed form solution implementation in order to compute RMSE on the full training data and perform 10-fold cross-validation and compare the value of RMSE on training with cross-validation RMSE. Furthermore, we use scikit-learn regression model from sklearn.linear_model with a subset of features to perform linear regression and using k-fold-cross-validation in the training data with k=5 to find the most informative variable. At the final step, we use the optimal percentile of features to find the RMSE on testing data.

To perform Ridge regression and Lasso Regression using modules from sklearn.linear_model in order to compare the RMSE on testing that use Ridge regression and the RMSE on testing that use Lasso Regression.

4.1.2 Diagnostics for Regression

In the performance of using the standard linear regression that using the closed form solution implementation in order to compute RMSE on the full training data the result shows that RMSE on Training is 9.84608356, and after performed 10-fold cross-validation the result show fold-1 to fold-10 with 9.8469.

In the performance of using scikit-learn regression model from a subset of features to perform linear regression, then we got the result of the optimal percentile of the features is 91 and the optimal number of the feature is 8. Using the number of the optimal percentile which is 91 to train the model on the full 80% on the training data and evaluate it using the set-aside 20% test partition, then the RMSE on testing data is 9.861479445959116.

In the performance of Ridge regression and Lasso Regression using modules from sklearn.linear_model. The Ridge regression shows the results in RMSE on testing is 9.860387795708471 and The Lasso Regression shows the results in RMSE on testing is 9.860380713047629.

4.1.3 Analysis of the result and discussion for Regression

According to the regression that using the number of the optimal percentile which is 91 to train model on the full 80% on the training data and evaluate it using the set-aside 20% test partition, then the RMSE on testing data is 9.861479445959116. Compared with Ridge regression which is 9.860387795708471 and Lasso regression is 9.860380713047629. These values show that it has almost the same number, but using percentiles got slightly a little bit higher than regularization regressions. According to the (*Figure 4.1.3.1)* show the plot between the mean absolute error on CV and the values. The plot has the highest value on mean absolute error at the lower value and getting lower when the value is getting big. In the graph below of Lasso (*Figure 4.1.3.3*), the lines of training and cross-val show similar patterns.

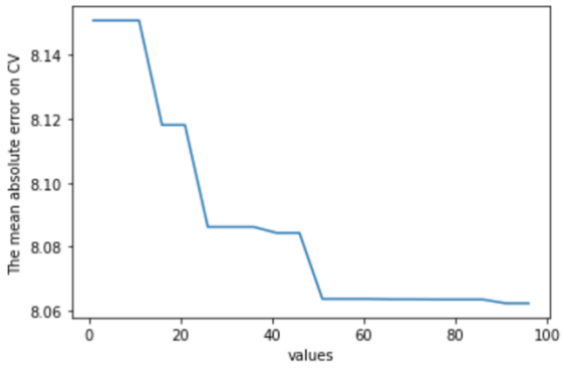

Figure 4.1.3.1 Linear Regression mean absolute error on CV

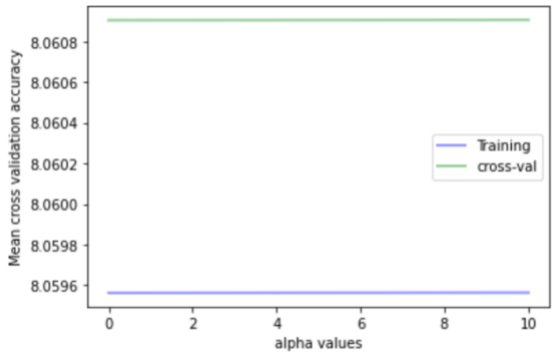

Figure 4.1.3.2 Ridge Regression

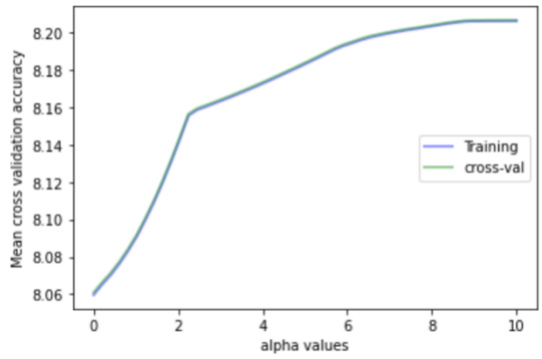

Figure 4.1.3.3 Lasso Regression

4.2.1 Application for KNN classifier, Scikit-learn's decision tree

To perform the classification using scikit-learn's KNN classifier. We normalize the data for all the attributes that will be the same scale (between 0 and 1). Then, we randomly separated training and testing for 80:20 respectively. We used a number of neighbors = 10 for the KNN classifier. In this case, the test data is the product category name. Then, experiment with different values of K and the weight parameter. Comparing with non-normalize training and testing that perform classification using scikit-learn's decision trees and Compare the average accuracy score on the test and the training data sets

4.2.2 Diagnostics for KNN classifier, Scikit-learn's decision trees

In the performance using KNN classifier. After the process of normalization in order to make all of the attributes be the same scale. We use a number of neighbors 10 and the classification report shows that the accuracy of the product category name in weighted avg of the precision is 0.81,

recall is 0.81 and f1-score is 0.81. Then, we experiment with weight distance in different K ranges (1 – 11) the results show that in K-1is 0.8423328079261427 which is the highest score and slightly decreased until K-11 which is 0.8037829317721233. The classification report shows that the accuracy of the product category name in weighted avg of the precision is 0.80, recall is 0.80 and f1-score is 0.80.

Then, we experiment without weight distance in different K ranges (1 – 11) the results show that in K-1is 0.8423328079261427 which is the highest score and slightly decreased until K-11 which is 0.7072731366809277. The classification report shows that the accuracy of the product category name in weighted avg of the precision is 0.70, recall is 0.71 and f1-score is 0.70.

In the end, we experiment with scikit-learn's decision trees with the non-normalized data. The classification report shows that the accuracy of the product category name in weighted avg of the precision is 0.87, recall is 0.87 and f1-score is 0.87. Therefore, the average accuracy score on test data is 0.8655708173834722 and the average accuracy score on train data is 0.9999437051048211.

4.2.3 Analysis of the result and discussion for KNN classifier, Scikit-learn's decision trees

According to the experiment that used a KNN classifier and performed with weight distance in different K ranges (1 – 11) the results show that in K-1 is the highest score and slightly decreased until K-11. Comparing the weight distance in different K ranges (1 – 11) the results shows that in K-1 is the highest score and slightly decreased until K-11. It shows that the neighborhood did not take the points from other neighborhoods when K is large, since we got K1 for both with and without weight distance.

According to the experiment with scikit-learn's decision tree with the non-normalized data. The classification report that we got shows that the accuracy result is the best one compared with previous experiment results. Also, the average accuracy score on test data is 0.8655708173834722 and the average accuracy score on train data is 0.9999437051048211. It shows that the scikit-learn's decision tree has the testing value lower than training value which means the model performs very good on training data with high variance but low bias.

4.3.1 Application for PCA

In order to perform PCA for reduced Dimensionality in clustering and the Kmeans implementation in scikit-learn, perform clustering on the image data. We randomly separated data for training and testing in 80:20 respectively, and we normalized the data by using a minmax scaler. Then, we perform Kmeans to perform clustering with K = 6. Then, we evaluate the clustering and use Silhouette analysis on the clustering. In order to compare values, we

compute the overall mean of Silhouette value and then we compare 6 clusters to 6 classes by computing Completeness and Homogeneity values of the generated cluster.

After the Kmeans implementation in scikit-learn perform clustering on the image data. We would like to compare it with PCA for reduced Dimensionality in clustering. The processes are the same with Kmeans, but we need to analyze the principal components to determine the number r of the principal components in order to capture at least 95% of variance in the data. After we get the number r component then use this number to transform the data into a reduced dimension.

4.3.2 Diagnostics for PCA

In performance of the Kmeans implementation in scikit-learn perform clustering on the data the results show that the overall mean silhouettes (*Figure4.3.2.1*) are 0.24161738098425942, the completeness score is 0.024503756787998543, and the homogeneity score is 0.010632049248690894. After we used a decomposition module to perform PCA. Then, we got the number r of the principal components in order to capture at least 95% of variance in the data. The result shows that PC from 1 until 6 can capture 96% of the variance which is at PC 6 is up to 97.34%. Then, we perform PCA with 6 components as a feature that we got to transform data into reduced dimensions. The result shows that Silhouette has overall mean (*Figure4.3.2.2*) is 0.25394330819802063, the completeness score is 0.02447060904776455, and the homogeneity score is 0.010618304631575795.

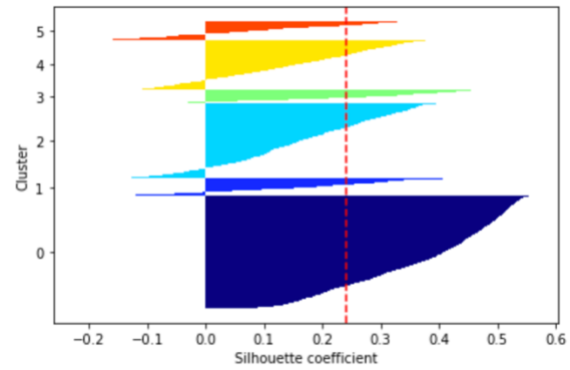

Figure 4.3.2.1 Silhouette Coefficient Using Kmeans

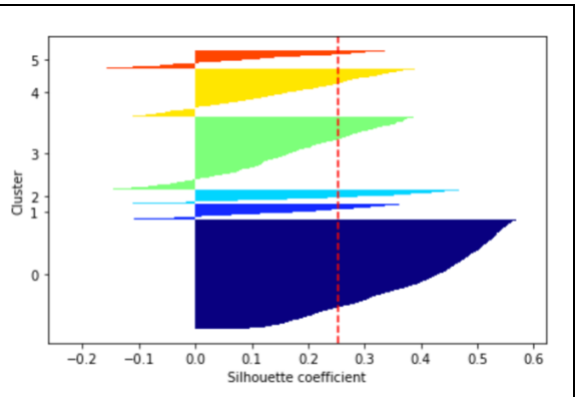

Figure 4.3.2.2 Silhouette Coefficient Using PCA

4.3.3 Analysis of the result and discussion for PCA

From the experiment on the Kmeans implementation in scikit-learn perform clustering on the data between PCA for reduced Dimensionality in clustering. The performance in PCA in overall mean of Silhouettes is almost same with the overall mean of Silhouettes that using Kmeans.

Also, the completeness score and the homogeneity score that using Kmeans and the completeness score and the homogeneity score that using PCA is almost same. It shows that after the reduce dimension in term of PCA, it still remains almost same value.

According to the (*figure 4.3.3.1)*, the plot of principal component number and percentage of the variance shows that at the lower PC number has the lower percentage of variance and it will be getting increased when the PC number is high. So, it will capture the number until it lasts of the principal component number.

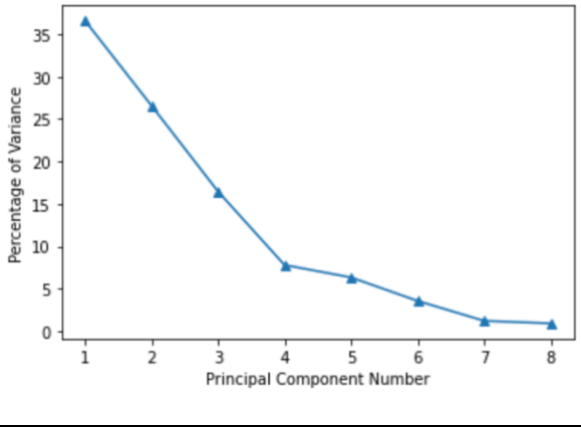

Figure 4.3.3.1 PCA Number

4.4.1 Application for Random forest

In order to perform ensemble classifier method Random Forest to compare with standard decision tree classifiers. We random separated training and testing for 80:20 respectively. Also, create a function in order to measure performance of the model. Then, we will get the accuracy for both classifiers.

After, we will investigate and compare model parameters. We use calculation parameter function (calc_params) to investigate the impact of particular parameters using cross-validation. Then, we will get the impact value of ‘min_samples_leaf’, ‘max depths’ and ‘N_estimators. These values will be used to perform the final model.

4.4.2 Diagnostics for Random forest

According to the experiment, at first, we used a standard decision tree classifier to compare with ensemble classifier method random forest. The result of the accuracy shows that, accuracy that using standard decision trees is 0.791 and accuracy that using random forest is 0.812.

Then, we investigate and compare the model parameters. The result of the impact of 'min_samples_leaf' the best one that works well is 1. The result of 'max depth' the best one is 20, and the result of 'n_estimator' the best one is 95. Therefore, the final model that uses 'min_samples_leaf' = 1, 'max depth' = 20, and 'n_estimator' = 95, has an accuracy of 0.810.

4.4.3 Analysis of the result and discussion for Random forest

From the experiment on the standard decision tree and random forest. The result of these values is obvious, which is the random forest is better than the standard decision tree. In order to perform the final model which will use the impact values. The result shows that after we pick the best one on the min_samples_leaf', 'max depth', and 'n_estimator'. The accuracy is 0.810. It can indicate that the random forest is more accurate than the standard decision tree classifier.

5. Explore How Sales Management Affect Sales Performance

In order to explore the relationship between sales management and sales performance, supervised methods of K-nearest-neighbor classification, regressions and AdaBoost Classifier are used. The transformed data set was grouped by seller id into 2970 rows. Each row is the sales management and sales performance per seller. Average delivery days (avg_del_days), average estimated delivery days (avg_est_del_days), unit per transaction (upt), freight value, review score, average response days (avg_resp_days) are the predictors while sales range is the target attribute. The whole dataset was split into training and test sets with 80:20 ratio.

5.1.1 Application of Nearest-neighbor Classification

In Nearest-neighbor Classification, the goal is to find the class label for each of the k neighbors, use a voting or weighted voting approach to determine the majority class among the neighbors and then assign the majority class label to the target attribute. Distance or similarity measure of Euclidean distance, cosine of the angle between vectors and cosine similarity with the inverse of seller frequency. The experiment is to find the best k value for the nearest neighbors with the most accurate measure. We will try a range of k values from 5 to 100 with an increment of 5.

5.1.2. Diagnostics for KNN Classification

KNN with Euclidean distance measure gives a range of accuracy rate from 39.56% of 15 nearest neighbors to 42.42% of 35 nearest neighbors. KNN with cosine of the angle between vectors (Cosine Similarity) gives a range of accuracy rate from 39.22% of 5 nearest neighbors to 43.77% of 20 nearest neighbors. According to the line graph in figure 5.1.2.1, we can see that Cosine Similarity has the highest accuracy at 20 nearest neighbors between the two methods

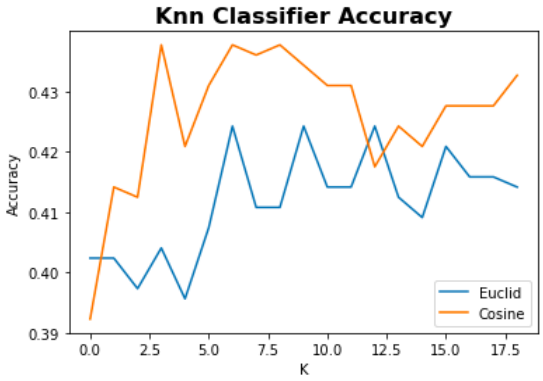

Figure 5.1.2.1 KNN Classification on Training Data

5.1.3. Analysis of the result and discussion for KNN Classification

Using the inverse of Cosine Similarity as a distance metric will give the most accurate results with the optimal k values of 20 nearest neighbors. We then use the obtained most optimal metrics to predict the classification of our test data. From figures 5.1.3.1 and 5.1.3.2, we can see that the initial classification on training data is very confused and basically it does not have clear classes. The classifier on test set is comparatively better at discriminating data points. The plot has shown that the classes are more segregated and a clear yellow class in the middle. In application, we can recommend if the sellers use this classifier to predict their sales range classes with all the required predators, the predicted class will be 43.77% accurate.

Figure 5.1.3.1 Cosine Similarity Classification on Training Data

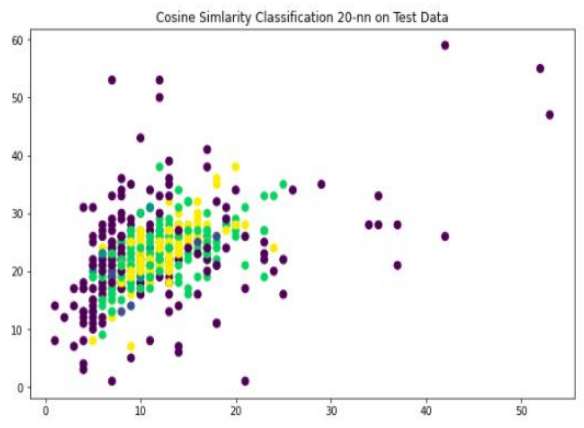

Figure 5.1.3.2 Cosine Similarity Classification on Test Data

5.2.1. Application of Standard Regression and Penalized Regressions

In this section, we experiment a variety of linear regressions: Standard, Ridge and Lasso, and Stochastic Gradient Descent for regression. In standard regression, we will try to find the most optimal percentile of features and the optimal number of features. After finding the most informative variables and their weights, we will build a predictive model based on standard regression. Next, we will find the most optimal alphas for both Lasso and Ridge regressions. Then, we will try to find the most optimal parameters: penalty, alpha and l1 ratio for elastic net penalty. In the end, we will evaluate the mean absolute error (MAE) on test data of each model.

5.2.2 Diagnostics for Standard Regression and Penalized Regressions

In the experiment, we find the optimal percentile of features is 21%. We apply the most percentile of features for feature selection. The most informative variables and their coefficients are average delivery dates of 32.21 and freight values of 96.48. Further we take these two selected variables to build a standard regression model. The MAE on test data is 1.396.
Next, we discover the optimal alpha for Ridge is 16.736326530612246 by comparing the smallest root mean square error (RMSE) with 10-fold cross validation on test and that for Lasso is 0.01. In figure 5.2.2.1 and 5.2.2.2, we can find the smallest Alphas as mentioned. With these two optimal alphas, we build two models respectively. The Ridge Regression MAE on test is 1.374. The Lasso Regression MAE on test is 1.375.

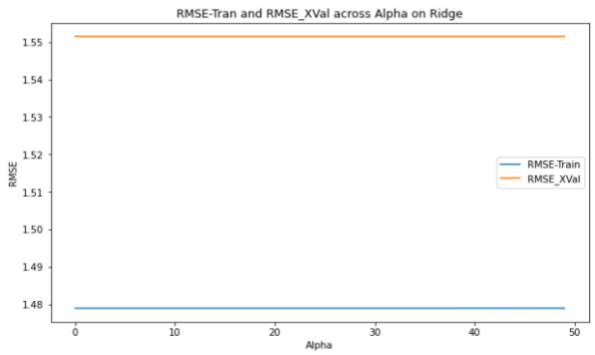

Figure 5.2.2.1 RMSE Train and RMSE Test across Alpha on Ridge

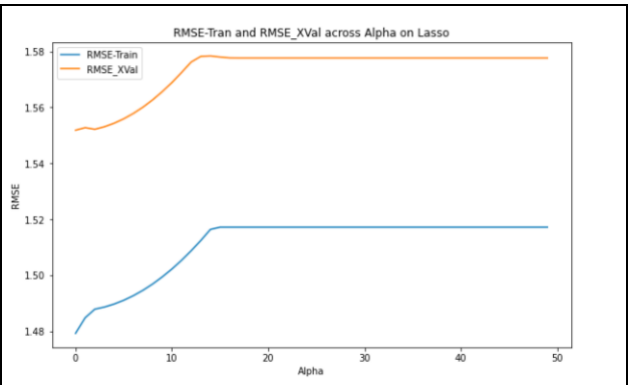

Figure 5.2.2.2 RMSE Train and RMSE Test across Alpha on Lasso

Then, we find that the optimal penalty is l1 and alpha is 0.0001 for Stochastic Gradient Descent by Grid Search Cross Validation. The SGD Regression MAE on test is 1.375. Lastly, we discover the optimal l1 ratio is 0.2631578947368421. Taking the optimal l1 ratio and alpha to Elastic Net Regression, The SGD Regression MAE on test is 1.377.

5.2.3. Analysis of the result and discussion for Standard Regression and Penalized Regressions

In table 5.2.3.1 has listed a summary of all MAE on test. Ridge model has the lowest error of 1.374. In fact, all the models are bad in terms of accuracy. Even if we select Ridge model, the r-squared score is 0.012 only. This means the model can only explain 1.2% of the variation in the response variable around its mean.

Method	MAE	R-Square
Standard	1.396	-0.009
Ridge	1.374	0.012
Lasso	1.375	0.012
SGD (penalty='l1', alpha=0.0001)	1.375	0.012
SGD (penalty='elasticnet', alpha=0.0001, l1_ratio=0.2631578947368421)	1.377	0.009

Table 5.2.3.1 Comparison of Standard, Ridge, Lasso and SGD of MAE and R-square

5.3.1. Application of AdaBoost Classifier

AdaBoost is one of ensemble boosting classifiers, which combines multiple classifiers to increase the accuracy of classifiers by an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers for high accuracy strong classifiers. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. We will try to find the optimal parameters: number of estimators and learning rate. Then we will apply the optimal parameters and build multiple models. In the end, we will evaluate the MAE on test data of each model and against regression models.

5.3.2. Diagnostics for AdaBoost Classifier

First, we run AdaBoost Classifier with a default number of 50 estimators and default learning rate of 1. The model will give an accuracy of 53% and MAE of 1.375. Then we try to find the optimal number of estimators in the ensemble by computer test scores from a range of estimators from 5 to 100. We found the optimal number of estimators is 25, in which it has the highest accuracy of 49.91%. With this parameter, the MAE on test is 1.375. Next, we have discovered

the optimal learning rate at 0.6 using a similar method. The accuracy is 52% and the MAE is 1.375.
In order to further optimize the parameters to build a model, we use Grid Search CV to find the best number of estimators and learning rate. The resulting parameters are 45 estimators and learning rate at 0.4. The accuracy is 49% and MAE is 1.375.

5.3.3. Analysis of the result and discussion for AdaBoost Classifier

The table 5.3.3.1 has listed a summary of all statistics on test of 20-NN Cosine Similarity, Ridge Regression model and multiple AdaBoost Classifiers. In general, AdaBoost Classifier performed a lot better than KNN Classifier and Regressions. AdaBoost Classifier with default number of estimators of 50 and learning rate at 1 gives the lowest MAE of 0.727, highest R-square of 40.2% and highest accuracy of 53%. It means that this AdaBoost Classifier can explain 40.2% of variations in the dataset and there would be a 53% chance to accurately classify the Sales Range classes.

Method	MAE	R-Square	Accuracy
20-NN Cosine Similarity	N/A	N/A	0.44
Ridge Regression	1.374	0.012	N/A
AdaBoost Classifier (default)	0.727	0.402	0.53
AdaBoost Classifier (n_estimators=25)	0.756	0.337	0.53
AdaBoost Classifier (learning_rate=0.6)	0.771	0.338	0.52
AdaBoost Classifier (n_estimators=45, learning_rate=0.4)	0.796	0.321	0.49

Table 5.3.3.1 Comparison of Ridge Regression, AdaBoost Classifiers of MAE, R-Square and Accuracy

6. Conclusion

As we mentioned, the Olist is the largest department store in Brazil and this is the dataset that we use for our entire project. We divided in three parts to focus on, which are “1) the relationship between location and delivery”, “Discover which product category is the best-selling”, and “Explore how sales management affects sales performance”

There are many techniques that we use in order to classify and analyze to find the answers for our scops. In the particular part, we indicate for the “the relationship between location and delivery” that for the first core research area, between delivery and location, we predicted the ‘delivery_grade’ using different models. For example, we used the models KneighborsClassifier, K-means clustering, PCA, Decision Tree and Random Forest. We got the most accuracy using KNN Classifier with Random Forest being next in accuracy. While the accuracy is not that high, we think that if we have more delivery case data that we could train the model to produce greater accuracy results.

For the “Discover which product category is the best-selling” indicates that using the Linear regression, Ridge, and Lasso, give us the similar result of the RMSE value for testing. The KNN classification is used in the part and the result shows that KNN with weight distance can give us better results compared with KNN without weighed distance. Also, PCA is used for reducing dimension. It shows that PC 1 until 6 can capture 95% of variance in the data, and the Silhouettes mean values still remain slightly the same value after using PCA. In addition, the Completeness score and Homogeneity score are almost the same. The last techniques that we use to perform in this part is Random forest. The result clearly shows that the random forest is better in terms of accuracy than the decision trees.

Our last line of analysis was to explore the relationship between sales management and sales range. We selected AdaBoost Classifier as our final model to explain the relationship. The accuracy is 53%, which is fairly low. The probability of the model to predict sales range classes accurately is just barely above 53%. The accuracy is generally low in all the models that we have built. It is probably not because of the inappropriate techniques that we used. But the variables are not significant/ relevant enough to answer the question.

C. Appendix

File Name	Attributes	Type
olist_customers_dataset	customer_id	numeric
	customer_unique_id	numeric
	customer_zip_code_prefix	numeric
	customer_city	categorical
	customer_state	categorical
olist_geolocation_dataset	geolocation_lat	categorical
	geolocation_lng	categorical
	geolocation_city	categorical
	geolocation_state	categorical
olist_order_items_dataset	order_id	numeric
	order_item_id	numeric
	product_id	numeric
	seller_id	numeric
	shipping_limit_date	date
	price	numeric
	freight_value	numeric
olist_order_payments_dataset	order_id	numeric
	payment_sequential	numeric
	payment_type	categorical
	payment_installments	numeric
	payment_value	numeric
olist_order_reviews_dataset	review_id	numeric
	order_id	numeric
	review_score	numeric
	review_comment_title	string
	review_comment_message	string
	review_creation_date	date
	review_answer_timestamp	date

Table 1 List of the raw data files

File Name	Attributes	Type
olist_orders_dataset	order_id	numeric
	customer_id	numeric
	order_status	categorical
	order_purchase_timestamp	date
	order_approved_at	date
	order_delivered_carrier_date	date
	order_delivered_customer_date	date
	order_estimated_delivery_date	date
olist_products_dataset	product_id	numeric
	product_category_name	categorical
	product_name_lenght	numeric
	product_description_lenght	numeric
	product_photos_qty	numeric
	product_weight_g	numeric
	product_length_cm	numeric
	product_height_cm	numeric
	product_width_cm	numeric
olist_sellers_dataset	seller_id	numeric
	seller_zip_code_prefix	numeric
	seller_city	categorical
	seller_state	categorical
product_category_name_translation	product_category_name	categorical
	product_category_name_english	categorical

Table 1-1 List of the raw data files