TDS 3301 Data Mining

Group Project

Dataset on Laundry

Group 13

Prepared by:

Student ID	Student Name	Email
1181103263	Chan Yun Hong	1181103263@student.mmu.edu.my
1181103320	Boe Chang Horn	1181103320@student.mmu.edu.my
1181103087	Tan Kai Yuan	1181103087@student.mmu.edu.my

Introduction

After taking a deep dive on understanding the dataset, we have decided to introduce two main questions to extract insight from the dataset to bring business value to the business stakeholders. The first question is **“What factors contribute to the change in the total money spent by customers?”**. This question is important as it can help the business stakeholders to focus on the factors that can increase the total money spent by customers, thereby uplifting the monthly revenue. The second question is **“What are the frequent types of customers who like to do laundry for blankets or clothes?”**. This question can help the business stakeholders to create targeted campaigns efficiently to target certain customer clusters based on the question’s answer.

Exploratory Data Analysis

To answer Question 1, we have included some of the important features such as the city, state, and postcode of the coordinate from the GeoPy library which utilized the OpenStreetMapAPI along with the weather data of the coordinate at the specific timestamp from the Visual Crossing’s Historical Weather Data API. Some city, state, and postcode data are found with missing values after the scrapping process, and being filled manually. After scrapping the mentioned features from external resources, we had merged the scraped data into a single dataframe. A few examples of the **extra data points** we added are ‘City', 'State', ‘Postcode’, 'conditions', ‘Temperature’, ‘windspeed’ and many more.

```
display(df[df["With_Kids"] == "yes"]
display(df[df["With_Kids"] == "no"][

# "With_Kids" conflicts with 'Kids_C
```

```
young       386
toddler     347
baby        336
no_kids     330
toddler     302
Name: Kids_Category, dtype: int64
```

```
no_kids     872
young       316
toddler     313
baby        297
toddler     289
Name: Kids_Category, dtype: int64
```

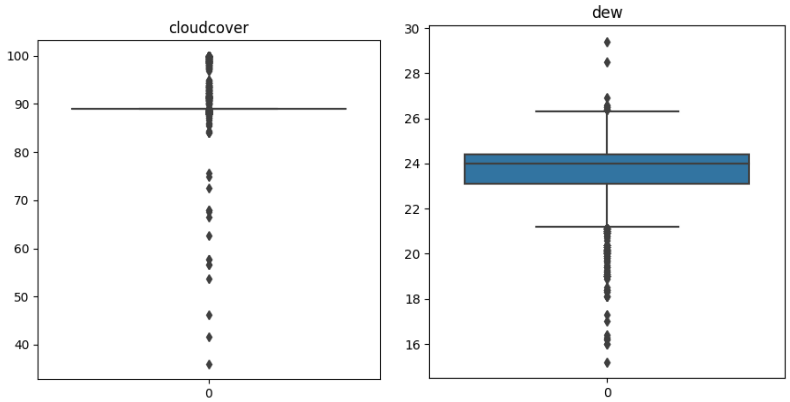
While looking at all the features from the dataset, we realized that the “With_Kids” feature is related to the feature of “Kids_Category”. We also noticed that there is a conflict of information with the dataset as the “Kids_Category” is not aligned with the “With_Kids” category. We decided to only keep the “Kids_Category” feature as it is more dominant than “With_Kids”. The “With_Kids” feature had also been utilized to fill in the missing value in “Kids_Catergory”.

Aside from this, we also found out that some of the values in the dataset come with a text format issue. There are some values being found with additional space which will make the computer treat it as a different value such as how “foreigner ” compares to “foreigner”. The data cleaning process has been conducted before continuing with the data transformation process.

During the **data transformation** process, we identified some of the features that could be transformed such as the “preciptype” will record the type of precipitation but in Malaysia, rain will be the only possible value. Therefore, the “preciptype” feature is transformed into “isRain”. The “DayOfWeek” and “HourInDay” features were extracted from the timestamp of each row to analyze when the customer visited the laundry.

Other than that, there are several features which contain a high amount of **missing values** such as “severerisk” and “windgust” from the weather dataset and also some of the unnecessary features such as the coordinate which had been utilized to scrape other important features were dropped. After dropping

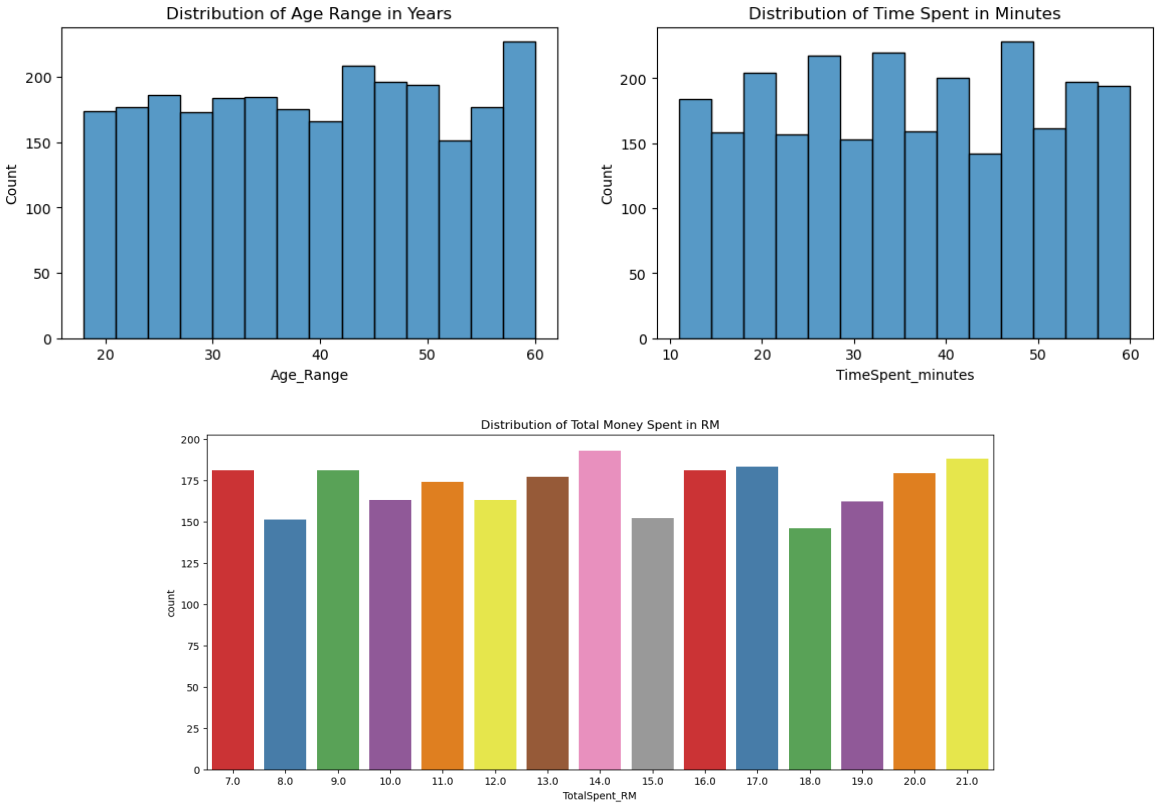
features with mentioned characteristics, rows with empty value were also dropped. The data left with 36 columns and 2574 entries.



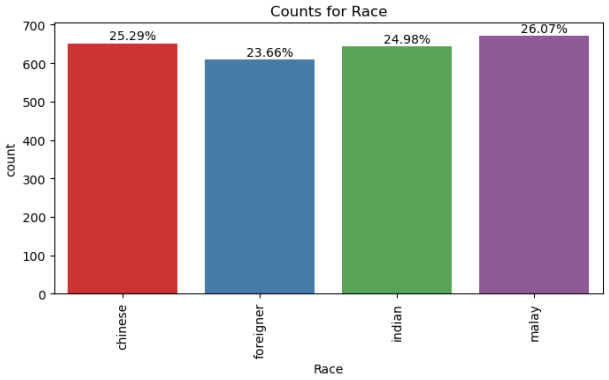
For the **outliers**, we use boxplots as they can quickly and readily display the distribution of the data and indicate any outliers that fall outside of the predicted range in order to discover outliers in a dataset. The figures above are some of the sample outputs of the boxplot. For handling outliers, we will drop the columns that have extremely high outliers, and use a machine learning model that is robust for dealing with outliers in the prediction task. The boxplots above are also a part of the **data visualization**.

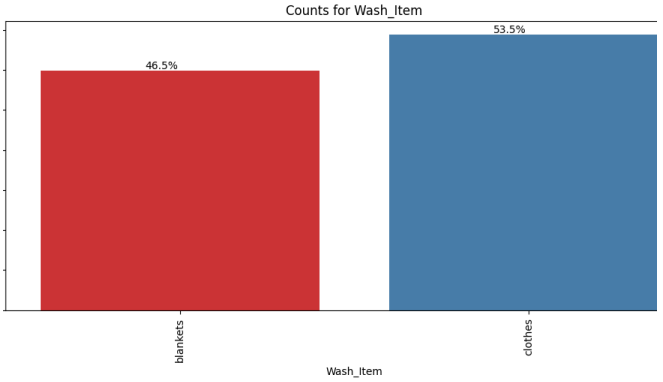
As the target variables for the two major questions, Total Money Spent and Wash Item, we draw barcharts for Univariate Analysis for Categorical and Numerical Data to further analyze the data. We can tell from the graphs that each column's general distribution is fairly balanced. As a result, we draw the conclusion that since the distribution of the data is fairly equal, **no data imbalance treatment is necessary**. Here are a few of the graphs that were produced to show how the data for the data variables were distributed.

Data Visualization for Univariate Analysis of Numerical Variables

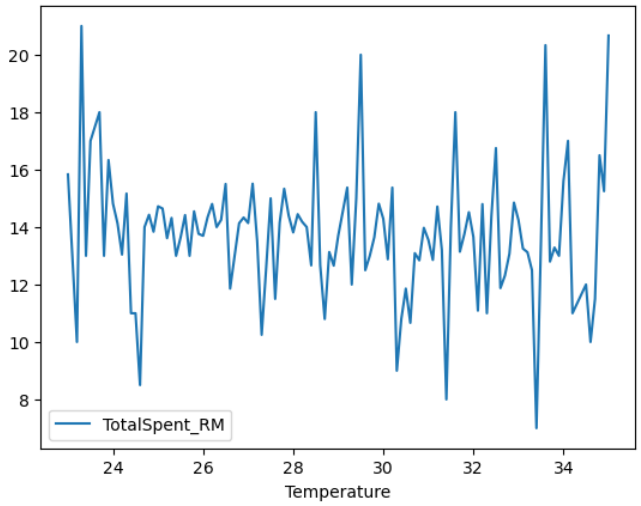


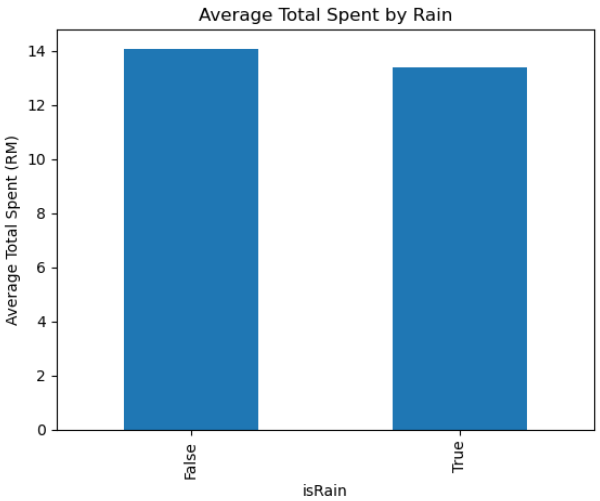
Data Visualization for Univariate Analysis of Categorical Variables





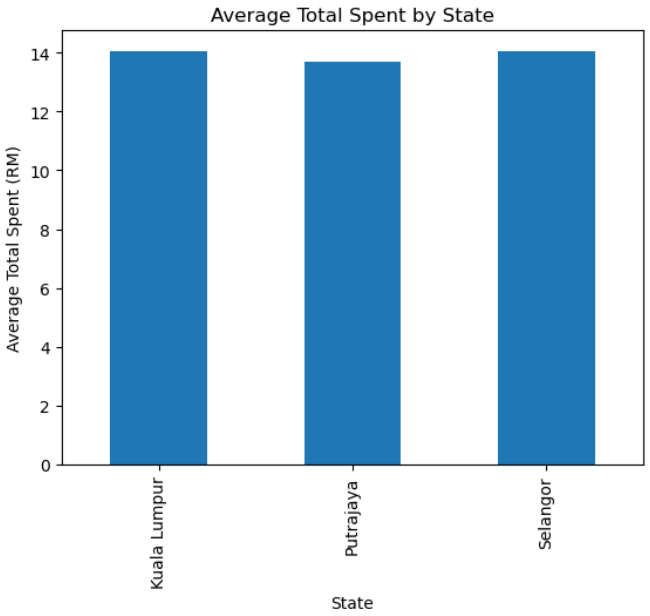
We formulate a few questions in order to better understand the **relationship between the variables**. The first question will be **"How does the weather affect the total money spent on average?"**. The question is important because it may aid the laundry shop in better understanding their customer preferences. To answer the question, we plot a line plot for the Temperature against the Total Spend, and also a barplot for the isRain variable against the average total spend of customers.





We may conclude that there is no significant difference found between Temperature and Total Spend based on the lineplot between those two variables, which shows that the line is fluctuating throughout the x-axis. Additionally, since the distribution of the isRain and Average Total Spend variables on the barplot is reasonably balanced, we can therefore conclude that there shouldn't be any significant difference found between the two variables.

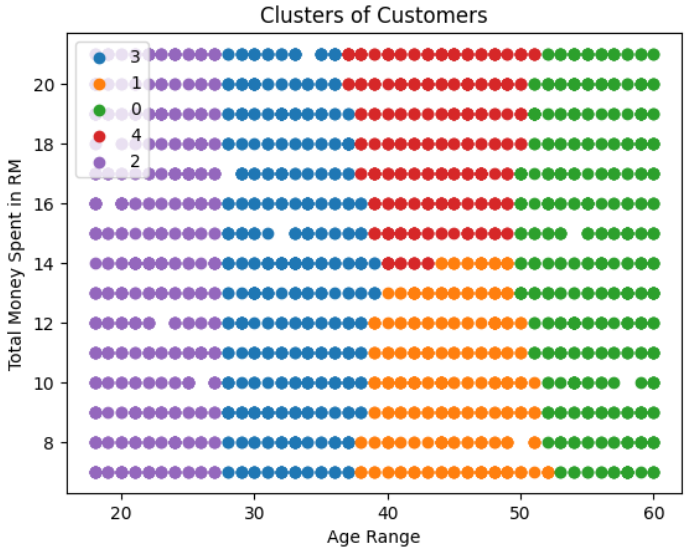
The second question is **"How does the location of customer affect the total money spent?"**. It is significant since the question could offer insightful information on customer location and in-store spending. To answer this question, we plot a barplot between the State and the average total spend of the customer to visualize its distribution.



We may assume that there is no significant difference found between the variable State and the Average Total Spend given that the distribution of the Average Total Spend for each of the States is fairly equal according to the barplot above.

Clustering

Next, we have chosen to perform customer segmentation based on age range by only including the age range and total money spent by the customers in the K-means clustering algorithm.



We have used the elbow method to determine the best k value for the clustering. Using the k value which is equal to 5, there is a red cluster which covers the high value of the y-axis, indicating that this customer cluster whose average age range is 43 tends to contribute more sales to the laundry shop. In contrast, there is an orange cluster which covers the low value of the y-axis, indicating that this customer cluster whose average age range is 45 tends to contribute fewer sales to the laundry shop.

Association Rule Mining

Utilizing Lift from ARM, we can determine the degree to which **customers' features** and the **type of wash item** they use are associated. This would help us respond to **Question 2**. We use the Apriori algorithm to the data after doing data preprocessing and data cleaning to identify the significant categorical features of the consumer. Using **`Lift(A -> B)`** , where A is the feature of customer (race,

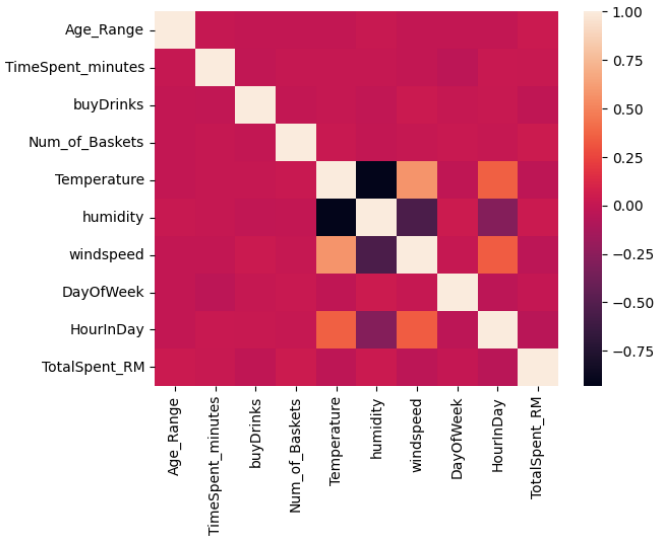
shirt_type and etc) and B is wash item (clothes or blankets), we could easily define what are the top features with the highest Confidence and Lift based on the output rules to support our claim. To display the output and provide a solution to the question, we print out each of the rules and choose the ones where B is either clothes or a blanket. Below are examples of the Top 5 Rules for All Customer Features and Top 5 Rules where B is Wash_Item.

Top 5 Rules of All Customer Features	Top 5 Rules where B is Wash_Item
(Rule 1) Putrajaya -> no_kids Support: 0.018 Confidence: 0.8679 Lift: 3.416 ========== (Rule 2) big_basket -> Putrajaya Support: 0.014 Confidence: 0.6981 Lift: 3.7281 ========== (Rule 3) casual -> Putrajaya Support: 0.006 Confidence: 0.283 Lift: 3.1266 ========== (Rule 4) Putrajaya -> clothes Support: 0.015 Confidence: 0.7358 Lift: 4.9713 ========== (Rule 5) Putrajaya -> fat Support: 0.005 Confidence: 0.2642 Lift: 3.6555 ==========	(Rule 1) Putrajaya -> clothes Support: 0.015 Confidence: 0.7358 Lift: 4.9713 ========== (Rule 2) Sepang -> clothes Support: 0.024 Confidence: 0.6562 Lift: 4.4336 ========== (Rule 3) no_specs -> blankets Support: 0.005 Confidence: 0.7222 Lift: 4.905 ========== (Rule 4) casual -> clothes Support: 0.007 Confidence: 0.2179 Lift: 3.4845 ========== (Rule 5) chinese -> clothes Support: 0.005 Confidence: 0.6 Lift: 4.1967 ==========

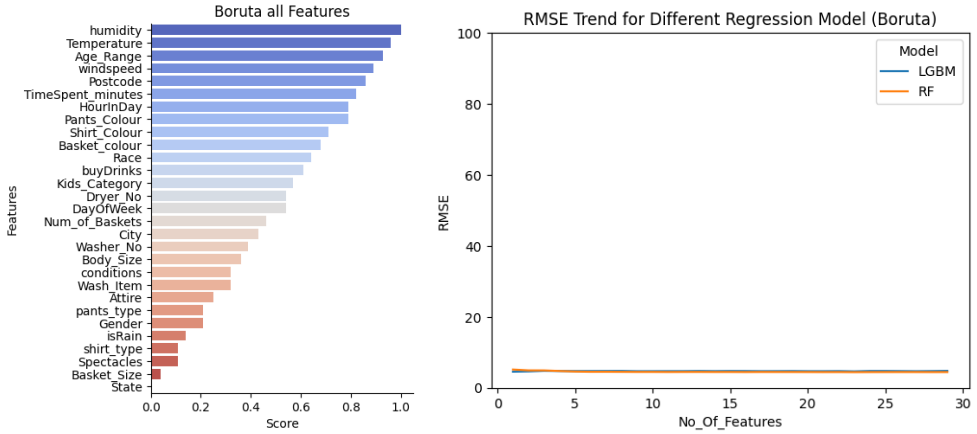
We need to look at the rules when B is Wash Item in order to respond to Question 2. Take rule 1 as an example. According to Support, Confidence, and Lift, 1.5% of Putrajaya customers do laundry for clothes, making them 4.9713 times more likely to do so than they would be if location and washing type were independent. The support, confidence, and lift of these rules imply that the linkages between the aspects of City, Attire, Spectacles, Race, and the sort of laundry they do are fairly strong. In summary, customers who don't wear glasses have the highest likelihood of washing blankets, according to the output, with a Lift value of 4.905, while those who live in Putrajaya have the highest chance of washing clothes with a Lift value of 4.9713.

Feature Selection

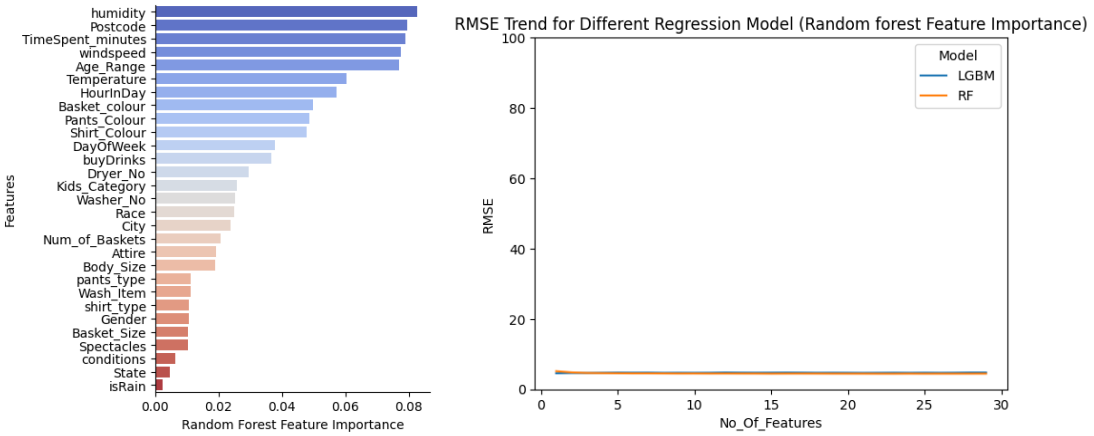
After doing exploratory data analysis, we start working on building a regression model to predict the total money spent by customers to answer Question 1. A regression model is used because the target variable is a numeric variable. Before feature selection, we **applied label encoding** on the dataset to ensure the data are in numeric value so that they can be fed into the machine learning model. For feature selection, we have chosen one method for each category of feature selection techniques which are **filter methods, wrapper methods and embedded methods**. For example, for filter methods, we have plotted the heatmap to show the **Pearson correlation** between the numeric variables to see whether there is a strong correlation between the dependent variable and independent variable but there isn't any as referred from the figure below.



For the wrapper method, we have used the **Boruta algorithm** to rank the features based on the returned score. In order to evaluate how the increase of features based on ranked order affects the model performance, we have also plotted the Root Mean Squared Error (RMSE) trend for different regression models when the number of features increases. As referred from the figure below, it seems like the increase of features based on ranked order determined by Boruta algorithm only brings very little difference to the model metrics, whereby indicates it is not useful for improving model performance.

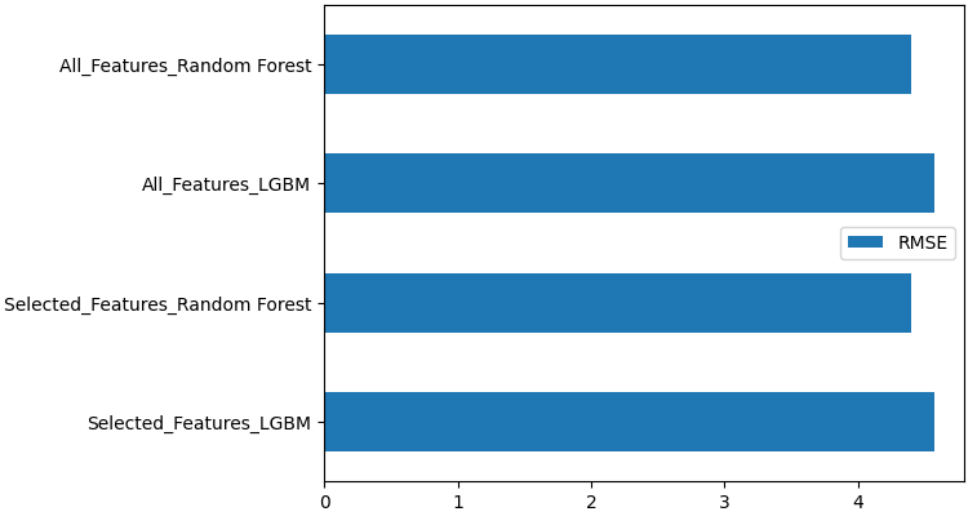


For embedded methods, **Random Forest feature importance** is also used to repeat the same process conducted when using Boruta algorithm. As referred from the figure below, it seems like the increase of features based on ranked order determined by Random Forest feature importance score only brings very little difference to the model metrics, which indicates it is not useful for improving model performance.



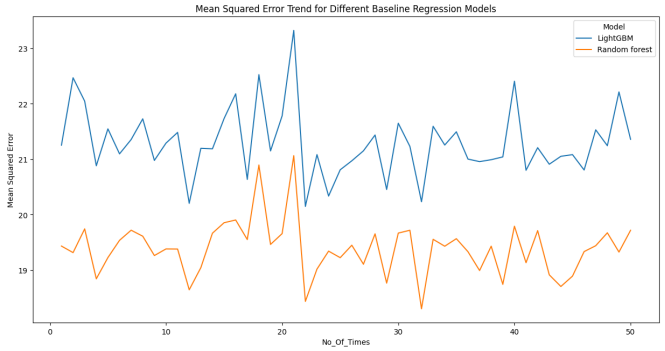
Next, we have **proposed a methodology to finalize the features** used for the regression model. First, we will get the set of features that have the best performance from both feature selection techniques

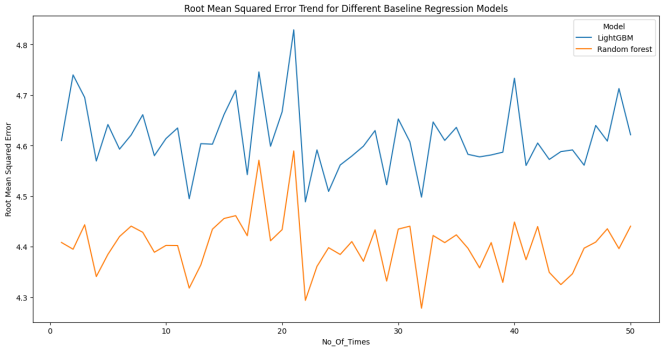
which are the Boruta algorithm and Random Forest feature importance, then retrieve the overlapped features of both sets of features as selected features. Lastly, we compare the performance between the model built using selected features based on the Boruta algorithm and Random Forest feature importance and the model built using all available features to determine the final set of features. As referred from the figure below, there is only extremely little difference between the RMSE of both models for each type of model. Since no significant difference is observed, we have decided to keep all the available features within model training to understand more about how each individual feature contributes to the model.



Model Construction and Comparison

For the regression problem, we have chosen **Random Forest** and **LightGBM** as the candidate models. Random Forest is chosen because it usually has decent performance, robust to overfitting that is faced by the decision tree algorithm while LightGBM is chosen because it has faster training speed and higher efficiency, lower memory usage and better accuracy. For **model validation**, we will perform a **train-test split** with test size which is equal to 0.3 to ensure the model is able to be evaluated on unseen data to see how well the model generalizes on unseen data. Also, we will use **k-fold cross-validation** for model comparison to **minimize sampling bias**.





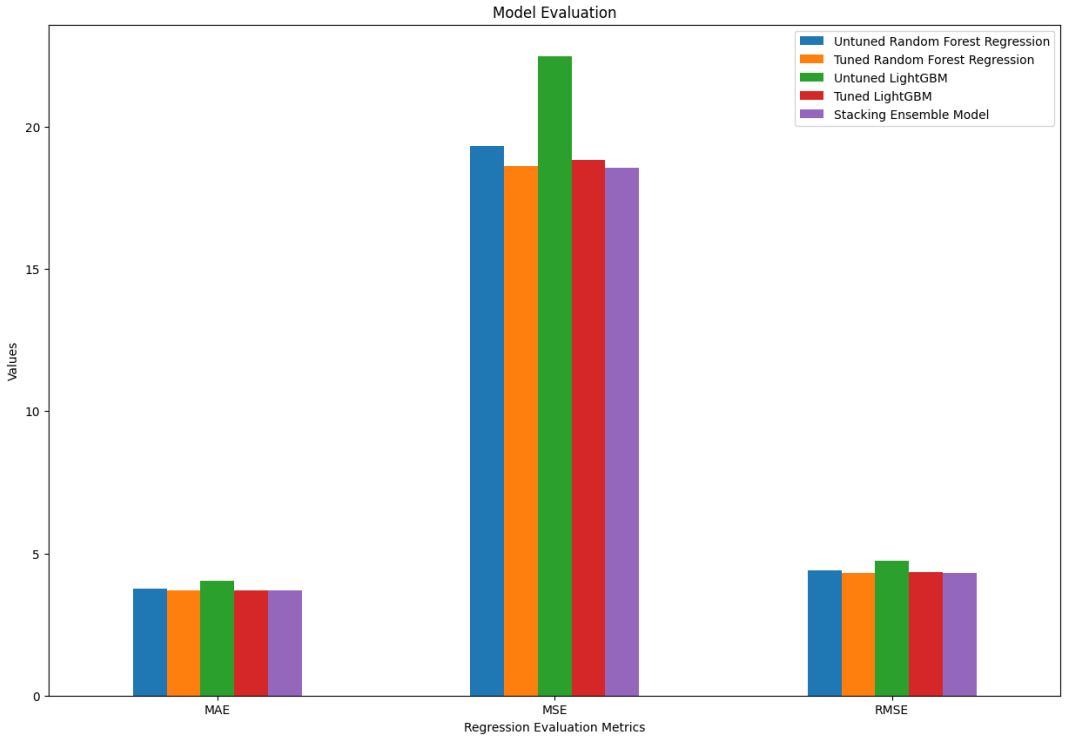
We **visualize the output of the model** by **plotting the trend of model metrics of baseline models in multiple iterations**. This is to also help us to get a basic understanding of how the model performance of baseline chosen models compare with each other. In this regression problem, the model metrics used will be Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). On average, all model metrics of the LightGBM model are higher than the Random Forest, which means Random Forest model is better than the LightGBM model.

Next, we **compare the difference between the RMSE of both models** using a statistical test called **Wilcoxon Test**. In this part, we have utilized k-fold cross-validation which k is equal to 15 to evaluate both models, resulting in 15 entries and each entry is a pair of model metrics from both models. These 15 entries will be used as samples for running the statistical test. According to the result of the statistical test, the p-value is lower than 0.05 which concludes that the difference between the RMSE of both models is statistically significant and the average RMSE of the Random Forest is lower so we can conclude that the Random Forest model is indeed better than the LightGBM model.

Next, we also would like to **understand what is the impact of SMOTE and non-SMOTE datasets on the model performance** and whether the difference is statistically significant. For the context, we have used the SMOTE library to create a new version of the dataset and **choose the Wilcoxon Test** as the statistical test to determine whether the difference is statistically significant, and also choose the baseline Random Forest model as the model to run this comparison. A single type of model is chosen as we would want to make sure everything other than the dataset type is constant. In this part, we have utilized k-fold cross-validation which k is equal to 15 to evaluate the model that uses SMOTE dataset and the model that uses the non-SMOTE dataset, resulting in 15 entries and each entry is a pair of model metrics from both models. These 15 entries will be used as samples for running the

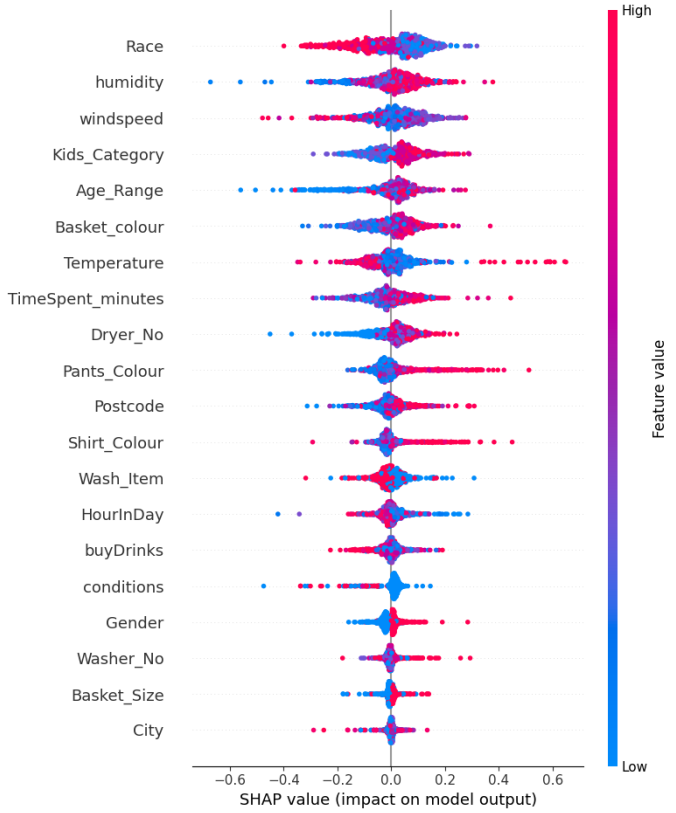
statistical test. According to the result of the statistical test, the p-value is lower than 0.05 which concludes that the difference between the RMSE of both models is statistically significant, thereby meaning **SMOTE dataset does impact the difference of RMSE** in a statistically significant way and the average RMSE of the model that uses SMOTE dataset is lower so we can conclude that using SMOTE dataset do help us improving model performance in this case. However, we did not proceed with using SMOTE dataset in later model training because we do consider that SMOTE has some disadvantages such as increasing the overlapping of classes and thus introducing additional noise as SMOTE does not take into consideration that neighboring examples can be from other classes while generating synthetic examples.

Furthermore, we perform **hyperparameter tuning** for the LightGBM model and Random Forest model. The number of boosted trees to fit, maximum tree depth for base learners, maximum tree leaves for base learners, minimum data in leaf and frequency for bagging are used for tuning the LightGBM model while the number of trees in the forest, the maximum depth of the tree, the number of features to consider when looking for the best split, the minimum number of samples required to be at a leaf node and the minimum number of samples required to split an internal node are used for tuning the Random Forest model. Overall, the numeric range of the parameter value would be relatively small as this is a small dataset. Then, the hyperparameter tuning procedure is then carried out using Grid Search algorithm, and the tuned models are compared to the untuned models. We layer the tuned Random Forest model on top of the tuned LightGBM model for the Stacking Ensemble Model. The graph below shows the comparison between 5 models using model metrics such as MAE, MSE and RMSE.



In conclusion, for whichever model, the tuned version of the model has better performance as those tuned models have lower errors. The tuned Random Forest model does perform the best in MAE but the Stacking Ensemble model performs the best among other models in MSE and RMSE. In conclusion, the **stacking ensemble model** does work better in this case as it has outperformed other models in most metrics, it will be chosen as the final regression model to predict the total money spent by customer.

Lastly, we also **interpret how the features impact the model prediction** by using SHAP values (SHapley Additive exPlanations) which is a method that is used for the interpretability of machine learning models. SHAP values can be visualized using the shap library as referred to the figure below.

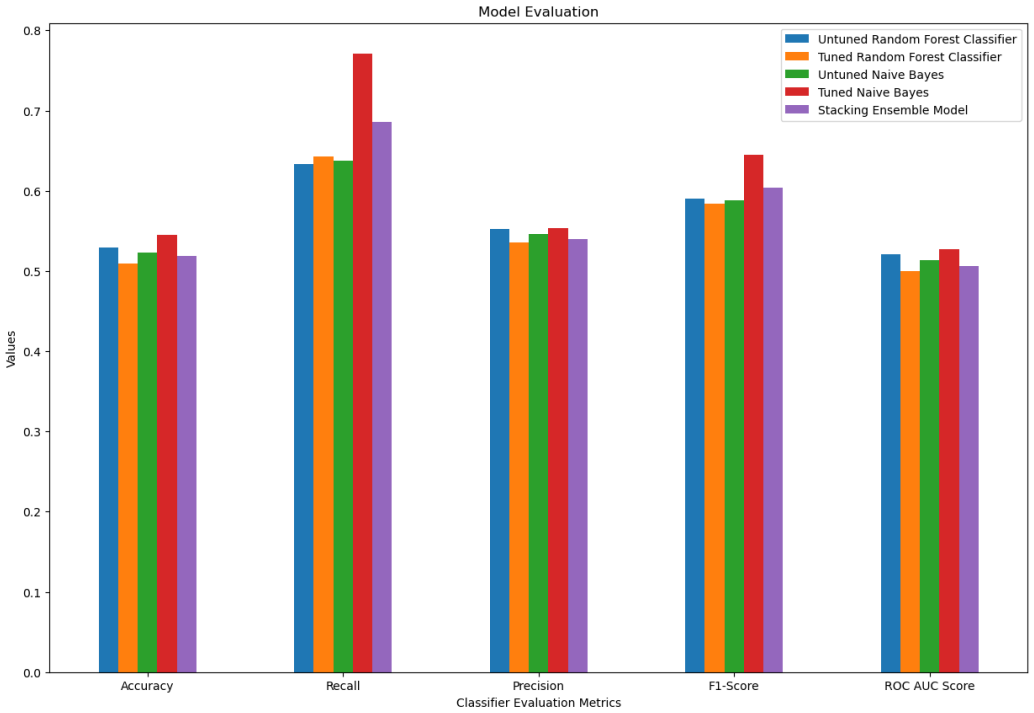


We can then use this graph to interpret how the changes in the features impact the model prediction. For example, high values of humidity will have a positive contribution to the prediction.

Classification

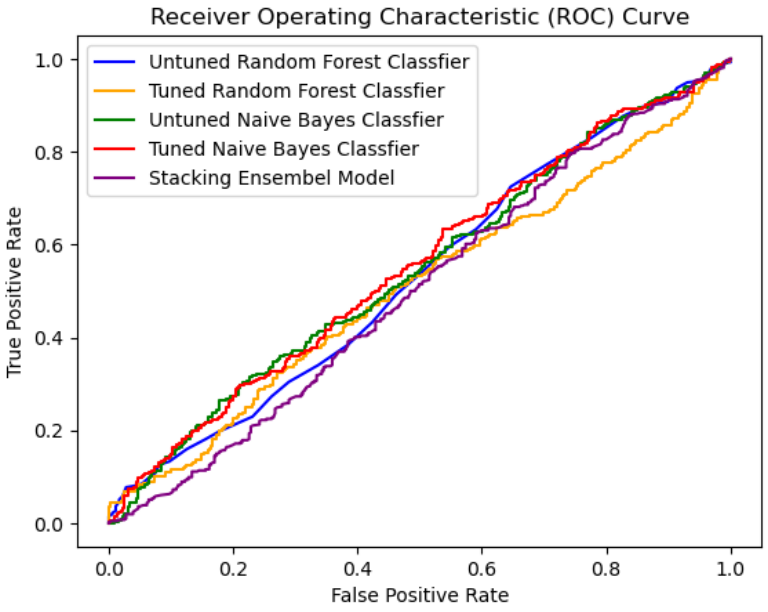
Random Forest Classifier is the first classification model we select because, as was already mentioned, it excels at handling overfitting and has a strong emphasis on features. In addition to Random Forest, **Naive Bayes** is the second classification model we use for classification because it is straightforward, efficient, and well-suited for tasks where the objective is to approximate the likelihood of each category with the given features or produce examples with comparable characteristics. To answer **Question 2**, we would have to choose Wash Item as the predicting target and the remaining columns as feature columns.

For hyper-parameter tuning the Naive Bayes model, we use var smoothing to regulate the level of regularization used, which can help avoid overfitting. In order to regulate the complexity and effectiveness of the model, we utilize the same parameters for the Random Forest Classifier that we use for the Random Forest Regressor above, such as number of estimators and maximum depth. The hyperparameter tuning procedure is then carried out using Grid Search algorithm, and the tuned models are compared to the untuned models. We layer the tuned Random Forest model on top of the tuned Naive Bayes model for the Stacking Ensemble Classifier. The graph below shows the comparison between 5 models (untuned RF, tuned RF, untuned NB, tuned NB, Stacking Ensemble).



We employ Accuracy, Recall, Precision, F1-Score, and ROC AUC Score to verify and compare classification models since they are effective at giving thorough assessment and managing imbalance classes. The percentage of accurate predictions among all of the model's predictions is known as accuracy. Recall measures a model's capacity to properly recognize each positive case. Precision is a metric for a model's ability to accurately identify positive instances and prevent false positives. In order to balance Precision and Recall, F1-Score is crucial. Lastly, the model's capacity to differentiate between the positive and negative classes is measured by the ROC AUC Score.

We can observe that the tuned model performs marginally worse than the untuned model for the Random Forest Classifier. For Naive Bayes, the tuned model performed much better than the untuned model. The Stacking Ensemble Model does not operate well since the tuned Naive Bayes model outperforms it in all measures. After that, we draw a ROC curve to assess how well the model can discriminate between positive and negative classes.

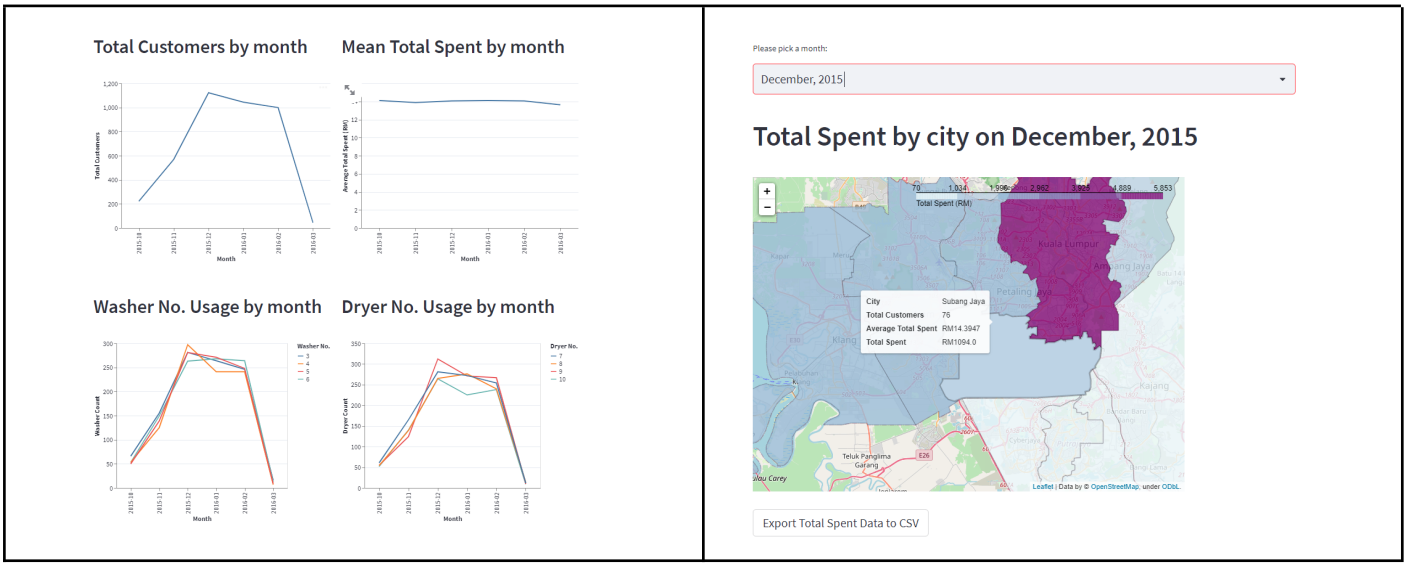


In conclusion, all of the models performed quite similarly to each other which are all very close to random guessing as they are all approaching the diagonal line. However, the tuned Naive Bayes model produced slightly better results than the other 4 models. Therefore, the tuned Naive Bayes model has been selected as the classification model to be utilized to predict the washed item for a customer.

To **improve the performance of both classification and regression models**, we would like to recommend a **data-centric strategy** which is about taking care of the quality of the dataset first before going deeper into adjusting the models. It is because as there is a data science norm goes, "garbage in garbage out". From the Pearson correlation, we already can see that there is no strong correlation between the feature variable and target variable and sometimes we can observe that the data is not inconsistent between columns by taking the "Kid Category" column as an example. We would like to propose that we should ensure the data collection is done in a systematic and discrete way to prevent human error whereby enhancing the data quality.

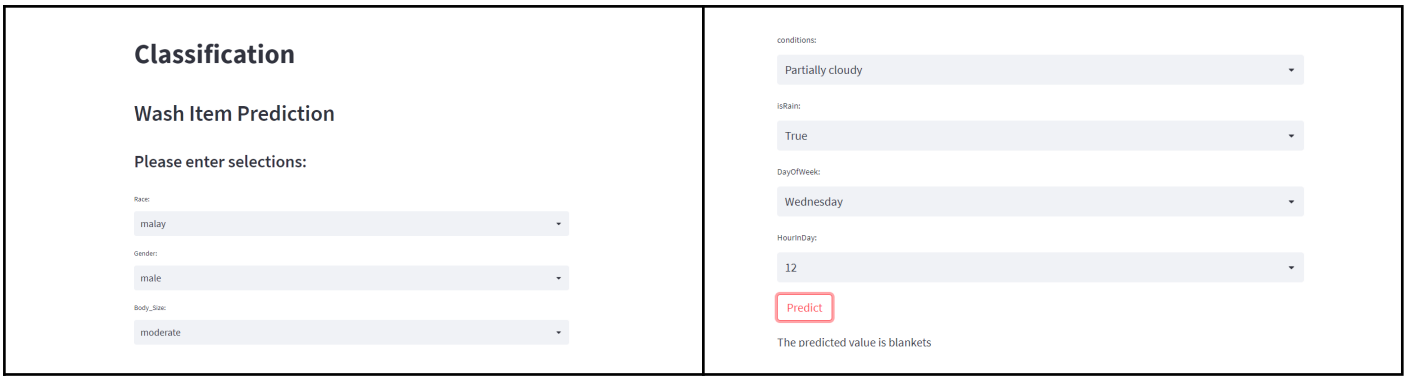
Deployment

a) Real-time Visualization



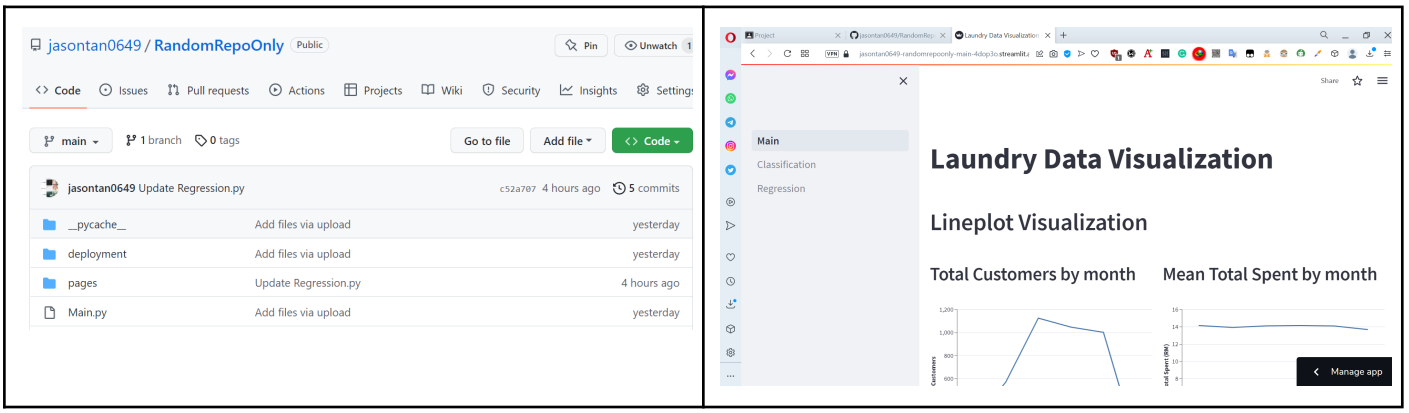
The real-time visualization of the laundry dataset consists of 4 line plots and a map with the city boundary. The 4 line plots show the monthly data of the total customers, the average spent of the customer and the usage of the washer and dryer. Other than that, an interactive choropleth map has been created to show the total spent, average spent of a customer, along with the total number of customers which allows the stakeholder to identify the location of its customer base. The map feature also allows analysis of the data by month to identify the spending pattern over time. Aside from this, the data could also be exported into a CSV file for further analysis.

b) Real-time Prediction



The real-time prediction consists of the best model which answers the questions that were asked in the introduction phase. The user will be required to fill up all the relevant information in order to predict the outcome. A regression model based on a tuned Stacking Ensemble Regressor model to predict the total spent of a customer and a classification model based on a tuned Naive Bayes model to predict the washed item of a customer were deployed on Streamlit.

c) Streamlit Cloud Hosting



In order for the Streamlit application to be able to be used online, the application is published and hosted on the Streamlit Cloud server. Therefore, a user could view the real-time data visualization along with real-time data prediction by visiting https://jasontan0649-randomrepoonly-main-4dop3o.streamlit.app.

d) Performance Improvement

One way to improve the performance of real-time prediction is that we should implement a load balancing mechanism in case there are tremendous incoming API requests. This is commonly done in the industry and it ensures the high availability of the API endpoint when there are users who want to consume our service.

Other than that, we could also utilized the Streamlit's caching method (st.cache) to store some of the values of the function that were called before during the same instance. This would reduce the usage of computing resources which could improve the performance during high traffic load.

Individual Contribution

Each member is considered to contribute equally to this project.

Student ID	Student Name	Distribution Percentage
1181103263	Chan Yun Hong	33.33%
1181103320	Boe Chang Horn	33.33%
1181103087	Tan Kai Yuan	33.33%