TDS 3301
DATA MINING

DATA MINING PROJECT

Prepared by

Muhammad Haziq Faiz Bin Mohd Ripin, 1201302740, 013-2061817
Anwar Ariff bin Mohamad Hassan, 1191302744, 011-56314969

1 Data Cleaning and Preparation

These are the steps we took to clean the data:

1. Remove the empty spaces, comma and correct the spelling.
2. For the empty values in numerical columns, we fill in with the mean value of the respective column.
3. For the empty value in categorical columns, we fill in with a random value from the unique values of that column.
4. Download an external dataset, a weather report csv from here.
5. Join the weather report csv with the original dataset using the date as key.
6. Normalized the numerical data.

2 Exploratory Data Analysis

2.1 Data Visualisation

These are some of the examples of data visualisation that have been made, the rest can be seen from streamlit.

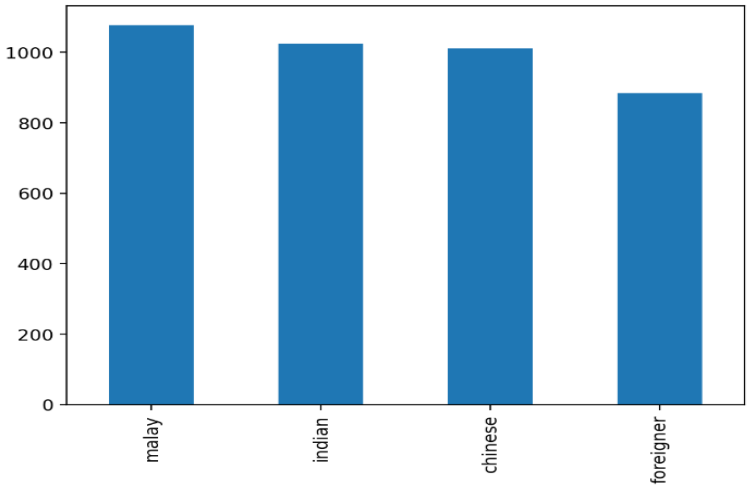

Figure 1: Bar graph to show frequency based on race

From the bar graph above, it can be seen that majority of the customers are Malay. The differences between each race are not too big.

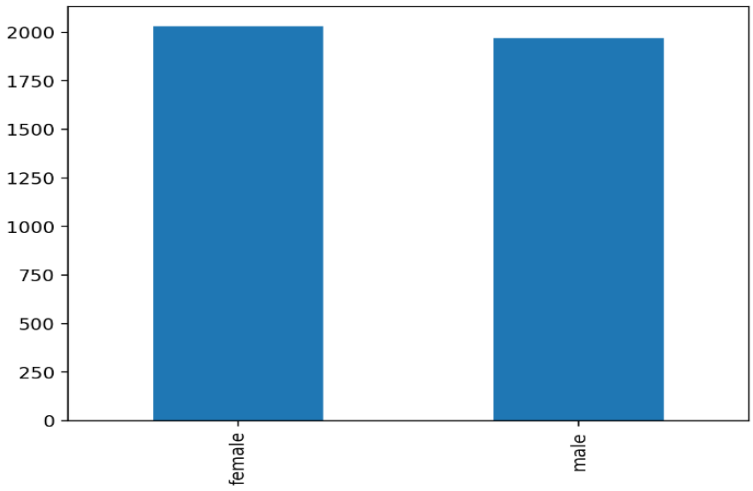

Figure 2: Bar graph to show frequency based on gender

From the bar graph above, it clearly shows that female is the majority customer for the laundry business.

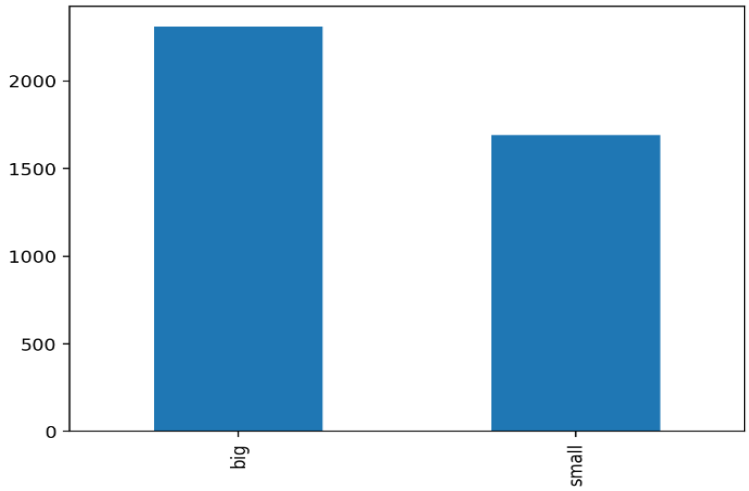

Figure 3: Bar graph to show frequency based on basket size

From the bar graph above, there is big gap between each basket size. This shows that most customers come to the laundry with big basket size. In addition, big basket size may indicate that customers prefer to clean many clothes at a time so that they can save money for travel cost and they do not have to go to laundry multiple times.

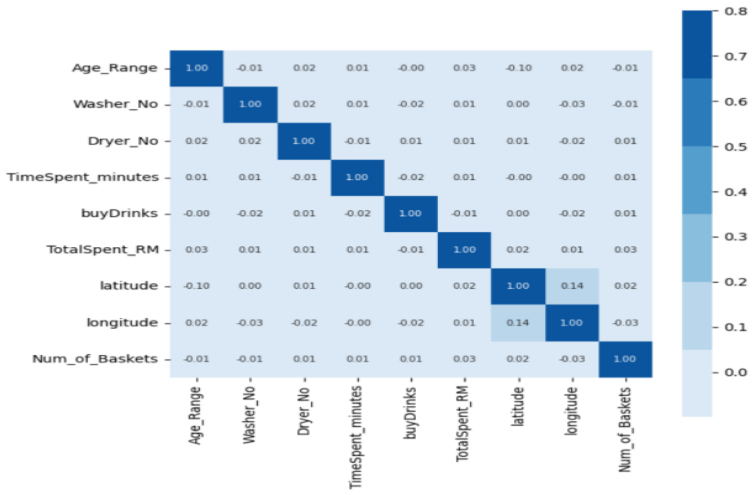

Figure 4: Heatmap to show correlation

Heatmap above has shown valuable informations that can be seen from the dataset. It clearly shows that most of the variables have correlation value close to zero which indicates that there is no linear trend between each variables.

2.2 Check for outlier

We plot boxplot graph for a few columns to check whether there exist any outlier in that specific column. Figure below shows that, there exist zero outlier in the specific columns.

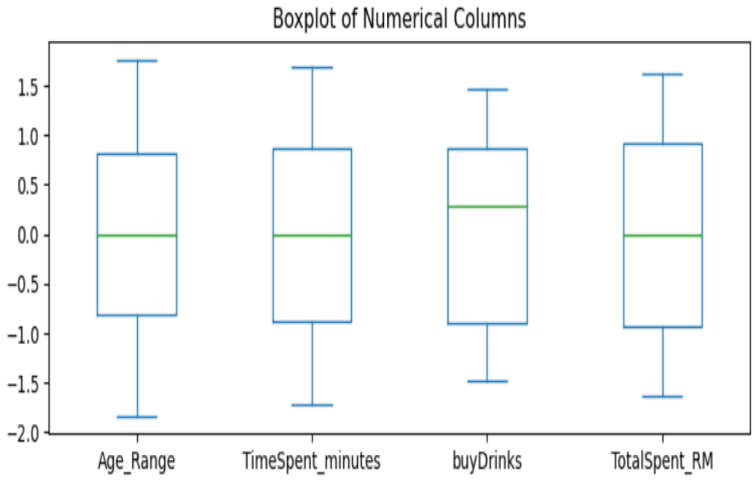

Figure 5: Boxplots for a few numerical columns

2.3 Relationship between variables

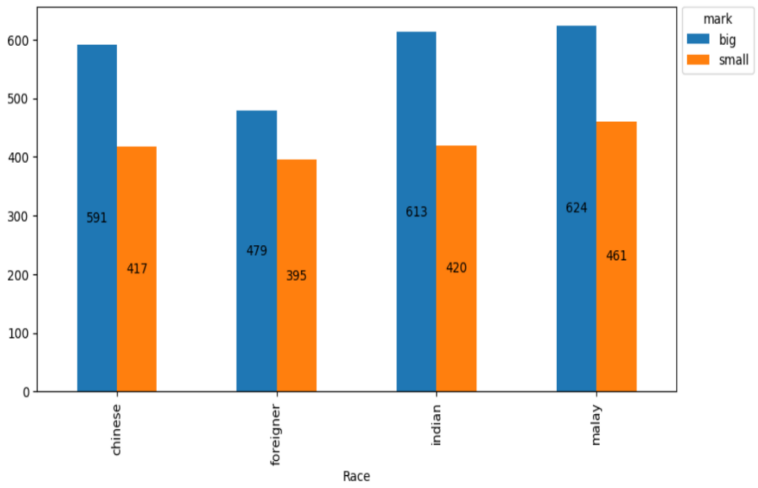

Figure 6: Grouped bar graph to show frequency of certain races at using basket size

We did Pearson's Chi-Square statistical hypothesis test to check for independence between basket size and race. Based on the test we did, both basket size and race are dependent to each other due to the p-value is lower than 0.05.

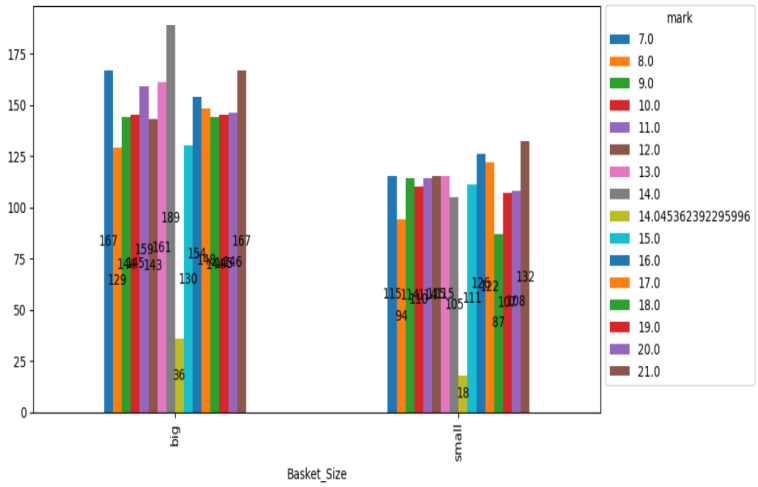

Figure 7: Total money spent based on basket size

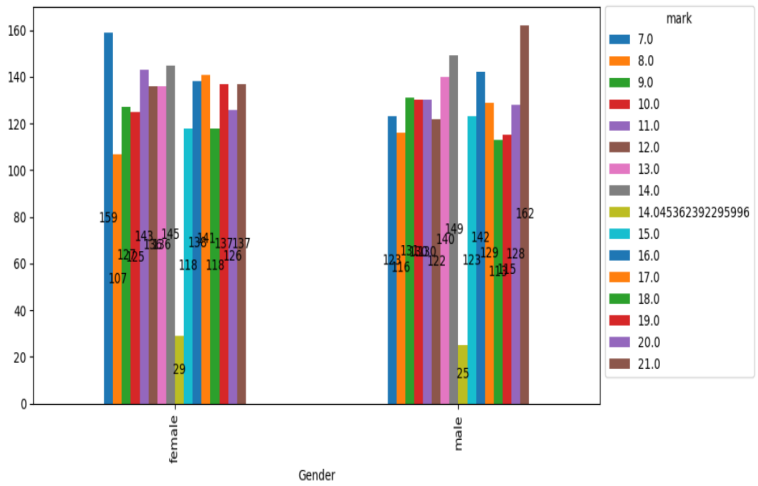

Figure 8: Total money spent based on gender

Next, we check whether basket size and gender can influence the total money spent by the customers using Two-Way ANOVA. Results shown that basket size and gender have no significant effect on total money spent due to their p-value being higher than 0.05. In addition, the interaction between basket size and gender on total money spent is not significant due to p-value is higher than 0.05.

3 Feature Selection

Feature selection refers to techniques that select a subset of the most relevant features for a dataset. This section we would describe the two feature selection algorithm that we have chosen, namely **Boruta** and **Recursive Feature Elimination**.

3.1 Boruta

Boruta is a wrapper-type feature selection algorithm which built around Random Forest. The mechanism in boruta can be explained in these few steps. First, boruta will duplicates the dataset and values in each column will be shuffled. These values here are called shadow features. After that, a classifier such as Random Forest Classifier will be trained on the dataset. This results to output that consists of importance for each of the features. Higher score indicates more important. Then, boruta will check each real features whether they have higher importance at every iterations. Boruta is great to use due to its advantage at doing very well in feature selection thanks to mechanism of shadow features. Boruta will be involved in the process for Question 1 and Question 2.

3.2 Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a wrapper-type feature selection algorithm. This means that a different machine learning algorithm that is used in the core of the method, is wrapped by RFE which helps select features. RFE works by searching for a subset of features by starting with all features in the training dataset and successfully removing features until the desired number remains. **RFE is used in our regression models**.

4 Model Construction & Comparison

4.1 Association Rule Mining

Association Rule Mining is good at showing interesting associations and relationships among large sets of data items. It shows how frequently an item set occurs in a transaction. Businesses usually use Association Rule Mining to identify relationships between the items that customers buy together frequently. Given a set of transactions, we can find rules that will estimate the presence of an item based on the occurrences of other items in the transaction. But as for our project, Association Rule Mining is not involved in any process of answering questions. We just create the model and display the output.

The result shows that (Rule 1) $Partly-cloudy-throughout-the-day. \rightarrow partly-cloudy-day$ has the highest support which is 0.217 and also the highest confidence which is 1.0.

4.2 K-Means Clustering

We used k-means clustering as our clustering algorithm to divide the datasets into k number of clusters to make it easier to detect anomalies. We observed that a higher number a clusters would result in smaller sum of squared error as shown in figure 10.

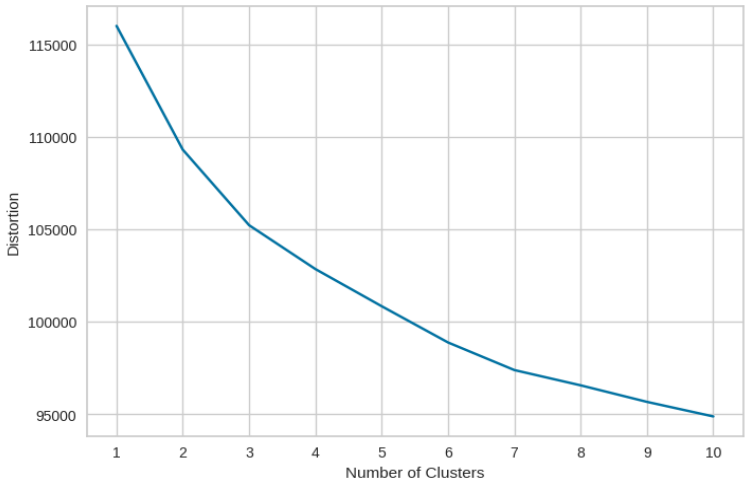

Figure 9: The Effect of Changing the Number of Clusters

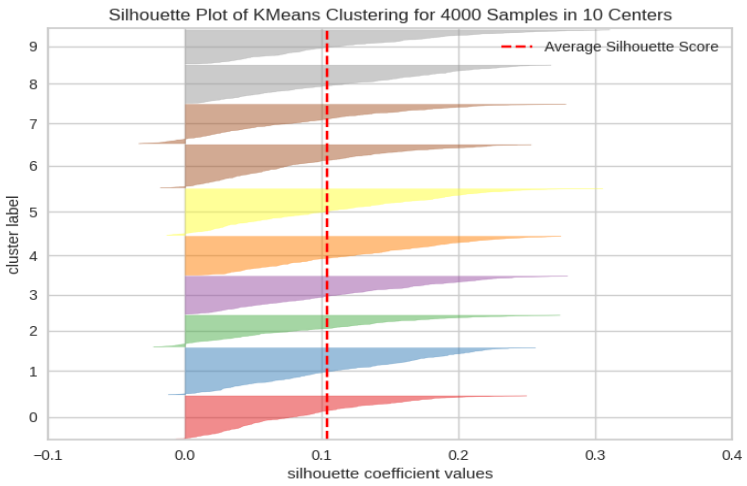

Figure 10: Silhouette Score of Each Cluster

We set the number of cluster to 10 and plotted the silhouette score as show in figure **??**. The clusters formed are quite good as they have similar thickness and the silhouette score for each cluster is above average silhouette scores.

4.3 Classification Algorithm

4.3.1 K-Nearest Neighbor

We chose k-nearest neighbor (kNN) as our clustering algorithm to **predict the weather on a given day**. The target feature has 4 classes namely: cloudy, rainy, fog and partially cloudy. We noticed that that the author of the dataset labelled sunny days as partially cloudy. We noticed there is a very huge class imbalance and we decided to use 3 different kind of sampling technique to transform the original dataset: **SMOTE, random undersampling and random oversampling**. Then, we used Grid Search to find the optimal value of k for each model trained on the transformed dataset. The evaluation metric we chose are the accuracy and f1 score of each class. Finally, we plot the ROC curve, confusion matrix and precision-recall curve for each optimal value of k. All the models are trained using shuffle split cross validation with the number of split set to 5 and test size to 0.3.

The accuracy and the optimal value of k by using different sampling techniques are shown in the table below.

Sampling	Accuracy	Optimal K
None	0.869	14
SMOTE	0.913	1
UNDERSAMPLING	0.940	22
OVERSAMPLING	0.939	1

Table 1: Effect of Using Different Sampling Techniques

The f1 score of each class when using different sampling technique is shown below.

We observed that only oversampling model tried to classify cloudy days and all models showed a very high score on partly-cloudy class.

Sampling	Cloudy	Fog	Rain	Partly-cloudy
None	0	0	0.661	0.910
SMOTE	0	0.473	0.569	0.864
UNDERSAMPLING	0	0	0.603	0.905
OVERSAMPLING	0.286	0.343	0.564	0.867

Table 2: F1 Score By Varying the Sampling Technique

We also investigated the effect of changing the value of k on the accuracy of each model and the result is displayed figure 11.

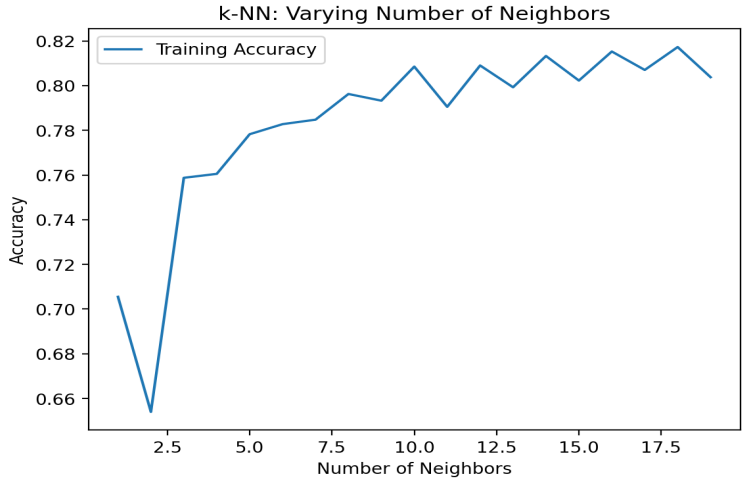

Figure 11: The Effect of Changing the Value of k With No Sampling Technique Applied

Finally, we plotted the confusion matrix, ROC curve and precision-recall curve for each of the model. As the target variable is a multi-label one, we had to binarize the weather column to plot the ROC curve and precision-recall curve. The plots for the model with no sampling technique applied is show in figure 12, 13 and 14. The rest of the plots can be viewed on our Streamlit app.

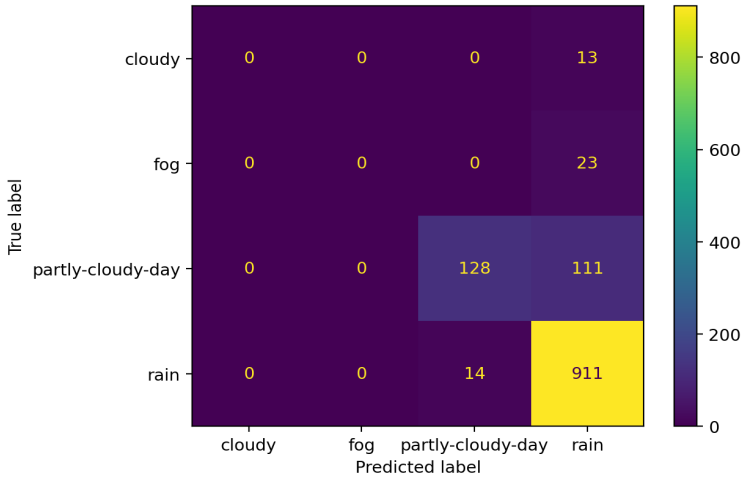

Figure 12: Confusion Matrix with No Sampling Technique

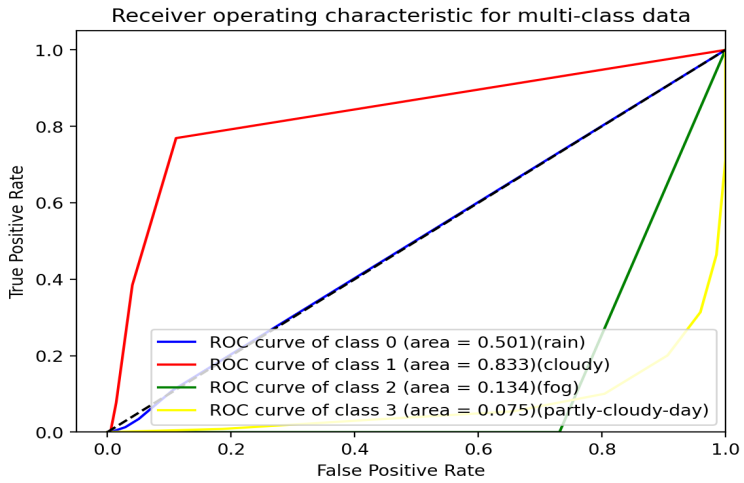

Figure 13: ROC Curve with No Sampling Technique

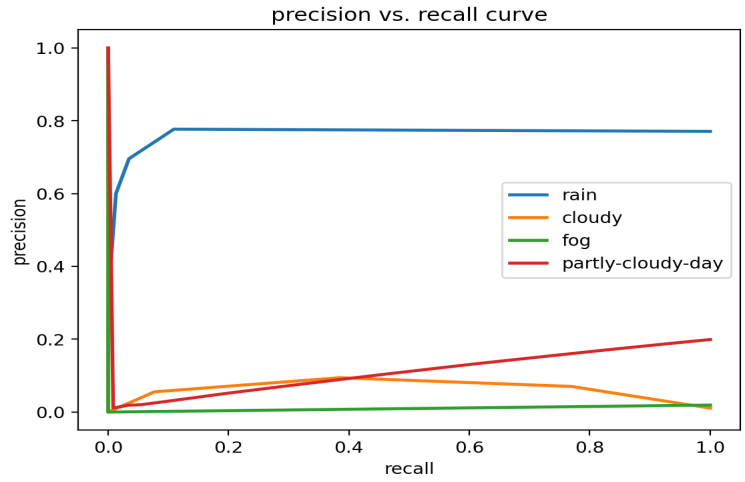

Figure 14: Precision-Recall Curve with No Sampling Technique

4.3.2 Naive Bayes

Naive Bayes has been chosen due to its fast at computing the accuracy. In addition, the results from Naive Bayes is reliable and Naive Bayes also easy to be implemented. Naive Bayes is included in Question 1, which trying to check whether number of features can affect accuracy results.

Results shown that Naive Bayes that used top 15 features from boruta score achieved higher accuracy which is 0.582 as compared to Naive Bayes that used all features with 0.577 accuracy. This shows that by dropping unnecessary features, classifier can achieve better results because unnecessary noises have been removed.

4.3.3 Random Forest

Random Forest has been chosen due to it is simple to be applied. Besides its good performance at classification tasks, Random Forest also can handle huge dataset efficiently. Random Forest is included in Question 2 which involves Grid Search as well. Figure below shows the confusion matrix for Random Forest after classification process.

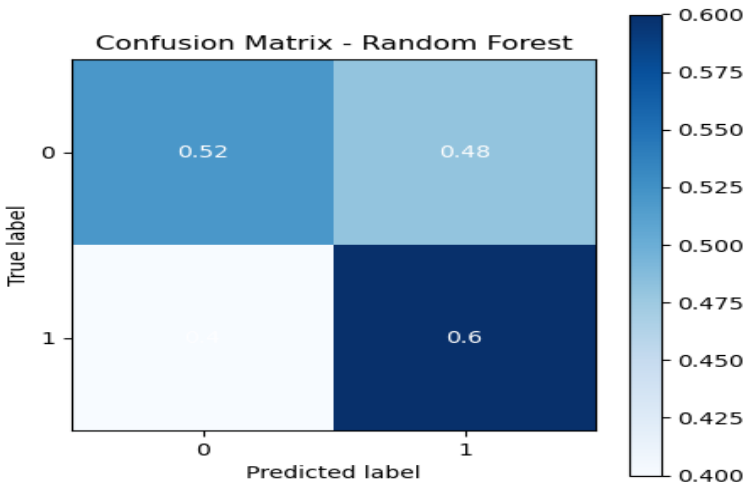

Figure 15: Confusion matrix for output of Random Forest

4.3.4 Decision Tree

Based on our early plan, we want to use Support Vector Machine with Grid Search due to its capability of getting excellent accuracy results. But, Support Vector Machine require much longer time to compute the accuracy results which is the reason we have to replace it with Decision Tree. Decision Tree is fast at classifying which helps us at saving our time to run the model. Figure below shows the confusion matrix for Decision Tree after classification process.

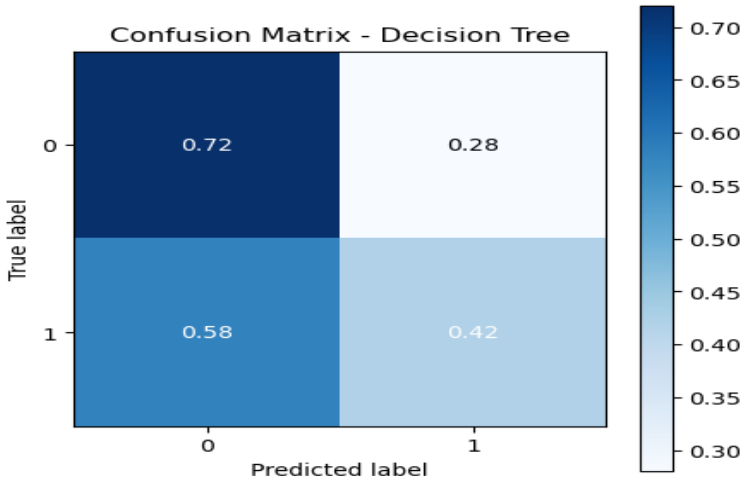

Figure 16: Confusion matrix for output of Decision Tree

4.4 Stacked Ensemble Modelling

Stacking Ensemble Classifier has been compared to Naive Bayes in Question 1. Result shows that stacking ensemble classifier with accuracy of 0.402, did not managed to outperform Naive Bayes with accuracy of 0.582 which has much higher accuracy score. Figure below shows how algorithms in the stacking ensemble classifier performs as a single model, the figure also shows boxplot for stacking ensemble classifier as well.

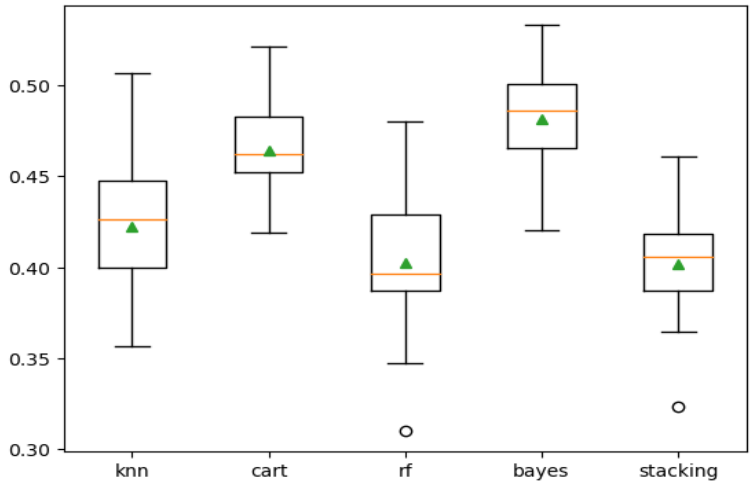

Figure 17: Boxplot to show outputs for Stacked Ensemble Modelling

4.4.1 Hyperparameter Tuning

We did our hyperparameter tuning by using GridSearchCV function in Question 2. Random Forest and Decision Tree are used together with Grid Search so that both models can achieve better results. Result shows that Random Forest that used together with Grid Search achieved 0.585 accuracy which outperforms Random Forest without Grid Search that achieved 0.568 accuracy only. The best parameter for Random Forest is 'max depth': 50, 'max features': 3, 'min samples leaf': 4, 'min samples split': 10, 'n estimators': 300.

Next, result shows that Decision Tree that used together with Grid Search achieved much better accuracy which is 0.591 as compared to Decision Tree without Grid Search that only achieved 0.554 accuracy. The best parameter for Decision Tree is 'max depth': 2, 'max features': 0.8, 'min samples leaf': 0.04.

4.5 Regression Algorithms

We build three regression models with the features chosen by RFE to investigate whether regression models can be used to **predict the humidity of a given day**. All the models are trained using shuffle split cross validation with the number of split set to 5 and test size to 0.3.

4.5.1 Linear Regression

First, we ran an RFE model to get the feature scores. The top 10 features are shown below.

Features	**Score**
weather	1.000
latitude	1.000
description	1.000
tempmin	1.000
tempmax	1.000
longtitude	1.000
Spectacles	0.950
Body_Size	0.910
Kids_category	0.820

Table 3: Top 10 RFE Features using Linear Regression

Then we run a linear regression with the top 10 features and get the r^2 score. We repeat the process with 5, 15 and 20 RFE features on our Streamlit app and the result is shown below.

RFE Features	**R2 Score of Linear Regression**
5	0.578
10	0.580
15	0.576
20	0.576

Table 4: r^2 Score of Linear Regression Using Different Number of RFE Features.

4.5.2 Lasso Regression

The same process are repeated for lasso regression, we ran an RFE model to get the features score before we build a lasso regression model with α of 0.20 . The top 10 features are shown below.

Features	Score
weather	1.000
description	1.000
tempmin	1.000
tempmax	1.000
latitude	1.000
Num_of_Baskets	0.960
longtitude	0.910
TotalSpent_RM	0.870
buyDrinks	0.820

Table 5: Top 10 RFE Features using Lasso Regression

Then we run a lasso regression ($\alpha = 0.20$) with the top 10 features and get the r^2 score. We repeat the process with 5, 15 and 20 RFE features on our Streamlit app and the result is shown below.

RFE Features	R2 Score of Linear Regression
5	0.464
10	0.464
15	0.464
20	0.464

Table 6: r^2 Score of Linear Regression Using Different Number of RFE Features.

Interestingly, there are no changes to the r^2 score when we changed the number of RFE features. So, we decided to adjust the value of α but maintain the number of RFE features to 10 and the result can be summarised in the table below.

Alpha	R2 Score
0.2	0.464
0.4	0.238
0.6	0.021
0.8	-0.02
1.0	-0.02

Table 7: r^2 Score by Changing the α Value of Lasso Regression with 10 RFE Features.

4.5.3 Polynomial Regression

We felt that the we could increase the r^2 score of our linear regression by changing it to a polynomial regression with degree 2. We transformed the data by using PolynomialFeature class, use the top 10 RFE features and run the models using a pipeline. The results can be seen in the next subsection.

4.5.4 Comparison of Regression Models

Table 8 shows the summary of the r^2 score obtained from each model using different number of RFE features. We can clearly observe that polynomial regression give the best

performance and it is not surprising as the performance will generally increases as the the number of features increases. However polynomial regression is very slow compared to the rest as the time complexity increases exponentially. Thus we believe that it is not suitable for real-time deployment.

RFE	**Linear Regression**	**Lasso Regression R2 (α=0.2)**	**Polynomial Regression**
5	0.578	0.464	0.724
10	0.580	0.464	0.720
15	0.576	0.464	0.720
20	0.576	0.464	0.719

Table 8: Summary of Different Regression Models

5 Model Deployment

The models developed has been deployed here. The sample screenshot is given below:

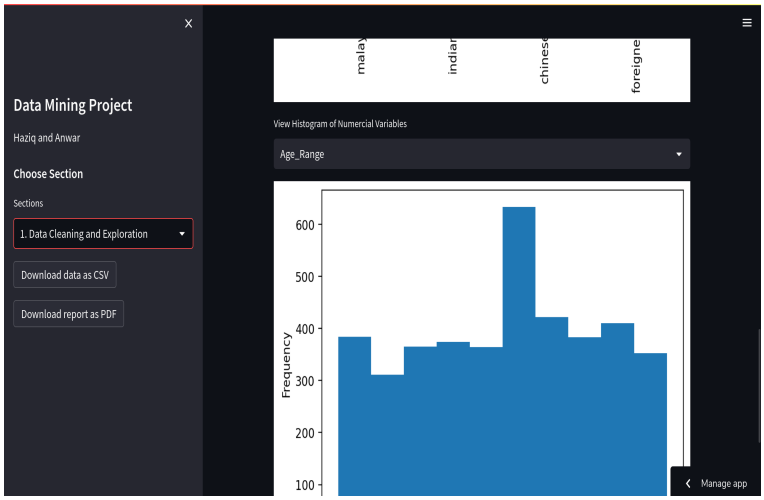

Figure 18: Example Screenshot of Deployed Streamlit App