MIS772 – Predictive Analytics
T2 2022

Assignment 2 – Individual
Student name: Aman Rajput
Student number: 221069377

Executive summary

AirbnbAI has provided a sample dataset of 6,000 rental listings and the related 104,000 customer reviews to leverage data and artificial intelligence to analyze and forecast consumer comments about their experiences staying in Hong Kong Airbnb rental units. The analysis would help AirbnbAI to understand how relevant are ratings when compared with actual feedback left by customers. Aside from this, AirbnbAI would also need some insights to understand relationships, patterns, or trends within segments that would allow them to predict the rating that a listing would likely get on the basis of its specifications and past reviews.

Following insights have been developed for AirbnbAI in response to this analytical project's stated business goals and proposition:

Question A: "Is there a significant correlation between the sentiment (positive vs negative) of customer reviews of a property, and their review score ratings?

With reference to figure 1, we can say there was unexpectedly a weak correlation (0.218) between the sentiment (positive vs negative) of customer reviews of a property and their review score ratings. This can be explained by either guests' decision to give a better or worse overall rating compared to the individual review scores and the process' inability to score positive and negative words in guest feedback.

Question B: "Can the review score ratings of properties be predicted (estimated) based on relevant attributes?

Yes, review score ratings can be predicted with selected attributes. We employ multiple techniques in multiple regression to get the most accurate predictions. While all the models were mostly statistically significant, this may not be true for independent variables. Few of the independent variables were found to be highly insignificant and correlated with each other, especially attributes related to the review scores. With changes in each iteration of the model, we filtered out values that were insignificant and had a low tolerance. In the final approach, we optimized feature selection using evolutionary optimization. The output of the best model (R^2 = 85.1%) indicates that the review_score_rating moves relatively in line with review_scores_cleanliness, review_scores_location, price_per_night, bedrooms, and room_type. Besides technical analysis, we must also consider the sentiments of the customers. For example, for each guest staying at a property. he may weigh different review score aspects differently while deciding on an overall score rating.

Question C: "What are the most meaningful different segments that exist in the retail properties?"

To understand different aspects of the properties, a lot of features are considered but the most important ones that separated these are the neighborhood, price per night, and property type. While the rating score of the property is also a key feature of any listing, it was found to be not very reliable when assessing a property by segment. Hence, a property can be best segmented by:

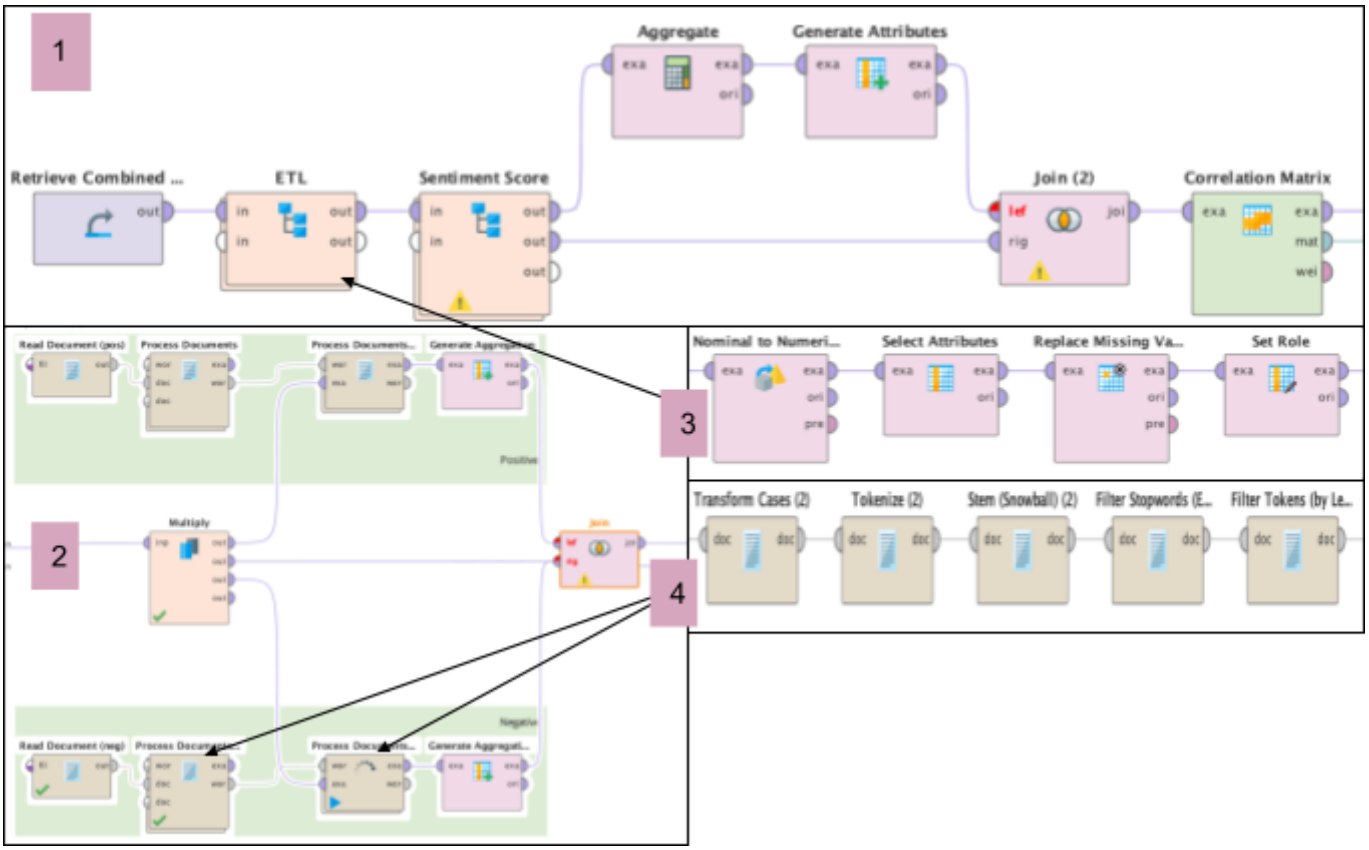
- Neighbourhood
- Bedrooms
- Accomodability
- Price
- Property Type / Room Type

AirbnbAI is recommended to vectorize such attributes for better suitability for data analysis. Any variation in wordings can lead to additional categories, making the process more complex and less reliable. Furthermore, to improve the overall prediction of the model, it is recommended to include emojis in the text analysis for a better understanding of sentiment. Lastly, review_score_rating should be a calculated average of other review score dimensions rather than keeping it as a separate score. With these changes in data pre-processing, better segmentation and prediction of review score ratings are achievable.

Data exploration, pattern discovery, and preparation

Through descriptive statistics and visualizations, we are keen on examining the correlation between all attributes to select the most relevant ones for the optimal regression model. The analysis also focuses on the correlation between the sentiment (positive vs negative) of customer reviews of a property, and their review score ratings. It should also be noted that the accuracy of the model highly depends on the processes being used, drawing conclusions from the data may lead to consumer bias, prejudice, damage, and imbalance to both the hosts and guests.

Figure 1: Correlation between Sentiment score and review_score_rating (and process)



1. The main process to calculate sentiment score consists of 2 subprocesses i.e. ETL and Sentiment Score. These processes return sentiment scores which are then aggregated for each property so that these can be compared to review score ratings.Process 2 is a subprocess named sentiment score, here the same dataset is used for text mining to retrieve the total positive or negative score in the customer's feedback. The dataset is multiplied and analyzed for a count of positive or negative separately and once the count is final, it is aggregated by the sum and stored for each property. Post-join, two new attributes i.e. Positive and Negative, are introduced to our dataset which could be used to carry out further operations as shown in Figure 1 (Process 1).
2. Process 3 is another subprocess that is used across almost all models to perform ETL operations on the dataset. The data needs to be transformed before we can use it for text analytics. First, all the relevant nominal attributes are converted to numerical as the regression model can only operate on numerical values. Then we filter out all the attributes except numerical ones using the "select attribute" operator. To make sure no missing information is passed further a "Replace missing Value" operator is used. Finally, we assign the ID attribute as ID. The process of ETL is similar for all processes, only the order of operators or operators used to convert an attribute to numerical value may vary depending upon the model, predictors, and label.
3. Process 4 is the most fundamental subprocess that is responsible for text mining. The process first converts all the text into lowercase (Transform cases), which are then split into smaller or root forms (Tokenize). From the remaining text, all the irrelevant, long, and short terms like adjectives are omitted (Stemming, Filter Stopwords, and Filter Tokens). Lastly, we are left with text as a vector of numbers i.e. no. of times a term has occurred.

Figure 2: Correlation matrix between computed sentiment score and review_scores_rating

Attributes	Sentiment Score	review_scores_rating
Sentiment Score	1	0.218
review_scores_rating	0.218	1

Sentiment Score = No. of positive words / Sum of positive and negative words

Using the aforementioned formula, we can calculate the sentiment scores. Since a better score rating and a better sentiment score indicates positivity, it is implied that these would go up with good feedback/good rating. The output correlation matrix shows a weak correlation between the positivity score calculated and the review score rating of the listing. This can be explained by either guests' decision to give a better or worse overall rating compared to the individual review scores or the model's inaccuracy to score.

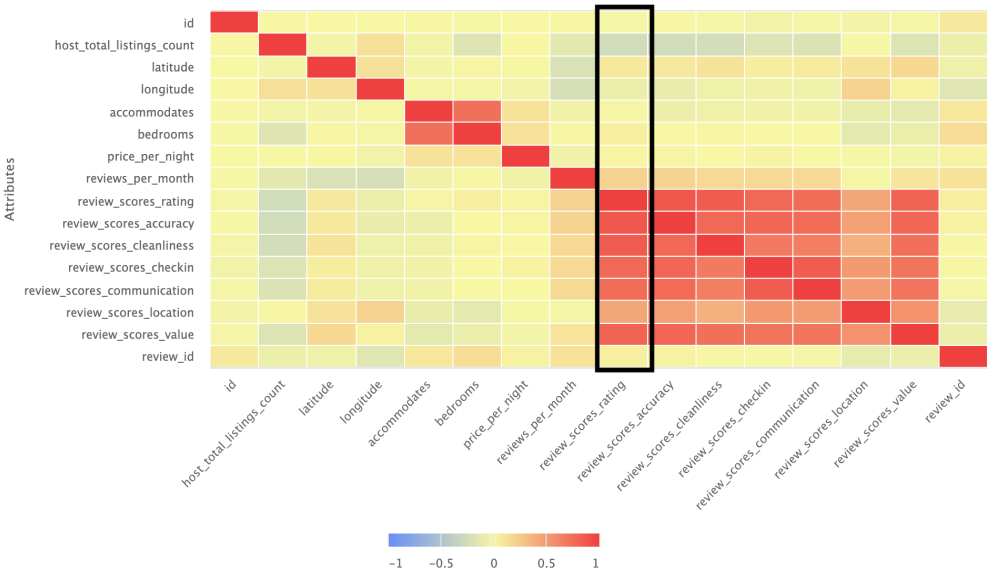
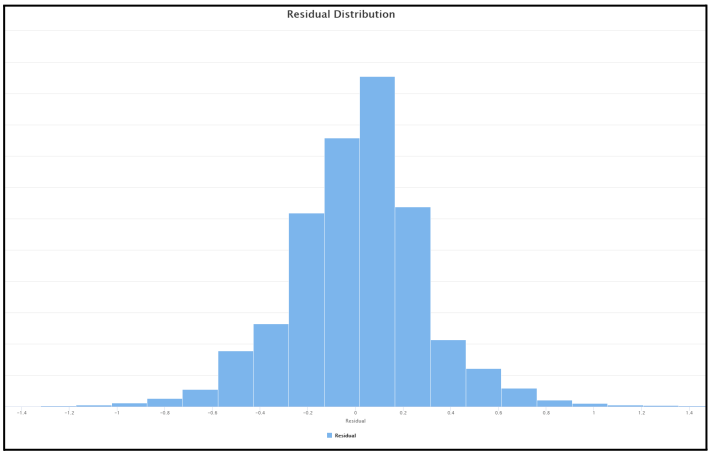


Figure 3: Correlation matrix for predicting review_score rating

Figure 3 shows a correlation between all the attributes within the dataset. All the values relating to review scores were not only highly correlated to rating but are also highly multicollinear with each other as a customer who doesn't like the cleanliness of the place may give the listing an overall bad rating. While these are the most relevant attributes, it is suggested to keep the tolerance of the model high or eliminate collinear features. With proper data transformation, when we run the regression model, a lot of low-tolerance attributes are removed while ensuring it keeps all the significant values.

Figure 4: Distribution of residuals of regression



The regression model can be validated by the analysis of residuals. It shows One of the major issues with heteroscedasticity and non-normality in the residuals is the inconsistency in the model's level of error throughout the observed data. This implies that the amount of predictive power predictor variables possess is not uniform across the whole range of the dependent variable. As a result, not all patterns in the dataset can be explained by the model. Evidently, on the basis of this argument, our regression model gave a near-normal residual plot, which reveals that the regression model was quite successful in making predictions.

Figure 5: Initializing cluster formation with k = 7

For the purpose of clustering, most relevant nominal attributes were assigned unique integers using unique integer coding. Alternatively, dummy coding could also be used for converting categorical data to numerical, however, the dummy coding method would provide for specific cluster traits but would also add more noise as it would test the group for each category. Hence, we initialize cluster formation with a random k=7. The clustering brings data groups or all the similar data points together, segregating groups with similar traits and assigning them into clusters. Further optimization is used to determine the optimal value of k and the centroid chart can be used to interpret clusters

Predictive modelling

Linear Regression is used here to predict the review score rating of properties by selecting appropriate predictive attributes. As we discovered certain relationships in the correlation matrix (See fig), we know that any attribute relating to the review score would impact our target variable. The regression model would require a balance of non-collinear and mutlicollinear attributes to predict with accuracy without introducing high multicollinearity. The min. tolerance for each model is set at 0.3 so that it does not eliminate all the predictors and returns fair accuracy.

Common Parameters across model:

Feature Selection: M5 Prime (selects a subset of attributes, which improves the Akaike information criterion the most)
Min. Tolerance: 0.3
Eliminate Colinear features: Yes

Figure 6: Illustration of full multi regression model

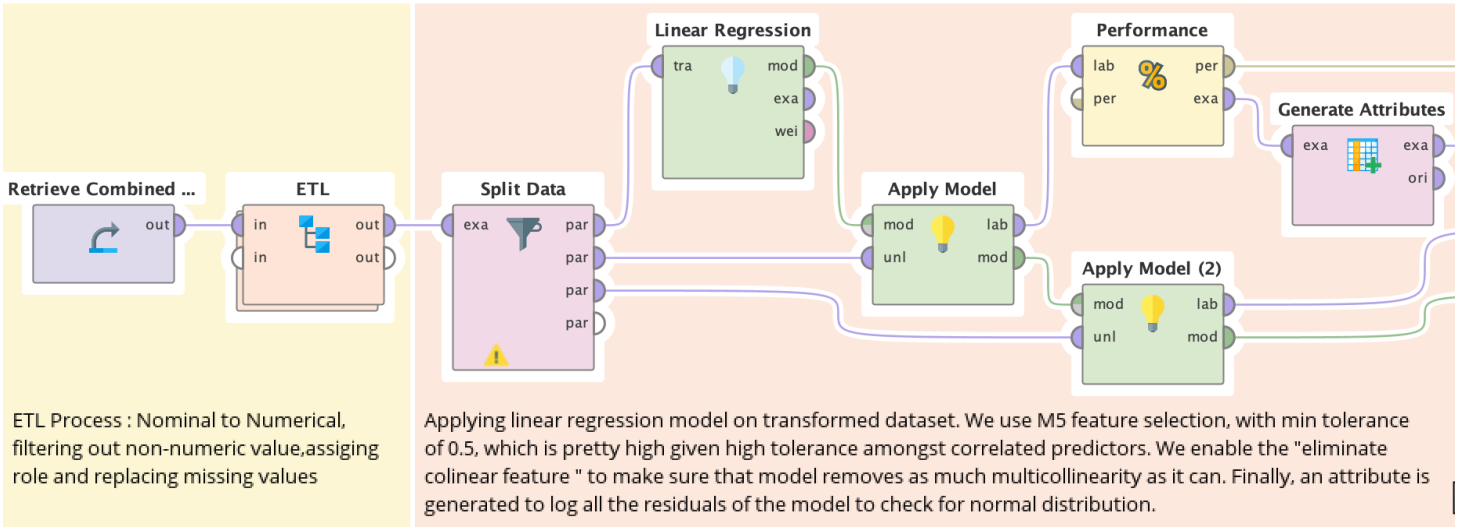
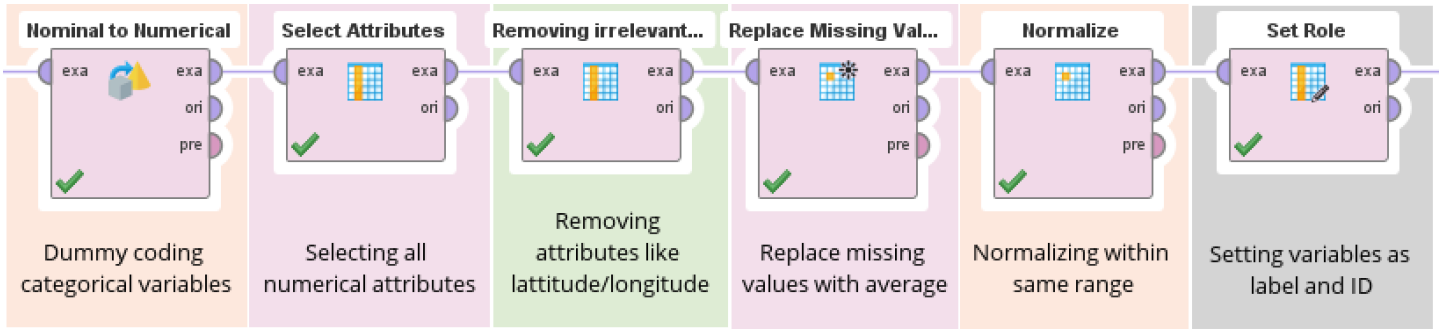


Figure 7: Sample ETL Process



Nominal attributes like room_type bring in more scope to improve the overall model, hence these are first to be encoded into binary form for the model to run. The Second and third operator filters out all the attributes but numerical and actual numerical values, like Review ID, latitude or longitude is a numerical value but cannot be used mathematically. To make sure the model is fed no missing value, we replace all the missing values with average. Till now, we have data with varying ranges across predictors, hence, we normalize all of the values within a range using range transformation under Normalize operator. Lastly, we assign the ID role to the ID attribute so as to mark each value without including it in the calculations by the model.

Figure 8: Clustering Process

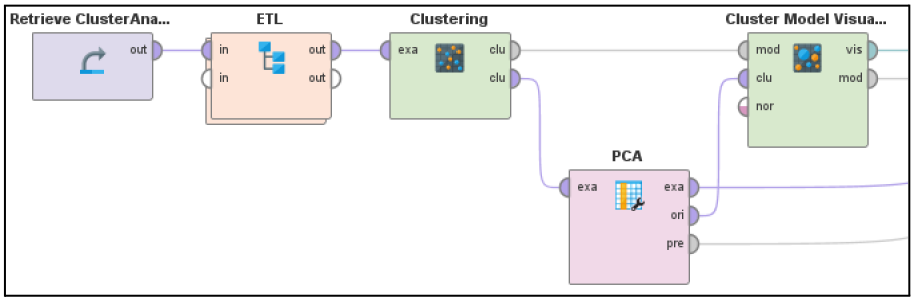


Figure 9: Example of Nominal to Numeric

prop_neighbourhood	
Central & Western	0
Wan Chai	1
Yuen Long	2
Islands	3
Kowloon City	4
Yau Tsim Mong	5
North	6
Southern	7
Tuen Mun	8
Eastern	9
Sham Shui Po	10
Sha Tin	11
Sai Kung	12
Wong Tai Sin	13
Tsuen Wan	14
Kwun Tong	15
Tai Po	16
Kwai Tsing	17

In figure 8, an illustration of the clustering process is shown. First, the numeric-coded dataset is retrieved, and basic ETL processes like normalization, filtration of numeric attributes, etc. are done to transform the data into the desired form. Nominal attributes like property_type and room_type were assigned numeric values for each category similar to how each neighborhood was assigned a unique value as shown in Fig 9. Then the clustering operator uses the k-means algorithm to segregate data into groups. "A cluster in the k-means algorithm is determined by the position of the center in the n-dimensional space of the n Attributes of the ExampleSet. This position is called the centroid."

A PCA operator follows an attribution reduction feature that upon addition to the model, reduces dimensionality in overall clusters. Using the covariance matrix, this operator conducts a Principal Component Analysis (PCA); it allows to define the amount of variance to retain while keeping the optimal number of the primary attributes in the original data. Lastly, a cluster model visualizer is used to capture essential details about the clusters and their centroids.

Figure 10: We can interpret each cluster by matching the value through which the cluster line passes. For example, we can say properties that are under cluster 5, mostly shared or hotel rooms in the Kwai Tsing neighborhood, hosted by hosts with fewer listings, accommodate fewer people, and get more reviews per month compared to others.

Figure 10: Centroid Chart

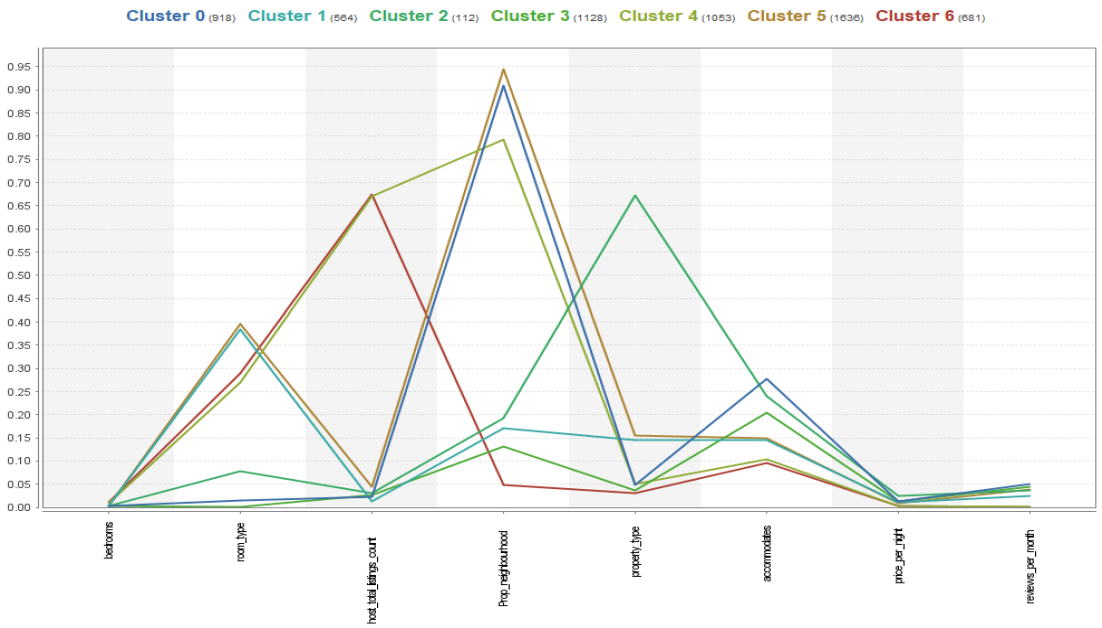
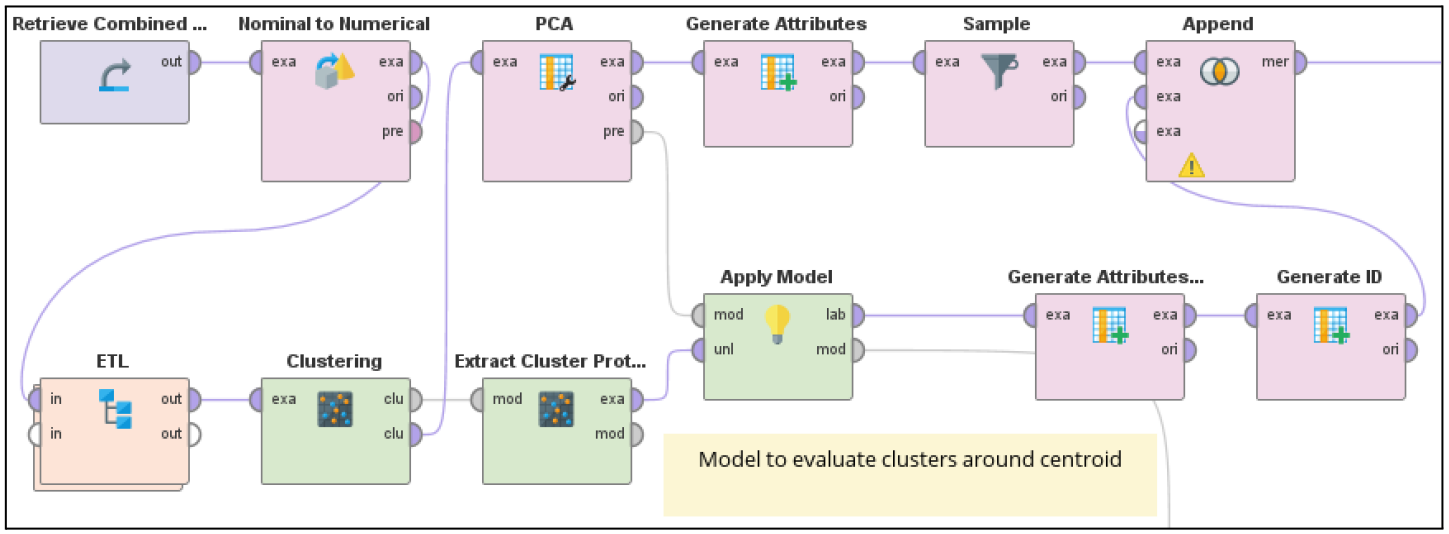


Figure 11: Cluster Evaluation Process

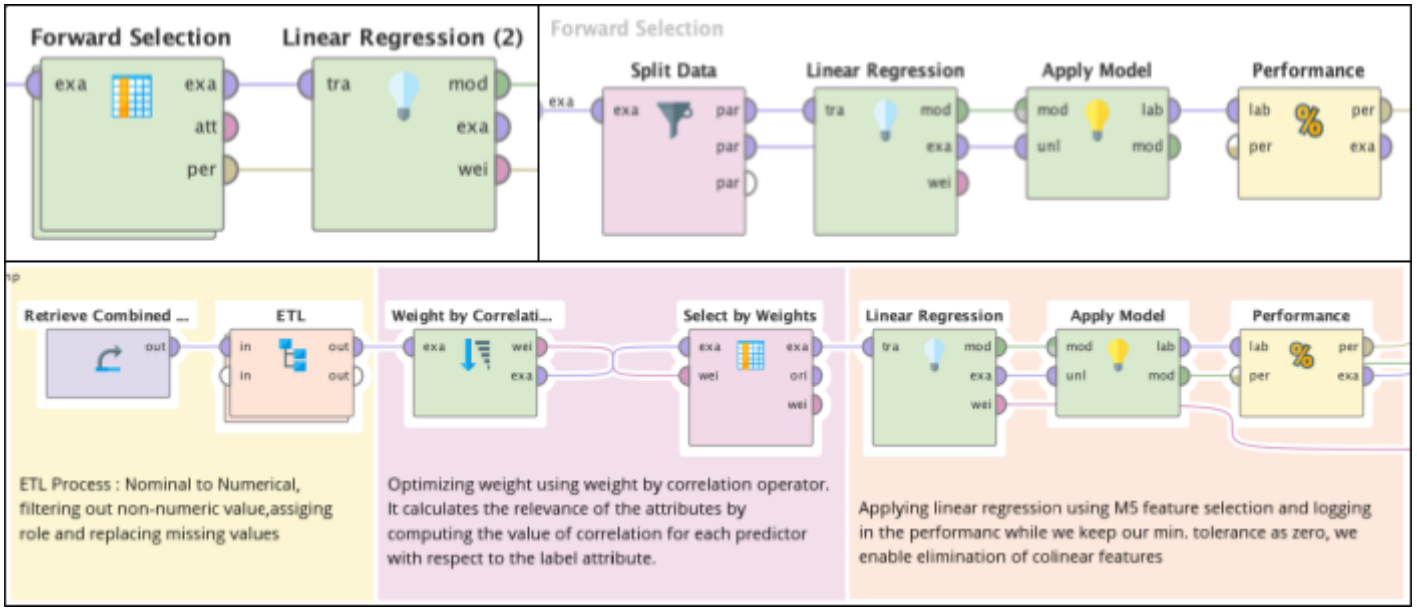


Model evaluation and improvement

Linear Regression Model

To optimize our model we can use different techniques ranging from stepwise selection to optimizing weights of predictors. Below are illustrations of two of those techniques i.e. forward selection, and weight optimization.

Figure 12: Optimizing regression model



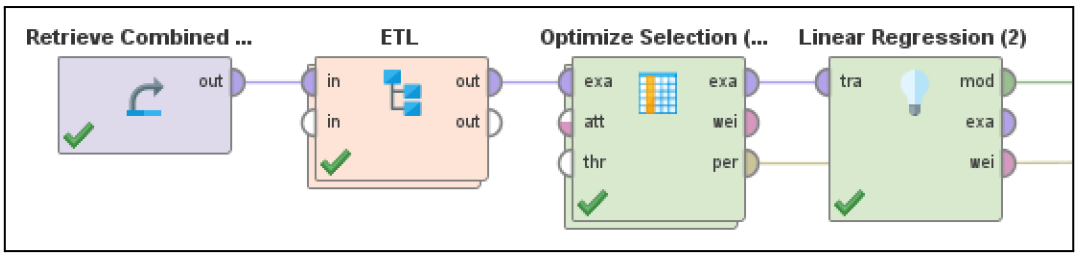
Through feature selection (weight optimization), where we only select the top 5 attributes (k=5) for the regression model, the squared correlation was not as good as the regular regression model which can be explained by fewer predictors. The downside of using a linear regression model is that the output R^2 will never decrease upon the introduction of a new predictor. This can affect the reliability of the model overall, hence, it is always suggested to have fewer and most significant variables.

Table 1: Performance of all regression models

Regression Specs	RMSE	Correlation	Squared correlation (R^2)
Simple Regression	0.177	0.773	0.598
Multiple Regression	0.171	0.792	0.628
Multiple Regression - Nominal	0.571	0.821	0.674
Multiple Regression - Feature Selection	0.161	0.817	0.667
Multiple Regression - Full	0.535	0.833	0.694
Multiple Regression - Forward selection	0.499	0.849	0.722
Multiple Regression - Optimize Selection (Evolutionary)*	0.337	0.928	0.851

As observed from the table on the left, we have come far from the initial accuracy of 0.598 which was achieved through simple regression with only one most correlated variable. The forward selection model is a combination of M5 feature selection and forward stepwise selection. It improves the performance of prediction by 4-5% from a full regression model. To improve the model further, we can go for more powerful techniques like brute force which consumes a lot of computing power. Due to limited resources, a sample of 10% is taken to test the predictive power of one such operator. We use evolutionary optimization selection features with default parameters to select the best feature. This operator uses a genetic algorithm for feature selection that involves switching and interchanging features. For this analysis, the tolerance of the linear regression model is set to 0.7. The process to implement the "Optimize Selection" operator has been shown below in Figure 13. Inside the "Optimize Selection" operator we use the same operators as used in forwarding selection (See Fig 12).

Figure 13: Using Optimize Selection Parameter



Clustering and Segmentation

Initially, we randomly assigned a value to k in k-means clustering and obtained the first set of clusters that we got. However, to determine the optimal value of clustering we can use two approaches known as David Bouldin and WSS. An "optimize parameter (grid)" operator (See fig 14) is used to iterate the model to find the best value of k and the value of David Bouldin at that k. We set the following parameters for the operator to iterate:

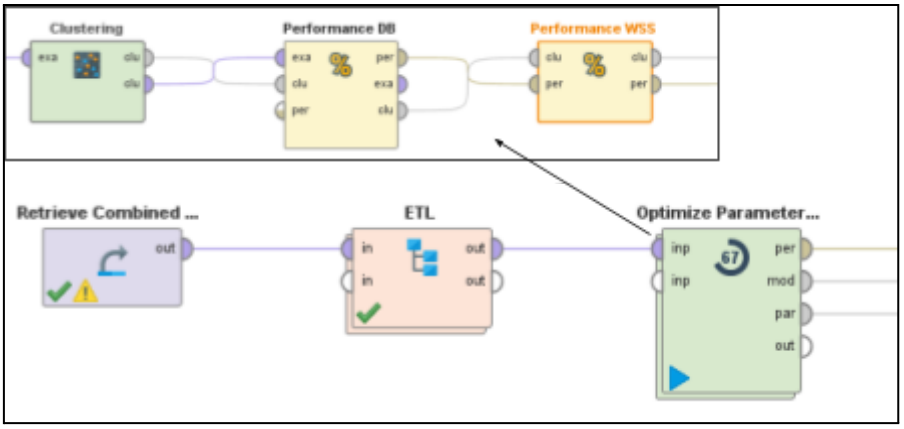
Parameters: Min 2; Max 50; Steps 10; Scale - Linear

and get the following output:

Output: Davies Bouldin: 1.138; Example distribution: 0.167; Optimal k = 7

We couldn't get a distribution plot showing WSS elbow clearly hence, the model is interpreted using optimal k returned. Since we ran the model at k =7, we would not have to rerun it to get new sets of clusters. It can also be concluded that the clusters generated in the first run of clustering were the most precise and optimal.

Figure 14: Clustering optimization,



In the end, we will evaluate our cluster model visually, for which we will plot centroid and data concentration around them. (See fig for the process)It is observed from Fig 15 that using the scatter plot we can see how well is data distributed around the centroid. We can observe that most of the variance in the data can be explained by the data distribution.

Figure 15: Cluster Evaluation (pc1 v. pc2)

MIS772 – Predictive Analytics

T2 2022

Assignment 1 – Individual
Student name: Aman Rajput
Student number: 221069377

Executive summary

Problem Statement:

The Portuguese Banking Institution (PBI), with a growing customer base, tasked Financial AI to employ data analytics and machine learning to analyse its customer data for the bank's direct marketing campaign. The marketing campaigns were based on phone calls and often, more than one contact with the same client was required. The bank needed insights into the effectiveness of its telemarketing campaign conducted on its clients and potential segment for them to target.

Solution Statement:

The customers of the PBI come from all ages, therefore customers were divided into different age groups - Adult (18-30); Middle-aged (30-55); and Senior Citizen (55+). During initial data exploration, it was observed that while most of the middle-aged and senior citizens were married, more than a third of young adults were single. Despite being the highest amongst married, middle-aged customers were also highest amongst single (60.8%) and divorced (79.6%) categories. This indicates that most of the customers of the bank are middle-aged and/or married and senior citizens and/or divorced were the least amongst clients. (See Figures 3 and 4). (Target Audience)

Secondly, after further exploration, it was noted that the top 5 most popular occupations amongst the bank customers were blue-collar (9732), management (9458), technician (7597), administration (5171) and services (4154). Out of these, clients from management held the highest average balance of $1763 amongst all occupations. Moreover, almost 82.5% of clients (7801) working in the management sector had tertiary education when compared to other sectors. (See Figure 5) The correlation analysis also strongly indicated that people who pursued management likely held tertiary qualifications. (See Figure 5)

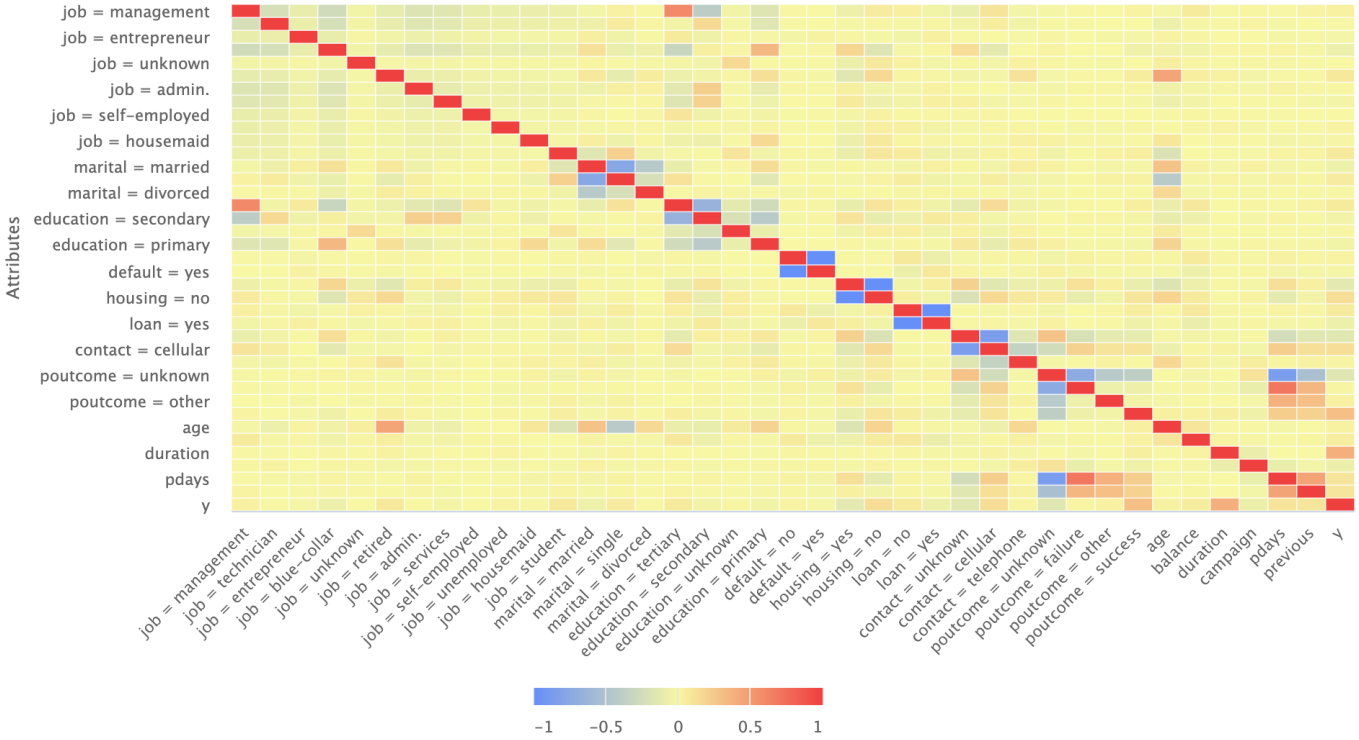
Using the two models i.e. kNN and decision tree, it was checked which model would be able to predict a positive customer's response to its campaign for term deposit subscription. However, it was the decision tree boosted with XGBoost which performed better in terms of accuracy and percentage of correct predictions. With the help of this classifier, the bank would be able to correctly predict with over 89% accuracy if the target client would subscribe to their term deposit. Few recommendations based on data analysis, classification and exploration to benefit from the campaign would be:

- Customers are likely to subscribe more in the months of March and September, hence the marketing campaign must align with that time of the year
- Telemarketers should aim for clients willing to know more, it is indicated that longer engagement of customers usually turned out to be successful conversions. This can also avoid undesirable advertisements leading to optimal customer satisfaction
- To tackle the imbalance in the dataset or, in simple terms - eliminate unknown details, telemarketers must target the right customer and gather as many details as possible about the target. With more accurate information, the algorithm would predict more precisely the outcome of the campaign for any client.
- By understanding its customer base better the bank is well equipped to provide better services. For example, depending on the individual customer's age and occupation can reveal when to approach them for loans, and the loan status will help assess their overall risk profile. This will help the bank provide good service to its customer at the right time (i.e. when they most require a loan).

Data exploration, pattern discovery, and preparation .

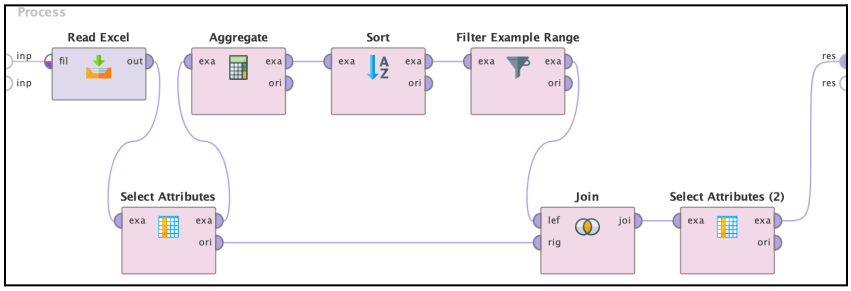
For the data exploration, different processes were implemented to answer different parts of the problem. Similarly, for each set of problems, different attributes are selected which are relevant to the problem. In Figure 1, a correlation matrix is derived by converting all the nominal values to the numerical (mostly binary), same operator is also used across other processes where numerical values would make predictions more accurate for rapidminer. Set Role operator, where used, changed the role of key attribute i.e. 'y' to label as in this analysis 'y' is our target attribute for learning.

Figure 1 - Correlation Matrice of Attributes



No value in the dataset is missing, however, there are certain nominal attributes like 'job', 'education', 'poutcome' and 'contact' where the information is unknown. With the option to replace, remove or retain, we'll choose to retain all the unknown polynomial values for further analysis as these values have almost no correlation to the target variable 'y'. Moreover, despite consisting of most values 'poutcome = unknown' has a correlation of -0.167; whereas 'poutcome = success' indicates (correlation = 0.307) that a customer is most likely to subscribe if the previous outcome for that client was successful. Apart from 'poutcome', another variable that highly affected the outcome of this current outcome is 'duration'. With a correlation of 0.395, it can be stated that a longer length of call indicated a successful conversion of clients for a subscription.

Figure 2 - RM Process for data exploration



To visualize customers' age distribution by marital status, two bar charts were used (Figures 3 and 4). In figure 3, values were stacked by percentage to show how age groups varied across marital categories while in figure 4, bars

display no. of singles, married and divorced across different age groups. This was achieved by discretizing age into different classes namely young adults, middle-aged and senior. This would also help the bank to identify its target segments in the future.

Figure 3 - Stacked age distribution across Marital status

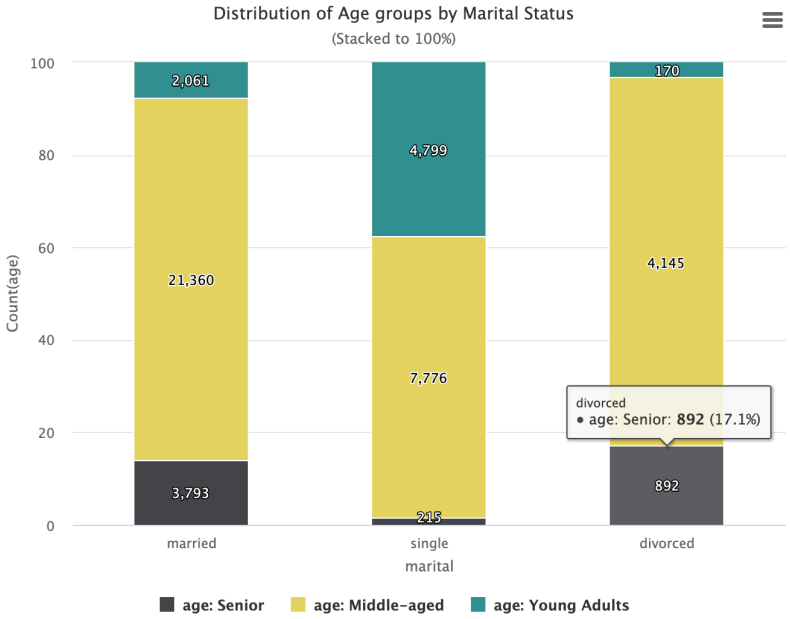
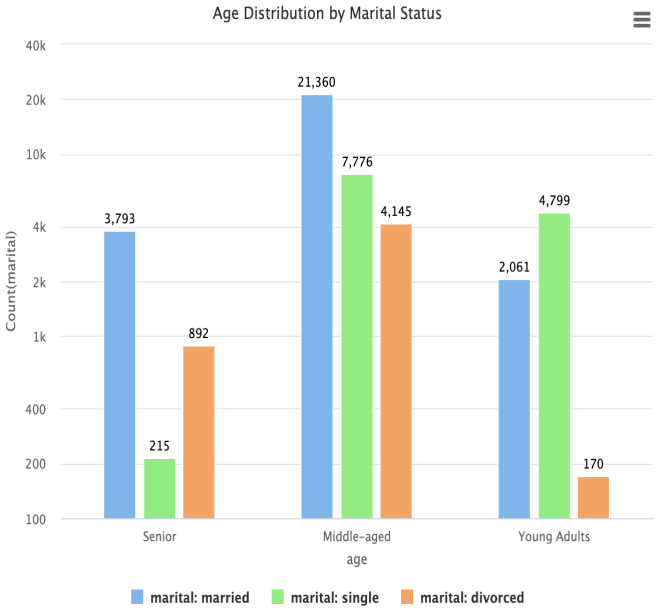
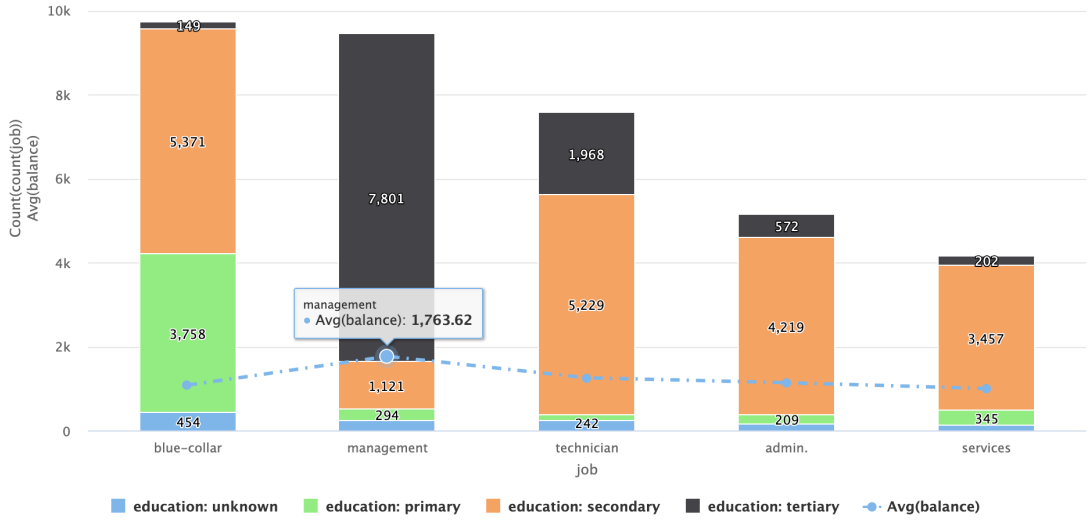


Figure 4 - Age Distribution by Marital Status



To bring out insights about occupation, education and average balance of clients, a thorough process was constructed in the Rapidminer. To begin with, relevant attributes (balance, education and job) are filtered from the dataset. For determining the most popular occupation amongst the client the data is aggregated by jobs; sorted by the number of clients in each occupation and then the top 5 out of those occupations are filtered by using the 'filter example range' operator. Now to focus on just these jobs, we use a 'join' operator to retrieve values from our main dataset only for the relevant jobs that are grouped and filtered. (See figure 2) With this process, a dataset will be generated with a new attribute 'count(job)'. This new attribute would be helpful for displaying the count of clients in each sector. Furthermore, the data is grouped by job and colour is set by education to generate a stacked and intuitive 'bar chart' under turboprep. An additional 'line' plot is added to check the highest average balance of clients working across different sectors. (See figure 5)

Figure 5 - Jobs and education of clients



Predictive modelling .

For the purpose of predictive modelling, on the dataset, kNN and decision tree models were used and validated using k-fold cross validation and hold-out method. A decision tree model usually splits into nodes or branches of values belonging to different classes. "The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features". In simple terms, it provides an alternative result for every "now what". While k-NN uses proximity to make predictions of an individual point. A random example of k-NN would be if we were to determine a random animal in a barn that is surrounded by goats, k-NN will likely predict that the said animal is a goat too.

Data Pre-processing:

As shown in figure 6, almost all the processes consist of similar pre-processing of data through the below operators:

Sample: This operator is used in selected examples (mostly k-NN) where, in absence of it, the process would take a long time to run. The parameter value ranged anywhere from 0.3-0.5 depending on the complication of the process.

Set role: This operator is used to set variable y as the target variable or the label. With this, the rest of the attributes are considered predictors.

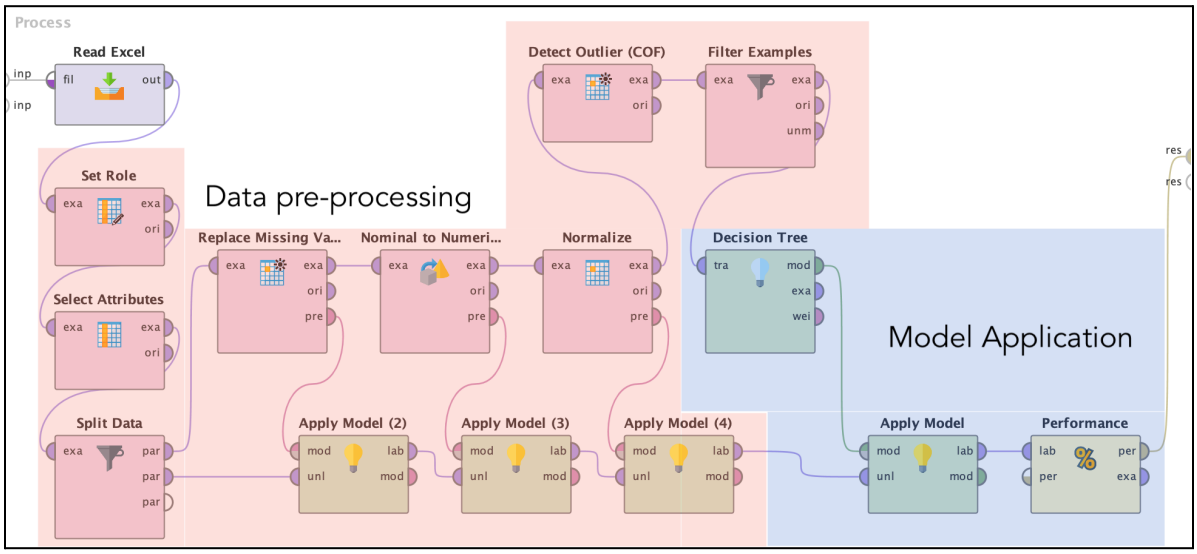
Replace missing values: It replaces any missing values in selected attributes with average.

Nominal to numerical: to map nominal attributes to numerical (usually binary) for a finer output

Normalize: this operator scales the values of attributes of different scales so that they fit in a specific range. This operator is quite important for k-NN models as it ensures fair comparison when dealing with euclidean distances.

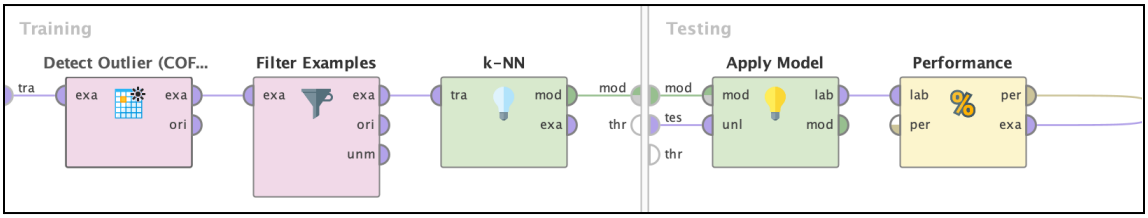
Detect Outlier: This operator ranks each instance by parameters N and k. It is ideal to identify any outlier (any point numerically distant) in the dataset and place it into a new attribute created - outlier.

Figure 6 - RM process Decision tree (Hold-out)



As you can see, 'Apply model' operators 2,3 and 4 are constantly deployed after each operator, this is done to prevent any data leakage. Similarly, in the case of cross-validation, it is also recommended to put relevant operators inside the 'cross-validation' operator. (See figure 7)

Figure 7 - RM process k-NN (Cross-validation operator)



The class imbalance here was addressed using sampling and weighting; rather than jumping to operators like 'SMOTE' and 'generate weight', the dataset was experimented with by two different sample operators to avoid sample bias - bootstrapping and stratified. With different models, as anticipated, it was found that bootstrapping samples gave better accuracy and value of kappa compared to stratified samples.

Table 1.1 Results from different methods and optimizations

	Hold-out		Cross validation (Sampled)	
Algorithm	Decision Tree	k-NN	Decision Tree	k-NN
Accuracy	89.90%	89.29%	90.33%	90.25%
Kappa	0.397	0.373	0.411	0.481
AUC	0.741	0.823	0.736	0.774

While both algorithms are non-parametric, the decision tree supports automatic feature interaction and is much faster due to KNN's expensive real-time execution. Initially, we employ decision tree and k-NN through the hold-out technique where we get a quite similar accuracy and kappa (See table 1.1). After running the same process through cross-validation, the decision tree model gave an increased accuracy of 90.33% and a slightly higher kappa of 0.411 whereas k-NN slightly outperformed it after the data had to be downsampled because of longer run-time. Nonetheless, a kappa below 0.6 is not considered optimum for a model and therefore there is a need to optimize by tackling imbalances in classes and poor kappa. It is further suggested to reinforce different techniques into cross-validation for best improvements. Apart from kappa and accuracy, another parameter that is useful to measure the ability of the classifier to distinguish between classes is AUC (area under the curve). It was evident from the individual ROC curves of both the decision tree and kNN that the kNN classifier performed better, which means kNN did a better job of classifying the positive class in the dataset.

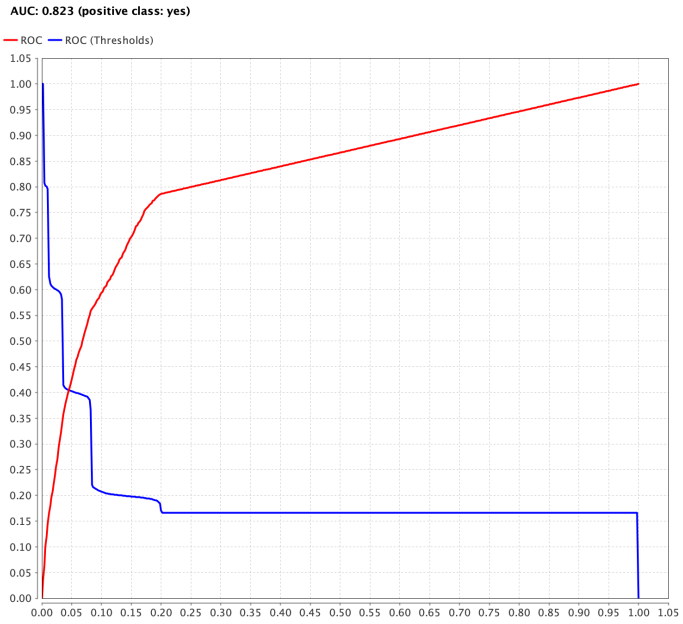


Figure 8 - ROC curve for kNN hold-out

As for the prediction regarding the subscription of term deposit, we can state that the current simple models perform better when we utilize cross validation for both the decision tree and k-NN. However, time expense by the k-NN model cannot be justified as it was not good enough compared to the decision tree.

Figure 9 - RM process decision tree kNN

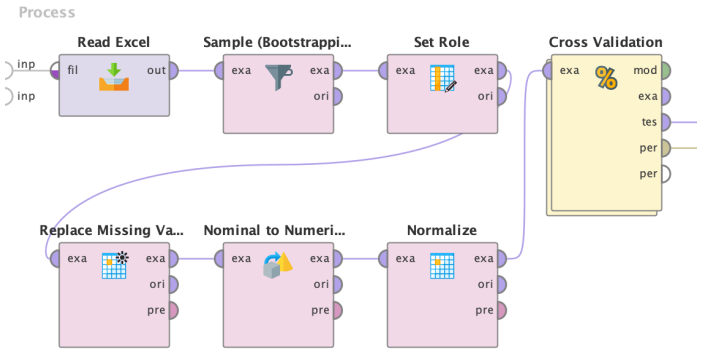
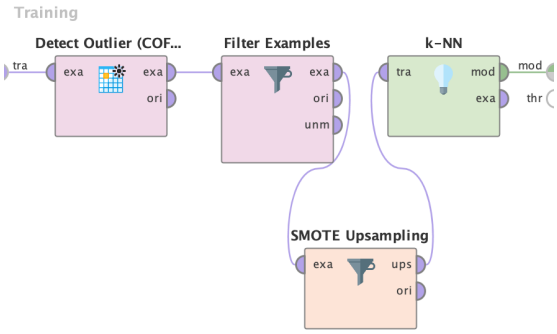


Figure 10 - Output of loop parameter



Model evaluation and improvement .

The methods used for evaluations are k-fold cross validation and hold-out. While both validations can't be compared to each other as they operate differently altogether, it is conclusive that hold-out simply splits the data into training and validation partitions and assumes they are representative of the population, which leads to single validation. However, when k-fold cross-validation is run on the dataset, the model is validated k-1 times. This makes sure that the model is trained on every data point and is more capable than the hold-out method. In the simple implementation of these validations, it was found that cross validation outperformed hold-out in all RM algorithms, and rightfully so. (See table 1.1)

Comparative analyses of models:

	Pros	Cons
Decision Tree	- Decision trees have great interpretability of the model; the outputs are simple to interpret and do not require statistical knowledge or complex concepts. - These models are quick at training and forecasting. - Perhaps, one of the best advantages is that this support both numerical and categorical data features. - A decision tree is also good for handling colinearity in the model.	- It is not recommended to use a decision tree where rare instances are important. It can easily prune those and eliminate them from the output - They are highly sensitive to the data, one small change can lead to a huge change in final trees - Overfitting is very common while using decision trees
kNN	- It's an easy and simple machine learning model that barely requires any hyperparameters - It is a non-parametric approach, therefore, it doesn't have any assumptions on the data distribution - Lazy learning- there is no need of model-fitting in advance, with the provided data point it can predict with good accuracy	- The interpretability of kNN is good in small datasets, however, with a dataset like here, the interpretability will deteriorate - Hyperparameters like k is to be carefully selected as a slight variation can create poor accuracy of the model - The biggest tradeoff of kNN for precision is the computational expense and time - "Proper scaling is expected for fair treatment among features", it is prone to class and weight imbalances

Conclusion: Although kNN performed slightly better than decision tree under both validations, the time consumed to run kNN could not justify the extra accuracy for prediction.

Approaches in sampling: Bootstrapping vs Stratification vs Eliminating Unknown values

To successfully run the kNN models, smaller samples were extracted from the main population. This was done using 'sample (bootstrapping)' operator. This operator was chosen over 'sample (stratified)' which ensures that the sample resembles the main population as bootstrapped sample gave a better result over stratified. In order to sophisticate the model, a radical approach is also experimented with, where 'declare missing values' operator was used (see figure 7) to eliminate all the unknown attributes; as a result, the kappa increased slightly at the expense of accuracy.

Table 2.1 Perfomance comparison of decision tree

Decision tree with cross-validation	Accuracy (%)	Kappa
With unknown value	90.08	0.404
Without unknown value	84.25	0.544

Model optimization:

As shown in table 2.2, different settings to the models and validation can generate different outputs. Mostly when class imbalance is tackled, it comes at the cost of accuracy. The best accuracy that was obtained was 90.33% through the decision tree with cross validation with a kappa of 0.411. After the deployment of various operators to balance the imbalanced classes, the accuracy fell significantly, however, there was an increase in kappa in all cases except ensemble. The kappa is a robust measure to see the efficiency of the model as it takes into account the correct prediction occurring by chance. Generally, a model with kappa above 0.6 is considered to be good, however, the best kappa that was obtained is 0.565 after using the gradient boosting trees technique.

Table 2.2 Model optimization results

RM techniques and validation	Optimization	Accuracy (%)	Kappa
Cross Validation	Decision Tree - Gradient Boosting	84.38	0.565
	Decision Tree- Generate Weight	83.97	0.533
	kNN SMOTE sampling	89.15	0.486
Hold-Out	Decision Tree Gradient Boosting	89.27	0.498
	Decision Tree Generate Weight	89.5	0.267
Ensemble	Stacking	89.21	0.211
	Voting	88.5	0.290

On the other hand, using SMOTE (see figure 10), the kappa increased to 0.486; this indicates that there may be a need to tune hyperparameters and extract relevant features from the data. In process Q3 kNN (opt), we use a loop parameter to determine optimum values of k. (See figure 11) With the highly varying lines on the chart, it can be concluded that there may be overfitting or underfitting of data.

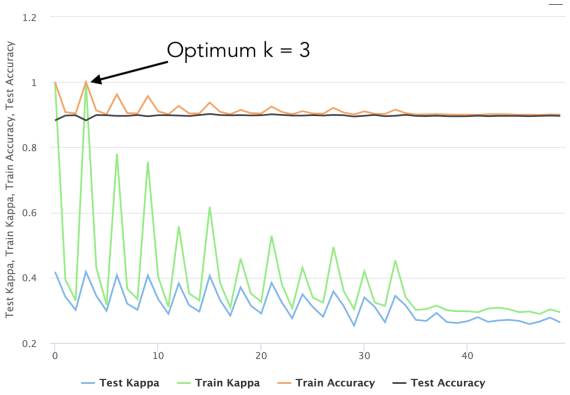


Figure 11 - Output of loop parameter for best k

Irrespective of better performance by kNN, we are not dealing with rare instances in the dataset, a decision tree boosted using XGBoost would prove to deliver the best results in general as looking at the confusion matrix (See figure 12) we can gather that it outperformed all models in terms of bias-variance tradeoff. Overall, these models are not as accurate as they can be, further addition to unknown details can highly boost the efficiency of the model by eliminating imbalances. All in all, due to the low value of kappa, sophisticated modelling may be required.

Figure 12 - GBT confusion matrix in holdout (top) vs cross validation (bottom)

	true no	true yes	class precision
pred. no	5394	563	90.55%
pred. yes	662	1223	64.88%
class recall	89.07%	68.48%	

	true no	true yes	class precision
pred. no	11187	666	94.38%
pred. yes	790	921	53.83%
class recall	93.40%	58.03%	