Predicting Customer Churn using a Sample Credit Card Customer Data Set from Selected Markets

By

Bhambhu, Alka
Hip, Hui Ling
Kwan, Vincent
Perez, Mark Anthony

Presented to

Prof. Resmi Ann Thomas
BA706 – Applied Analytic Modeling

Centennial College

17 December 2021

Outline of the Paper

1.0 INTRODUCTION

It is a rule of thumb in marketing to the retail segment that it costs 5x more to acquire a new customer than to retain one. Thus, developing an end-to-end automated solution that helps predict customer attrition is essential to many marketing organizations. This is especially through for financial institutions, in general, and credit card companies, in particular, since (a) credit cards are essentially a homogenous product that has limited differentiation and (b) the addressable market of credit-worthy individuals that meet both customer lifetime value and individual risk characteristics involve a finite population for which each financial institution needs to compete. It should be unsurprising, therefore, that the group has decided to focus its efforts on this area of research.

2.1 Problem Statement and Objectives

This paper primarily addresses the research question: *What analytics model (or combination of models) might best predict the attrition of credit card customers?*

Responding to this research question would achieve the following objectives:
1) Identify the analytic model/s that best predict credit card customer churn;
2) Identify features/variables that are most associated or have a significant causal relationship with the customer defection; and,
3) Explain how the analytics model/s may applied to business process and decision making.

2.2 Significance of Work

The work can contribute significantly in the following ways:
1) Since the data used by the study is in the public domain, the resulting models are easily replicable;
2) The results of the study can be used to inform as well as fine-tune existing models;
3) New data can be introduced to train the models and expand its applicability;
4) Finally, learnings from the study can be used to avoid repeating any mistakes or errors once the same is applied to larger production data sets.

2.3 Scope and Limitations

The data set used in this study is cross-sectional rather than longitudinal. Thus, no analysis over time is provided. Moreover, the data used needs to be treated as hypothetical and may have been subjected to treatment ahead of its use in this study. No information is accessible to back-test the veracity of the data nor is documentation available to trace data lineage.

Behavioral data, such as transactions usage, are absent in the development of these models. Access to and use of such information could greatly enrich not only the predictive power but also the nuances of actual customer behavior the eventually leads to customer attrition.

Methodology-wise, the study is conducted almost exclusively using SAS Enterprise Miner 13.20. Models developed include decision tree, multivariate regression, and neural network that for the most part use algorithms that are predefined in the SAS application and which were part of the instruction provided during the course of the semester.

Finally, no data on the customer lifetime value is included. Consequently, the models cannot be used to determine the profitability of attriting customers. Knowing such information

would have allowed researchers to determine whether the defecting customers are worth retaining in the first place.

2.4 Survey of Related Studies

Customer churn is a well-studied area and the application of multivariate analysis as well as machine learning models have been proven to be useful. It should be noted that while multivariate analysis has traditionally been favored by financial institutions for their maturity, stability and interpretability, there is a growing body of knowledge that has introduced the use of machine learning into the credit card marketer-modeler's toolkit.

For example, Zhang et al (2008) showed how back propagation neural networks can be used on customer transaction data to predict customer exit and how a set of decision rules "can be extracted from the trained neural networks in order to improve the clarity and explicability of the loyalty model." [1] Kumar and Ravi (2008), on the other hand, developed an ensemble model that included Multilayer Perceptron (MLP), Logistic Regression (LR), decision trees (J48), Random Forest (RF), Radial Basis Function (RBF) network and Support Vector Machine (SVM).[2] To achieve the same objective, Tsai and Lu (2009) considered hybrid models by combining back-propagation artificial neural networks (ANN) and self-organizing maps (SOM) for churn prediction. Their "experimental results showed that the two hybrid models outperform the single neural network baseline model in terms of prediction accuracy and Types I and II errors over the three kinds of testing sets."[3] Celik and Osmanoglu (2019), in a review, examined the use of machine learning algorithms such as artificial neural networks (ANN), decision trees, support vector machines (SVM), naive bayes, k-nn and extreme gradient boosting (XGBoost), Cox proportional hazard model and deep learning techniques in the development of customer churn models.[4] More recently, Poveda & Galpin (2020) used a selection of supervised Machine Learning models (LightGBM, XGBoost, Random Forest and Logistic Regression) to identify the likelihood of credit card customer attrition using data about clients of a bank in Colombia. Of the models developed, LightGBM was selected based on the analysis of the ROC curves using AUC. [5]

What follows is a much simpler version of all these studies (and many more like them), but which has been developed to test the basic understanding of the authors about the content taught in the course.

[1] Tao Zhang, Bo Yuan, & Wenhuang Liu (2008). *Predicting Credit Card Customer Loyalty Using Artificial Neural Networks*. Proceedings of the 11th Joint Conference on Information Sciences.
[2] Dudyala Anil Kumar & Vadlamani Ravi (2008). *Predicting credit card customer churn in banks using data mining*. International Journal of Data Analysis Techniques and Strategies 1(1):4-28
[3] Chih-Fong Tsai, Yu-Hsin Lu (2009). *Customer churn prediction by hybrid neural networks*. Expert Systems with Applications, Volume 36, Issue 10, Pages 12547-12553.
[4] Ozer Celik & Usame Osmanoglu (2019). *Comparing to Techniques Used in Customer Churn Analysis*. Journal of Multidisciplinary Developments. 4(1), 30-38, 2019
[5] Carlos Alvaro Rico-Poveda & Ixent Galpin (2020). *Forecasting Credit Card Attrition using Machine Learning Models*. ICAIW 2020: Workshops at the Third International Conference on Applied Informatics 2020, October 29–31, 2020, Ota, Nigeria.

3.0 DATA AND METHODOLOGY

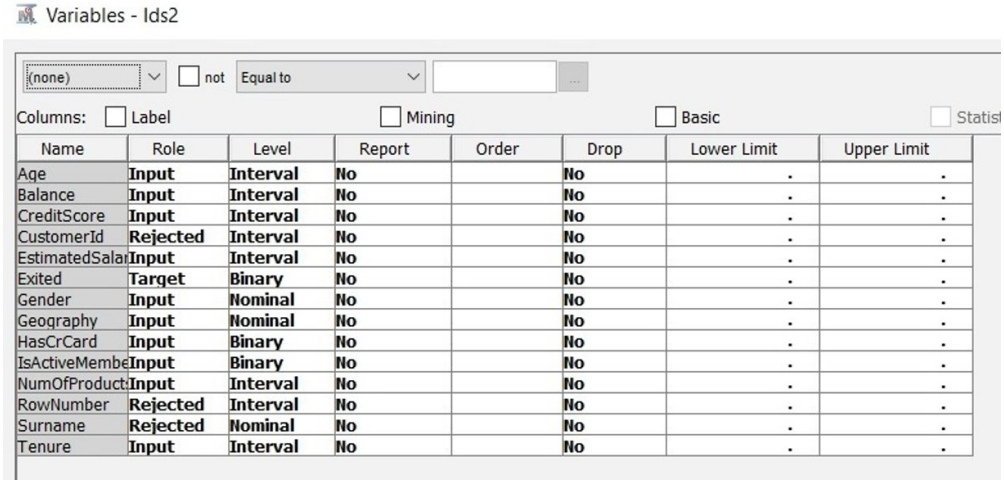
3.1 Data Description, Extraction, and Pre-processing

Data Description

For this study, group explored several data sets available in the public domain in sites such as kaggle.com. It was from the said site that the Bank Customer Churn[6] data set was selected.

This particular data set contained details of a bank's customers where the observed variable indicated whether a customer dropped out of an undisclosed bank credit card portfolio (i.e., closed his or her account) or continued to be a customers.

The data set consisted of 10,000 observations and 14 variables with no missing values despite zero balances reported for certain observations in selected locations. The basic features of the data are described in the following table.


Variables - Ids2

(none) not Equal to

Columns: Label Mining Basic Statis

Name	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
Age	Input	Interval	No		No	.	.
Balance	Input	Interval	No		No	.	.
CreditScore	Input	Interval	No		No	.	.
CustomerId	Rejected	Interval	No		No	.	.
EstimatedSalar	Input	Interval	No		No	.	.
Exited	Target	Binary	No		No	.	.
Gender	Input	Nominal	No		No	.	.
Geography	Input	Nominal	No		No	.	.
HasCrCard	Input	Binary	No		No	.	.
IsActiveMembe	Input	Binary	No		No	.	.
NumOfProduct	Input	Interval	No		No	.	.
RowNumber	Rejected	Interval	No		No	.	.
Surname	Rejected	Nominal	No		No	.	.
Tenure	Input	Interval	No		No	.	.

Note that for the purpose of the study, 'CustomerID' and 'RowNumber' was dropped since SAS autogenerates its own reference ID. Moreover, 'Surname' was dropped for lack of relevance to the study objectives as well as data privacy purposes.

To further describe the data, a simple data dictionary is provided in the following table. The comments column was from the observations of the researchers.

Element or value display name	Description	Data type	Data Level	Required ?	Accepts null value?	Comments
Geography	Location of customer	Character	Nominal	Y	Y	Data related to France, Germany, Spain
Age	Age	Integer	Interval	Y	Y	Right skewed
Gender	Gender	Character	Binary	Y	Y	Only Male/Female values
RowNumber	Index artifact	Integer	Interval	Y	N	Rejected
CustomerId	Index artifact	Integer	Interval	Y	N	Rejected
Surname	Index artifact	Character	Nominal	Y	N	Rejected
NumOfProducts	No. of products owned by customer	Integer	Interval	Y	Y	Between 1 to 4
CreditScore	Customer's credit score	Integer	Interval	Y	Y	Very slightly left-skewed
Balance	Amount stored in account	Float	Interval	Y	N	Zero values exist
Tenure	Integer	Integer	Interval	Y	Y	Length of relationship

[6] https://www.kaggle.com/shivan118/churn-modeling-dataset

Data Extraction

Data was downloaded from the source website in *.csv format. Thereafter, it was imported into a SAS table using the following procedure:

a. From Sample tab, drag File import node onto the diagram workspace.
b. In Train section, select advanced advisor to 'Yes'.
c. Click on Import file, to locate and import data from the device.
d. To save data in SAS format, import Save Data node from the Utility tab.
e. Create a new library and go to Output Format and click on SAS library name.
f. From the open window, select the create library and run the Save Data node.

Data Partitioning

As part of data preparation, the data set is divided into three non-overlapping sets: the training set , the validation set , and the test set . This is performed in order to use only a portion of the data for developing the primary models with the remaining subsets of the data set aside for model validation and testing.

Given the size of the data set, the data was partition in a 50:30:20 split where:

- 50% of the data is devoted to training for preliminary model fitting;
- 30% of the data is devoted for validation purposes to assess the adequacy of the model;
- 20% of the data is devoted to final testing to obtain a final, unbiased estimate of the generalization error of the model.

Data Set Allocations	
Training	50.0
Validation	30.0
Test	20.0

Data Pre-processing

Replacement/Bucketing

The SAS Replacement node was applied to collapse the Tenure variable into 3 categories to reduce the variable's redundancy. The change is reflected in the following table.

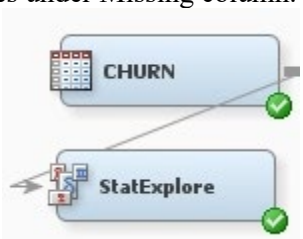
Tenure	Category
0-3	1
4-6	2
7-10	3

Missing Values

One of the most important aspects to consider before running the Regression models is to check whether there is any missing value in the dataset. Missing values can significantly reduce the amount of training data for the modelling and lead to the inability of the prediction formulas to score cases. Consequently, it introduces biased outcome, makes the analysis of

data more difficult, and leads to inefficient model. The primary remedy for missing values in Regression models is through imputation.

The procedure applied to check for missing values in interval variables was as follows:
1. From the Explore tab, select, and drag the StatExplore node onto the diagram workspace.
2. Connect the dataset to the StatExplore node.
3. Run and view the results of the StatExplore node.
4. Go to View > Summary Statistics > Interval Variables.
5. Check the missing values under Missing column.



Interval Variables

Data Role	Target	Target Level	Variable	Missing
TRAIN	Exited	0	Balance	0
TRAIN	Exited	1	Balance	0
TRAIN	Exited	0	Age	0
TRAIN	Exited	1	Age	0
TRAIN	Exited	0	NumOfProducts	0
TRAIN	Exited	1	NumOfProducts	0
TRAIN	Exited	0	EstimatedSalary	0
TRAIN	Exited	1	EstimatedSalary	0
TRAIN	Exited	0	CreditScore	0
TRAIN	Exited	1	CreditScore	0

Since there was no missing value in the dataset, imputation is not applicable.

Outliers and Skewed Values

Regression models are sensitive to extreme values in the input space. Since the outliers and skewed values can be selected over inputs that produce better predictions, the model performance may be affected. Therefore, treating the outliers and skewed values is necessary so long proper investigation was done. There are several techniques to treat outliers and skewed values. Cap and Floor and Log Transformation have been used in the models creation. The limits method for Cap and Floor is based on standard deviation from the mean by default. Log Transformation can reduce skewness and make patterns more interpretable by replacing the affected variable x with log(x).

The skewness of the interval variables in the dataset was verified using the StatExplore node.

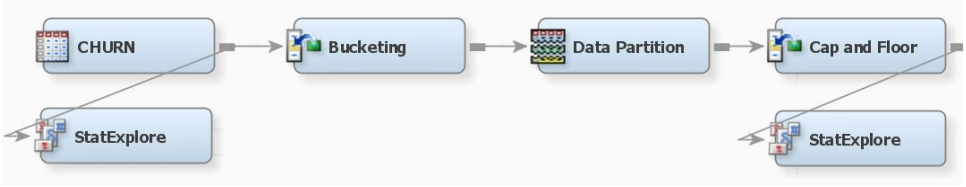
Interval Variables

Data Role	Target	Target Level	Variable	Skewness ▼
TRAIN	Exited	1	NumOfProducts	1.57032
TRAIN	Exited	0	Age	1.01132
TRAIN	Exited	0	NumOfProducts	0.745568
TRAIN	Exited	1	Age	0.077978
TRAIN	Exited	0	EstimatedSalary	0.002085
TRAIN	Exited	1	EstimatedSalary	-0.0331
TRAIN	Exited	0	CreditScore	-0.07161
TRAIN	Exited	1	CreditScore	-0.14108
TRAIN	Exited	0	Balance	-0.14111
TRAIN	Exited	1	Balance	-0.51273

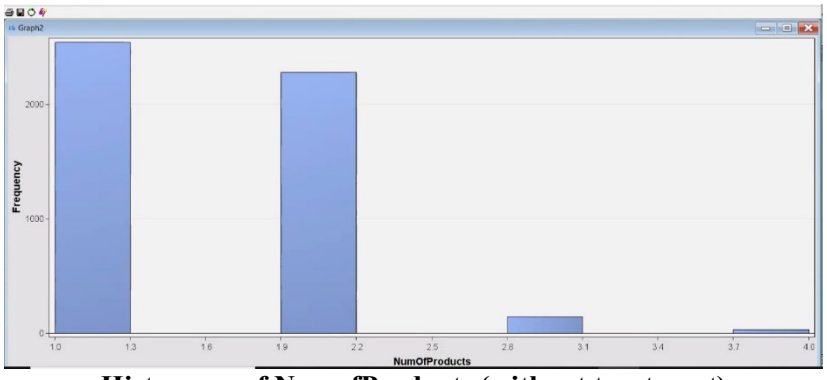
Cap and Floor

NumOfProduct with Target Level 1 exhibited the highest Skewness. Thus, the procedure to Cap and Floor was applied as a possible remedy in the following manner:

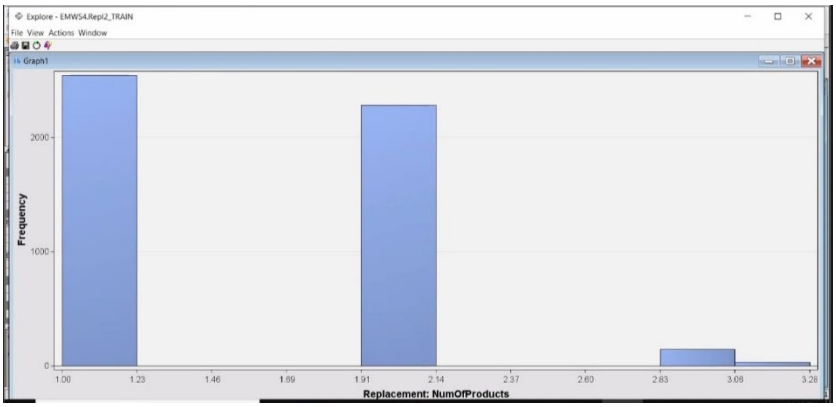
1. From the Modify tab, select, and drag the Replacement node onto the diagram workspace.
2. Rename the Replacement node to Cap and Floor.
3. Connect the Data Partition node to the Cap and Floor node.
4. Run the Cap and Floor node.
5. From the Explore tab, select, and drag the StatExplore node onto the diagram workspace.
6. Connect the Cap and Floor node to the StatExplore node.
7. Run and view the results of the StatExplore node.
8. Go to View > Summary Statistics > Interval Variables.
9. Check the skewness level under Skewness column.



After running the cap and floor, see the following histograms of 'NumProduct' for comparison:



Histogram of NumofProducts (without treatment)



Histogram of NumofProducts (with treatment)

Apparently, applying Cap and Floor on 'NumOfProduct' produced dubious results and added no value since observably the distribution of the variable before and after the Cap and Floor treatment did not reasonably change. Therefore, 'NumofProduct' was no longer subjected to this treatment. Moreover, it should be noted that NumOfProduct is a count – it counts how many products a customer uses from 1 to 4.

On the other hand, Cap and Floor improved the skewness of 'Age'. Thus, the treatment was retained for this variable.

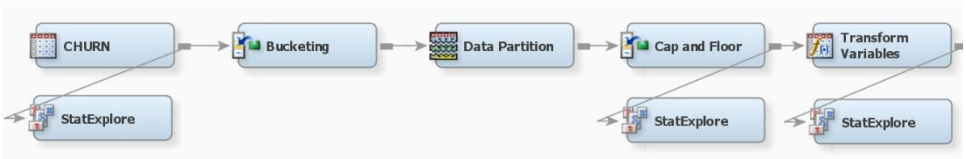
Interval Variables

Data Role	Target	Target Level	Variable	Skewness
TRAIN	Exited	0	REP_Balance	-0.14739
TRAIN	Exited	1	REP_Balance	-0.5279
TRAIN	Exited	0	REP_Age	0.870428
TRAIN	Exited	1	REP_Age	0.032388
TRAIN	Exited	0	NumOfProducts	0.746356
TRAIN	Exited	1	NumOfProducts	1.500731
TRAIN	Exited	0	REP_CreditScore	-0.04693
TRAIN	Exited	1	REP_CreditScore	-0.05286
TRAIN	Exited	0	REP_EstimatedSalary	-0.00202
TRAIN	Exited	1	REP_EstimatedSalary	-0.00359

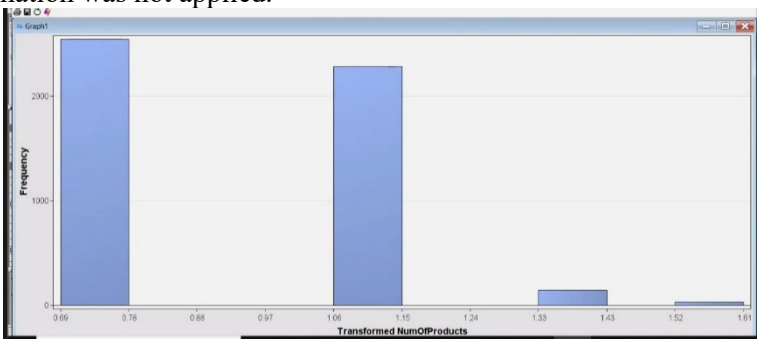
Log Transformation

Next, the use of the Log Transformation node was explored. The procedure for Log Transformation is as follows:

1. From the Modify tab, select, and drag the Transform Variables node onto the diagram workspace.
2. Connect the Cap and Floor node to the Transform Variables node.
3. Right click on Transform Variables node and select Edit Variables…
4. Change only the Method from Default to Log.
5. Run the Transform Variables node.
6. From the Explore tab, select, and drag the StatExplore node onto the diagram workspace.
7. Connect the Transform Variables node to the StatExplore node.
8. Run and view the results of the StatExplore node.
9. Go to View > Summary Statistics > Interval Variables.
10. Check the skewness level under Skewness column.



As in Cap and Floor, log transformation did not improve 'NumOfProducts' and the distribution didn't change. Later in the process, it was also observed that model performance worsened when the transformed version of NumOfProducts was used. Thus, log transformation was not applied.

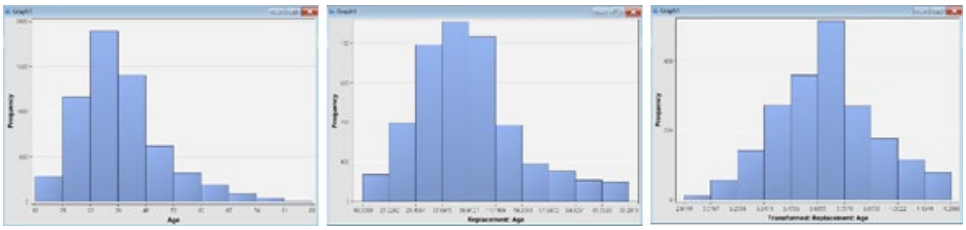


The level of skewness of Age variable improved further with the Log Transformation.

Interval Variables

Data Role	Target	Target Level	Variable	Skewness
TRAIN	Exited	0	REP_Balance	-0.14739
TRAIN	Exited	1	REP_Balance	-0.5279
TRAIN	Exited	0	LOG_REP_Age	0.15071
TRAIN	Exited	1	LOG_REP_Age	-0.55985
TRAIN	Exited	0	NumOfProducts	0.746356
TRAIN	Exited	1	NumOfProducts	1.500731
TRAIN	Exited	0	REP_CreditScore	-0.04693
TRAIN	Exited	1	REP_CreditScore	-0.05286
TRAIN	Exited	0	REP_EstimatedSalary	-0.00202
TRAIN	Exited	1	REP_EstimatedSalary	-0.00359

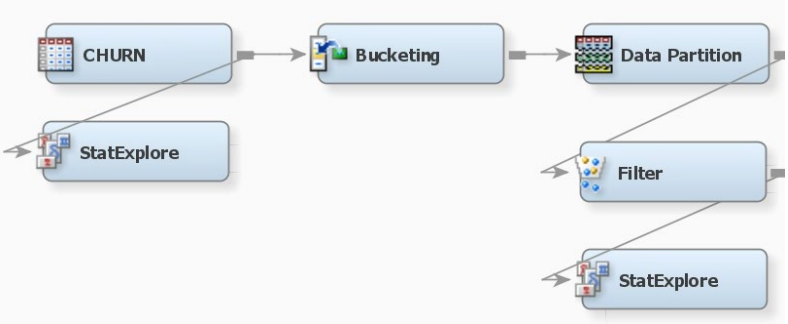
Below are the plots on Age variable: Original (left), after Cap and Floor (middle), and after Log Transformation (right), which provide a clear visualization on how the skewness of Age variable is improved throughout and becomes more normally distributed.



Filter

In addition to Cap and Floor and Log Transformation, the Filter node was also applied to explore an alternative technique to treat the outliers and skewed values. Filter excludes extreme outliers that to be excluded from the analysis. By filtering the extreme outliers from the training data, the parameter estimates become more stable, thus produce model with better performance. The procedure to Filter is as follows:

1. From the Sample tab, select and drag the Filter node onto the diagram workspace.
2. Connect the Data Partition node to the Filter node.
3. Run the Filter node.
4. From the Explore tab, select, and drag the StatExplore node onto the diagram workspace.
5. Connect the Filter node to the StatExplore node.
6. Run and view the results of the StatExplore node.
7. Go to View > Summary Statistics > Interval Variables.
8. Check the skewness level under Skewness column.



Filtering deteriorates the level of skewness of 'Age' unlike when Cap and Floor and Log Transformation is applied.

Interval Variables

Data Role	Target	Target Level	Variable	Skewness
TRAIN	Exited	0	Balance	-0.14794
TRAIN	Exited	1	Balance	-0.52325
TRAIN	Exited	0	Age	0.753617
TRAIN	Exited	1	Age	-0.01759
TRAIN	Exited	0	NumOfProducts	0.440439
TRAIN	Exited	1	NumOfProducts	1.362356
TRAIN	Exited	0	EstimatedSalary	-0.00473
TRAIN	Exited	1	EstimatedSalary	0.002602
TRAIN	Exited	0	CreditScore	-0.04154
TRAIN	Exited	1	CreditScore	-0.03101

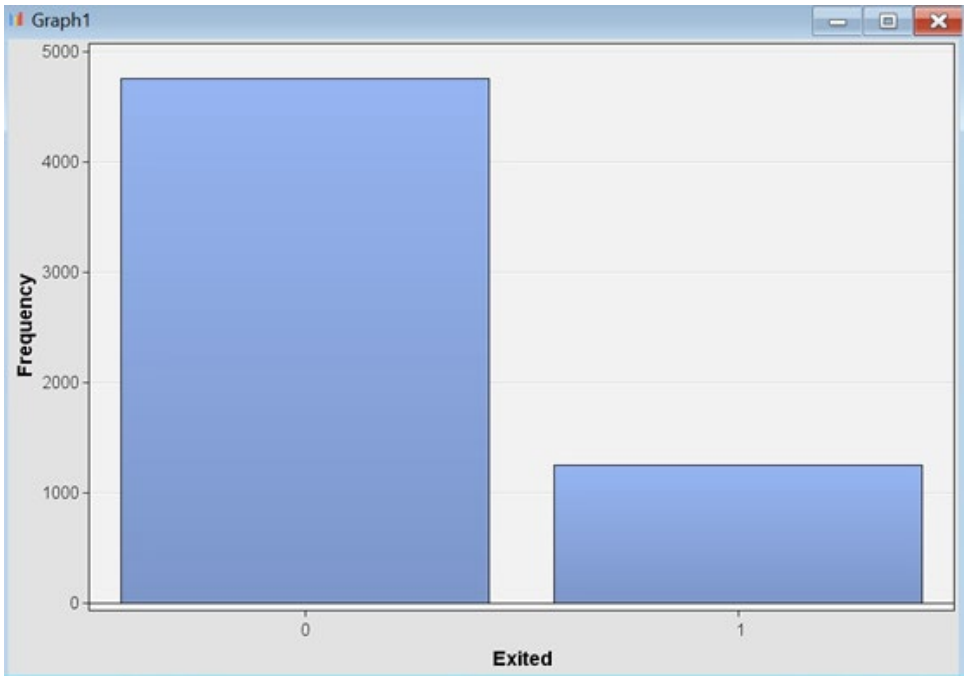
Moreover, filtering results in dropping observations just because they are outliers even though such may be legitimate observations. To minimize the loss of valuable data, only Cap and Floor and Log Transformation were used in building the models.

3.2 Exploratory Data Analysis

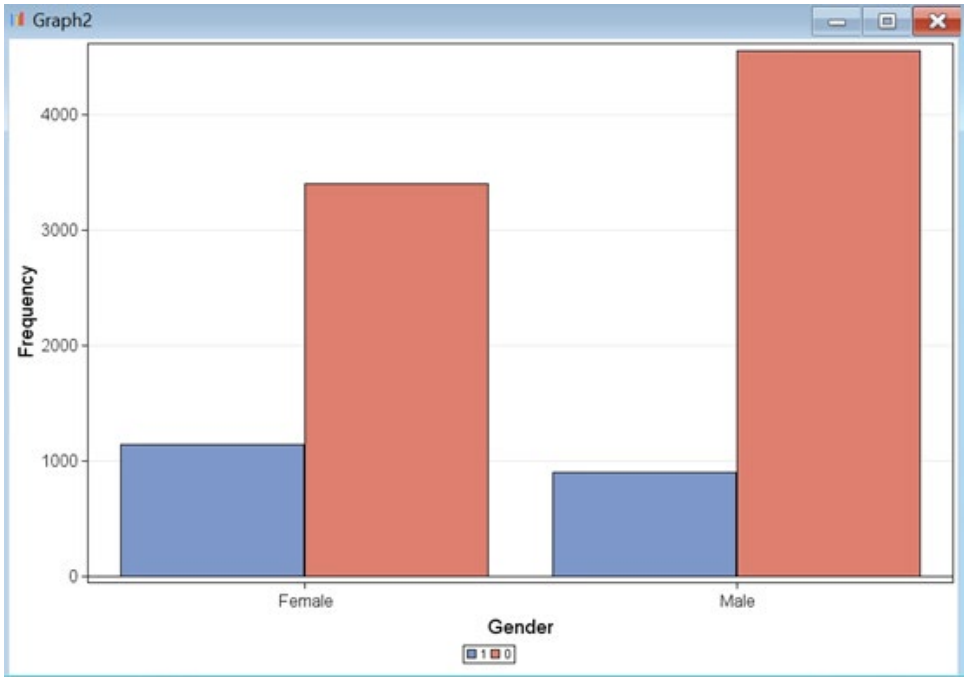
Exploring data prior to modelling is essential to discover initial patterns and characteristics of the data. It gives an overview of the significant points and important trends as well as identifies hidden insights and potential relationships of the data.

The following are some choice visualizations that describe the relationship of the variables with the number of customers who exited the bank.

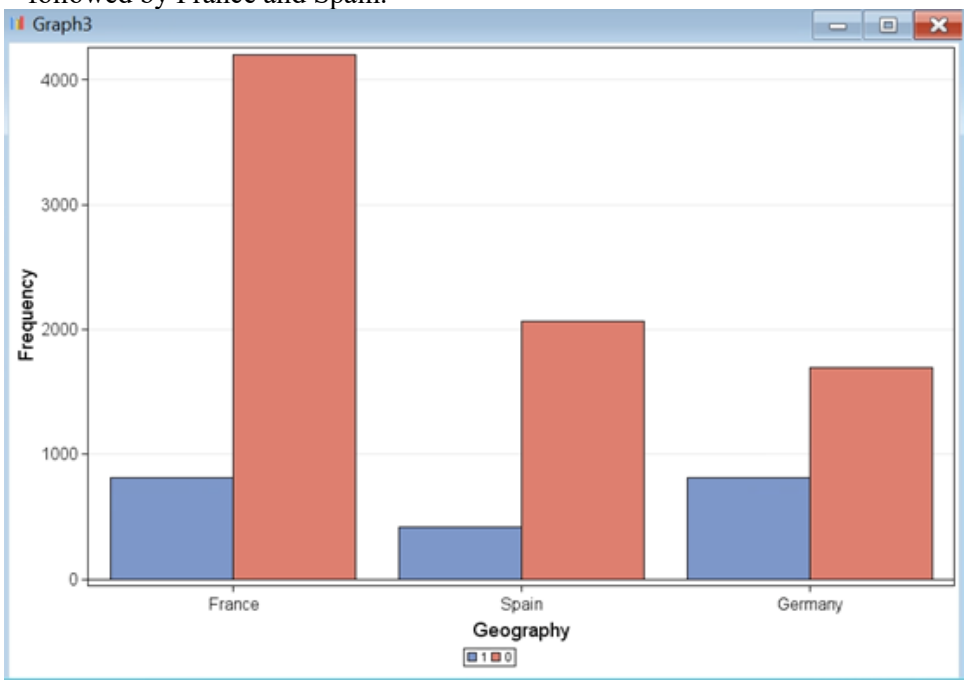
- The number of customers who exited the bank, which is 2037 is lower than the number of customers who did not exit the bank, which is 7963. Therefore, the proportion of customers who exited the bank is 0.2037.



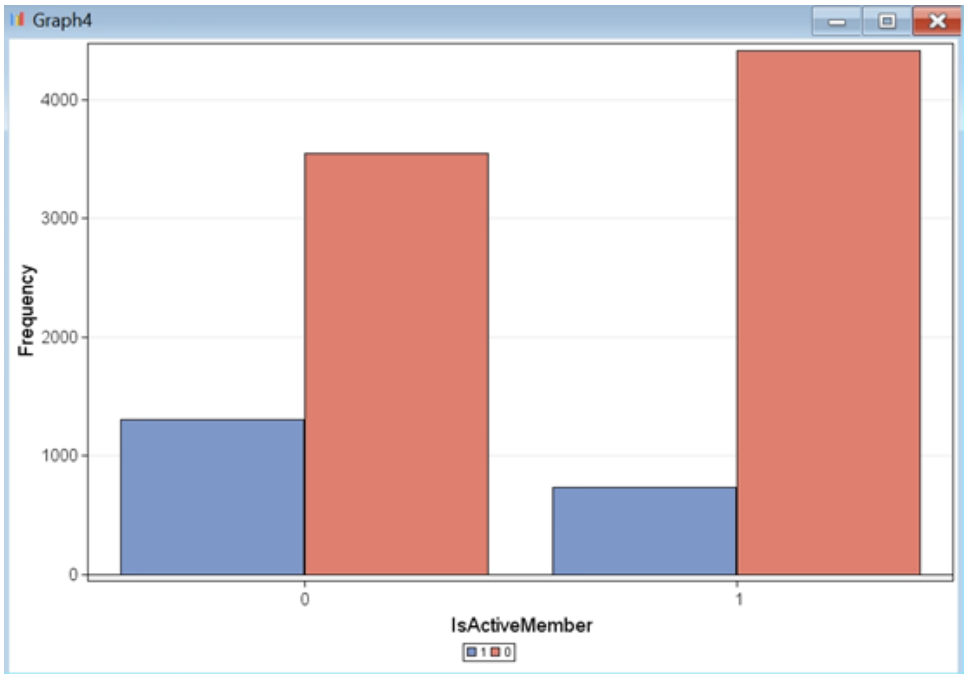
- The female customers exited the bank more often than the Male customers.



- Among the 3 countries, the exited rate of the customers in Germany is highest, followed by France and Spain.

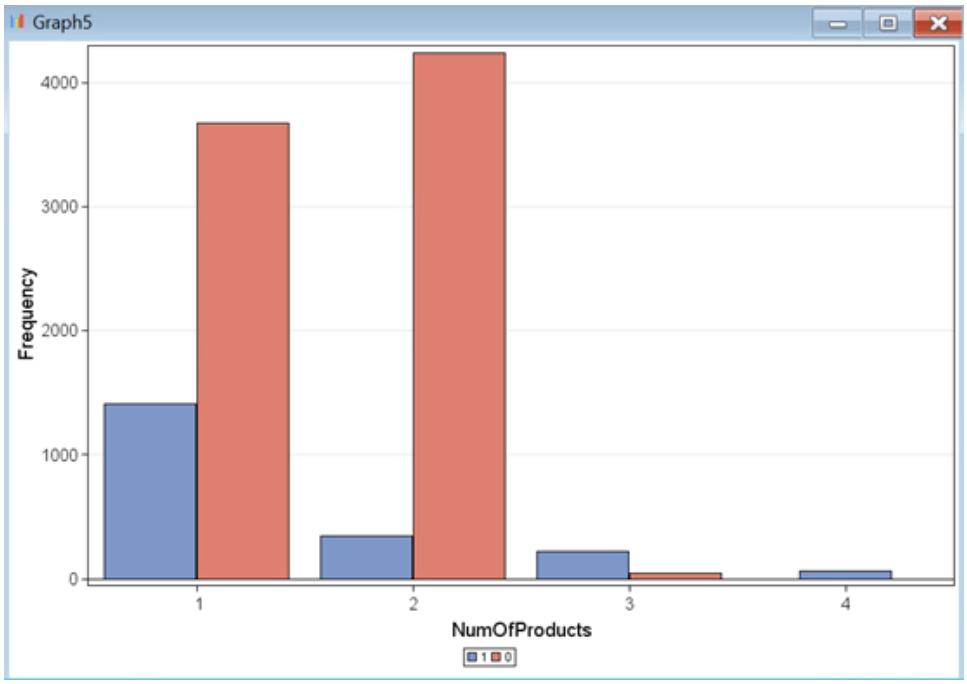


- The inactive customers exited the bank more often than the active customers. In other words, customers who are not using the bank services frequently exited the more.

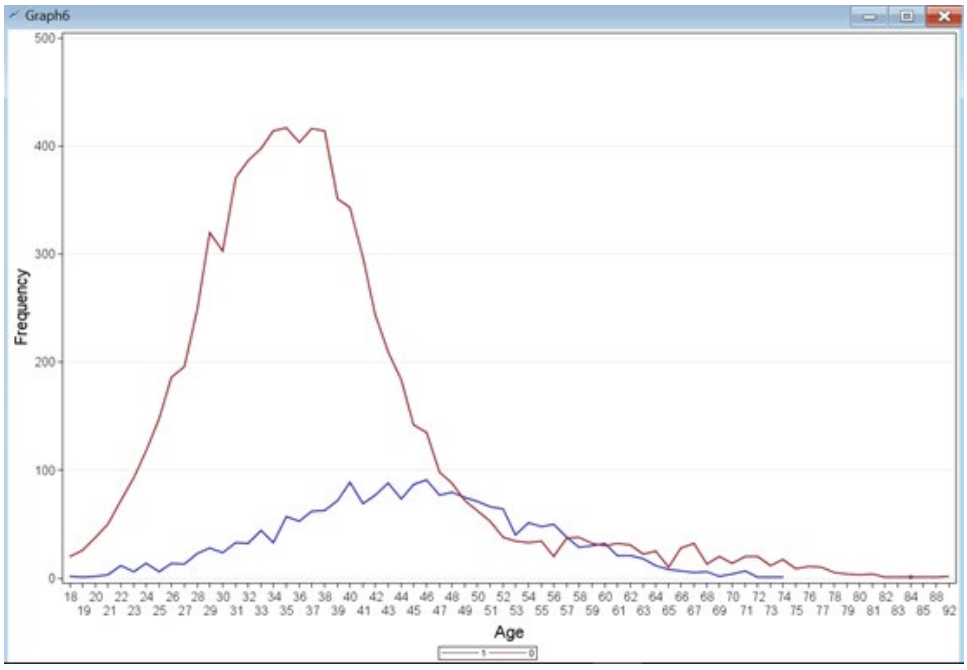


- Customers who purchased only 1 product have higher rate of exiting. On the other hand, there is a small number of customers who purchased 4 products exited the bank. This group of customers might be long-term customers who have been

benefited from different products years ago, but the products are no longer available to them now.

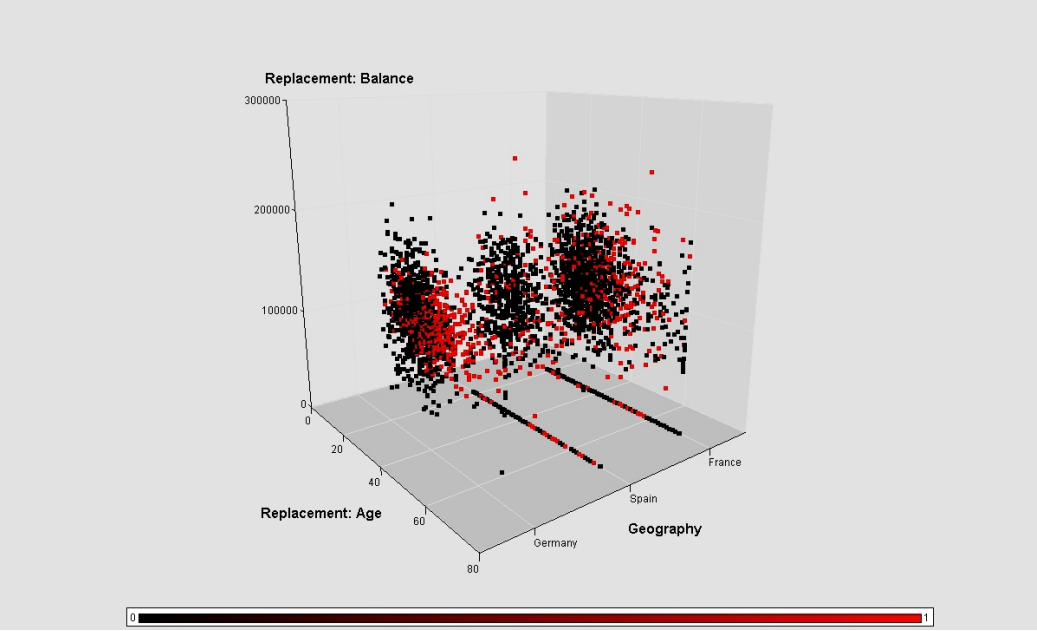


- Customers between age 40 and 50 exited bank more often than younger or older customers.



The team also conducted further exploratory analysis to understand the data's underlying structure. The purpose was to reveal whether there could be a structural pattern among variables such as 'Balance', 'Age', and 'Geography' using a 3D scatterplot where the

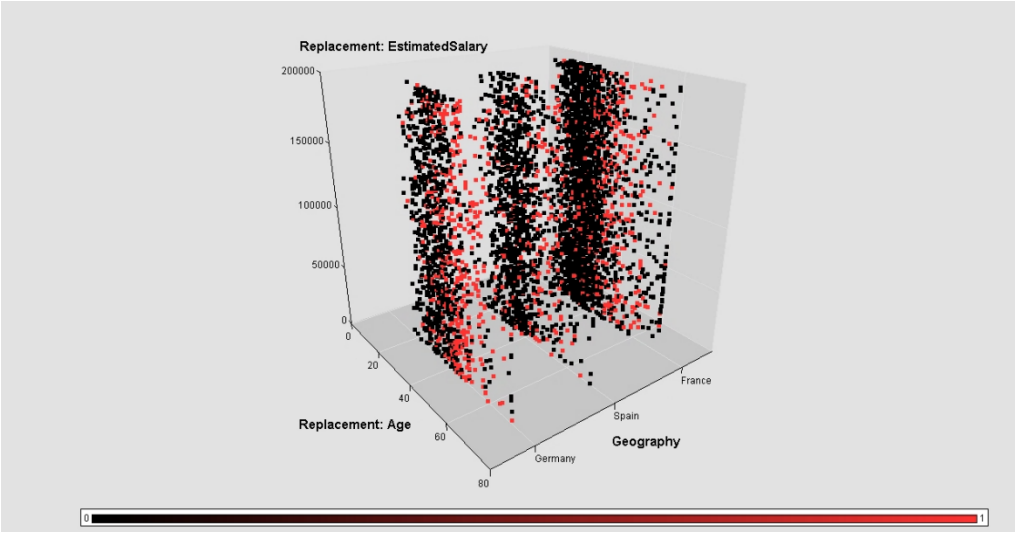
observations were coloured either Red or Black depending on whether they Exited (1, Red) or not (0, Black).



Observations:

1. There are three clusters according to the region the data originated from (Germany, Spain, or France). There is a clear pattern in the distribution of the Target variable (Exited) according to the region the data originates from. It can be clearly seen that older German customers exit more often. French customers who Exited have a more scattered distribution, but it can still be seen that older customers exit more often. The same pattern exists for Spanish customers, but it is more subtle.
2. On first examination, Balance doesn't seem to have a material effect on Exited status. However, careful inspection shows a slight pattern with regards to relationship between Balance vs Exited.
3. There is a clear pattern with regards to Age – older customers are more likely to Exit than younger customers.
4. The long line of observations on the floor of the graph are customers with a balance of zero. From a first glance, these accounts share the patterns as the other accounts.

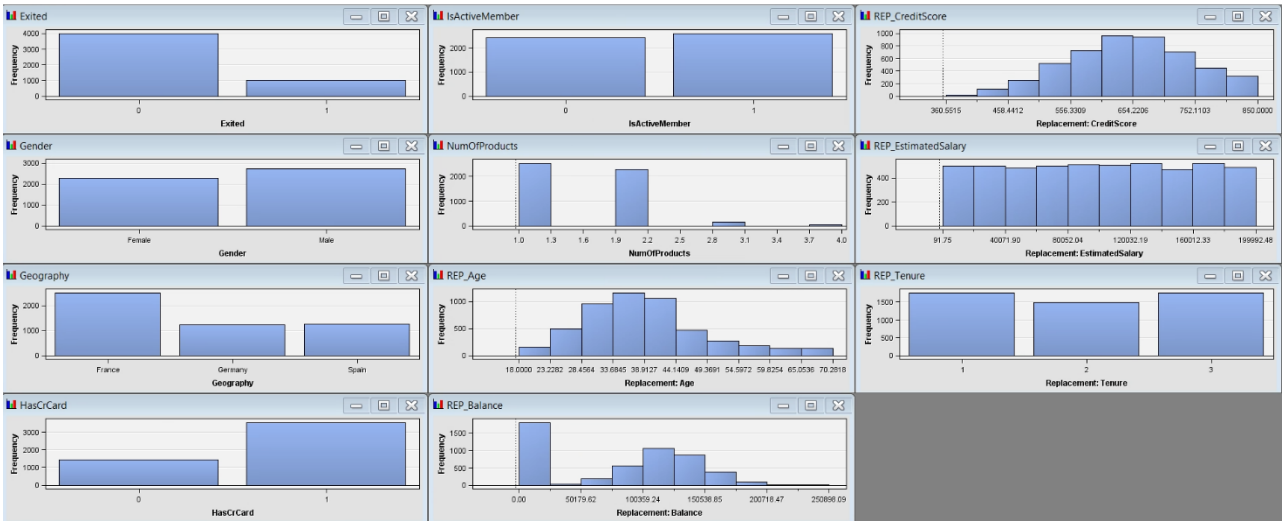
Additionally, a 3D scatterplot with 'Age', 'Geography' and 'EstimatedSalary'.



Observations:

1. The correlation between Exited and Age is more apparent in this graph.
2. There seems to be a slight pattern with regards to Exited vs EstimatedSalary. However, it is hard to discern from this graph alone.
3. The distribution of EstimatedSalary is uniform across all Salary ranges. This is reinforced by a subsequent visualization of the histogram of the EstimatedSalary variable (shown below).

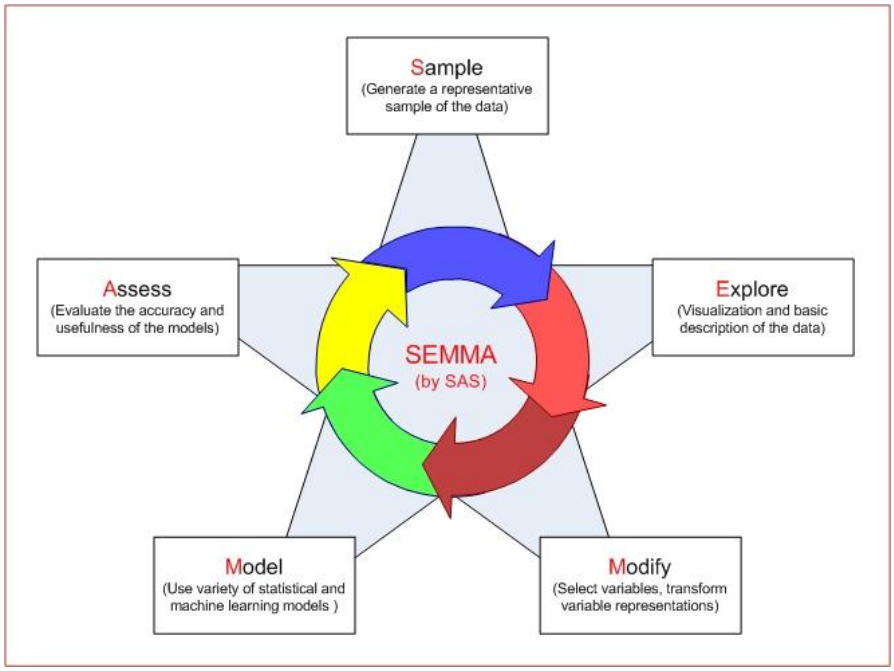
Finally, a visualization was created for all variables after applying 'Cap and Floor' and 'Transformation' operations.



It can be observed that the data transformation procedures were successful for 'Age' which is not as significantly skewed without the transformation. The transformation also applied to 'Balances' but affected only those whose balances that were non-zero. It should be noted that zero balances are naturally occurring and are taken as reflections of reality-on-the-ground rather than indicative of a bias in the data that needs to be corrected.

3.3 Conceptual Framework

The study follows the SAS trademarked SEMMA to complete the model development aspects of its data mining activities.



To comply with the course requirements for the completion of this paper, the models to be developed to answer the research question must include decision tree, regression, and neural network.

As for selection criteria, this paper also applies what has been prescribed in the course – Average Squared Error, Misclassification Rate, and the ROC Index.

Model Assessment (Goodness of Fit)

Statistics are used to determine which model fits best.

	Type of Prediction	Statistic
Decision Tree Logistic Regression Neural Network	Decision	Misclassification Rate
Decision Tree Logistic Regression Neural Network	Estimate	Average Squared Error
Linear Regression	Estimate	R-Square

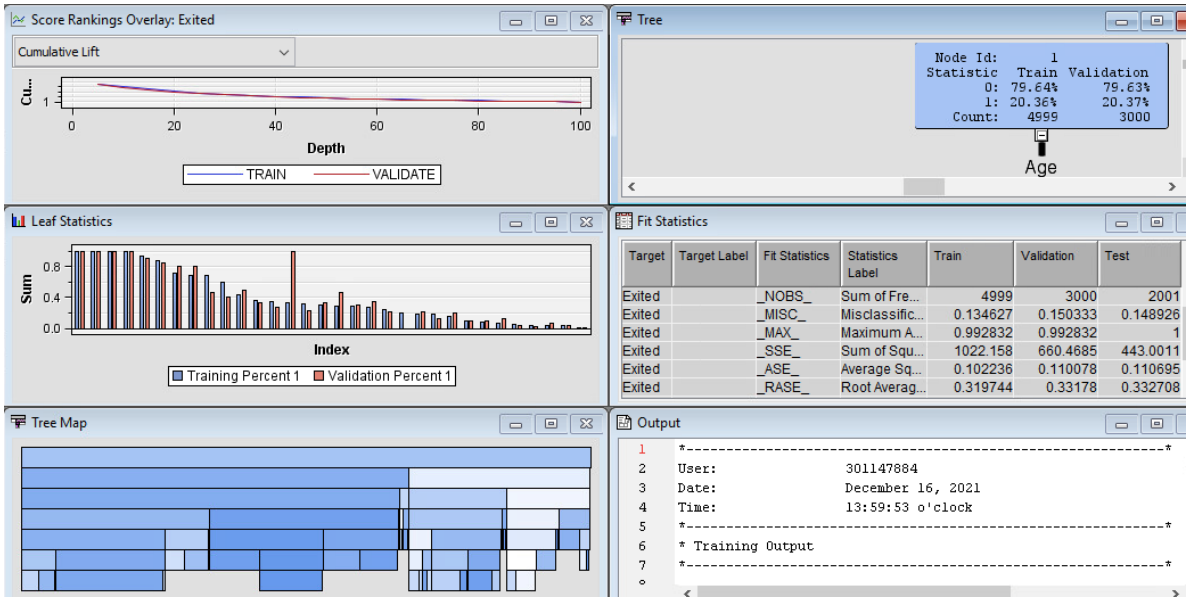
4.0 MODEL DEVELOPMENT AND TESTING

4.1 Decision Trees

Maximal Tree

The maximal tree is the tree grown before any pruning takes place. Procedure to build a maximal tree:

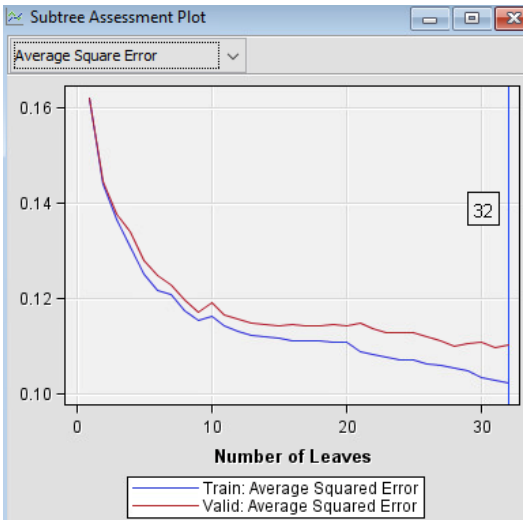
1. From the Model tab, select and drag the Decision tree node onto the diagram workspace.
2. Connect it to data partition node and rename it 'Maximal tree'.
3. Click on Interactive tool in the Train section, keeping all other properties as default.
4. Instead of selecting each split sequentially, right click on the age node and select Train node, an automated way to grow maximal tree.
5. In Train section, select 'Yes' for Use Frozen Tree tab, to prevent the maximal tree from being changed by other property settings when the flow is run.



First split: Age

Competing splits: NumofProducts, Geography, isActiveMember and Balance.

Using the Validation Average Square Error criterion, the performance on the validation sample only improves up to a tree of, approximately, 28 leaves.

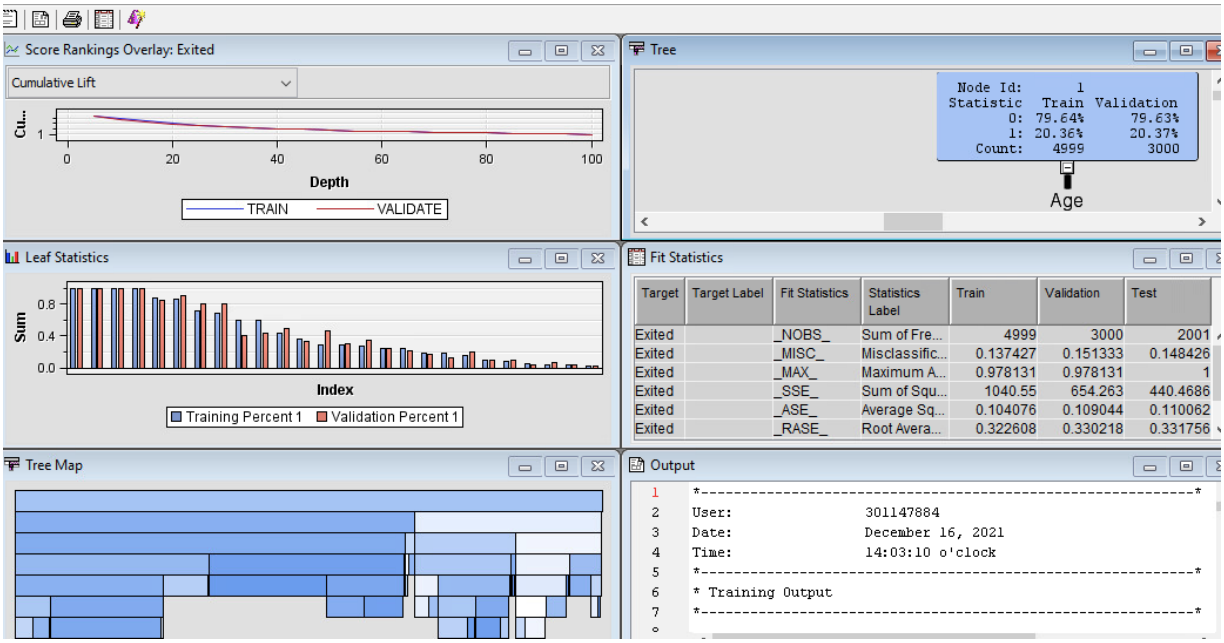


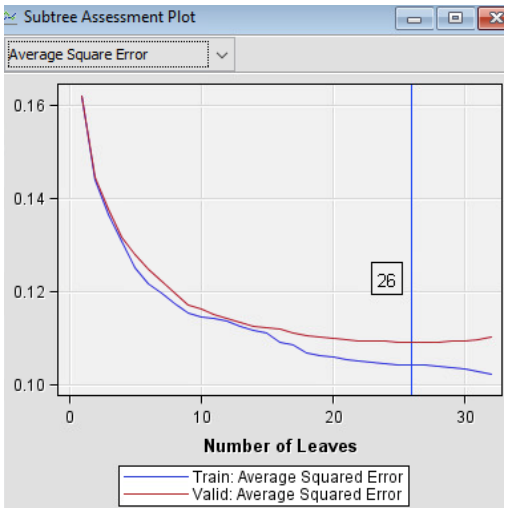
Average Square Error Tree (2 Way Split)

To reduce the model complexity and improve the performance of the model, maximal tree will be pruned. The default method used to prune the maximal tree is Assessment. The Assessment Measure property specifies the optimality measure used to select the best tree in the sequence. For the first model, the Average square Error as an assessment measure is used.

Procedure to build Probability tree:

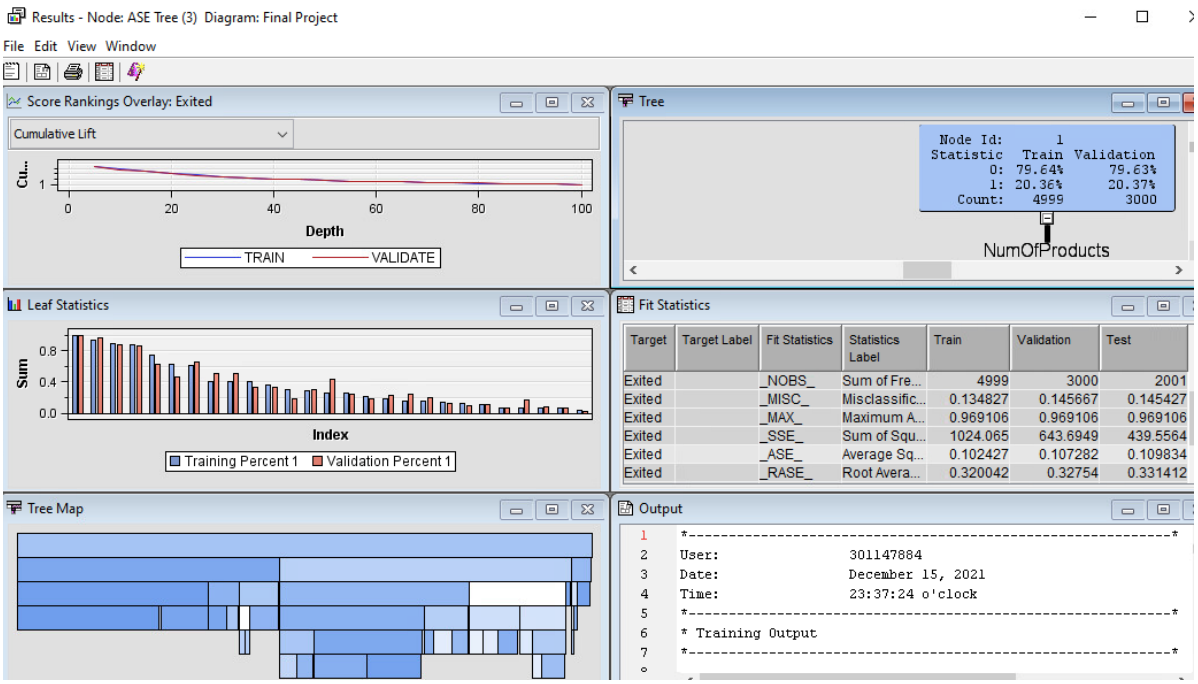
1. From the Model tab, select and drag the Decision tree node onto the diagram workspace.
2. Connect it to data partition node and rename it 'ASE tree (2)'.
3. Change the assessment measure to Average Square Error.
4. With maximum 2 branches and keeping other properties as default, run the model.

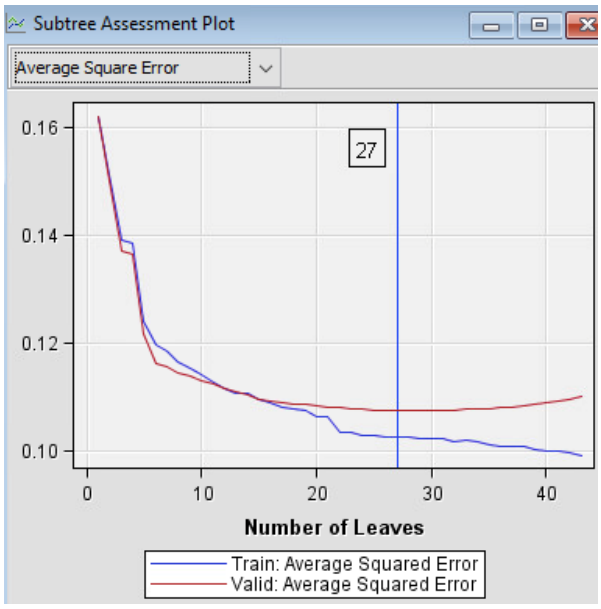




Like maximal tree, age is the first split and competing splits are NumofProducts, Geography, isActiveMember and Balance. The number of leaves in optimal tree has decreased to 26 in 2-way split.

Average Square Error Tree (3 Way Split)
Repeat the procedure on creating Decision tree by naming the node as 'ASE Tree (3)' and update the maximum branches to 3.

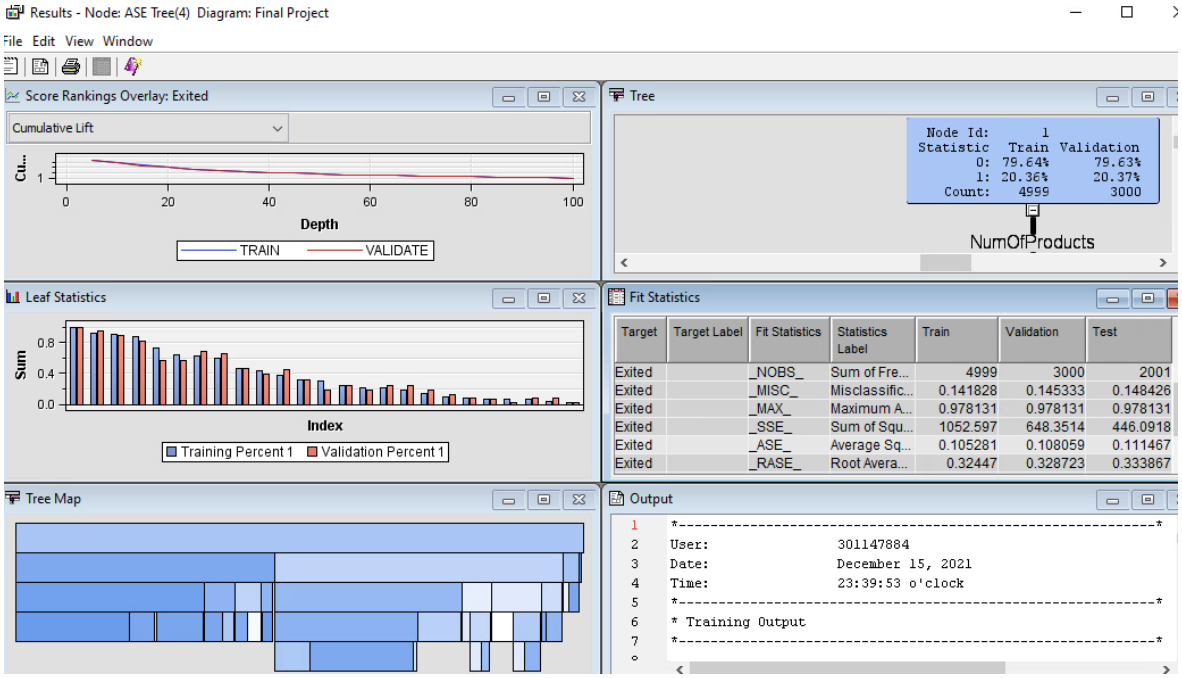


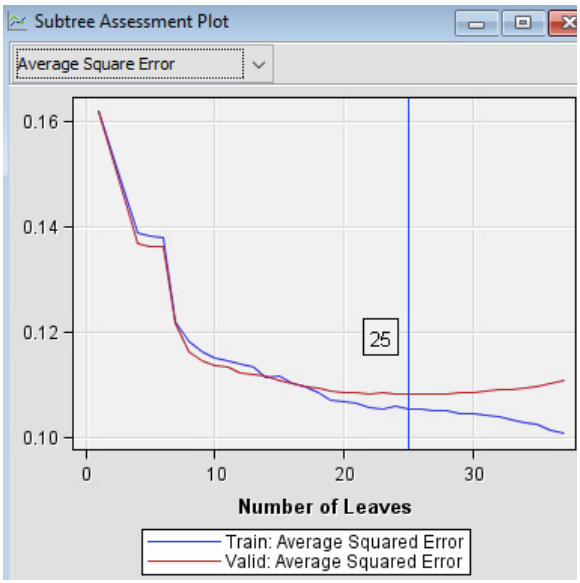


For this tree, NumofProducts is the first split with Age, Geography, is ActiveMember and Balance as competing splits. Due to increase in number of branches, number of products owned by customer has become the most important variable.

Average Square Error Tree (4 Way Split)
Repeat the procedure on creating Decision tree by naming the node as 'ASE Tree (4)' and update the maximum branches to 4.

Sub tree assessment plot shows that 25 leaf tree is optimal. In terms of first split and competing splits,4 ways split tree is similar to 3 ways probability tree.

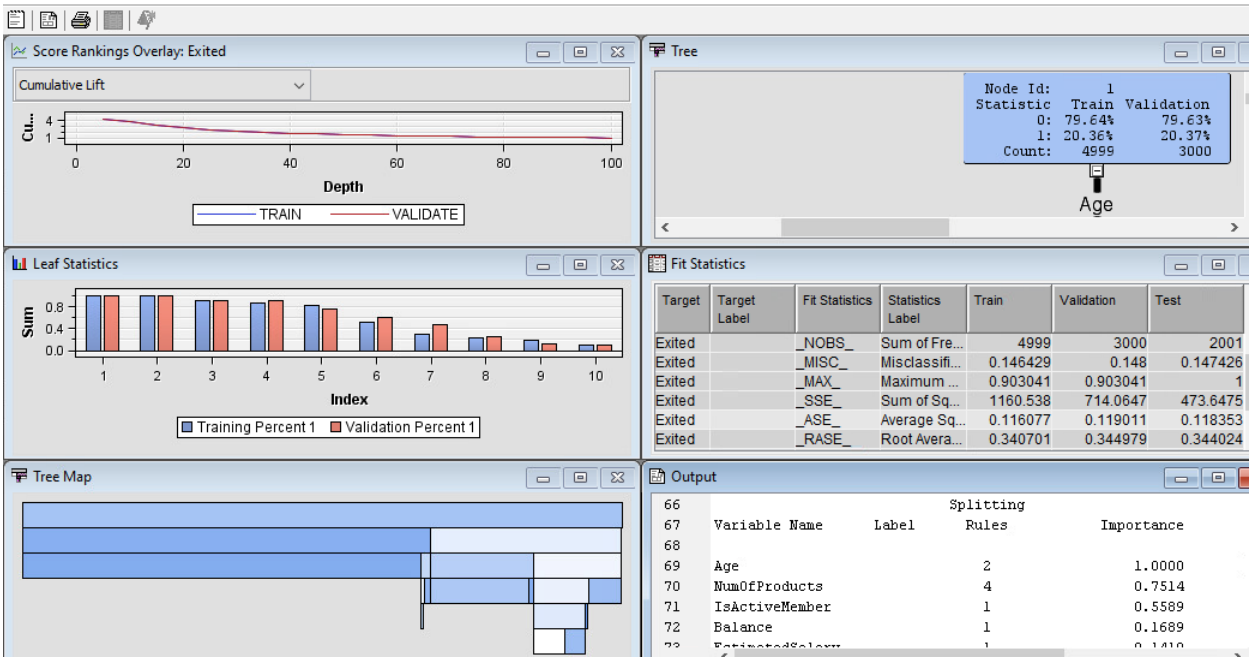


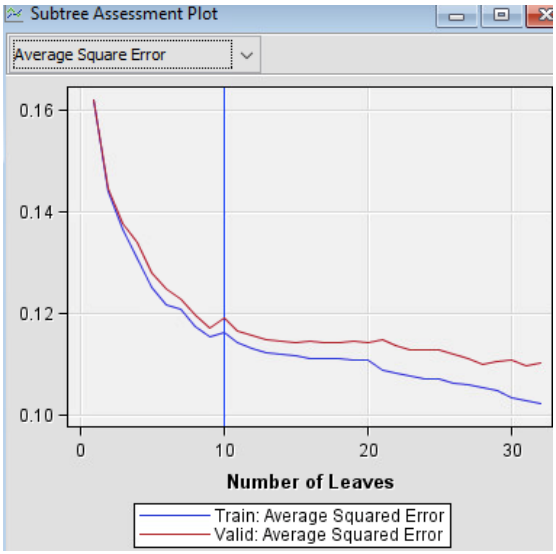


Misclassification Tree (2 Way Split)

Procedure to build Probability tree:

1. From the Model tab, select and drag the Decision tree node onto the diagram workspace.
2. Connect it to data partition node and rename it 'Misclassification tree (2)'.
3. Change the assessment measure to Misclassification rate.
4. With maximum 2 branches and keeping other properties as default, run the model.

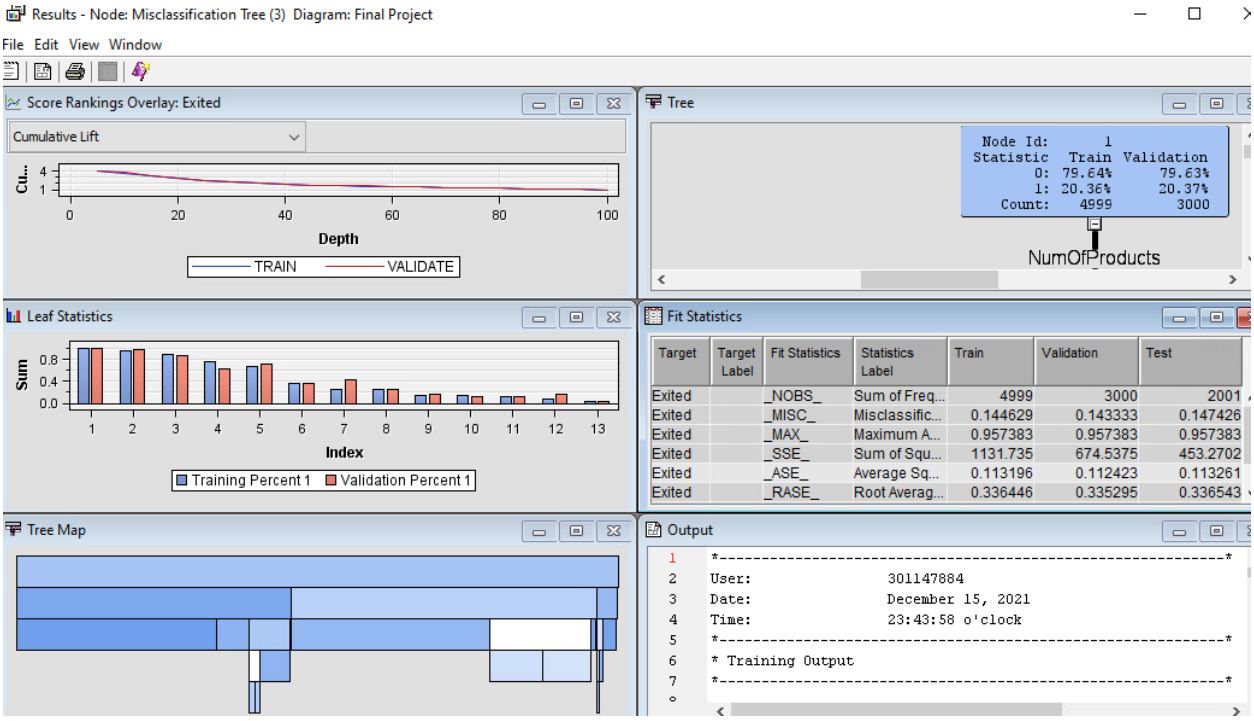


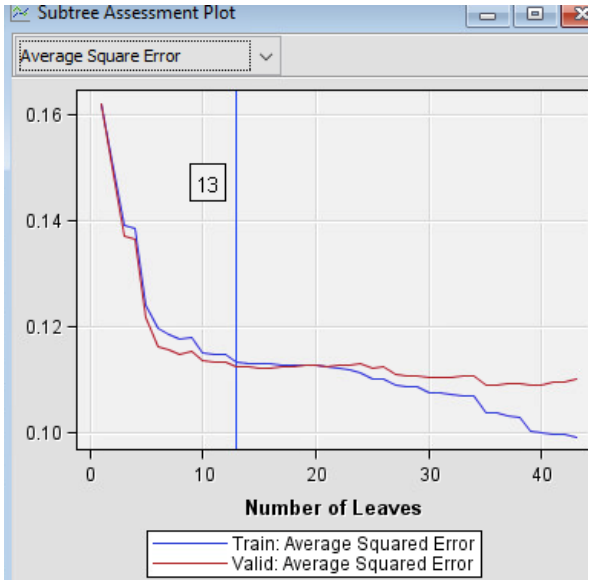


Like other 2-way split trees, age is the first split and competing splits are NumofProducts, Geography, isActiveMember and Balance. Number of leaves in the optimal tree has decreased drastically in misclassification tree in comparison to the probability tree.

Misclassification Tree (3 Way Split)

Repeat the procedure on creating Decision tree by naming the node as Misclassification Tree (3)' and update the maximum branches to 3.

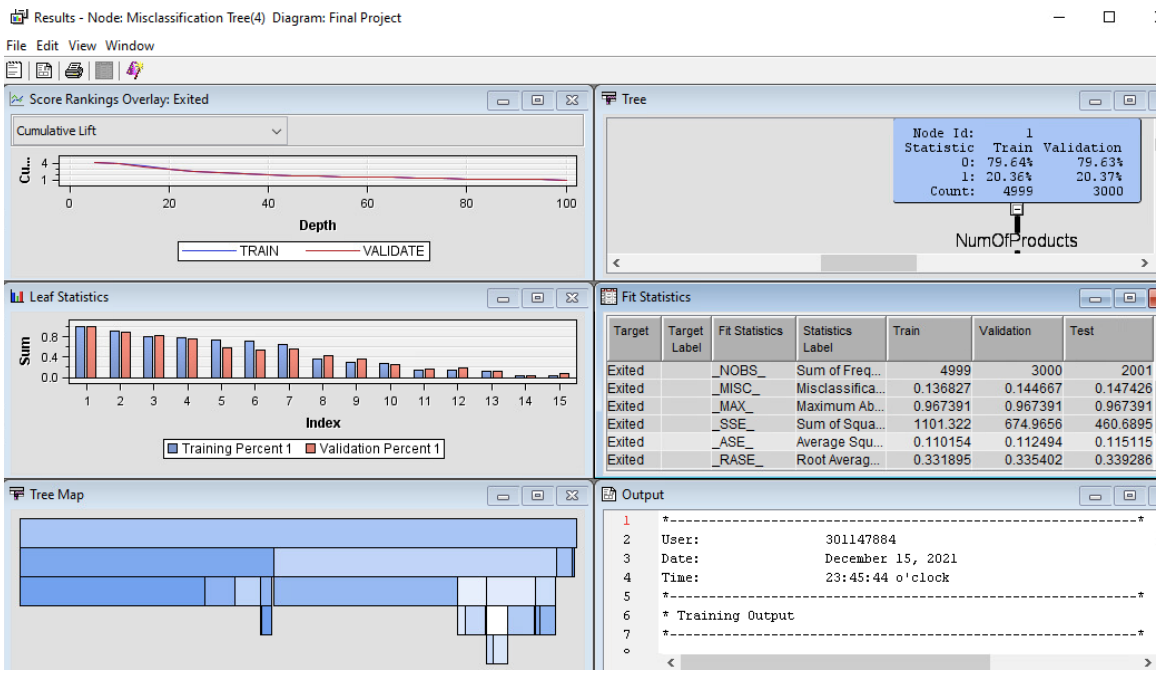


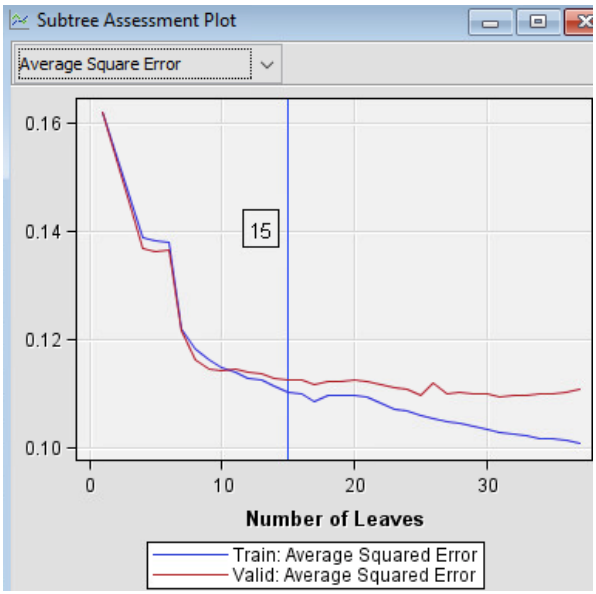


The Subtree Assessment Plot above shows that with the increase in number of branches age is no more the first split. Similar is the case with misclassification 3-ways split tree. Number of leaves have increased to 13 in the optimal tree.

Misclassification Tree (4 Way Split)

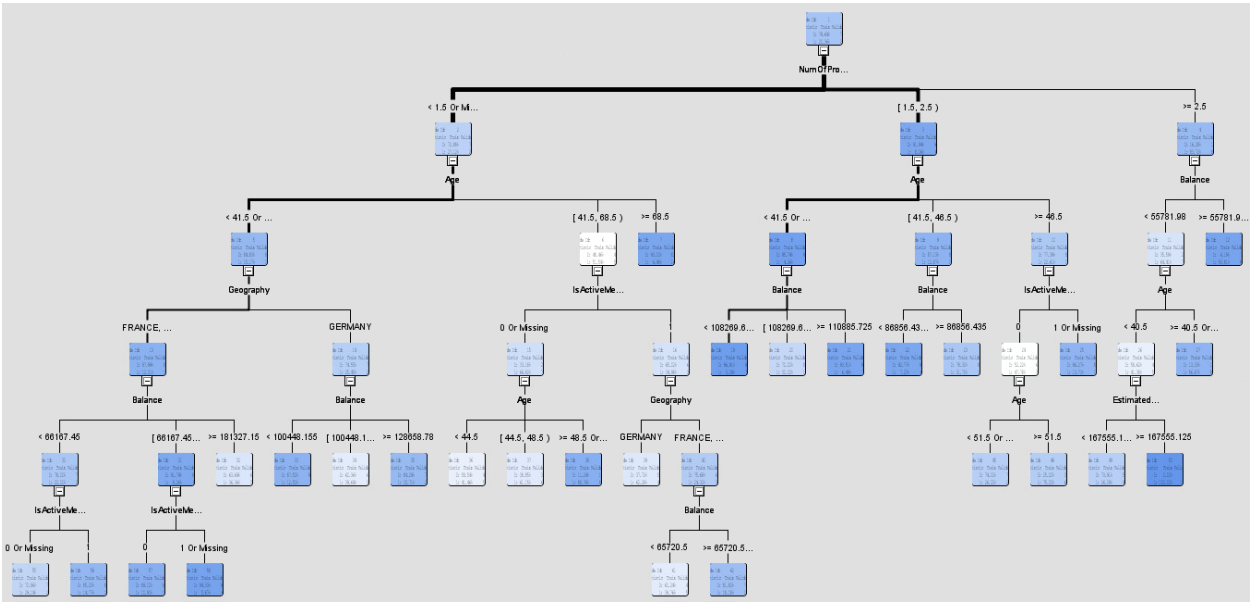
Repeat the procedure on creating Decision tree by naming the node as Misclassification Tree (4)' and update the maximum branches to 4.





Model Comparison & Interpretation of results

Based on Average square error the result shows that the Average Square Error tree 3-way split has the lowest test squared error of .109834 in comparison to all other trees, which makes it a better model. Similarly, if all the decision trees were compared based on misclassification rate, Probability tree 3-way split still performs better than the rest.

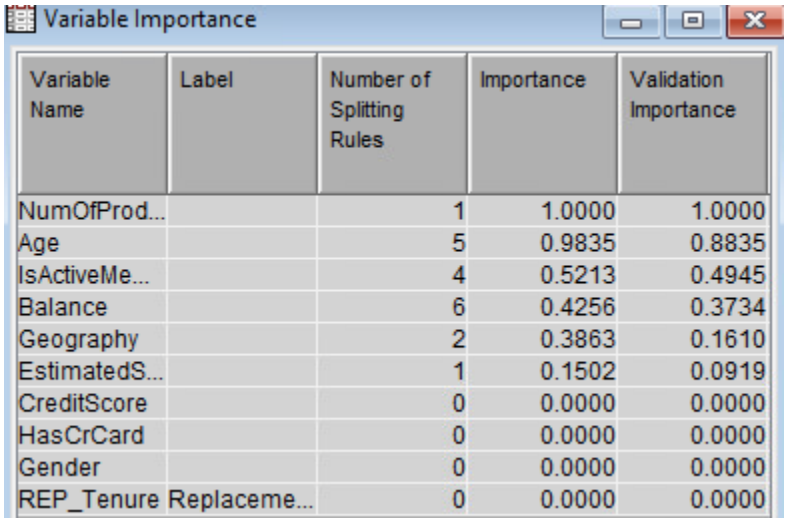


Chosen Decision Tree Model: ASE Tree(3)/Probability Tree 3

Moreover,

- Number of the products (NumOfProducts) is the most important variable in the decision of the customer to exit the bank. It is an interval variable. Customers who owned more than 3 products are more likely to leave the bank.

- Age of the customer is the second most important factor. It is also an interval variable. People in the age group of 41-69 years, irrespective of the number of products owned, are more likely to exit from the bank.
- Active status is the third most important factor. It is a binary variable. Older people (40 years or more) who have been inactive for a long time are more likely to leave the bank.
- Balance seems like an important factor for German customers who are less than 41 years old. Customers who have balance of $100,450 or more are more likely to leave.
-


Variable Importance

Variable Name	Label	Number of Splitting Rules	Importance	Validation Importance
NumOfProd...		1	1.0000	1.0000
Age		5	0.9835	0.8835
IsActiveMe...		4	0.5213	0.4945
Balance		6	0.4256	0.3734
Geography		2	0.3863	0.1610
EstimatedS...		1	0.1502	0.0919
CreditScore		0	0.0000	0.0000
HasCrCard		0	0.0000	0.0000
Gender		0	0.0000	0.0000
REP_Tenure	Replaceme...	0	0.0000	0.0000

4.2 Multivariate Regression Models

Since the target variable is a binary variable, which is to predict whether the customer left or continues with the bank, Logistic Regression model is used rather than Linear Regression model. A cost function named Maximum Likelihood Estimation function is used to measure the accuracy, indicating how close the values are from the actual. Consequently, five regression models were created:

Model 1: Full Regression

Procedure to Full Regression:

1. From the Model tab, select and drag the Regression node onto the diagram workspace.
2. Rename the Regression node to Full Regression.
3. Connect the Transform Variables node to the Full regression node.
4. Run the Full regression node.

Model 2: Forward Regression

Repeat the procedure on creating Full Regression by naming the node as Forward Regression and update the node property as below before running the node:

Model Selection	
Selection Model	Forward
Selection Criterion	Validation Error
Use Selection Defa	Yes
Selection Options	...

Model 3: Backward Regression

Repeat the procedure on creating Full Regression by naming the node as Backward Regression and update the node property as below before running the node:

Model Selection	
Selection Model	Backward
Selection Criterion	Validation Error
Use Selection Defa	Yes
Selection Options	...

Model 4: Stepwise Regression

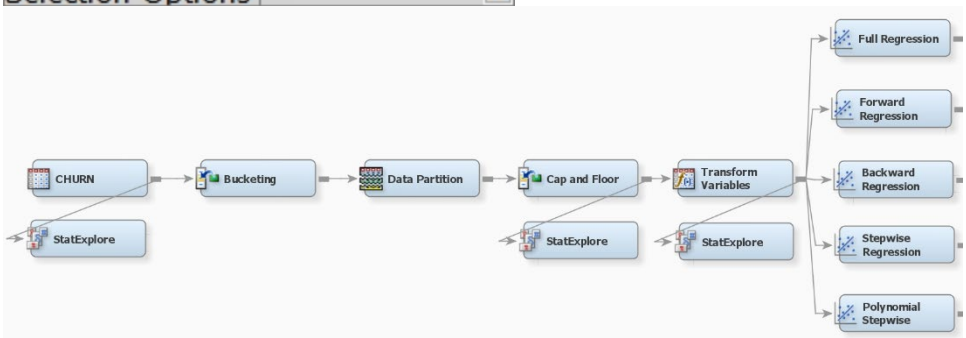
Repeat the procedure on creating Full Regression by naming the node as Stepwise Regression and update the node property as below before running the node:

Model Selection	
Selection Model	Stepwise
Selection Criterion	Validation Error
Use Selection Defa	Yes
Selection Options	...

Model 5: Polynomial Stepwise Regression

Repeat the procedure on creating Full Regression by naming the node as Polynomial Stepwise and update the node property as below before running the node:

Equation	
Main Effects	Yes
Two-Factor Intera	Yes
Polynomial Terms	Yes
Polynomial Degre	2
User Terms	No
Term Editor	...
Model Selection	
Selection Model	Stepwise
Selection Criterion	Validation Error
Use Selection Defa	Yes
Selection Options	...



Fit Statistics

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Test: Average Squared Error
Y	Reg5	Reg5	Polynomial Stepwise	Exited		0.109077
	Reg3	Reg3	Backward Regression	Exited		0.131684
	Reg	Reg	Full Regression	Exited		0.131685
	Reg2	Reg2	Forward Regression	Exited		0.131711
	Reg4	Reg4	Stepwise Regression	Exited		0.131711

Model Comparison and Interpretation of Results

The result shows that Polynomial Stepwise Regression is the best model in term of Average Squared Error in the Test data, which is 0.109077. Below are the important variables in this model:

1. Gender
2. Geography
3. IsActiveMember
4. LOG_REP_Age
5. NumOfProducts
6. REP_Balance
7. LOG_REP_Age*LOG_REP_Age
8. NumOfProducts*NumOfProducts
9. NumOfProducts*REP_Balance
10. REP_Balance* REP_Balance

To express the effect of an input variable on the likelihood a target variable to occur in terms of probability, the Odds Ratio Estimates are interpreted.

Odds Ratio Estimates

Effect		Point Estimate
Gender	Female vs Male	1.696
Geography	France vs Spain	0.883
Geography	Germany vs Spain	2.637
IsActiveMember	0 vs 1	2.992

- For **Gender (Female vs Male)**, the point estimate equals 1.696. This means that cases with a Female value for Gender are 69.6% more likely to exit than cases with a Male value for Gender.
- For **Geography (France vs Spain)**, the point estimate equals 0.883. This means that cases with a France value for Geography are 11.7% less likely to exit than cases with a Spain value for Geography.
- For **Geography (Germany vs Spain)**, the point estimate equals 2.637. This means that cases with a Germany value for Geography are 2.6 times more likely to exit than cases with a Spain value for Geography.
- For **IsActiveMember (0 vs 1)**, the point estimate equals 2.992. This means that cases with a 0 value for IsActiveMember are approximately 3 times more likely to exit than cases with a 1 value for IsActiceMember.

4.3 Neural Network Models

Preparatory Notes
Unless otherwise stated, all other properties were left at default values. For instance, the training technique used in Optimization was left as Default since SAS Enterprise Miner automatically chooses the appropriate technique depending on the number of weights.

The Model Selection Criteria was set to Average Square Error as discussed in class and consultations. After the best Neural Network was obtained, it was also trained with Misclassification set as the Model Selection Criteria to compare results.

It must be noted that finding the optimal Neural Network model is a task of trial and error. It takes a lot of experimentation and tinkering to get the details right. And often, the result is marginal improvements in performance. While there are algorithms that exist to find the best Neural Network (NEAT is an evolutionary algorithm designed to evolve Artificial Neural Networks), a simple iterative trial and error search is employed here.

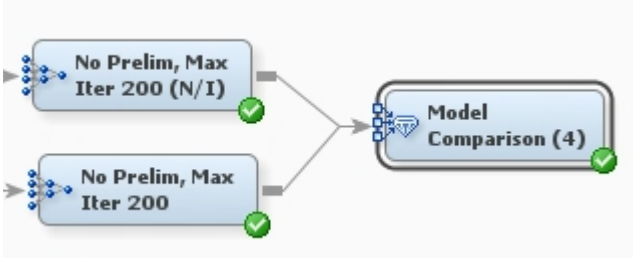
Many different parameters are tried out and the ones that work best are selected. This way, the model is built from the ground up – choosing the options which deliver the most promising results.

Finally, in every step of the way, the decisions made were trade-offs between improved performance and time/computing resources taken to run the Neural Networks.

Input Selection
In SAS Enterprise Miner, the Neural Network node doesn't have a built-in way to select useful inputs. This problem can be remedied by connecting the Neural Network node to other Nodes like Regression. In this case, the variables chosen by the Regression Node are passed forward to the Neural Network for training. Essentially, an ensemble model is built where different learning algorithms are used together/combined to produce predictions.

The Polynomial Regression node is used to feed inputs into the Neural Networks in the analysis, since the Polynomial Regression node has the lowest ASE out of all the other Regression Nodes. The variables chosen were as follows:
1) Gender
2) Geography
3) IsActiveMember
4) LOG_REP_Age
5) NumOfProducts
6) REP_Balance
7) LOG_REP_Age* LOG_REP_Age
8) NumOfProducts*NumOfProducts
9) NumOfProducts*REP_Balance
10) REP_Balance*REP_Balance

No Input Selection
The Neural Network node connected to the Polynomial Regression node is compared with another Neural Network node (N/I) which was identical in all respects except it was connected to the Transform node directly (No Input Selection).



(N/I) means No Input Selection

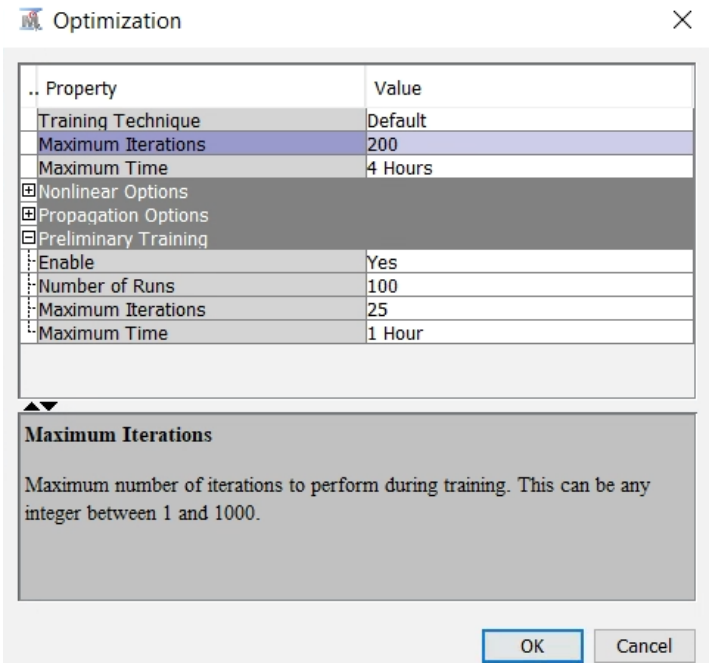
Selected Model	P	Model Node	Model Description	Target Variable	T	Selection Criterion: Test: Average Squared Error	Test: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights
Y	...	Neural13	No Prelim, Max Iter 200	Exited		0.102579	0.142429	4999	4971	28	28
	...	Neural14	No Prelim, Max Iter 200 (N/I)	Exited		0.105991	0.149925	4999	4956	43	43

As expected, the Average Square Error of the model with input selection was lower. Therefore, the variables selected by the Polynomial Regression Node will be used to train the Neural Networks moving forward. Choosing to connect the Neural Networks to the Polynomial Regression node is a practical decision that improves training times and produces better networks.

In a later section, the performance of the User Defined Neural Network Full Logistic Model with and without Input Selection will be compared.

Optimization

Maximum iterations
The Maximum Iterations value is set to 200. This value is a good balance between training the model until convergence criteria is satisfied, and time spent training the model.



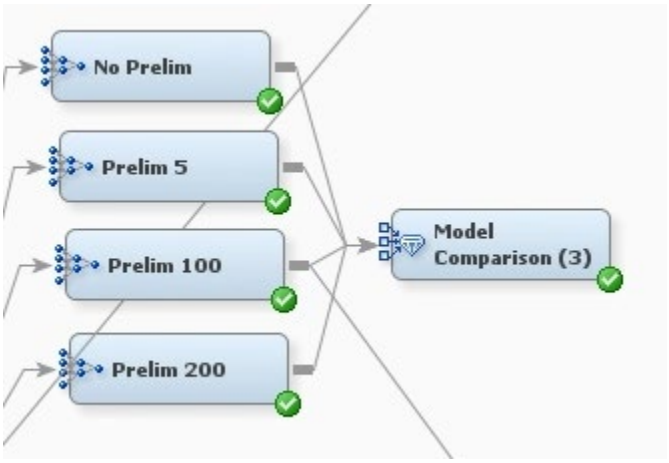
Keep in mind that the training procedure will not run for 200 iterations always. If convergence is satisfied before the Maximum Iteration limit is reached, then the training procedure will halt. This is the case for all the Neural Network models. The convergence criteria are satisfied after about 150 iterations for most models.

Why use a convergence criterion?
Running a Neural Network for many iterations doesn't guarantee good results. In fact, it might generate problems of its own like the Vanishing Gradient problem where the gradient is so small as to *vanish*, rendering further training/iteration meaningless. Furthermore, resources are finite, and a Neural Network simply cannot be ran forever/for large number of iterations.

Preliminary Training
Local minima are a problem which plagues every optimization problem. Neural Networks suffer from the same problem. SAS EM has a Preliminary Training option which trains the network for a few iterations from random initializations of the weight parameters. After this initial training, the best weights/estimates obtained here are used for further training of the Neural Network.

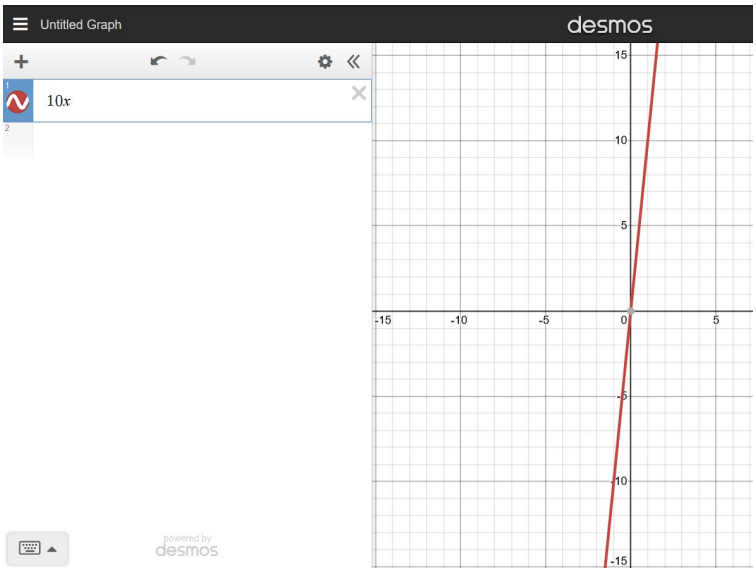
Number of Runs

How do the ideal number of Runs for Preliminary training, and the ideal number of iterations to train the Neural Network for during each run being determined? The basic procedure to find out is Trial and Error. That is trying out different settings, comparing them to each other, and choosing the best one.

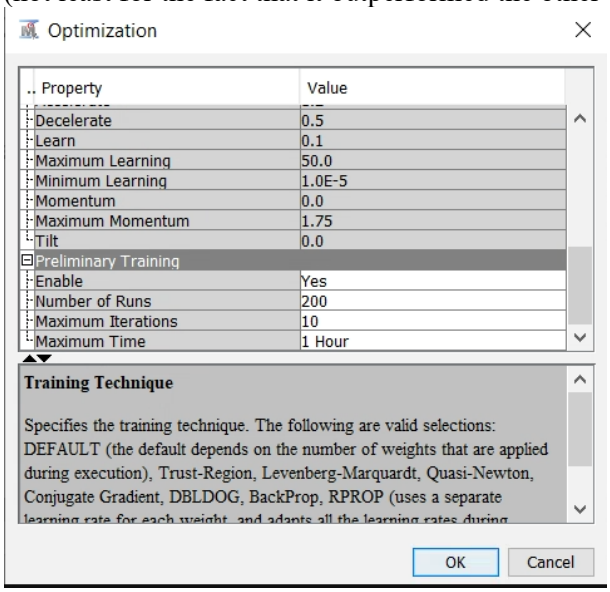


Selected Model	F	Model Node	Model Description	Target Variable	T	Selection Criterion: Test: Average Squared Error	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights
Y	...	Neural11	Prelim 100	Exited		0.104552	4999	4968	31	31
	...	Neural12	Prelim 200	Exited		0.104559	4999	4968	31	31
	...	Neural10	Prelim 5	Exited		0.104732	4999	4968	31	31
	...	Neural	No Prelim	Exited		0.106671	4999	4968	31	31

A NN is ran with no Preliminary training, and then subsequently NNs with Preliminary Training set to 5 runs, 100 runs and 200 runs respectively (all of them had 10 iterations for each run). Observations: As the number of runs for prelim training being increased – the time required for the NN training to complete increases drastically. With 5 runs and 10 iterations, the team must wait for $5 \times 10 = 50$ computing steps (simple estimation) to complete. With 200 runs and 10 iterations, the team must wait for $200 \times 10 = 2000$ computing steps. Essentially, the team is dealing with an $n \times 10 = 10n$ function, where n is the number of runs (keeping iterations fixed).

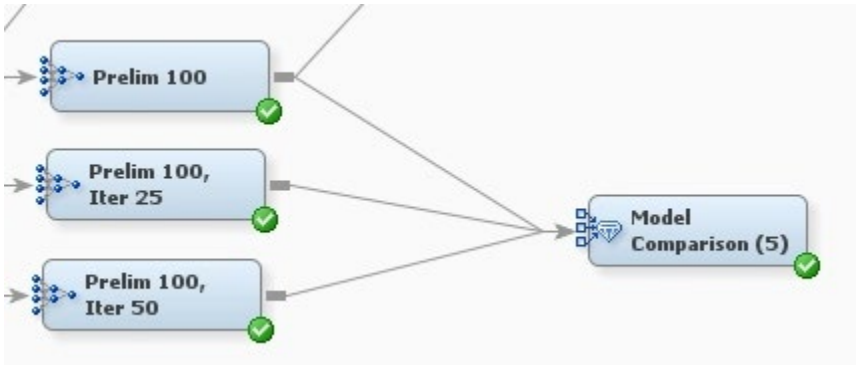


Therefore, in the interest of optimizing computing time and resources compared to predictive power received from the model – we've decided to settle on Preliminary Training runs of 100. Running the NN node with Runs set to 200 took a long period of time. Comparing the time and benefits received from increasing the number of Preliminary Training runs, a choice of 100 seems logical (not least for the fact that it outperformed the other options).



Number of Iterations

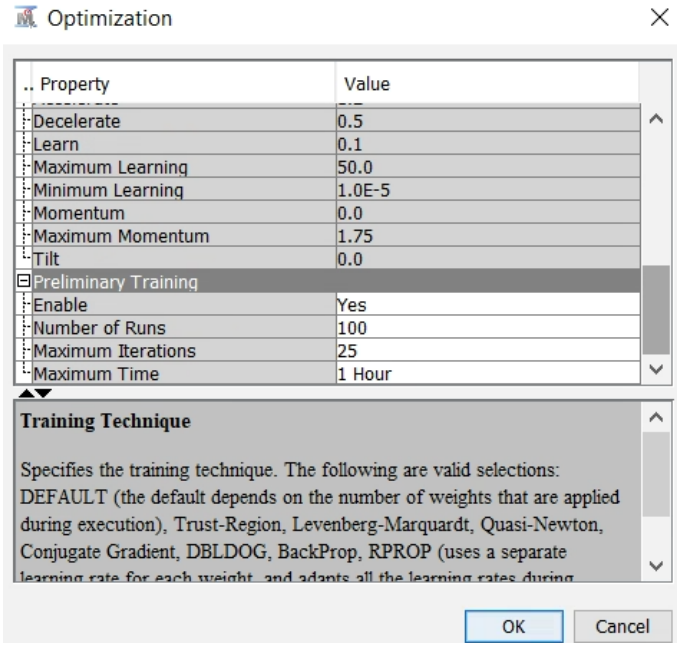
As demonstrated earlier, increasing the Number of Runs increases the time spent training the model drastically. Naturally, this is true if the Number of Iterations increased too. NN nodes are tested with Number of runs set to 100 and Number of Iterations set to 10, 25, and 50.



Selected Model	P	Model Node	Model Description	Target Variable	T	Selection Criterion: Test: Average Squared Error ▲	Test: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights
Y	...	Neural11	Prelim 100	Exited		0.104552	0.14043	4999	4968	31	31
	...	Neural16	Prelim 100, Iter 50	Exited		0.104607	0.142929	4999	4968	31	31
	...	Neural15	Prelim 100, Iter 25	Exited		0.104626	0.142929	4999	4968	31	31

Results of Comparison

There is little difference between the models with Number of Iterations set to 50 and 25. Therefore, 25 is chosen for the Number of Iterations since it took a long time to run the model with 50 number of iterations.

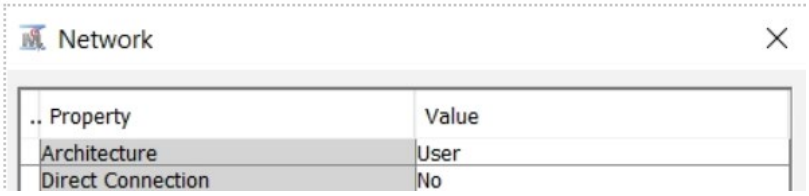


After settling on the Optimization parameters, different Network Architectures are explored to determine the best Neural Network to use.

Network Architecture

In a Neural Network, the value produced by the Combination Function is passed on to the Activation Function. Basically, the combination function determines how the weight-input combinations produced from predecessor layers will be combined before being passed on to the Activation function. Since the Logistic and Hyperbolic Tangent activation functions are being used, the team decided to use the Linear option for Combination.

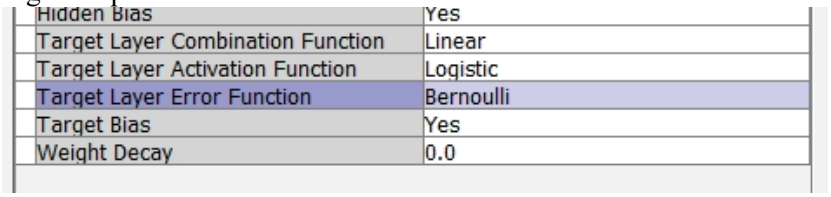
The model architecture is changed to User Defined – this allowed modification of additional parameters, and experimentation with various settings.



Target Layer Error Function

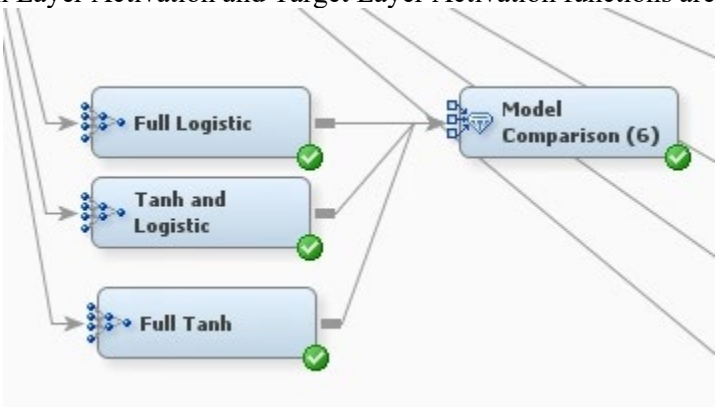
It is important to note that the Target Layer Error Function is to be set to Bernoulli for all the User Defined Neural Networks. The Bernoulli Error function is used when the Target variable takes on the value 0 or 1 (Exited is the target variable, and it takes on the values 0 or 1).

Leaving this parameter as Default will return in incorrect training of the Neural Network, and very high Average Squared Error and Misclassification Rate. Therefore, it is important not to forget to change this parameter.



Hidden and Target Layer Activation Functions

Different Hidden Layer Activation and Target Layer Activation functions are experimented.



Hidden Layer Activation function and Target Layer Activation functions compared

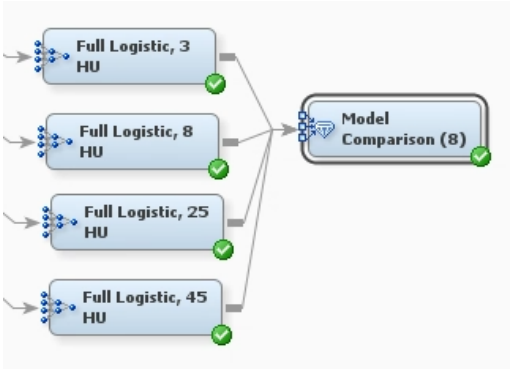
Selected Model	F	Model Node	Model Description	Target Variable	T	Selection Criterion: Test: Average Squared Error ▲	Test: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights
Y	...	Neural17	Full Logistic	Exited		0.102097	0.144928	4999	4967	32	32
	...	Neural4	Tanh and Logistic	Exited		0.102243	0.143928	4999	4967	32	32
	...	Neural2	Full Tanh	Exited		0.124661	0.168916	4999	4967	32	32

The model with the Logistic function for both its Hidden Layer Activation function and Target Layer Activation function had the lowest Average Squared Error out of all the other models, and therefore this Network Architecture is settled. The selected model also had the second lowest Misclassification rate. There is very little difference between the selected model and the Tanh and Logistic model in which the Tanh function was used for the Hidden Layer and the Logistic function was used for the Target Layer.

Why was the Tanh and Logistic Model made?
Mixing Activation functions in Neural Networks is a standard practice – doing so helps networks learn non-linear relationships in the data better.

Tanh (Hyperbolic Tangent) is a good activation function as it has a stronger gradient (derivative) and gives negative and positive values ([-1, 1]). The Logistic Function is a good activation function for the target layer since the Sigmoid maps values to the [0, 1] space, and its mathematical derivation is also relatively simple.[7]

Hidden Units
Hidden Units set to 3, 8, 25 and 45 are experimented for the Full Logistic Neural Network. 45 Hidden Units are experimented because the team wanted to see whether large networks would improve performance compared to small and lean networks.



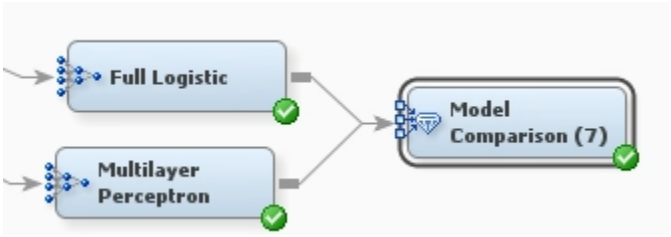
[7] Evolving Parsimonious Networks by Mixing Activation Functions, https://arxiv.org/pdf/1703.07122.pdf
Improving prediction of secondary structure, local backbone angles and solvent accessible surface area of proteins by iterative deep learning, https://www.nature.com/articles/srep11476

Selected Model	P	Model Node	Model Description	Target Variable	Ta	Selection Criterion: Test: Average Squared Error	Test: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom
Y	...	Neural7	Full Logistic, 3 HU	Exited		0.102097	0.144928	4999	4967	32
	...	Neural8	Full Logistic, 8 HU	Exited		0.102178	0.134933	4999	4917	82
	...	Neural20	Full Logistic, 45 HU	Exited		0.103454	0.149425	4999	4547	452
	...	Neural9	Full Logistic, 25 HU	Exited		0.10367	0.142429	4999	4747	252

Results of comparison

The model with just 3 Hidden Units performed the best, both in terms of Average Squared Error and Misclassification Rate. The explanation for this is that the Neural Networks with more Hidden Units were overfitting on the Training dataset. The additional Hidden Units encoded for the noise in the dataset rather than the signal – leading to poor generalization and performance on the Test dataset. Therefore, Hidden Units are opted to set to 3 for the NN moving forward.

Comparing the User Defined Network to the Default Multi-layer Perceptron
As a final test of the User Defined Network, it is compared to the default Multi-Layer Perceptron in SAS EM.

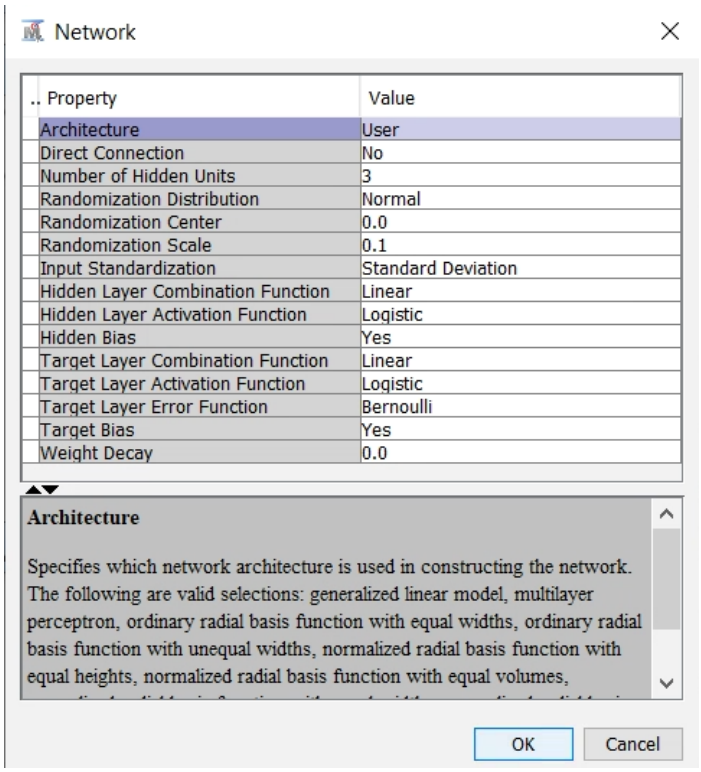


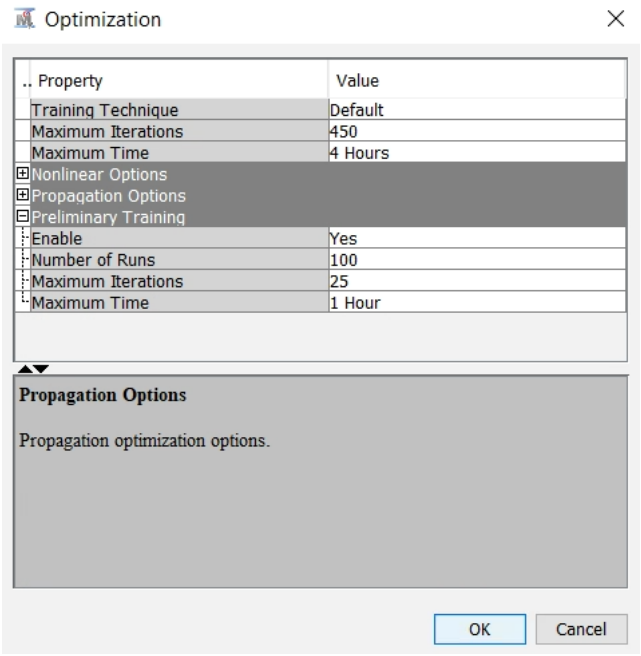
Selected Model	F	Model Node	Model Description	Target Variable	T	Selection Criterion: Test: Average Squared Error	Test: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights
Y	...	Neural6	Full Logistic	Exited		0.102097	0.144928	4999	4967	32	32
	...	Neural5	Multilayer Perceptron	Exited		0.102292	0.143928	4999	4971	28	28

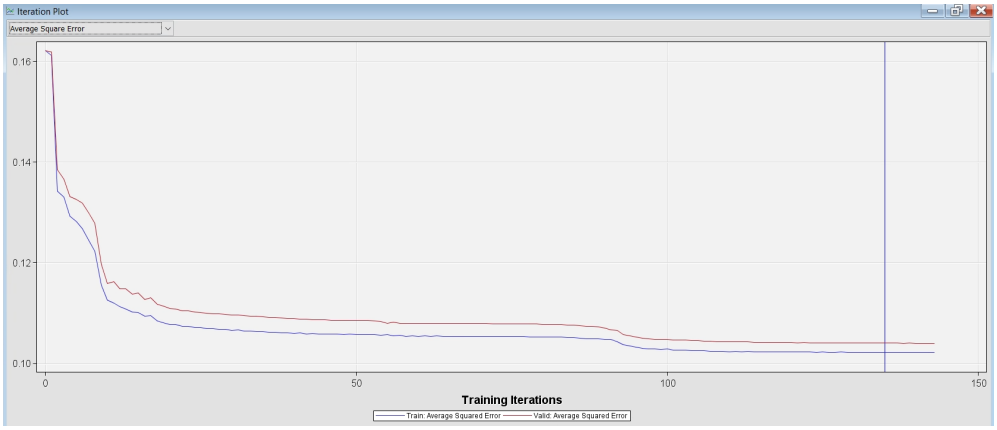
The User Defined Network performed better than the default Multilayer Perceptron model in terms of the Average Squared Error. It had slightly worse performance compared to the default model in terms of Misclassification Rate. It must be noted that the team is dealing with a Customer Churn problem, where every improvement no matter how marginal, is of vital importance. Therefore, experimenting to find the Full Logistic Model was worth the effort.

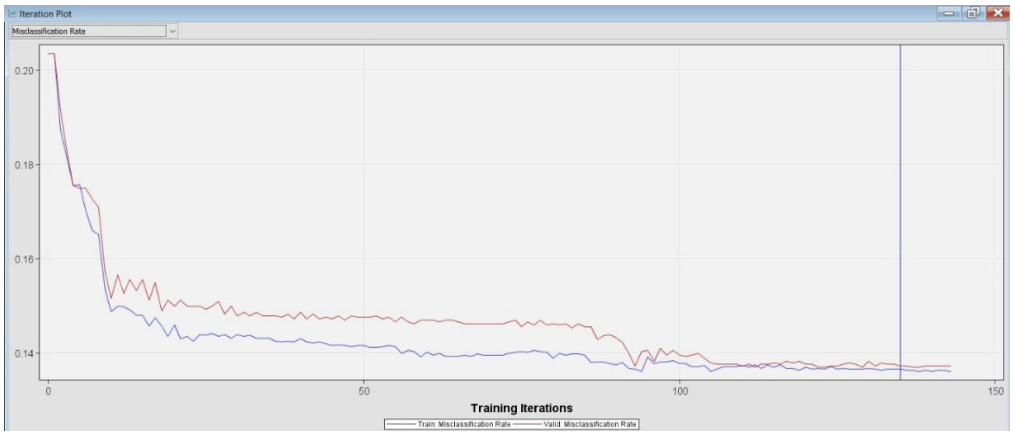
The Final User Defined Neural Network architecture
After running various experiments to find the optimal parameters for the Neural Network Architecture and Optimization targets, a Network with the Logistic function for both the Hidden Layer Activation Function and Target Layer Activation Function is settled on.
The Logistic Function is a good activation function for the target layer since the Sigmoid maps values to the [0, 1] space, and its mathematical derivation is also relatively simple. The

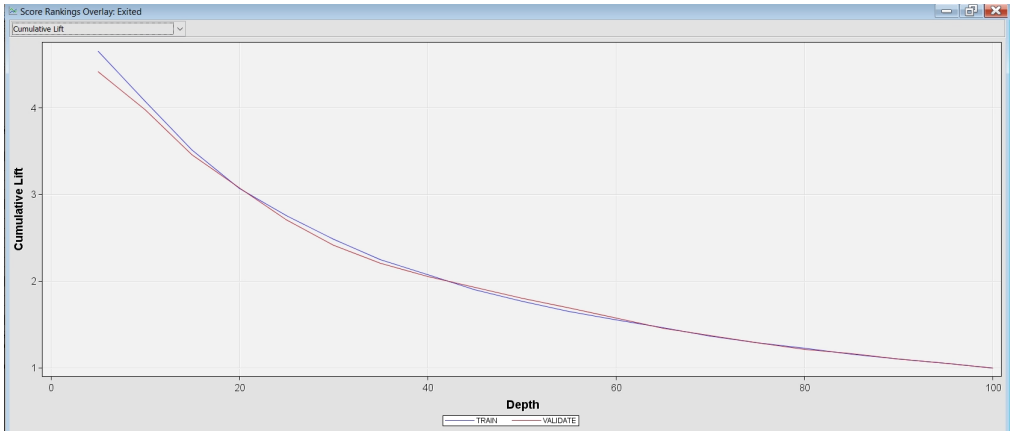
simplicity of the Logistic Function also means that training takes place quicker. However, it must be noted that the Logistic Function suffers from the Vanishing Gradient problem. The Target Layer Error Function was set to Bernoulli since we're dealing with a Target variable which takes on the values [0, 1]. The Randomization Distribution parameter is not changed since the team is transforming their inputs to make them Normal. Additionally, a Weight Decay is not used since there is no large number of inputs being passed to the Neural Network.

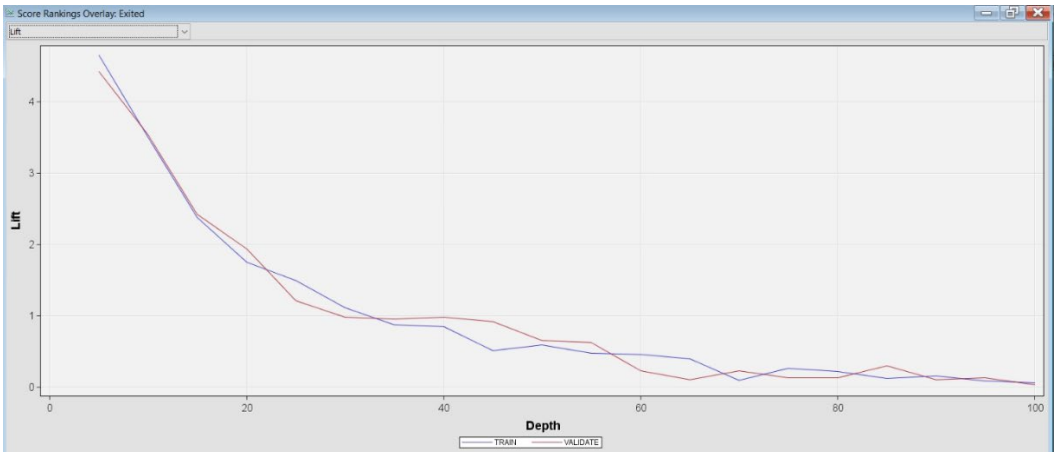



Optimization
.. Property
Value
Training Technique
Default
Maximum Iterations
450
Maximum Time
4 Hours
Nonlinear Options
Propagation Options
Preliminary Training
Enable
Yes
Number of Runs
100
Maximum Iterations
25
Maximum Time
1 Hour
Propagation Options
Propagation optimization options.
OK
Cancel

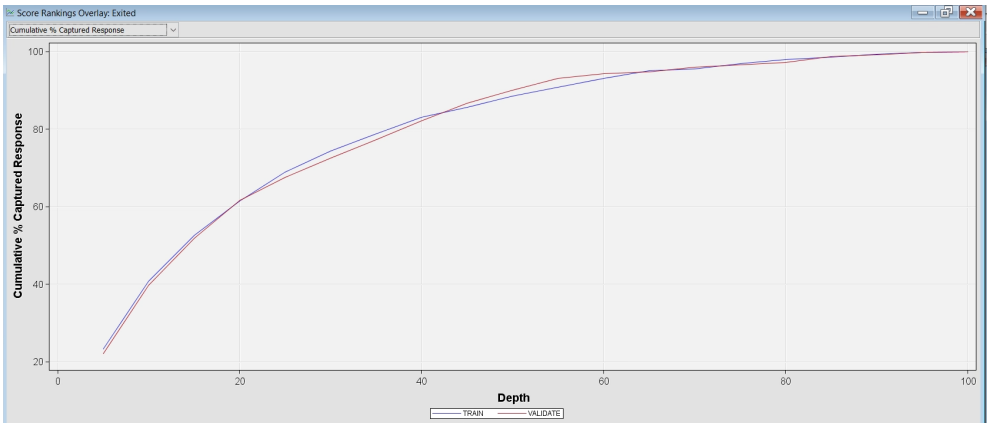

Iteration Plot
Average Square Error
0.16
0.14
0.12
0.10
0
50
100
150
Training Iterations
Train: Average Squared Error
Valid: Average Squared Error

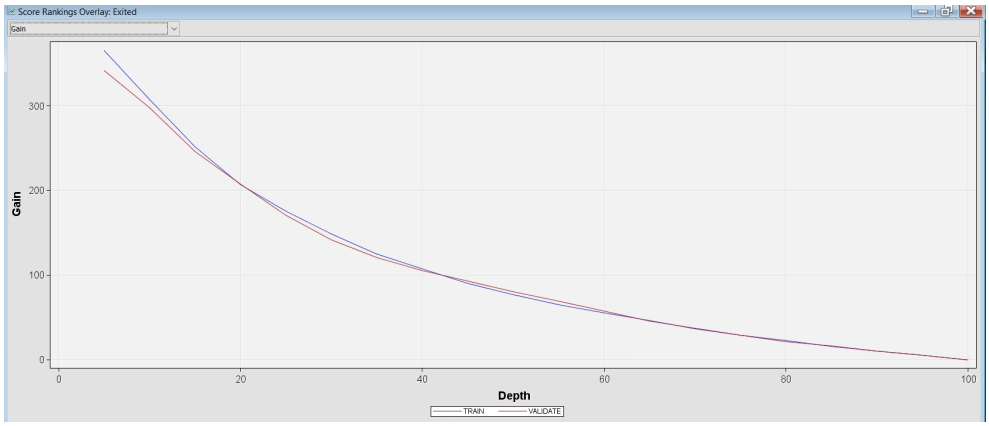

Iteration Plot
Misclassification Rate
0.20
0.18
0.16
0.14
0
50
100
150
Training Iterations
Train: Misclassification Rate
Valid: Misclassification Rate


Score Rankings Overlay: Exited
Cumulative Lift
Cumulative Lift
4
3
2
1
0
20
40
60
80
100
Depth
TRAIN
VALIDATE

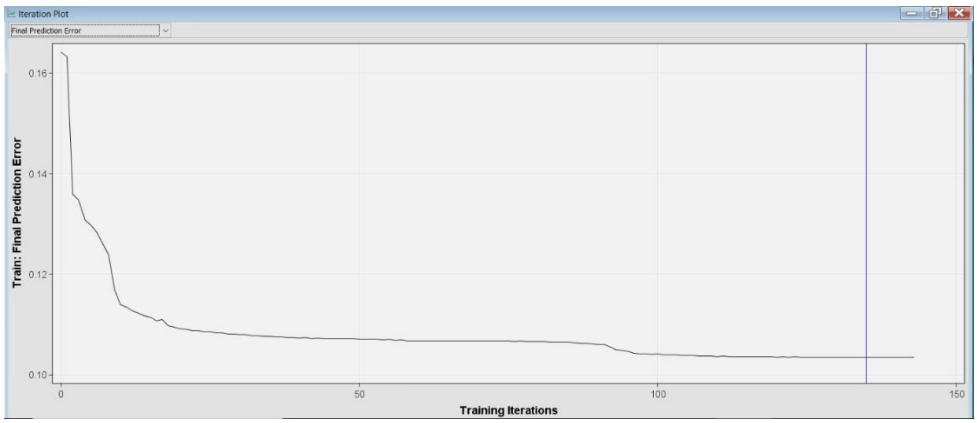

Score Rankings Overlay: Exited
Lift
Lift
4
3
2
1
0
20
40
60
80
100
Depth
TRAIN
VALIDATE

Lift Chart





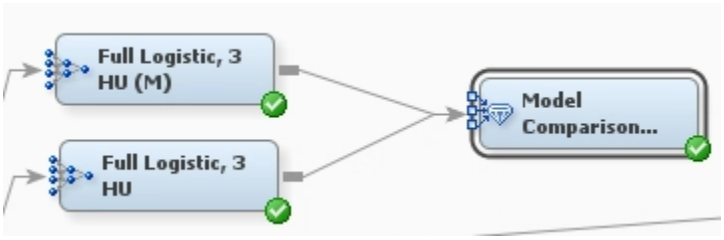
Gain Chart



Final Prediction Error Chart

Misclassification as Model Selection Criteria

Since the team are dealing with a Classification problem – a version of the best Neural Network is trained with Misclassification set as the Model Selection Criteria. Then, it is compared to the model trained with Average Square Error set as Model Selection Criteria.



Selected Model	P	Model Node	Model Description	Target Variable	T	Selection Criterion: Test: Misclassification Rate	Test: Average Squared Error	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights	Train: Akaike's Information Criterion	Train: Schwarz's Bayesian Criterion	Train: Average Squared Error
Y	...	Neural19	Full Logistic, 3 HU	Exited		0.144928	0.102097	4999	4967	32	32	6826.156	7034.7	0.102618
	...	Neural3	Full Logistic, 3 HU (M)	Exited		0.144928	0.102097	4999	4967	32	32	6826.156	7034.7	0.102618

Results of Comparison

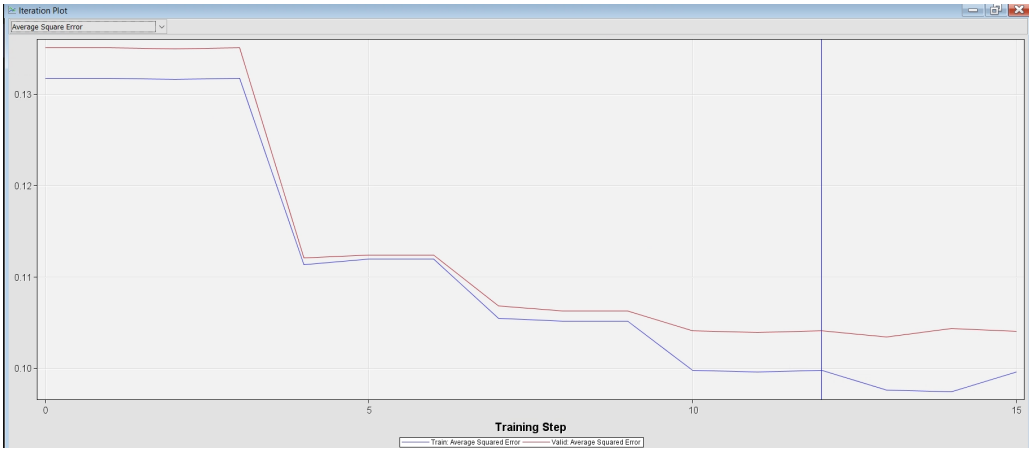
Both the models had the same performance in all aspects. Therefore, in the case – it made no difference to train a model with Misclassification or Average Squared Error as Selection Criteria.

AutoNeural

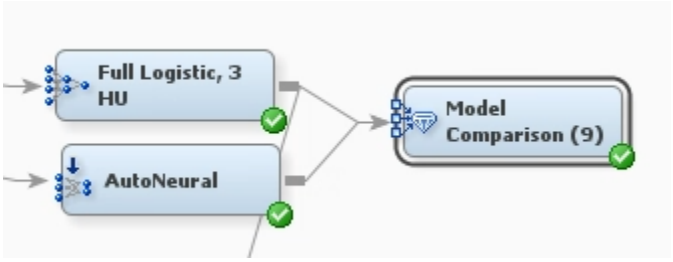
The AutoNeural Node automatically searches through different architectures and settings to find the best Neural Network. This tool allows automatically perform all the steps done earlier and choose the best results. However, there are certain differences from manually created networks, and networks created by the AutoNeural node. For instance, even if a manually created network shares the same number of hidden units and layers as an AutoNeural node – it is not necessary that the weights chosen are the same.
Similar to how the team tried combinations of different activation functions, and hidden units in the Neural networks – the AutoNeural node iterates through a wide range of different parameters and chooses the one that performs the best.

The Number of Hidden Units is set to 1 – this way, the AutoNeural Node will add 1 hidden unit for each iteration pass. Additionally, for Activation Functions - the Logistic and Tanh Activation Functions are only set to Yes – the other functions are not relevant to the problem at hand. The Maximum Iterations is set to 10 for the AutoNeural Node and set Tolerance to Medium. Tolerance dictates the Preliminary training of the Neural Network trained by the AutoNeural node.

Model Options	
Architecture	Single Layer
Termination	Overfitting
Train Action	Search
Target Layer Error Fur	Default
Maximum Iterations	10
Number of Hidden Unit	1
Tolerance	Medium
Total Time	One Hour
Increment and Search	
Adjust Iterations	Yes
Freeze Connections	No
Total Number of Hidde	30
Final Training	Yes
Final Iterations	5
Activation Functions	
Direct	No
Exponential	No
Identity	No
Logistic	Yes
Normal	No
Reciprocal	No
Sine	No
Softmax	No
Square	No
Tanh	Yes
Score	
Hidden Units	No
Residuals	Yes
Standardization	No
Status	



The Neural Network generated by the AutoNeural node is compared to the User Defined Network.



Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Test: Average Squared Error	Test: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights
Y	AutoNeural	AutoNeural	AutoNeural	Exited		0.10043	0.13943	4999	4962	37	37
	Neural7	Neural7	Full Logis...	Exited		0.102097	0.144928	4999	4967	32	32

Results of comparison

The AutoNeural node soundly beat the Neural Network. It outperformed in terms of the Average Squared Error, as well as the Misclassification Rate.

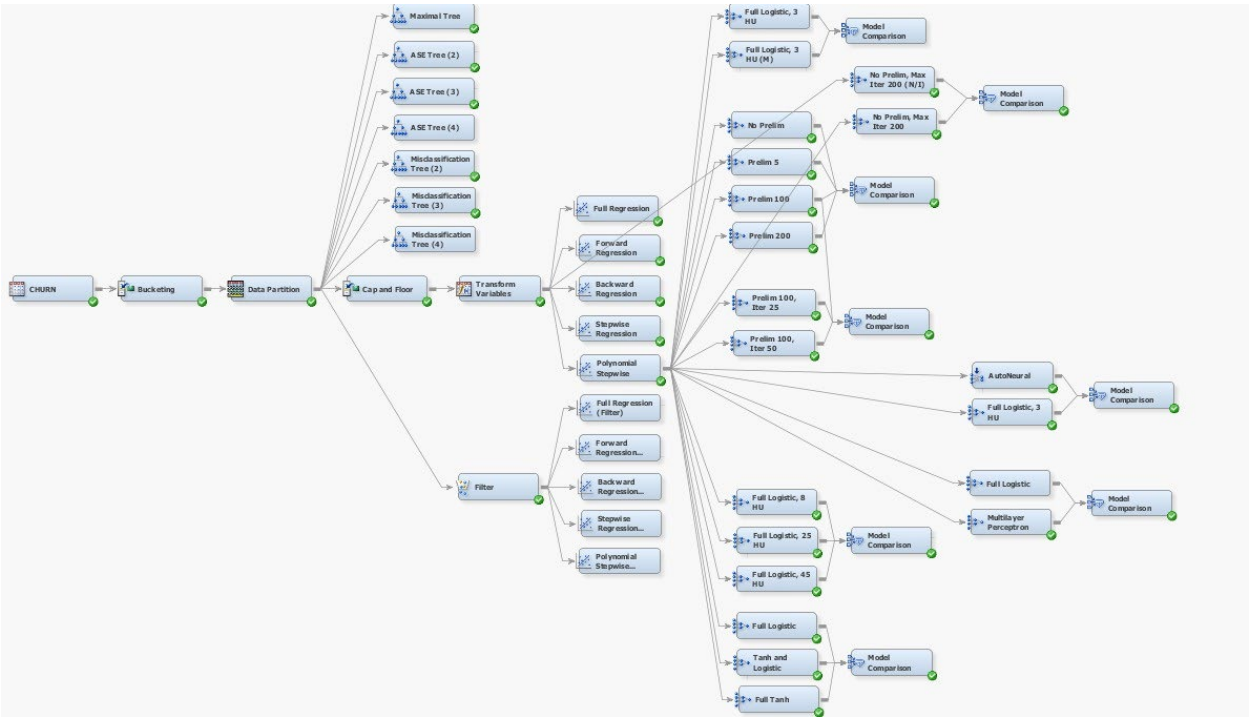
Final Remarks on Neural Network
It is important to note that a lot of time and effort are spent finding the optimal parameters for the User Defined Neural Network (Full Logistic). The team received superior results from running the AutoNeural Node, and similar results from a default Multilayer Perceptron. However, it was important to establish the parity of results by experimenting before making conclusions based on the default models provided by SAS EM. This is a Customer Churn problem where every customer that failed to be predicted churn for costs a lot of resources to the bank. Every improvement greatly saves costs and resources of the bank. Therefore, effort was rewarded by a model which performed better in the end compared to all other models.

The AutoNeural Node and the User Defined Neural Network (Full Logistic) are the two resulting models of all the trial and experimentation here.

5.0 DISCUSSION

As explained in the Conceptual Framework section, the team followed the SEMMA, the logical organization of SAS Enterprise Miner, to complete the data mining tasks required to develop and identify the predictive model that could best address the problem statement. Multiple experiments and iterations were conducted to determine the suitable models that would best represent each track.

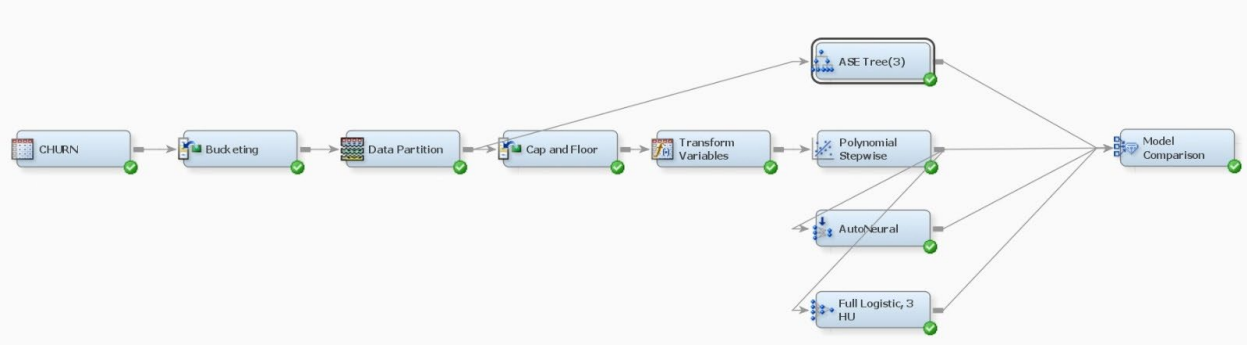
The following diagram indicates the multiple processes that the research team underwent to produce the best-in-class models for the selection.



5.1 Final Models Comparison

Choosing the best performing models from the decision tree, regression, and neural network tracks from the Model Development and Testing phase, the study produced four (4) champion models from which the final model is selected. The resulting best-in-class models are:

a) ASE Tree (3-way split)
b) Polynomial Stepwise Regression
c) User Defined Neural Network (Full Logistic)
d) AutoNeural Node



Using the SAS EM Model Comparison node with the selection criteria set to 'Average Squared Error', the following Fit Statistics output was generated:

Fit Statistics

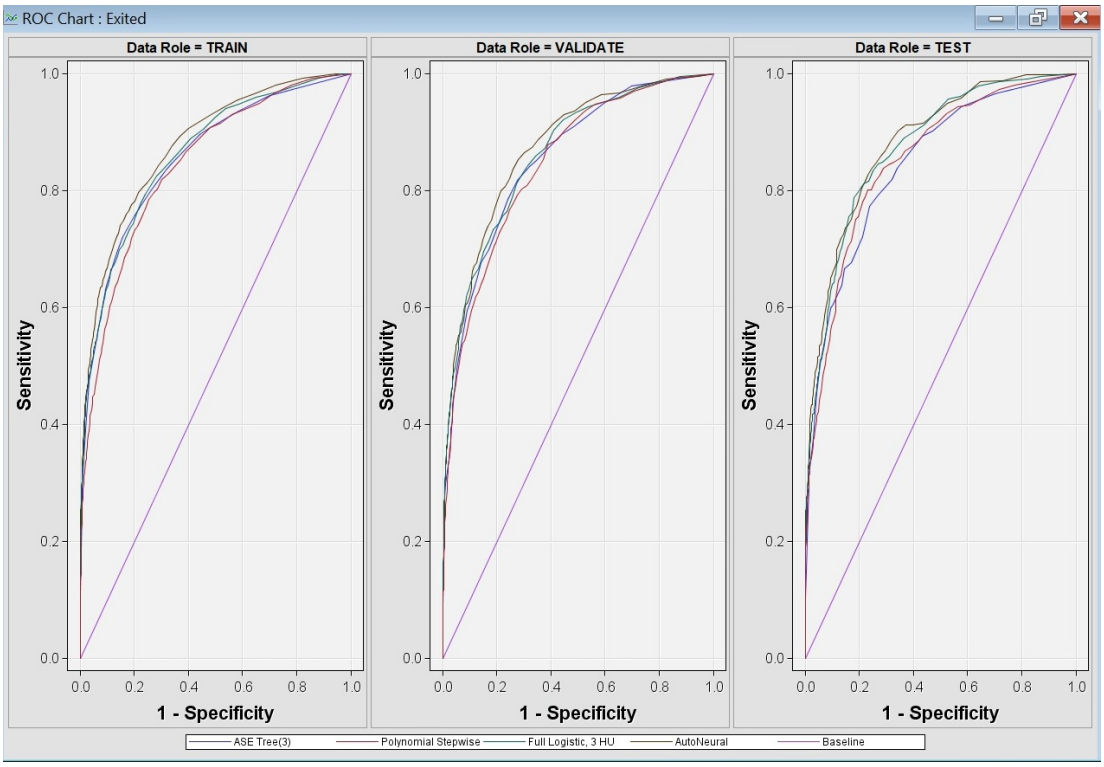
Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Test: Average Squared Error ▲	Test: Misclassification Rate	Test: Roc Index
Y	AutoNeural	AutoNeural	AutoNeural	Exited	0.10043	0.13943	0.88
	Neural	Neural	Full Logistic, 3 HU	Exited	0.102097	0.144928	0.873
	Reg5	Reg5	Polynomial Stepwise	Exited	0.109077	0.154923	0.852
	Tree	Tree	ASE Tree(3)	Exited	0.109834	0.145427	0.845

Although ASE would be the sufficient criteria for selection for the purposes of this report, the Misclassification Rate and the ROC Index are also displayed and reviewed for comparison.

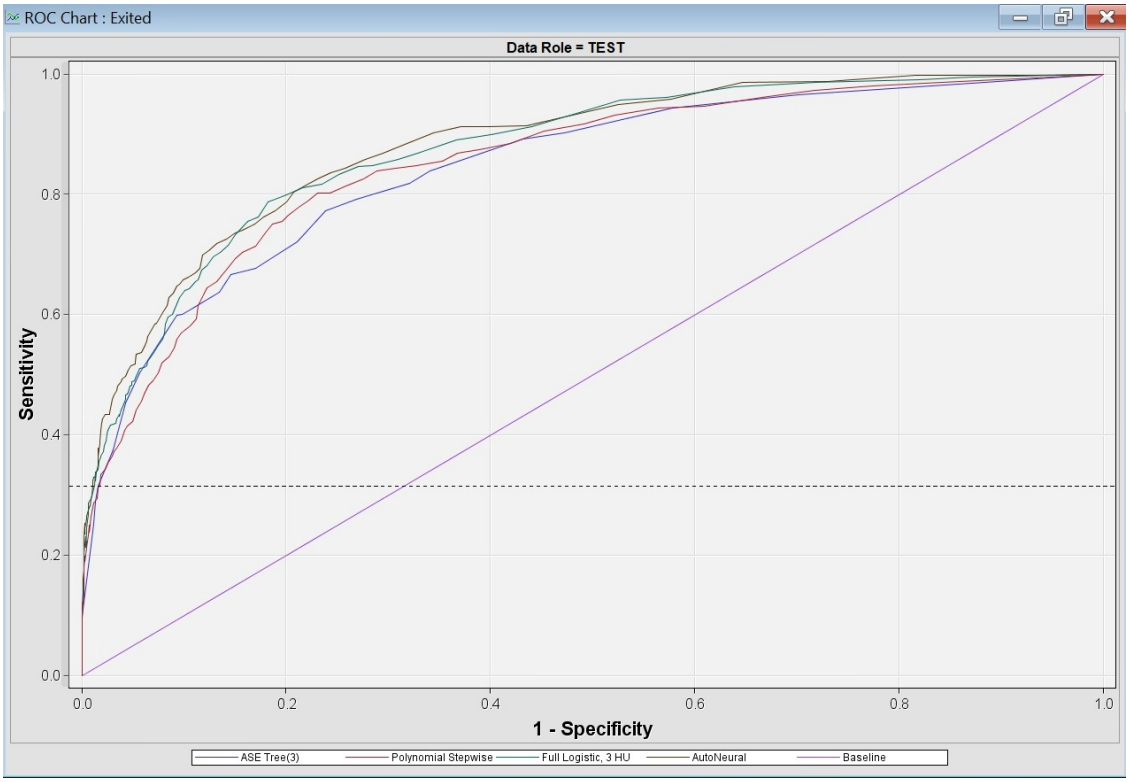
5.2 Reading the Results

Given that earlier discussions on the best-in-class models are already provided, this section will only cover the overall selection and result and will no longer get into the specifics features of each individual model.

The results show that the AutoNeural (AN) model provides the best prediction of customer attrition as it outperforms all the other models based on ASE, but also on Misclassification Rate and ROC Index. The ASE of 0.10043, the AN model is the one that provides predictions with the least difference between actual and estimated outcomes. With the Misclassification Rate at 0.13943, the AN model provides a prediction with the lowest fraction of predictions that are incorrect. With an ROC of 0.88, the AN model is able to distinguish between the Positive and the Negative class points correctly.



ROC Curves for all the models in Model Comparison



From inspection of the ROC Curve chart, the team concluded that the AutoNeural model outperforms all other models. Furthermore, it is also performing much better than baseline (diagonal line through the chart). Therefore, the AutoNeural node is the best model with an ROC of 0.88.

Purely based on ASE, the User Defined Neural Network (Full Logistic) provides the next best predictions, with the Polynomial Stepwise Regression and ASE Tree (3-way split) coming behind with weaker capacities to minimize the difference between actual and estimated outcomes. In terms of Misclassification Rates, however, ASE Tree (3-way split) does better than the Polynomial Stepwise Regression which provides the worst misclassification of the set. The ROC Index performance of the models mimic the pattern of the ASEs thereby providing support for confirming the final model.

5.3 Business Implications

Based on the result, the study proposes that the use of an AutoNeural network can provide the best predictor for customer attrition for a credit card population with similar characteristics as the data set. By applying the model, the business may be able to identify earlier on which customers on their portfolio are likely to defect and to preventively take action through campaigns and portfolio actions.

That said, there will likely be some concern about the 'black box' nature of a Neural Network based model since the 'logic' behind the prediction becomes difficult to explain beneath the first layer. The business would have an idea which customer would defect but not why. Nor will it know which variables or levers to use to reduce the potential for defection particularly for highly profitable customers.

For this reason, it may be important to support neural network results with strategic actions guided by the traditional models – namely decision tree and multivariate regression – if only to identify and prioritize which levers are most appropriate to address. The traditional models can provide marketers and campaign managers with some levers that could be used individually or in conjunction with each other to develop promotional strategies that reduce the likelihood of defection.

6.0 CONCLUSION

As mentioned in the beginning of the study, it costs more to acquire new customers than to retain existing ones. This makes the development of an end-to-end automated solution to predict customer attrition practical, useful and beneficial. This study worked on a research question that would lead to developing a predict analytics model (or combination of models) for credit card customer defection.

In the process, the study produced four (4) champion models from which the final model is selected -- ASE Tree (3-way split), Polynomial Stepwise Regression, User Defined Neural Network (Full Logistic) and AutoNeural Node. Of these, the results demonstrated that the AutoNeural (AN) model provides the best prediction of customer attrition as it outperforms all the other models based on ASE, but also on Misclassification Rate and ROC Index.

However, it also recognized that there may likely be some concern about the interpretability of a Neural Network based model. Thus, the study also recommends the use of the traditional models – namely decision tree and multivariate regression – in conjunction with the Neural Network model, if

only to identify and prioritize which levers are most appropriate to address through marketing campaigns and portfolio actions.