

# WQD7005 Data Mining

## ALTERNATIVE ASSESSMENT 1

Name	SID
Danial Mirza bin Madrawi	22063607

## 1. OBJECTIVE

The competitions in financial industries are getting harder in the next decade. One of this industry main source of revenue are interest income which they could get by giving loan or credit payment facilities to customer. Therefore, the more the credit are given, the more interest they get. Since the data are collected by every credit activity, the company hope they could get some insight by processing the data.

In this report the aim is to predict the balance of customer's credit card. The balance is a suitable target for prediction because it is a continuous variable that represents the total amount of money that the customer owes to the credit card company. This attribute is a good target for prediction because it is a key indicator of the user's creditworthiness and financial stability. By predicting the balance, we can gain insights into the user's spending habits, payment behaviour, and credit utilization. This information can be used to develop personalized financial products and services that meet the user's needs and preferences. Additionally, the balance column is a good target for prediction because it is a primary factor in determining the user's credit score, which is a key metric used by lenders to evaluate the user's creditworthiness and ability to repay loans.

## 2. DATASET

The dataset used in this report is Customer Dataset sourced from: [Customer DataSets \(kaggle.com\)](https://www.kaggle.com/datasets/ashishpatel26/customer-dataset)  
This dataset contains the customer's card transaction from a bank. The datasets consist of 2 separates CSV files.

Below are the columns of the first dataset:

*Table 1: First Dataset column.*

Attribute	Data Type	Description
cust_id	Object	Identification of Credit Card holder (Categorical)
balance	Float	Total amount of money that you owe to your credit card company
balance_frequency	Float	How frequently the balance is updated, score between 0 and 1 (1 = frequently updated, 0 = not frequently updated)
purchases	Float	Amount of purchases made from account (oneoff_PURCHASE+installments_purchases)
oneoff_purchases	Float	Maximum purchase amount done in one-go
installments_purchases	Float	Amount of purchase done in installment

Attribute	Data Type	Description
cash_advance	Float	Cash in advance given by the user
purchases_frequency	Float	How frequently the purchases are being made, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased)
oneoff_purchases_frequency	Float	How frequently purchases are happening in one-go (1 = frequently purchased, 0 = not frequently purchased)

Below are the columns of the second dataset:

*Table 2: Columns of Second Dataset*

Attribute	Data Type	Description
cust_id	Object	Identification of Credit Card holder (Categorical)
cash_advance_frequency	Float	How frequently the cash in advance being paid
cash_advance_trx	Integer	Number of Transactions made with "Cash in Advanced"
purchases_trx	Integer	Number of purchase transactions made
credit_limit	Float	Limit of Credit Card for user
payments	Float	Amount of payment done by user
minimum_payments	Float	minimum amount of payments made by user
prc_full_payment	Float	Percent of full payment paid by user
tenure	Integer	tenure of credit card service for user in years

### 3. DATA IMPORT AND PREPROCESSING

#### 2.1. Dataset Import

Since there are 2 datasets, we will be using Talend Open Studio for Data Integration. Talend Open Studio for Data Integration is a popular choice for data preprocessing because it offers a wide range of features, including the ability to add data quality, big data integration, and processing resources.

These nodes are arranged as below:

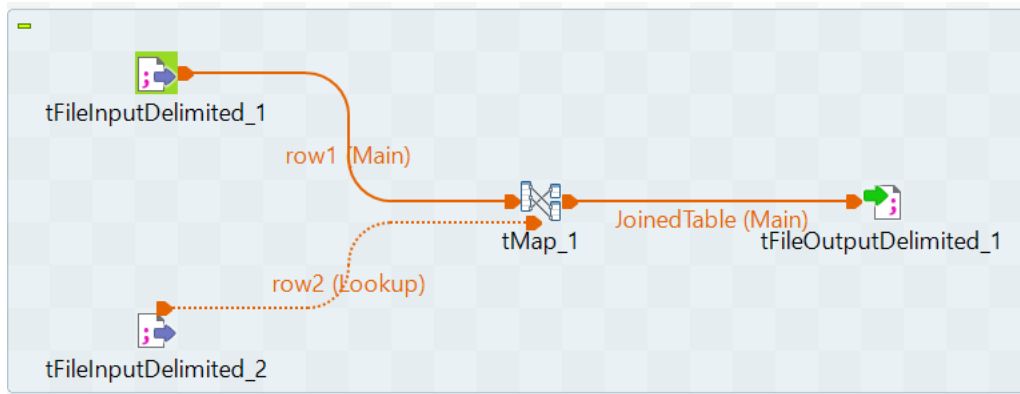


Figure 1: Talend Open Studio for Data Integration nodes arrangement.

There are 2 tFileInputDelimited nodes for each dataset. Both node settings are as below:

Row Separator:  Field Separator:  \*

☐ CSV options

Header:  Footer:  Limit:

☒ Skip empty rows ☐ Uncompress as zip file ☐ Die on error

Figure 2: tFileInputDelimited node settings.

The header will be the first row and their Filed Separator is “,”.

After the datasets are imported. tFileInputDelimited nodes will be set their schemas, below are the schema for both datasets. All is float except CUST\_ID will be string:

Column	K...	Type
CUST_ID	<input type="checkbox"/>	String
BALANCE	<input type="checkbox"/>	Float
BALANCE_FREQUENCY	<input type="checkbox"/>	Float
PURCHASES	<input type="checkbox"/>	Float
ONEOFF_PURCHASES	<input type="checkbox"/>	Float
INSTALLMENTS_PURCHASES	<input type="checkbox"/>	Float
CASH_ADVANCE	<input type="checkbox"/>	Float
PURCHASES_FREQUENCY	<input type="checkbox"/>	Float
ONEOFF_PURCHASES_FREQUENCY	<input type="checkbox"/>	Float

Figure 3: Schema of first dataset

Column	K...	Type
CUST_ID	<input type="checkbox"/>	String
PURCHASES_INSTALLMENTS_FREQUENCY	<input type="checkbox"/>	Float
CASH_ADVANCE_FREQUENCY	<input type="checkbox"/>	Float
CASH_ADVANCE_TRX	<input type="checkbox"/>	Float
PURCHASES_TRX	<input type="checkbox"/>	Float
CREDIT_LIMIT	<input type="checkbox"/>	Float
PAYMENTS	<input type="checkbox"/>	Float
MINIMUM_PAYMENTS	<input type="checkbox"/>	Float
PRC_FULL_PAYMENT	<input type="checkbox"/>	Float
TENURE	<input type="checkbox"/>	Float

Figure 4: Schema of second dataset

## 2.2. Dataset Join

Once the datasets are imported and their schemas been set, they both need to be joined. To join the datasets, connect the tFileInputDelimited to tMap\_1 node. Since both datasets have CUST\_ID in common, CUST\_ID will be used for as the key to map both datasets.

Below are the settings of the mapping:


row1	
Column	
CUST_ID	
BALANCE	
BALANCE_FREQUENCY	
PURCHASES	
ONEOFF_PURCHASES	
INSTALLMENTS_PURCHASES	
CASH_ADVANCE	
PURCHASES_FREQUENCY	
ONEOFF_PURCHASES_FREQUENCY	
row2	
Expr. key	Column
 row1.CUST_ID	CUST_ID
	PURCHASES_INSTALLMENTS_FR...
	CASH_ADVANCE_FREQUENCY
	CASH_ADVANCE_TRX
	PURCHASES_TRX
	CREDIT_LIMIT
	PAYMENTS
	MINIMUM_PAYMENTS
	PRC_FULL_PAYMENT
	TENURE

Figure 5: tMap node mapping settings.

After mapping both datasets, we can set the outcome of the joined dataset be as the table below:

joinedtable	
Expression	Column
row1.CUST_ID	CUST_ID
row1.BALANCE	BALANCE
row1.BALANCE_FREQUENCY	BALANCE_FREQUENCY
row1.PURCHASES	PURCHASES
row1.ONEOFF_PURCHASES	ONEOFF_PURCHASES
row1.INSTALLMENTS_PURCHASES	INSTALLMENTS_PURCHASES
row1.CASH_ADVANCE	CASH_ADVANCE
row1.PURCHASES_FREQUENCY	PURCHASES_FREQUENCY
row1.ONEOFF_PURCHASES_FREQUENCY	ONEOFF_PURCHASES_FREQUENCY
row2.PURCHASES_INSTALLMENTS_FREQUENCY	PURCHASES_INSTALLMENTS_FREQ...
row2.CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_FREQUENCY
row2.CASH_ADVANCE_TRX	CASH_ADVANCE_TRX
row2.PURCHASES_TRX	PURCHASES_TRX
row2.CREDIT_LIMIT	CREDIT_LIMIT
row2.PAYMENTS	PAYMENTS
row2.MINIMUM_PAYMENTS	MINIMUM_PAYMENTS
row2.PRC_FULL_PAYMENT	PRC_FULL_PAYMENT
row2.TENURE	TENURE

Figure 6: Joined dataset after mapping.

### 2.3. Dataset Produce

To produce the joined dataset, connect tMap\_1 node to tOutputDelimited. What we want is the joined dataset to be exported as a single CSV file. Below are the settings of the node, the output will include their headers.

File Name "D:\out.csv" ...

Row Separator "\n" Field Separator ","

☐ Append ☒ Include Header ☐ Compress as zip file

Schema Built-In Edit schema ☐ Sync columns

Figure 7: tOutputDelimited node settings.

The separator is set to “,” as usual CSV file.

### 2.1. Data Cleaning

To clean the data, we will use Talend Data Preparation. Talend Data Preparation is a great choice for data preprocessing because it offers a wide range of features, including the ability to quickly identify errors and apply rules that you can easily reuse and share, even across massive data sets. This is the result after importing the dataset:

	CUST_ID	BALANCE	BALANCE_FREQ...	PURCHASES	ONEOFF_PURCH...	INSTALLMENTS...	CASH_ADVANCE	PURCHASES_FR...	ONEOFF_PURCH...	PURCHASES_IN...	CASH_ADVANCE...	CASH
		text	decimal	decimal	decimal	decimal	decimal	decimal	geo_coordinate	decimal	decimal	
7418	C17619		33.88002	0.909091	3421.12	3000.55	412.57	209.6194	0.916667	0.833333	0.25	0.166667
7419	C17620		21.881529	1.0	89.04	89.04	0.0	0.0	0.083333	0.083333	0.0	0.0
7420	C17621		1223.631	1.0	2065.5	628.8	1436.7	0.0	1.0	0.166667	1.0	0.0
7421	C17622		1441.9335	0.909091	0.0	0.0	0.0	2464.6848	0.0	0.0	0.0	0.333333
7422	C17623		1070.6313	1.0	651.91	651.91	0.0	40.07681	0.416667	0.416667	0.0	0.083333
7423	C17624		594.26886	0.909091	1836.76	966.0	69.96	0.0	0.583333	0.416667	0.25	0.0
7424	C17625		25.715061	0.363636	449.32	449.32	0.0	0.0	0.166667	0.166667	0.0	0.0
7425	C17626		1439.4102	1.0	184.7	184.7	0.0	0.0	0.166667	0.166667	0.0	0.0
7426	C17628		543.8152	1.0	0.0	0.0	0.0	45.47558	0.0	0.0	0.0	0.083333
7427	C17629		36.979202	0.181818	0.0	0.0	0.0	4262.12	0.0	0.0	0.0	0.166667
7428	C17630		9220.534	1.0	1420.31	777.66	642.65	9816.782	1.0	0.25	1.0	0.833333
7429	C17631		6631.79	1.0	74.17	74.17	0.0	4618.935	0.090909	0.090909	0.0	0.454545
7430	C17632		39.251217	1.0	1066.62	339.1	727.52	204.61537	0.833333	0.833333	0.916667	0.083333
7431	C17633		2.02217	0.181818	0.0	0.0	0.0	438.9244	0.0	0.0	0.0	0.090909
7432	C17634		38.06822	0.727273	799.92	0.0	799.92	0.0	1.0	0.0	1.0	0.0
7433	C17635		578.7958	0.545455	994.32	105.0	889.12	2578.5637	0.833333	0.166667	0.833333	0.75
7434	C17636		1325.6799	1.0	1084.87	687.57	397.3	3330.9136	0.833333	0.333333	0.833333	0.416667
7435	C17637		743.52826	1.0	158.95	158.95	0.0	0.0	0.375	0.375	0.0	0.0
7436	C17638		0.0	0.0	180.66	54.0	126.66	0.0	0.583333	0.083333	0.5	0.0
7437	C17639		41.78926	1.0	821.51	581.95	319.56	109.91813	0.833333	0.083333	0.833333	0.083333
7438	C17640		987.1106	1.0	0.0	0.0	0.0	1139.393	0.0	0.0	0.0	0.142857
7439	C17641		11.981427	0.727273	187.55	0.0	187.55	0.0	0.75	0.0	0.75	0.0
7440	C17642		11.316443	0.181818	112.23	112.23	0.0	0.0	0.083333	0.083333	0.0	0.0
7441	C17643		0.0	0.0	300.0	0.0	300.0	0.0	0.083333	0.0	0.083333	0.0

Figure 8: Customer Datasets in Talent Data Preparation.

Some columns have wrong data types assigned to them, some of them becomes Integer and 1 becomes a Geo\_Coordinate. Hence why the colour below the column names is not green. Indicated some data produce errors. Below are the columns and their datatypes:

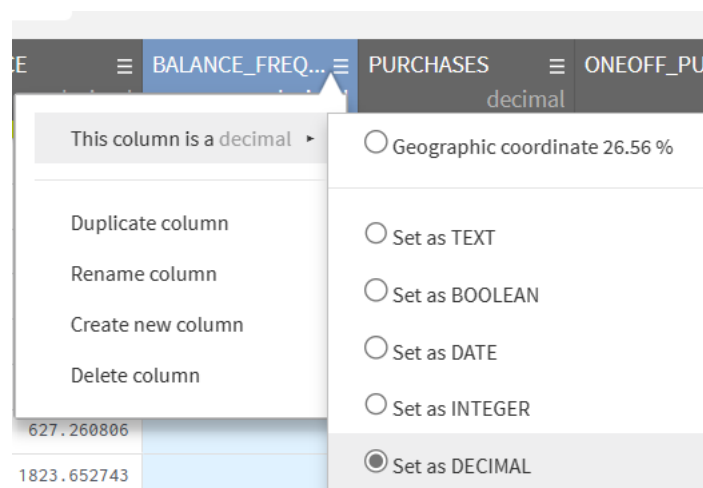
BALANCE	BALANCE_FREQ...	PURCHASES	ONEOFF_PURCH...	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FR...	ONEOFF_PURCH...
decimal	integer	decimal	integer		integer	decimal	decimal
						geo_coordinate	

PURCHASES_IN...	CASH_ADVANCE...	CASH_ADVANCE...	PURCHASES_TRX	CREDIT_LIMIT	PAYMENTS	MINIMUM_P...	PRC_FULL_PAY...	TENURE
integer	decimal	integer	integer	integer	decimal	decimal	integer	integer

Figure 9: The datatypes of Customer Datasets columns in Talent Data Preparation.

We can set them to correct datatypes, which is Decimal by clicking the columns such as below:



By correcting all datatypes, now we can see all colour turns into green. Below are current datatypes of the columns:

BALANCE	BALANCE_FREQ...	PURCHASES	ONEOFF_PURCH...	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FR...	ONEOFF_PURCH...	PURCHASES_IN...
decimal	decimal	decimal	decimal		decimal	decimal	decimal	decimal

CASH_ADVANCE...	CASH_ADVANCE...	PURCHASES_TRX	CREDIT_LIMIT	PAYMENTS	MINIMUM_P...	PRC_FULL_PAY...	TENURE
decimal	integer	integer	decimal	decimal	decimal	decimal	integer

Figure 10: The datatypes of Customer Datasets columns in Talent Data Preparation after cleaning.

## 2.2. Missing Data

By clicking the columns one by one, we can see the value of missing data of the column, this is the result of CREDIT\_LIMIT and MINIMUM\_PAYMENTS:

CHART	VALUE	PATTERN	ADVANCED	CHART	VALUE	PATTERN	ADVANCED
Count: <b>8950</b>			Min: <b>50</b>	Count: <b>8950</b>			Min: <b>0.02</b>
Distinct: <b>206</b>			Max: <b>30000</b>	Distinct: <b>8637</b>			Max: <b>76406.21</b>
			Mean: <b>4494.45</b>				Mean: <b>864.21</b>
Duplicate: <b>8744</b>			Variance: <b>13240979.88</b>	Duplicate: <b>313</b>			Variance: <b>5628502.9</b>
			Valid: <b>8949</b>				Valid: <b>8637</b>
			Median: <b>3000</b>				Median: <b>312.34</b>
Empty: <b>1</b>			Lower quantile: <b>1600</b>	Empty: <b>313</b>			Lower quantile: <b>169.1</b>
Invalid: <b>0</b>			Upper quantile: <b>6500</b>	Invalid: <b>0</b>			Upper quantile: <b>825.51</b>

Figure 11: Aggregate values of CREDIT\_LIMIT (left) and MINIMUM\_PAYMENTS (right).

We can see that there are 1 missing data in CREDIT\_LIMIT and 313 missing data in MINIMUM\_PAYMENTS. However, we cannot replace CREDIT\_LIMIT and MINIMUM\_PAYMENTS with 0 or null. CREDIT\_LIMIT is the limit of customer's credit and MINIMUM\_PAYMENTS is the minimum payment made by customer. This column will be imputed in SAS Enterprise Miner.

Now the cleaned dataset can be exported to a CSV file.



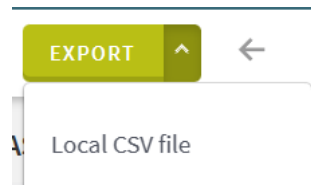


Figure 12: Exporting the dataset.

## 4. PREDICTION

### 3.1. Data Import

Now the dataset is ready for prediction. The prediction will be done using SAS Enterprise Miner. SAS Enterprise Miner is a great choice for prediction because it offers a wide range of features, including the ability to generate models using SAS Rapid Predictive Modeler, which is an easy-to-use GUI that steps you through a workflow of data mining tasks. We can start by importing the dataset using File import node.

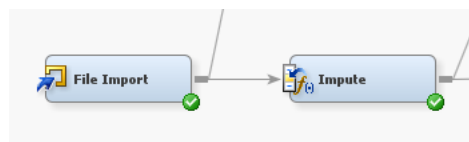


Figure 13: File Import node.

Notes	
<b>Train</b>	
Variables	
Import File	D:\Master Data Science\Data Mining\Cus ...
Maximum Rows to Import	1000000
Maximum Columns to Import	10000
Delimiter	,
Name Row	Yes
Number of Rows to Skip	0
Guessing Rows	500
File Location	Local
File Type	csv
Advanced Advisor	No
Rerun	No
<b>Score</b>	
Role	Train

Figure 14: File Import node settings.

Above are the settings of the node. The Import File is the Customer Datasets file we exported after cleaning in Talend Data Preparation, the Maximum Row to Import will be set to 1000000 so all row will be imported, the Delimiter of this dataset is “,” so it will set to “,”. Since the file contains headers. Name Row will be “Yes”. The role of this dataset will be “Train” as it will be used for training the models.

### 3.2. Variables Settings

The roles of each attributes need to be decided. The roles of attributes define the purpose of each attribute in the dataset, which is essential for accurate prediction. By setting the roles of each attribute,

we can ensure that the data is properly prepared for prediction and that the model is trained on the correct variables.

The CUST\_ID will be set to ID as it is the ID of the Customers. Thus, not really contribute for prediction. The target role will be assigned to Balance.

(none)

▼

☐
not

Equal to

▼

...

Columns:

☐ Label

☐ Mining

☐ Basic

Name ▲	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
BALANCE	Target	Interval	No		No	.	.
BALANCE FREQUENCY	Input	Interval	No		No	.	.
CASH ADVANCE	Input	Interval	No		No	.	.
CASH ADVANCE FREQUENCY	Input	Interval	No		No	.	.
CASH ADVANCE TRX	Input	Interval	No		No	.	.
CREDIT LIMIT	Input	Interval	No		No	.	.
CUST ID	ID	Nominal	No		No	.	.
INSTALLMENTS PURCHASED	Input	Interval	No		No	.	.
MINIMUM PAYMENTS	Input	Interval	No		No	.	.
ONEOFF PURCHASES	Input	Interval	No		No	.	.
ONEOFF PURCHASES FREQUENCY	Input	Interval	No		No	.	.
PAYMENTS	Input	Interval	No		No	.	.
PRC FULL PAYMENT	Input	Interval	No		No	.	.
PURCHASES	Input	Interval	No		No	.	.
PURCHASES FREQUENCY	Input	Interval	No		No	.	.
PURCHASES INSTALLMENTS	Input	Interval	No		No	.	.
PURCHASES TRX	Input	Interval	No		No	.	.
TENURE	Input	Interval	No		No	.	.

Figure 15: Customer Datasets Variable settings.

### 3.3. StatExplore

To measure how much data is missing, we can use StatExplore node. The StatExplore node in SAS Enterprise Miner is a multipurpose tool that can be used to examine variable distributions and statistics in your data sets. It generates summarization statistics and provides a wide range of features, including the ability to select variables for analysis, compute standard univariate distribution statistics, compute standard bivariate statistics by class target and class segment, and compute correlation statistics for interval variables by interval input and target.

We will connect File Import node to StatExplore and run it. By running the node, we can see the result.

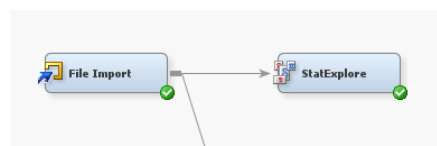


Figure 16: StatExplore node.

Below are the results from StatExplore:

Variable	Role	Mean	Standard Deviation	Non Missing	Missing
BALANCE_FREQUENCY	INPUT	0.877271	0.236904	8950	0
CASH_ADVANCE	INPUT	978.8711	2097.164	8950	0
CASH_ADVANCE_FREQUENCY	INPUT	0.135144	0.200121	8950	0
CASH_ADVANCE_TRX	INPUT	3.248827	6.824647	8950	0
CREDIT_LIMIT	INPUT	4494.449	3638.816	8949	1
INSTALLMENTS_PURCHASES	INPUT	411.0676	904.3381	8950	0
MINIMUM_PAYMENTS	INPUT	864.2065	2372.447	8637	313
ONEOFF_PURCHASES	INPUT	592.4374	1659.888	8950	0
ONEOFF_PURCHASES_FREQUENCY	INPUT	0.202458	0.298336	8950	0
PAYMENTS	INPUT	1733.144	2895.064	8950	0
PRC_FULL_PAYMENT	INPUT	0.153715	0.292499	8950	0
PURCHASES	INPUT	1003.205	2136.635	8950	0
PURCHASES_FREQUENCY	INPUT	0.490351	0.401371	8950	0
PURCHASES_INSTALLMENTS_FREQUENCY	INPUT	0.364437	0.397448	8950	0
PURCHASES_TRX	INPUT	14.70983	24.85765	8950	0
TENURE	INPUT	11.51732	1.338331	8950	0
BALANCE	TARGET	1564.475	2081.532	8950	0

Figure 17: StatExplore results.

From the result, we can see CREDIT\_LIMIT is missing 1 data while MINIMUM\_PAYMENTS lost 313 data similar in Talend Data Preparation.

### 3.3.1. Correlation

The result also produced Correlation chart of the attributes with target, BALANCE.

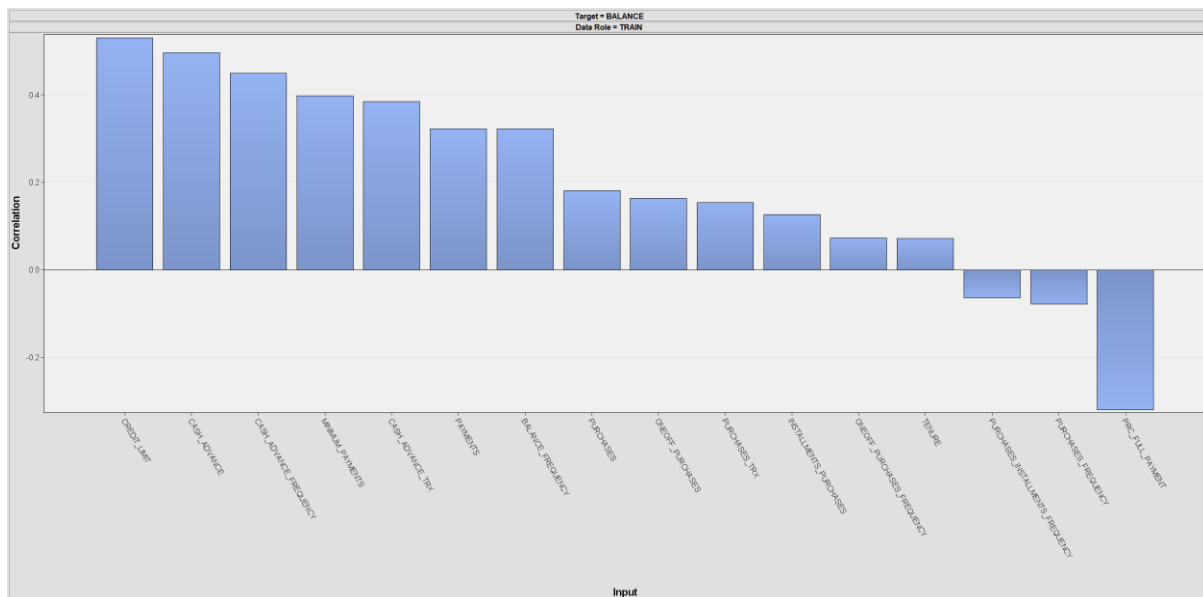


Figure 18: Corelation chart for target: balance.

The result of CORRELATION\_CHART is:

*Table 3: Correlation value for each variables to target.*

Input	Correlation
CREDIT_LIMIT	0.53128
CASH_ADVANCE	0.49669
CASH_ADVANCE_FREQUENCY	0.44922
MINIMUM_PAYMENTS	0.39868
CASH_ADVANCE_TRX	0.38515
PAYMENTS	0.3228
BALANCE_FREQUENCY	0.32241
PURCHASES	0.18126
ONEOFF_PURCHASES	0.16435
PURCHASES_TRX	0.15434
INSTALLMENTS_PURCHASES	0.12647
ONEOFF_PURCHASES_FREQUENCY	0.07317
TENURE	0.07269
PURCHASES_INSTALLMENTS_FREQUENCY	-0.06319
PURCHASES_FREQUENCY	-0.07794
PRC_FULL_PAYMENT	-0.31896

We can see that CREDIT\_LIMIT has the highest correlation to BALANCE, while PRC\_FULL\_PAYMENT has the lowest correlation.

### 3.4. Imputation

Imputation is the process of replacing missing data with substituted values. It is a common technique used in data preprocessing to handle missing data. There are several reasons why data may be missing, including data entry errors, equipment failure, and survey non-response. Imputation is used to replace missing values with estimated values based on the available data. This can help to reduce bias and improve the accuracy of statistical analyses.

To apply imputation to our dataset, Impute node will be use.



Figure 19: Impute node.

In the case of the MINIMUM\_PAYMENTS attribute, mean imputation is a suitable method for replacing missing values because it is a simple and effective method that can be used for continuous variables. By replacing missing values with the mean value of the non-missing values in the same column, we can ensure that the data is complete and can be used for analysis.

The Default Input Method for Interval Variables will be set to Mean for mean imputation.

Train	
Variables	...
Nonmissing Variables	No
Missing Cutoff	50.0
Class Variables	
Default Input Method	Count
Default Target Method	None
Normalize Values	Yes
Interval Variables	
Default Input Method	Mean
Default Target Method	None
Default Constant Value	
Default Character Value	
Default Number Value	.
Method Options	
Random Seed	12345
Tuning Parameters	...
Tree Imputation	...

Figure 20: Mean Imputation node settings.

After running the Impute node, we can look at the result using StatExplore node, we can see CREDIT\_LIMIT and MINIMUM\_PAYMENTS became IMP\_CREDIT\_LIMIT and IMP\_MINIMUM\_PAYMENTS after imputed. Now no missing data exist. However, due to the mean imputation, the minimum value of MINIMUM\_PAYMENTS is slightly above 0, but this is acceptable as no customer will have 0 minimum payments.

Variable	Role	Mean	Standard Deviation	Non Missing	Missing	Minimum
BALANCE_FREQUENCY	INPUT	0.877271	0.236904	8950	0	0
CASH_ADVANCE	INPUT	978.8711	2097.164	8950	0	0
CASH_ADVANCE_FREQUENCY	INPUT	0.135144	0.200121	8950	0	0
CASH_ADVANCE_TRX	INPUT	3.248827	6.824647	8950	0	0
IMP_CREDIT_LIMIT	INPUT	4494.449	3638.612	8950	0	50
IMP_MINIMUM_PAYMENTS	INPUT	864.2065	2330.588	8950	0	0.019163
INSTALLMENTS_PURCHASES	INPUT	411.0676	904.3381	8950	0	0
ONEOFF_PURCHASES	INPUT	592.4374	1659.888	8950	0	0
ONEOFF_PURCHASES_FREQUENCY	INPUT	0.202458	0.298336	8950	0	0
PAYMENTS	INPUT	1733.144	2895.064	8950	0	0
PRC_FULL_PAYMENT	INPUT	0.153715	0.292499	8950	0	0
PURCHASES	INPUT	1003.205	2136.635	8950	0	0
PURCHASES_FREQUENCY	INPUT	0.490351	0.401371	8950	0	0
PURCHASES_INSTALLMENTS_FREQUENCY	INPUT	0.364437	0.397448	8950	0	0
PURCHASES_TRX	INPUT	14.70983	24.85765	8950	0	0
TENURE	INPUT	11.51732	1.338331	8950	0	6
BALANCE	TARGET	1564.475	2081.532	8950	0	0

Figure 21: Imputation StatExplore result.

### 3.5. Data Splitting

Before we deploy prediction models to our dataset, the dataset must be split into training, validation and testing.

The dataset must be split into training, validation, and testing sets to ensure that the machine learning model is generalizable and accurate. The training set is used to train the model, the validation set is used to tune the model's hyperparameters, and the testing set is used to evaluate the model's performance on new, unseen data.

To do this Data Partition node will be used.

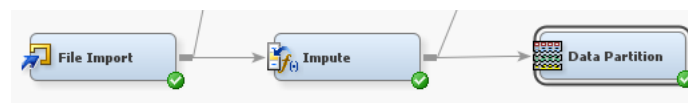


Figure 22: Data Partition node.

Train	
Variables	
Output Type	Data
Partitioning Method	Default
Random Seed	12345
Data Set Allocations	
Training	60.0
Validation	10.0
Test	30.0
Report	
Interval Targets	Yes
Class Targets	Yes

Figure 23: Data Partition node settings.

In the settings, training, validation and testing will be 60%, 10% and 30% respectively from the dataset.

Now we are ready to apply prediction model.

### 3.6. Decision Tree Analysis

#### 3.6.1. Decision Tree

Decision Tree node will be used to apply Decision Tree to predict BALANCE. The Decision Tree node in SAS Miner is used to create decision trees that can classify observations based on the values of nominal, binary, or ordinal targets, predict outcomes for interval targets, or predict the appropriate decision when you specify decision alternatives.

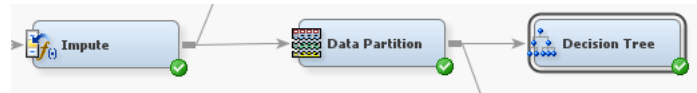


Figure 24: Decision Tree node.

Train	
Variables	
Interactive	
Import Tree Model	No
Tree Model Data Set	
Use Frozen Tree	No
Use Multiple Targets	No
Splitting Rule	
Interval Target Criterion	ProbF
Nominal Target Criterion	ProbChisq
Ordinal Target Criterion	Entropy
Significance Level	0.2
Missing Values	Use in search
Use Input Once	No
Maximum Branch	3
Maximum Depth	6
Minimum Categorical Size	5
Node	
Leaf Size	5
Number of Rules	5
Number of Surrogate Rules	0
Split Size	.

Figure 25: Decision Tree node settings.

The Maximum Branch will be set from 2 to 3 due to most of attributes are interval variables. By having more branch, it can lead to a more complex model that can fit the training data better. The maximum depth will be 6.

After running the node, these are the result:

### 3.6.2.Tree

This is the tree created from the node. The tree is cropped from left to right and viewed from top to bottom.



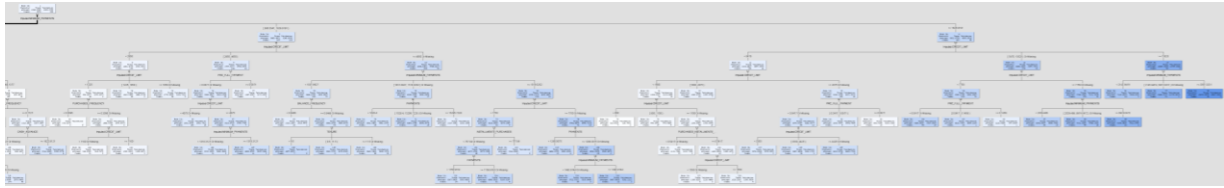


Figure 26: Tree created from Decision Tree result.

### 3.6.3. Score Rankings Matrix: Balance

A Score Rankings Matrix is a graphical representation of the performance of different models in a machine learning task. It is used to compare the performance of different models based on their accuracy, precision, recall, and other metrics.

For this node, blue will be Predicted value and red will be Target value. Top chart will be training dataset while bottom will be Validation dataset.

#### 3.6.3.1. Mean

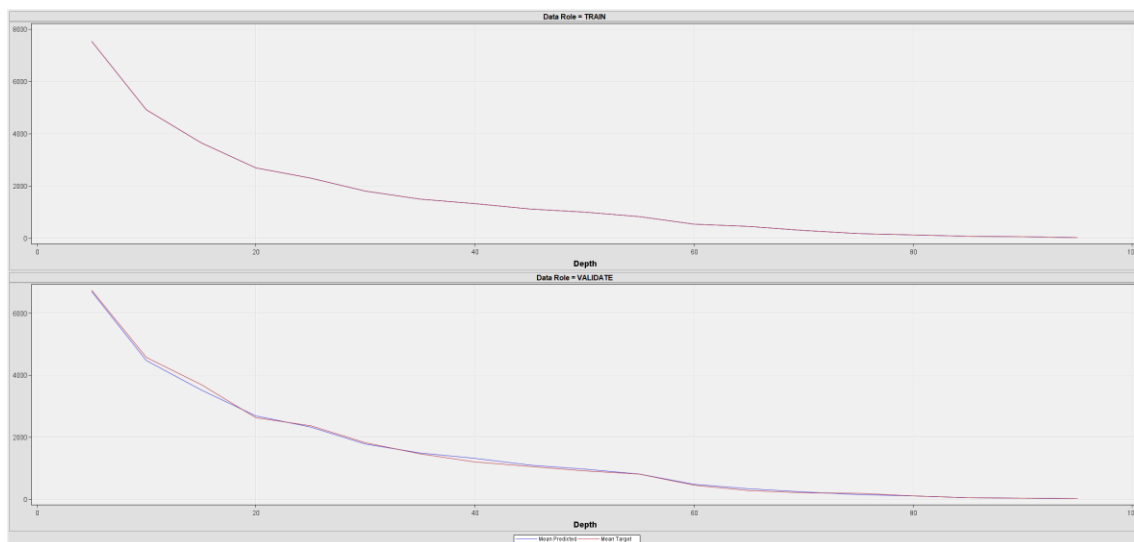


Figure 27: Score Rankings Matrix (Mean) for Decision Tree.

Both charts show a decline in values as depth increases. The predicted mean and target mean align perfectly in both the Training and Validation data sets, indicating that the predictions are extremely accurate or even perfect.



### 3.6.3.2. Max

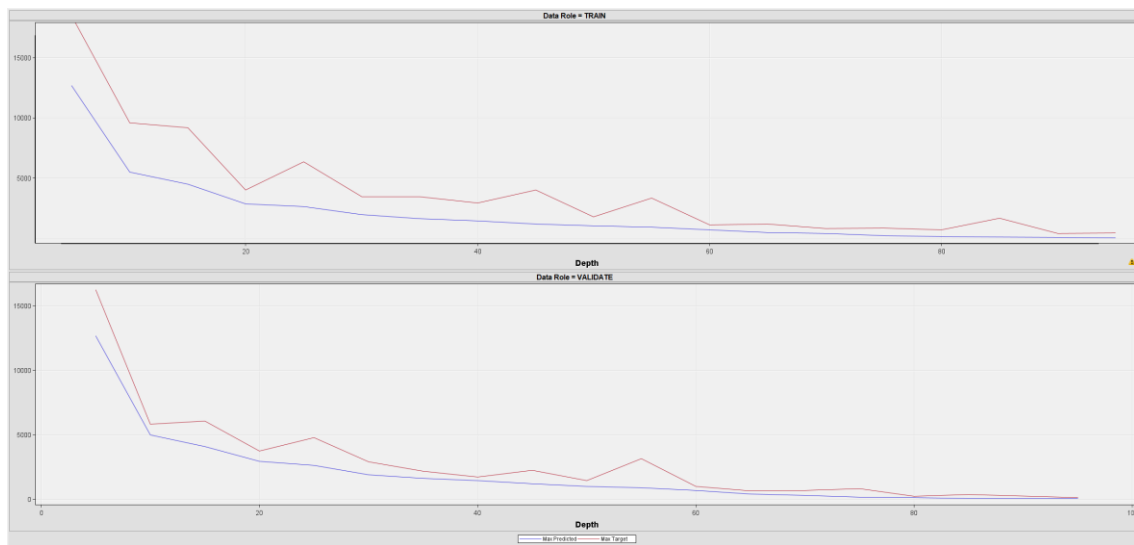


Figure 28: Score Rankings Matrix (Max) for Decision Tree.

The max predicted are consistently lower than the max target in both the training and validation data. Although the two lines are close to each other, they do not intersect. This indicates that the model is underpredicting the target variable. In other words, the model is not accurately capturing the relationship between the input features and the target variable. This could be due to a variety of reasons such as insufficient data, poor feature selection, or an inadequate model architecture. The validation set however shows better result than training set as the max predicted and max target are closer.

### 3.6.3.3. Min

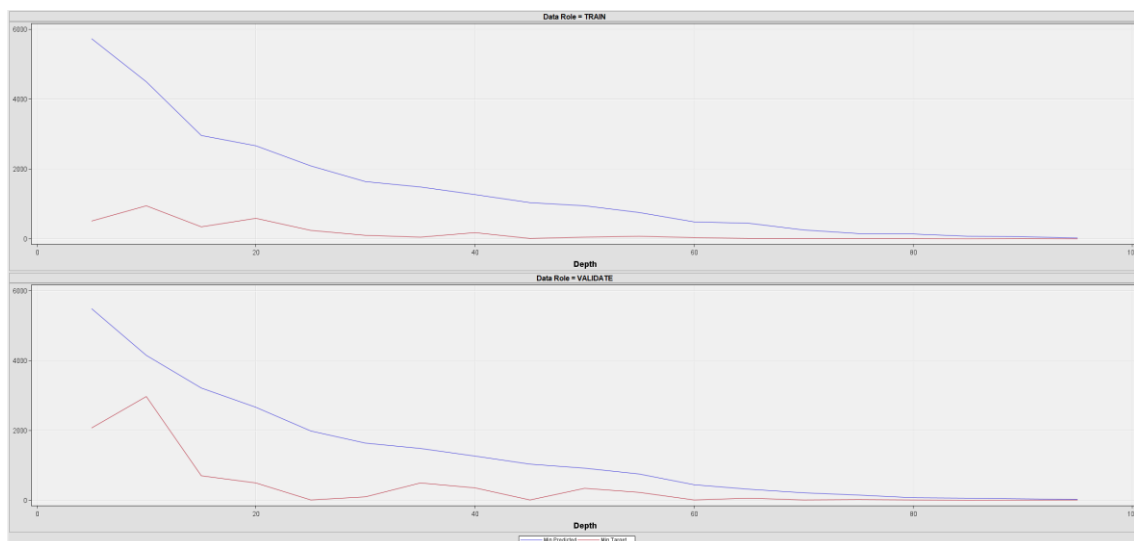


Figure 29: Score Rankings Matrix (Min) for Decision Tree.

In both the training and validation data, the min predicted starts at a higher point and declines more steeply than the min target, which starts at a lower point and has a less steep decline. This could indicate that the model initially overestimates the target balance but becomes more accurate as depth increases, with predictions and targets converging.

The min predicted in both charts starts from a higher point and declines more steeply compared to the min target. The red line in training starts at a much lower point than in validation. The validation data has some increases at depth before going down. They gradually get closer and closer as the depth increases.

This could indicate that the model is overfitting to the training data, which means that it is too complex and is fitting the noise in the data instead of the underlying pattern. As a result, the model is not generalizing well to new data, which is why the validation data has a less steep decline in the red line. The increases in the validation data could be due to the model learning new patterns in the data that were not present in the training data.

The document highlights some of the strengths and weaknesses of the decision tree model, such as its accuracy, simplicity, interpretability, and ability to handle different types of variables. The document also identifies some of the potential problems of the decision tree model, such as overfitting, underpredicting, and overestimating.

### 3.7. Random Forest

#### 3.7.1. Iteration Plot

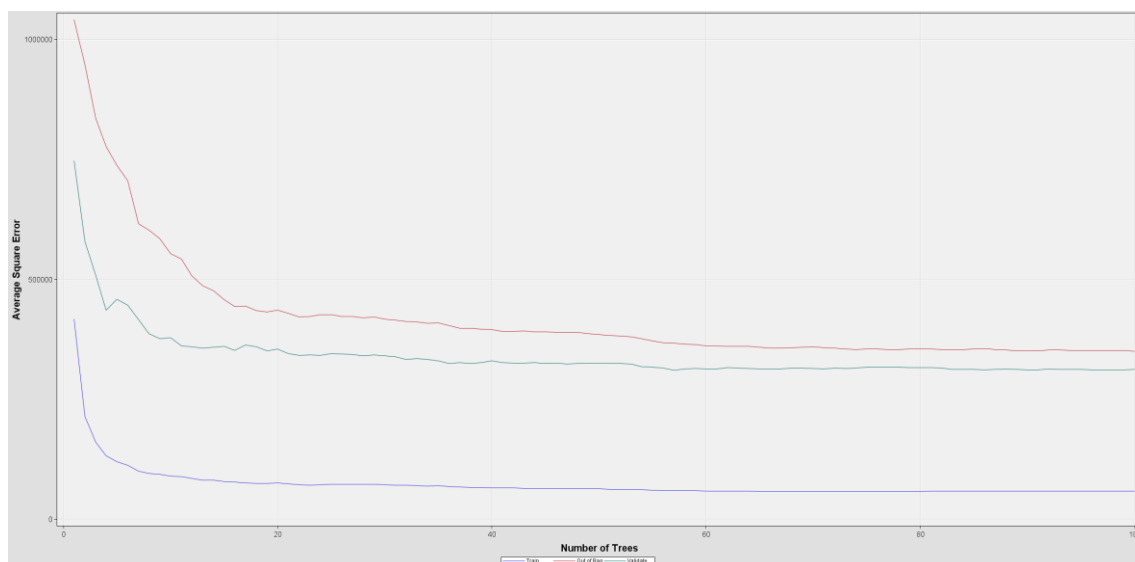
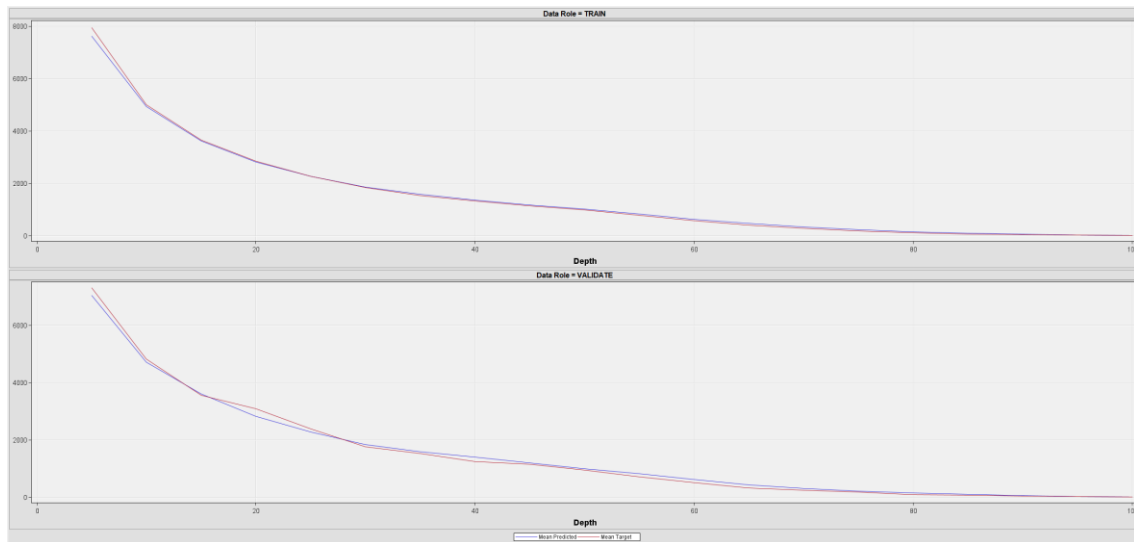


Figure 30: Iteration Plot Random Forest.

As the number of trees increases, the average square error for train, out of bag, and validate decreases. All three metrics seem to stabilize around 60 trees, indicating that increasing the number of trees beyond this point may not result in a significant reduction in error.

### 3.7.2.Score Rankings Matrix: Balance

#### 3.7.2.1. Mean



*Figure 31: Score Rankings Matrix (Mean) for Random Forest.*

Similar to Decision Tree. Both charts show a decline in values as depth increases. The predicted mean and target mean align perfectly in both the Training and Validation data sets, indicating that the predictions are extremely accurate or even perfect.

### 3.7.2.2. Max

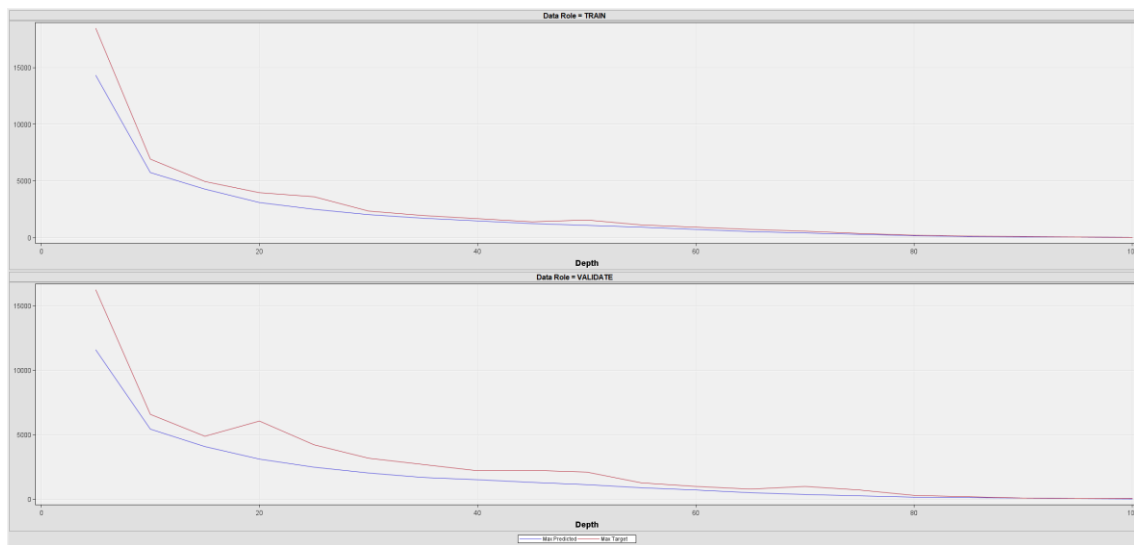


Figure 32: Score Rankings Matrix (Max) for Random Forest.

The max predicted is consistently below the mean target. However, as depth increases, particularly at depth 20 and 80, the mean predicted approaches closer to the max target indicating an improvement in prediction accuracy. This could suggest that as the model is trained with more depth, its predictions become more accurate and align closely with the target values.

### 3.7.2.3. Min

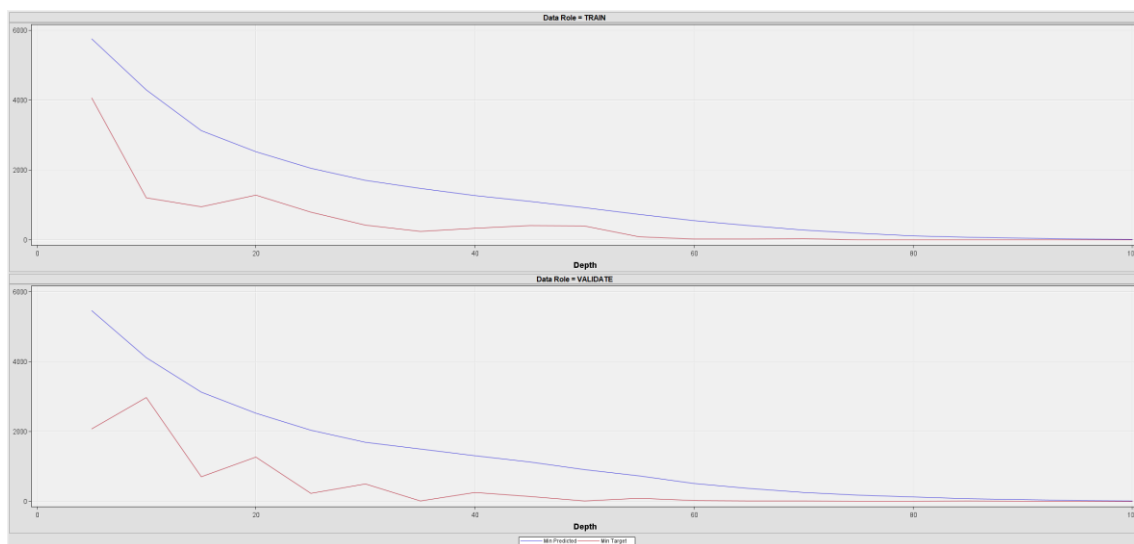


Figure 33: Score Rankings Matrix (Min) for Random Forest.

The min predicted starts at a higher point than the min target, indicating a higher initial score ranking. As depth increases, both lines decline, but they do so at similar rates, indicating a consistent reduction in score rankings over increased depth.

For the Training data, both min decline smoothly without significant fluctuations. However, in the Validation data graph, there are noticeable spikes in the min target, indicating occasional increases in score ranking before continuing its decline. These spikes occur at depths 10, 20, and 30. This could be due to overfitting or noise in the training data that affects model performance during these specific depths.

Over increased depth, the gap between min predicted and min target narrows down; this could be due to model optimization or learning where predictions are becoming more accurate or aligned with targets as depth increases.

## **5. Analysis**

### **4.1. Decision Tree**

The model has a perfect mean prediction for both the training and validation data sets, indicating a high accuracy. However, the model also has some limitations, such as underpredicting the max balance, overfitting to the training data, and overestimating the min balance. These limitations could be due to the model being too complex or not capturing the nonlinear relationships in the data.

### **4.2. Random Forest**

The model has a lower average square error than the decision tree model, indicating a better fit to the data. The model also has a better max prediction, especially at higher depths, indicating a better capture of the outliers. However, the model still has some limitations, such as overestimating the min balance and having spikes in the validation data, indicating some overfitting or noise in the data. These limitations could be reduced by tuning the hyperparameters or using feature selection techniques.

## **6. Conclusion**

The balance of credit card reflects the customer behaviour and the credit utilization. Customers with higher credit limits and more purchases tend to have higher balances, as they use their credit cards more often and for larger amounts. Customers with higher cash advances also tend to have higher balances, as they may face cash flow problems or debt issues. Customers who pay their minimum or full payments regularly tend to have lower balances, as they are more responsible and disciplined with their credit card usage.

Balance prediction is a useful tool for various business applications, such as customer segmentation, marketing optimization, risk management, and fraud detection. By predicting the balance of each customer, businesses can customize their products, services, messages, and offers to match their customers' needs and preferences. They can also track their customers' balance and

behaviour over time, and detect any potential issues or opportunities. This way, they can enhance their customer satisfaction, loyalty, and profitability, while lowering their credit risk and losses.

**GITHUB LINK:**

[https://github.com/dannymirxa/WQD7005\\_Data\\_Mining\\_Alternative\\_Assessment\\_1](https://github.com/dannymirxa/WQD7005_Data_Mining_Alternative_Assessment_1)