Safe Driver Prediction

Problem

Nothing ruins the excitement of buying a brand new car more quickly than getting a new insurance bill. It may not seem fair that you have to pay such a high price of car insurance especially when you know you have been cautious on the road for years. Insurance companies have been overcharging the good driver and reducing the cost of insurance for the reckless ones due to inaccuracies in insurance claim predictions. A reliable forecast on the future insurance claim may enable them to further tailor their prices accordingly and make auto insurance coverage more accessible to more drivers.

Analysis Goal

The goal of this data mining project is to predict whether a driver will file an insurance claim using decision tree and logistic regression in SAS Enterprise Miner.

Analytical Workflow

Sample

The dataset used in this analysis was retrieved from Kaggle website. It consists of 13 columns and 30240 rows where each row corresponds to a policy holder and the columns contain information of the holder including their personal details and driving habits. The target variable signifies whether a claim was filed (1) or was not filed (0).

Original source of data: https://www.kaggle.com/mu202199/safe-driver-prediction/data

Before the dataset was loaded into the project, it was first converted into SAS dataset format as it was originally in Excel format. Using the Data Source Wizard, the dataset and its column metadata were loaded and defined.

Table 1: Data Metadata

Name	Model Role	Measurement Level	Description
ID	ID	Nominal	Identity
Gender	Input	Nominal	Gender
EngineHP	Input	Interval	Engine Horsepower
Credit_History	Input	Interval	Credit score
Years_Experience	Input	Interval	Years of driving experience
Annual_Claims	Input	Interval	Number of claims filed annually
Marital_Status	Input	Nominal	Marital status
Vehicle_Type	Input	Nominal	Type of vehicle
Miles_Driven_Annually	Input	Interval	Miles driven annually
Size_of_Family	Input	Interval	Family size
Age_Bucket	Input	Nominal	Age bucket
State	Input	Nominal	State of residence
Target	Target	Binary	Insurance claim flag

Explore

In the exploration stage, several visualization techniques were used to uncover initial pattern and create a broad picture of the characteristics of the dataset.

Obs #	ID	target	Gender	EngineHP	credit_history	Years_Experience	annual_claims	Marital_Status	Vehical_type	Miles_driven_annually	size_of_family	Age_bucket	State
1	1	1	F	522	656	1	0	Married	Car	14749	5	<18	IL
2	2	1	F	691	704	16	0	Married	Car	15389	6	28-34	NJ
3	3	1	M	133	691	15	0	Married	Van	9956	3	>40	CT
4	4	1	M	146	720	9	0	Married	Van	77323	3	18-27	CT
5	5	1	M	128	771	33	1	Married	Van	14183	4	>40	WY
6	6	1	F	144	722	18	1	Married	Truck	12208	8	>40	DE
7	7	1	F	151	788	31	3	Married	Truck	13957	2	>40	NJ
8	8	1	F	88	747	21	1	Single	Car	14200	5	>40	ME
9	9	1	F	653	717	34	0	Married	Car	17084	1	>40	CA
10	10	0	F	120	785	19	1	Married	Truck	0	1	35-40	NJ
11	11	1	M	150	836	20	1	Single	Utility	9485	8	>40	KS
12	12	0	F	423	718	5	3	Single	Truck	11898	2	35-40	CT
13	13	1	F	104	793	36	1	Single	Car	12385	6	>40	WV
14	14	1	F	87	577	2	1	Married	Truck	41473	4	<18	CT
15	15	1	M	191	635	5	0	Single	Car	11869	5	18-27	CT
16	16	1	M	138	734	18	1	Married	Truck	9280	3	35-40	NM
17	17	1	M	94	644	6	2	Married	Truck	92463	2	35-40	SC
18	18	1	F	82	686	22	1	Married	Car	6986	4	>40	WY
19	19	1	M	338	640	1	3	Married	Car	95547	4	18-27	CT
20	20	0	F	213	733	8	3	Single	Truck	13484	2	35-40	CA

Figure 1: First 20 rows of data

Table in Figure 1 above shows the first 20 rows of observations to give a glimpse of the data.

Output

```

Data Role=TRAIN Variable=EngineHP

             Target                          Non                                       Standard
Target       Level     Median    Missing   Missing    Minimum    Maximum       Mean   Deviation   Skewness   Kurtosis   Role     Label

_OVERALL_                 141          0     30240         80       1005   196.6043     132.347   1.668214   2.276157   INPUT   EngineHP
target         0          141          0      8844         80       1005   198.5768    133.4707   1.621589   2.063918   INPUT   EngineHP
target         1          141          0     21396         80        996   195.7889    131.8742    1.68808   2.368869   INPUT   EngineHP


Data Role=TRAIN Variable=Miles_driven_annually

             Target                          Non                                       Standard
Target       Level     Median    Missing   Missing    Minimum    Maximum       Mean   Deviation   Skewness   Kurtosis   Role             Label

_OVERALL_               12277          0     30240          0      99943   17411.42    17478.34   2.891775   7.846618   INPUT   Miles_driven_annually
target         0        12256          0      8844          0      99943   17515.71    17701.47   2.900329   7.890862   INPUT   Miles_driven_annually
target         1        12286          0     21396          0      99826   17368.31    17385.51   2.887651   7.823561   INPUT   Miles_driven_annually


Data Role=TRAIN Variable=Years_Experience

             Target                          Non                                       Standard
Target       Level     Median    Missing   Missing    Minimum    Maximum       Mean   Deviation   Skewness   Kurtosis   Role           Label

_OVERALL_                  10          0     30240          1         40   13.25437    9.890621   0.825245   -0.14444   INPUT   Years_Experience
target         0           10          0      8844          1         40   13.26594    9.932251   0.837495    -0.1199   INPUT   Years_Experience
target         1           10          0     21396          1         40   13.24958     9.87359   0.820121   -0.15476   INPUT   Years_Experience

```

Figure 2: Variables summary output

'StatExplore' node from the Explore tab was ran to provide a summary of the variables in the dataset. From the output window as shown in Figure 2, all variables do not have any missing value, however, the Miles_Driven_Annually variable has minimum value of 0 which is considered unusual as the minimum years of driving experience is 1. These values can be treated as missing values and will be handled in the pre-processing stage later.

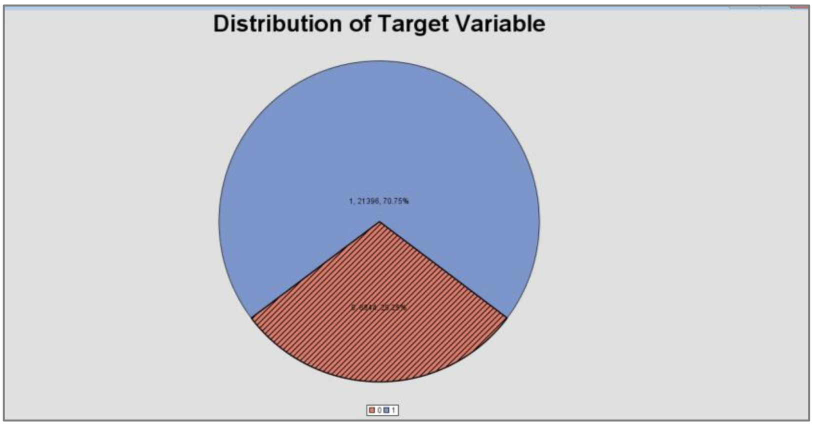


Figure 3: Pie chart of target variable distribution

The pie chart in Figure 3 shows the distribution of the target variable. The number of insurance claim filed, represented by 1 is in the majority with a percentage of 70.75% while the number of claim not being filed, represented by 0 takes up only 28.25%.

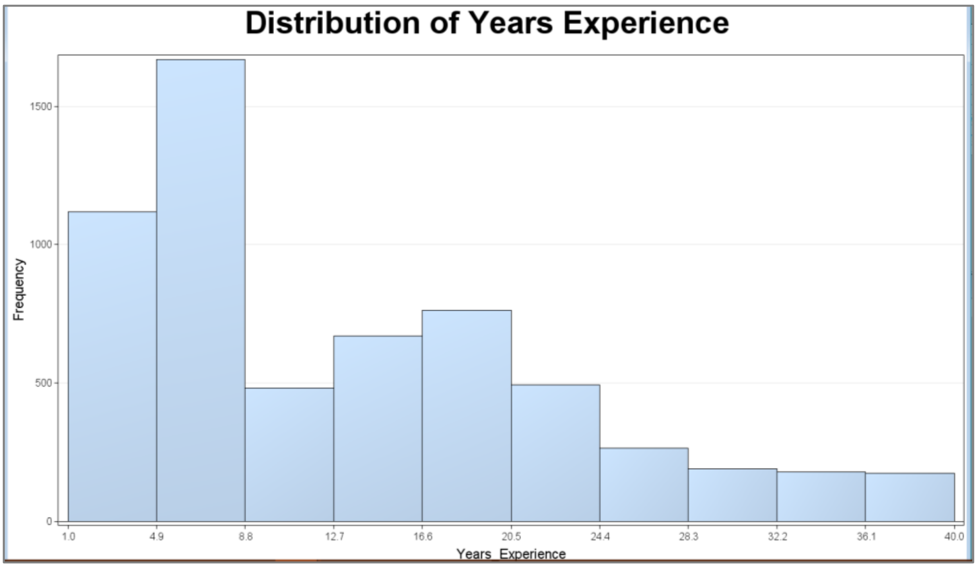


Figure 4: Histogram of years of experience with driving

Figure 4 illustrates the distribution of experience in years with driving of the policy holders who filed an insurance claim. Most of these people have driving experience between 1 to 8.8 years, followed by 16.6-20.5 years. Another interesting thing to note is that people with more driving experiences are less likely to initiate a claim than the ones with fewer years of driving experiences.

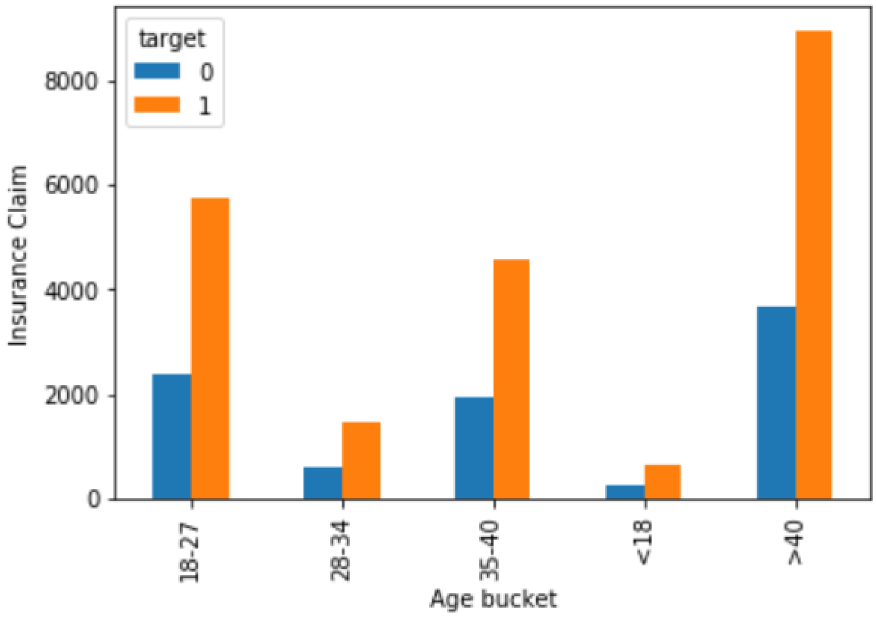


Figure 5: Clustered bar chart of age bucket by target variable

From the clustered bar chart above, it seems that there is way more individuals belong to age group of more than 40 years old, whereas individuals who belong to the age group of 28-34 and below 18 years old are in the minority. The distribution of the target (0 and 1) for different age bucket is similar.

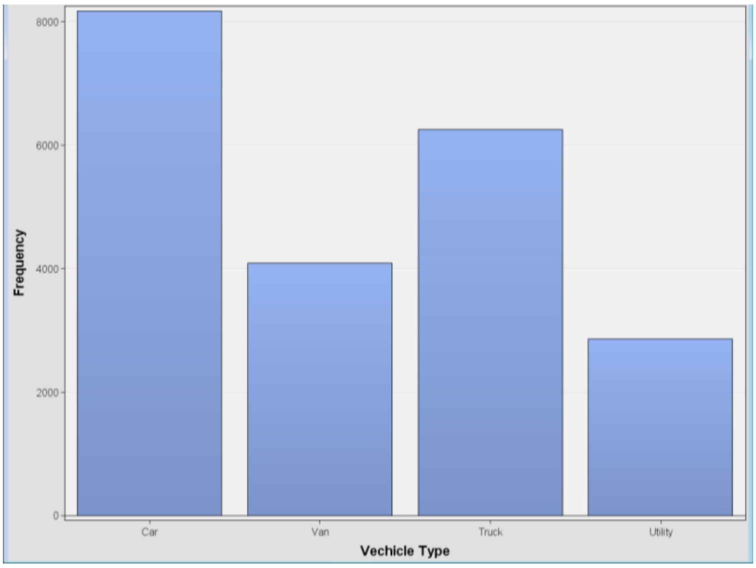


Figure 6: bar chart of vehicle type owned by policy holder who filed a claim

The bar chart above demonstrates the distribution of individuals who files a claim for different vehicle type

Modify

SAS Enterprise Miner includes a number of useful tools that can be used in modifying data source preparation for data modelling. Several pre-processing methods including data cleansing and variable transformation was performed using these tools.

Replacement

As shown in the summary table in Figure 2, the variable Miles_Driven_Annually has unusual minimum value of 0. The 0 values are often used as a substitute for missing or unknown value. With the use of "Replacement" tool, the 0 values can be replaced with a true missing value indicator. In this way, the SAS Enterprise Miner tools will be able to recognize the missing value and respond to them correctly.

The "Replacement" tool was connected to the data node in the workspace window. Before executing the replacement process, several settings need to be changed and specified, including which variable has unwanted value, what are the values that need to be replaced and what is the replacement value. In this case, only the variable "Miles_Driven_Annually" values are replaced with missing or "." when it is equals to zero.

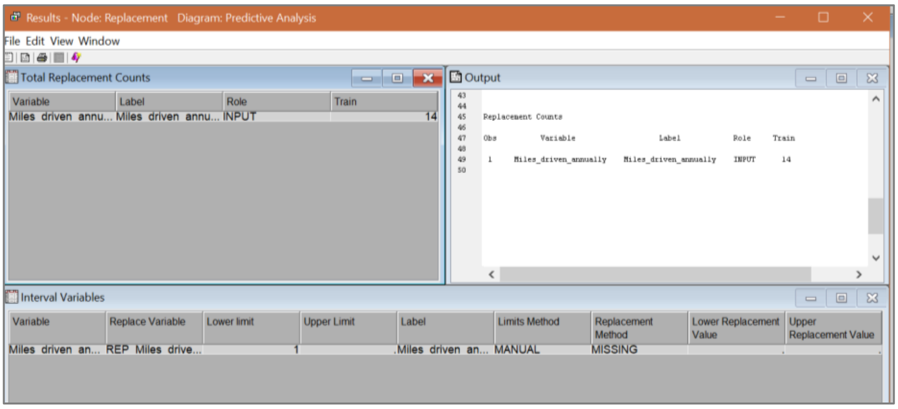


Figure 7: Replacement result window

The node was run and the result is demonstrated in Figure 7. The total replacement counts window shows that 14 observations of "Miles_Driven_Annually" variables were modified by the node. The interval window summarizes the replacement process that was conducted.

Data Partition

Partitioning dataset into train and test set is imperative in predictive modelling. It allows independent gauging of model performance and helps in improving the generalization performance of the model. In this project, 50% of the data is used as training dataset, 40% as validation dataset and 10% as test dataset. This is considered as a good practice of data partitioning as half of the data is used for generating the predictive model and nearly half of the data is used for tuning and optimizing the model. The "Data Partition" node is responsible for dividing up the dataset.

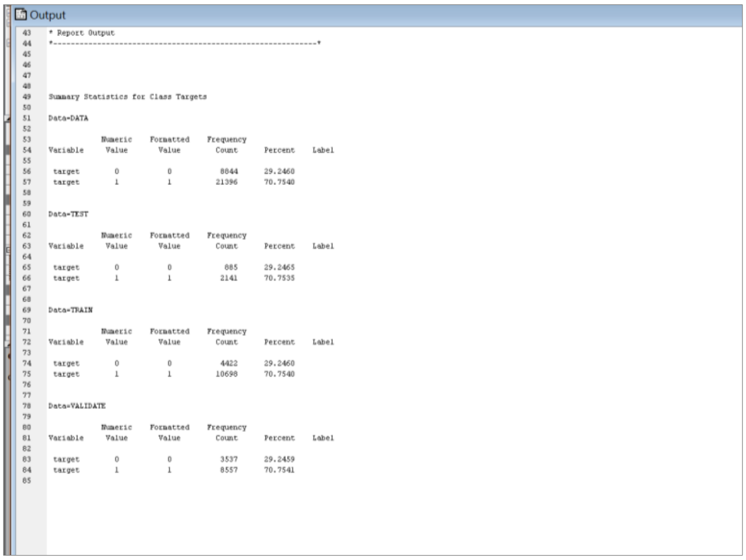


Figure 8: Data Partition Node Output

The above output window provides a frequency table that shows the distribution of the target variable in raw, training, validation and test data set.

Imputation

Imputation is very important in improving the model performance especially for logistic regression. If the missing values are not handled properly, one may end up drawing an inaccurate inference from the data. As mentioned above, there are 14 missing values found in the variable "Miles_Driven_Annually". These missing values can be replaced with synthetic values. "Impute" tool was used to impute the missing value with the mean of non-missing values in the variable "Miles_Driven_Annually".

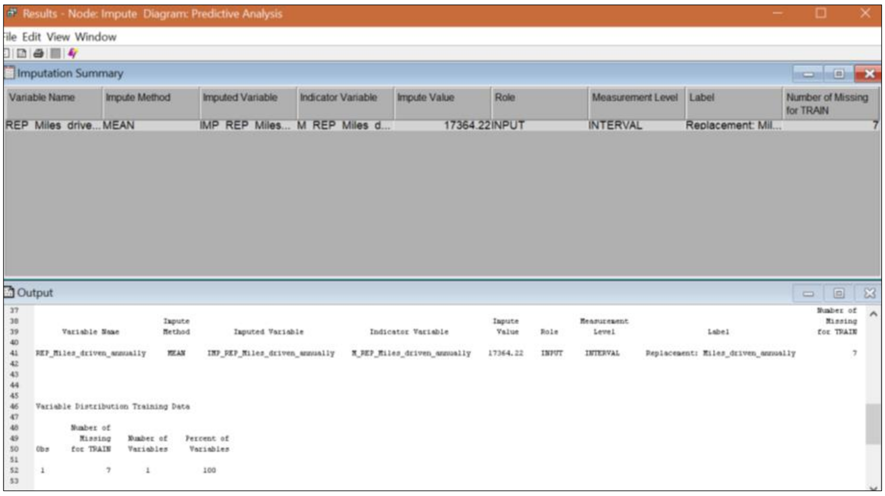


Figure 9: Impute Node Output

A new input named IMP_REP_Miles_Driven_Annually was generated with synthetic values filled in.

Transformation

Regression model are very sensitive to outliers. Inputs with highly skewed distributions can be selected over inputs that yield better overall predictions. To prevent this problem, the input distribution often requires regularization using simple transformation to optimize model performance.

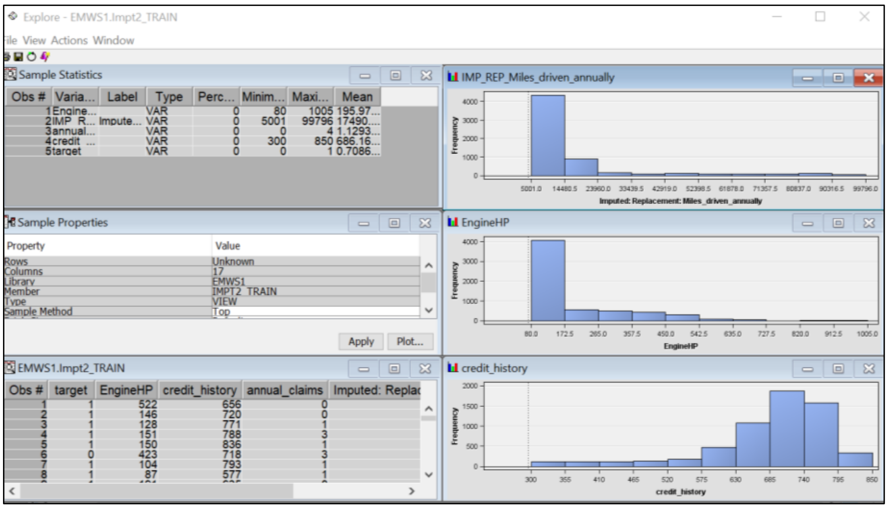
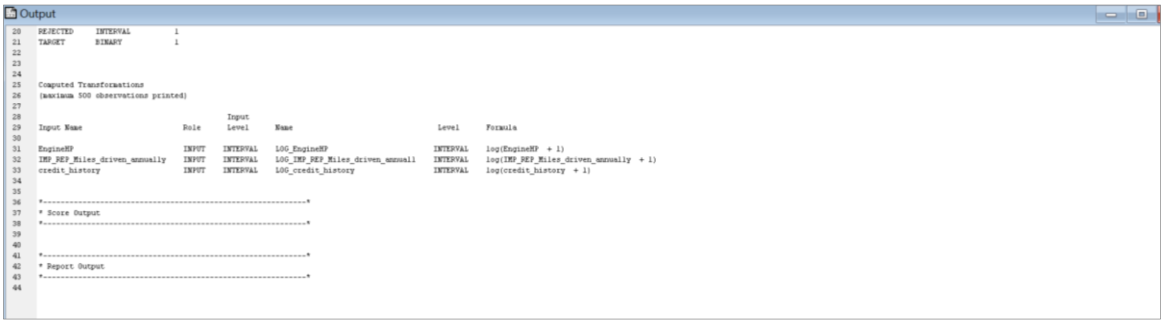


Figure 10: Exploration on variables' distribution

"Transform Variables" tool was used to explore the distribution of the variables and perform transformation. As show in Figure 10, variables IMP_REP_Miles_Driven_Annually, EngineHP and Credit_History show some degree of skewness in their distribution. Log transformation was chosen to regularize the skewed distribution.



```
Output
20  REJECTED    INTERVAL     1
21  TARGET      BINARY       1
22
23
24
25  Computed Transformations
26  (maximum 500 observations printed)
27
28                                         Input
29  Input Name                      Role   Level     Name                              Level     Formula
30
31  EngineHP                        INPUT  INTERVAL  LOG_EngineHP                      INTERVAL  log(EngineHP  + 1)
32  IMP_REP_Miles_driven_annually   INPUT  INTERVAL  LOG_IMP_REP_Miles_driven_annuall  INTERVAL  log(IMP_REP_Miles_driven_annually  + 1)
33  credit_history                  INPUT  INTERVAL  LOG_credit_history                INTERVAL  log(credit_history  + 1)
34
35
36  *------------------------------------------------------------*
37  * Score Output
38  *------------------------------------------------------------*
39
40
41  *------------------------------------------------------------*
42  * Report Output
43  *------------------------------------------------------------*
44
```

Figure 11: Transform Variable Node Output

Notice the Formula column from the output window in Figure 11, the actual transformation used was log (input+1). This action of the transform variables tool avoids problems with 0 values of the underlying inputs.

Model

Two models, logistic regression and decision tree have been trained separately to classify the insurance claim.

Logistic Regression

Logistic regression is a type of regression analysis to conduct when the dependent variable is binary. It is a widely used predictive analysis that helps to describe data and explain relationship between many different types of independent variables and one dependent variable. The output of a logistic regression model is a value between 0 and 1 denoting the probability of whether the insurance claim is filed.

A “Regression” tool was used to train the model by connecting it to the “Transform Variable” node where it feeds the regression node with pre-processed data that is ready to be trained. The “Regression” node would determine the target’s measurement level and decides the right regression type to execute, that is, if the target variable is binary, then it will performs logistic regression, if the target variable is continuous, then it will performs linear regression.

Decision tree

Unlike logistic regression, decision tree uses a tree-like model to make decisions. Specifically, it ranks input variables based on the strength of their contributions to the tree.

The decision tree model was constructed using “Decision Tree” tool. This tool allows multiway splitting of your data based on nominal, ordinal and continuous variables. The tool supports both automatic and interactive training. Automatic training was chosen to train decision tree. Instead of building the tree by selecting each split subsequentially, tree is grown automatically until the stopping rules prohibit further growth, or when a maximal tree is formed. By default, the algorithm splits the data on the variables with highest logworth.

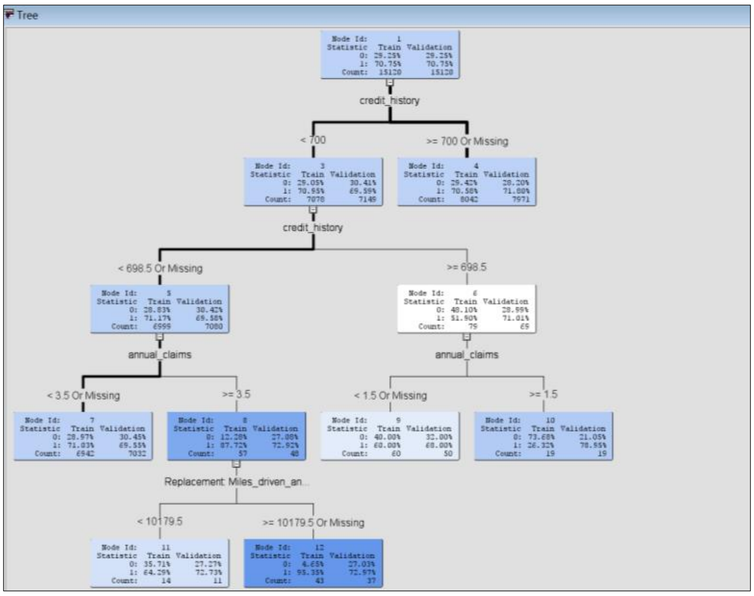


Figure 12: Decision Tree view

Figure 12 shows the maximal tree built automatically by the decision tree tool.

Assess

Performance of both logistic regression model and decision tree model were compared and evaluated using several metrics to choose the best performing classifier.

Model Comparison

SAS Enterprise Miner allows us to compare two or more models using “Model Comparison” tool and select the champion model based on the value of a single statistic.

The “Control Point” node connects the Logistic Regression and Decision Tree node and passing the data or model output to “Model Comparison” node. The “Control Point” node helps to better organize the process flow diagram.

Fit Statistics

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Misclassification Rate	Train: Akaike's Information Criterion	Train: Average Squared Error	Train: Average Error Function	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Total Degrees of Freedom	Train: Divisor for ASE	Train: Error Function	Train: Final Prediction Error	Train: Maximum Absolute Error	Train: Mean Square Error	Train: Sum of Frequencies	Train: Number of Estimate Weights	Train: Root Average Sum of Squares	Train: Root Final Prediction Error	Train: Root Mean Squared Error	Train: Schwarz's Bayesian Criterion	Train: Sum of Squared Errors
Y	Reg	Reg	Regres...	target		0.2922...	18333....	0.2059...	0.6019...	15054	66	15120	30240	18201....	0.2077...	0.8128...	0.2068...	15120	66	0.4537...	0.4557...	0.4547...	18836.7	6227.2...
	Tree	Tree	Decisio...	target		0.2933...	.	0.2064...	.	.	.	15120	30240	.	.	0.9534...	.	15120	.	0.45437	.	.	.	6243.1...

Figure 13: Fit Statistic window

The model comparison node computes several assessment measures that help us to evaluate how well the models fit the observations. From the Fit Statistics window, regression model was selected as the champion model. The champion model has the value Y in the Selected Model Column. The selection criteria used for model selection is misclassification rate in the validation data. The misclassification rate for logistic regression model is 0.2922, which is slightly lower than the misclassification rate for decision tree of value 0.2933. In other words, the logistic regression model has made less incorrect classifications of observations than decision tree model. Model accuracy can be interpreted using misclassification rate where accuracy = 1-misclassification rate. The accuracy of logistic regression model is roughly 79% whereas the accuracy of decision tree model is about 71%. The goal is to minimize the misclassification rate and maximize the accuracy of the model.

Event Classification Table
Model Selection based on Valid: Misclassification Rate (_VMISC_)

Model Node	Model Description	Data Role	Target	Target Label	False Negative	True Negative	False Positive	True Positive
Reg	Regression	TRAIN	target		0	2	4420	10698
Reg	Regression	VALIDATE	target		0	2	3535	8557
Tree	Decision Tree	TRAIN	target		5	14	4408	10693
Tree	Decision Tree	VALIDATE	target		14	3	3534	8543

Figure 14: Confusion matrix

Other assessment measures that we can look at are the derivations from a confusion matrix including sensitivity, specificity and precision. From the table in Figure 21, we can construct a confusion matrix that contains information about actual and predicted classifications in a 2 by 2 contingency table.

<u>Logistic regression</u>

Table 2: Confusion matrix

	Predicted: No	**Predicted: Yes**
Actual: No	TN = 2	FP = 3535
Actual: Yes	FN = 0	TP = 8557

Sensitivity or true positive rate = TP/TP+FN = 8557/8557 = 1

Specificity or true negative rate = TN/TN+FP = 2/2+3535 = 0.0005

Precision = TP/TP+FP = 8557/8557+3535 = 0.7077

Decision tree

Table 3: Confusion matrix

	Predicted: No	**Predicted: Yes**
Actual: No	TN = 3	FP = 3534
Actual: Yes	FN = 14	TP = 8543

Sensitivity or true positive rate = TP/TP+FN = 8543/8543+14 = 0.9984

Specificity or true negative rate = TN/TN+FP = 3/3+3534 = 0.0008

Precision = TP/TP+FP = 8543/8543+3534 = 0.7074

Sensitivity or true positive rate is the number of correct positive predictions divided by the total number of positives. A sensitivity value closer to 1 is better than the value closer to 0. Logistic regression classifier has sensitivity of 1, which is higher than decision tree classifier with value 0.9984. Specificity or true negative rate is calculated as the number of correct negative predictions divided by the total number of negatives. A specificity value closer to 1 is better than the value closer to 0. Logistic regression classifier has sensitivity of 0.0005, which is lower than decision tree classifier with value 0.0008. Precision is the number of correct positive predictions divided by the total number of positive predictions. A precision value closer to 1 is better than the value closer to 0. Logistic regression classifier has precision value of 0.7077, higher than the decision tree classifier with precision value of 0.7074 which indicates a higher precision of classifications.