

The **commsdata** data set contains information about customers of a telecommunications company and their use of the company's services. The input variables include demographic variables, variables that describe product usage and type, billing data, and customer service and call center information. The main goal is to use this set of input variables and train supervised learning models to predict the churn event. To sum;

- ID variable is Customer ID; Target variables are Churn and Upsell_xsell
- There are 18 Categorical-valued inputs and 117 Interval-valued inputs

The screenshot shows the 'Model Studio - Build Models' interface. On the left, a table lists variables with columns: Variable Name, Label, Type, and Role. The 'churn' variable is selected with a checkmark in the first column. On the right, a configuration panel for the 'churn' variable is shown, with an orange border. The panel includes dropdowns for Role (set to Input), Level (set to Binary), Order (set to Default), Transform (set to Default), and Impute (set to Default).

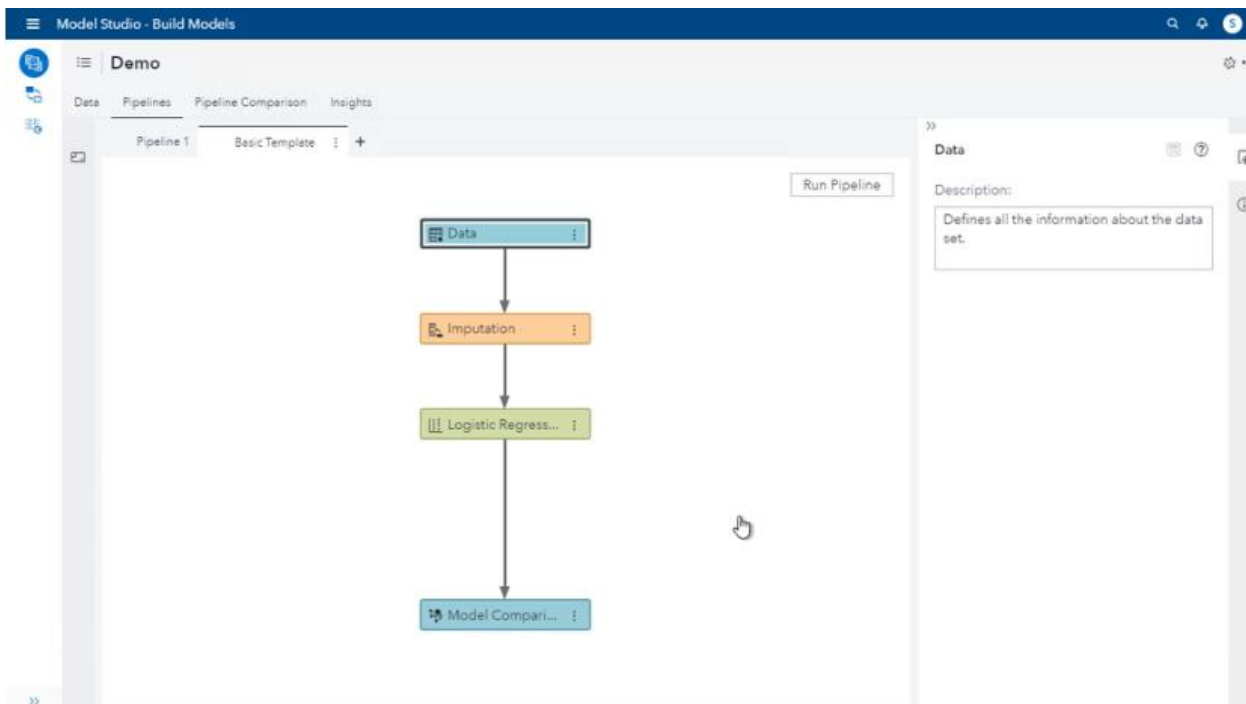
Variable Name	Label	Type	Role
<input type="checkbox"/> calls_total	Total Calls Curr	Numeric	Input
<input type="checkbox"/> calls_TS_acct	Number Calls Tech Support	Numeric	Input
<input checked="" type="checkbox"/> churn	Churn Flag	Numeric	Input
<input type="checkbox"/> city	Account City	Character	ID
<input type="checkbox"/> city_lat	Account City Latitude	Numeric	Input
<input type="checkbox"/> city_long	Account City Longitude	Numeric	Input
<input type="checkbox"/> count_of_suspensions_6m	Times Suspended Last 6M	Numeric	Input
<input type="checkbox"/> credit_class	Credit Class	Character	Input
<input type="checkbox"/> cs_afr_amer	Census Area African-American	Numeric	Input
<input type="checkbox"/> cs_caucasian	Census Area Caucasian	Numeric	Input
<input type="checkbox"/> cs_hispanic	Census Area Hispanic	Numeric	Input
<input type="checkbox"/> cs_med_home_value	Census Area Median Home Value Index	Numeric	Input

Modifying the Data Partition

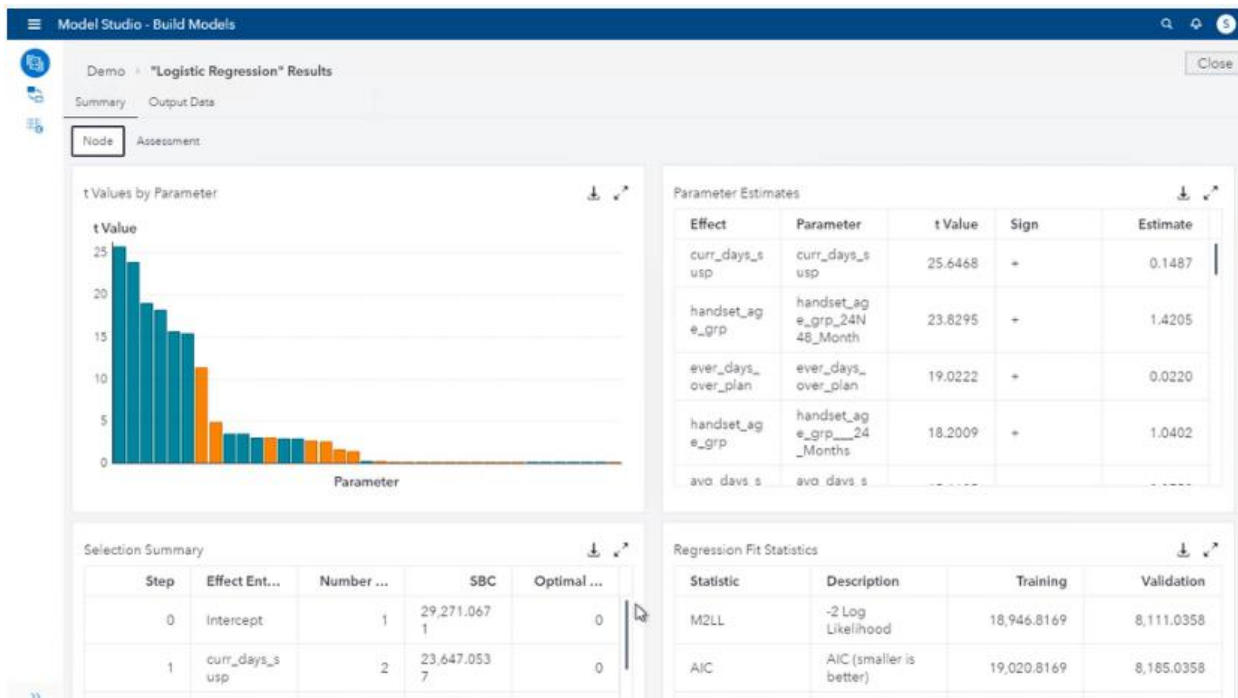
We want to reject 11 variables in the commsdata data set. Rejected variables will not be used in our models. Default partitioning method is Stratify. We're going to specify 70% of the data for training and 30% for validation, with no test.

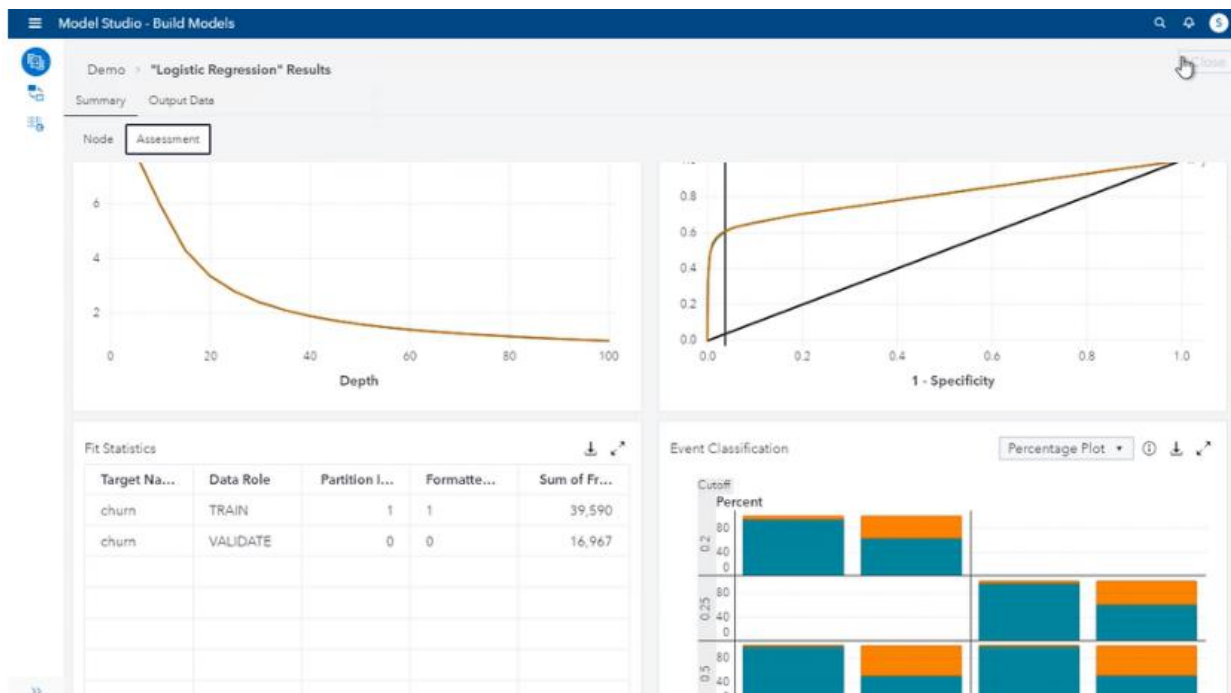
The screenshot shows the 'Project Settings' dialog box with the 'Partition Data' tab selected. The 'Create partition variable' checkbox is checked. The 'Method' is set to 'Stratify'. The 'Trainings' percentage is 70.00% (with a value of 70 in the input field). The 'Validation' percentage is 30.00% (with a value of 30 in the input field). The 'Test' percentage is 0.00% (with a value of 0 in the input field). A note states: 'Note: These settings are active only when a partition variable is not set with the data. Using a data source with a pre-defined partition variable or manually selecting a partition variable will override these settings.'

We start with 4 basic nodes first: the Data node; one node for data preparation, which is Imputation; one model, Logistic Regression; and then Model Comparison. Even when there is only one model in a pipeline, a Model Comparison node is included by default, with a class target is KS -- Kolmogorov-Smirnov.



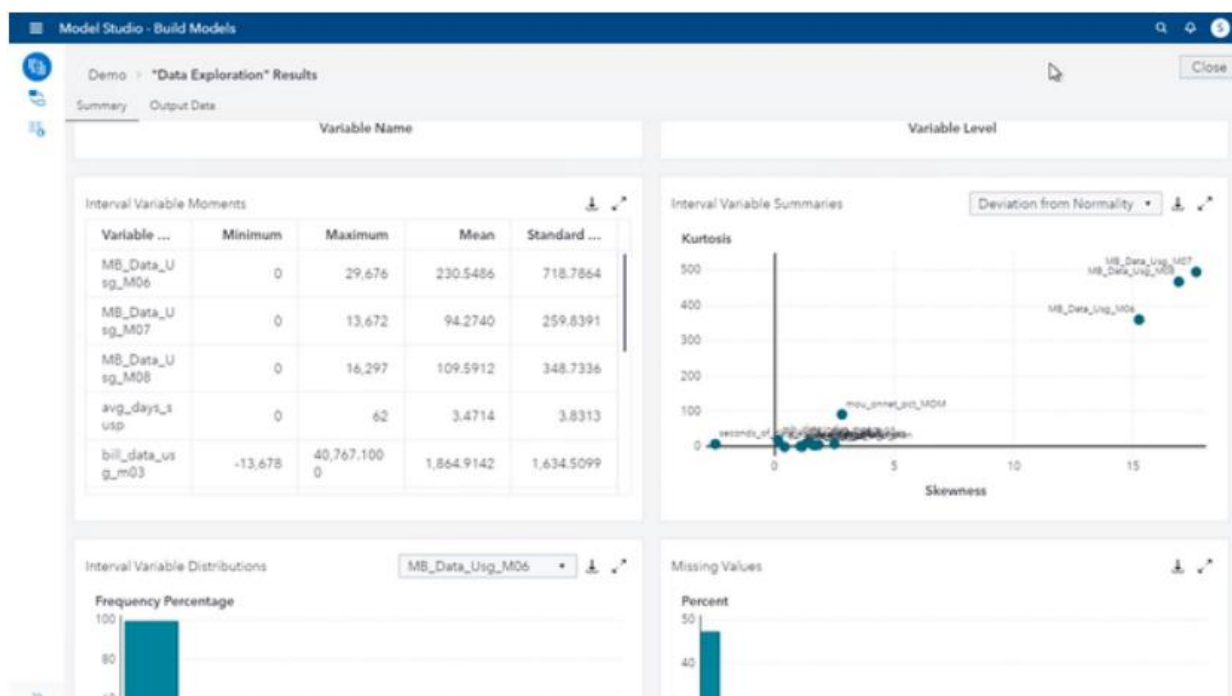
In the results of the Logistic Regression node, in the upper left corner, notice that there are two tabs: Node and Assessment. Some of the windows on the Node tab are the t Values by Parameter table, the Parameter Estimates table, and the Selection Summary table. Some of the windows available on the Assessment tab are the Lift reports, the ROC reports, and the Fit Statistics table.





Exploring the Data

At this point we add Data Exploration node and connects it to the Data node. By default, a maximum of 50 of the most important variables will be selected, data like cutoff for flagging variables with a high percentage of missing values, high-cardinality class variables, class variables with dominant levels, class variables with rare modes, skewed interval variables, peaky interval variables, and interval variables with thick tails can be eliminated. The scatter plot of skewness against kurtosis for all the interval input variables. Notice that we have a few interval input variables in the upper right corner that are suspicious based on high kurtosis and high skewness values.



Replacement node can be used to replace outliers and unknown class levels with specified values. This is where you invoke the metadata property of the lower limit that we set before. In the Properties pane of the Replacement node, set the Default limits method to Metadata limits.

Interval Variables

Name	Variable ...	Replace ...	Limits Me...	Lower Limit
BILL_DATA_USG_M03	3M Avg Billed Data Usage	REP_BILL_D ATA_USG_M03	METALIMIT	0
BILL_DATA_USG_M06	6M Avg Billed Data Usage	REP_BILL_D ATA_USG_M06	METALIMIT	0
CALLS_IN_OFFPK	Calls Incoming Off-Peak	REP_CALLS_IN_OFFPK	METALIMIT	0
CALLS_IN_PK	Calls Incoming Peak	REP_CALLS_IN_PK	METALIMIT	0

Replacement Counts

Name	Variable ...	Train	Role	Variable ...
BILL_DATA_USG_M03	3M Avg Billed Data Usage	2,668	INPUT	INTERVAL
BILL_DATA_USG_M06	6M Avg Billed Data Usage	1,856	INPUT	INTERVAL
CALLS_IN_OFFPK	Calls Incoming Off-Peak	3,424	INPUT	INTERVAL
CALLS_IN_PK	Calls Incoming Peak	4,050	INPUT	INTERVAL

Node Score Code

```

1
2
3
4 /***** Interval Variables *****/
5 /*          Interval Variables          */
6 /*****
7
8 /**** BILL_DATA_USG_M03 *****/
9 Length 'REP_BILL_DATA_USG_M03' n 0;
10 Label 'REP_BILL_DATA_USG_M03' n 0;

```

Properties

Property Name	Property Value
unknownLevel	IGNORE
calcMethod	METALIMIT
altMethod	NONE
stddevCutoff	3

Adding Text Mining Features

Here we'll be using Text Mining node. There are currently five text variables in the commsdata data set, but we'll only make use of one of them, which is called Verbatims. Two of the other text variables are already rejected. We'll need to manually reject the other two, which we'll do using the Data tab. Several windows are available, including tables of Kept Terms and Dropped Terms. These tables include terms used and ignored respectively during the text analysis. In the Kept Terms table, notice that several terms include a plus sign. The plus sign next to a word indicates stemming. For example, + service represents service, services, serviced, and so on.

Kept Terms

Term	Role	Attribute	Freq	Number ...
very	ADV	Alpha	12,160	10,481
+ service	N	Alpha	9,188	7,982
not	ADV	Alpha	7,830	6,139
mtt	PN	Alpha	7,023	6,034
+ phone	N	Alpha	6,280	4,963
helpful	A	Alpha	5,023	4,946
+ customer	N	Alpha	5,577	4,899
+ call	V	Alpha	4,097	3,500

Dropped Terms

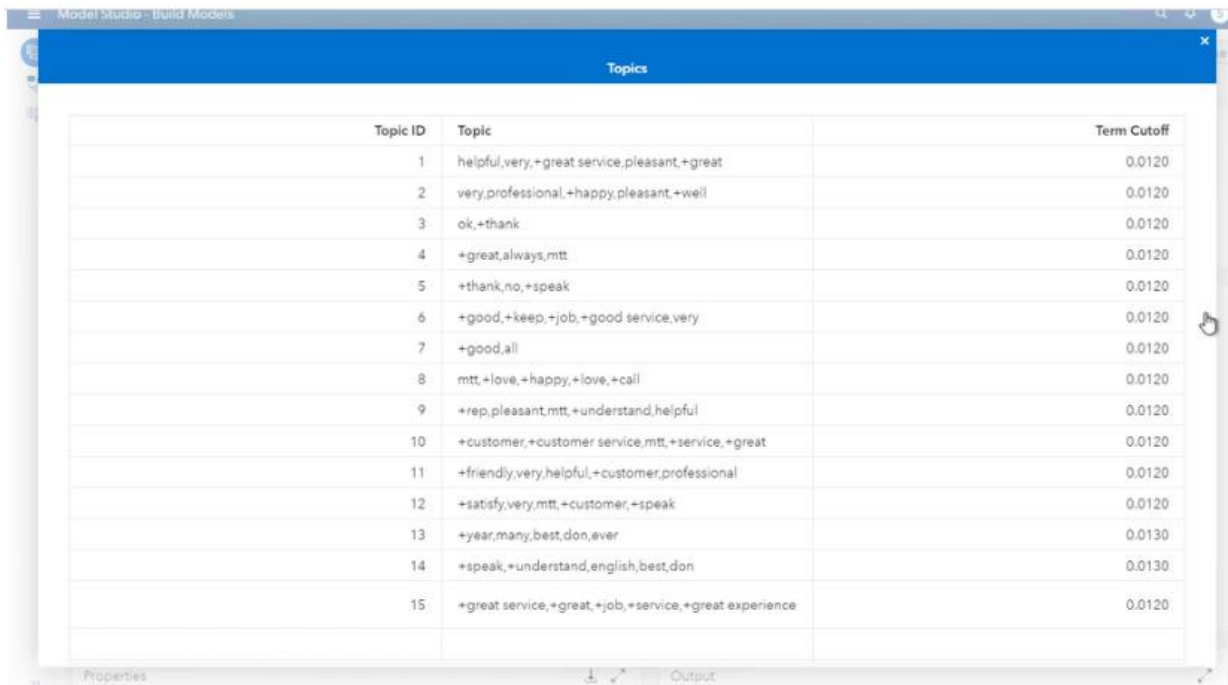
Term	Role	Attribute	Freq	Number ...
+ be	V	Alpha	47,246	28,040
+ have	V	Alpha	14,782	10,197
+ do	V	Alpha	5,438	4,571
+ get	V	Alpha	5,300	4,487
+ will	V	Alpha	5,095	4,260
t	N	Alpha	4,907	4,056
+ can	V	Alpha	3,698	3,252
i	N	Alpha	2,992	1,975

Terms: Role by Frequency

Topics

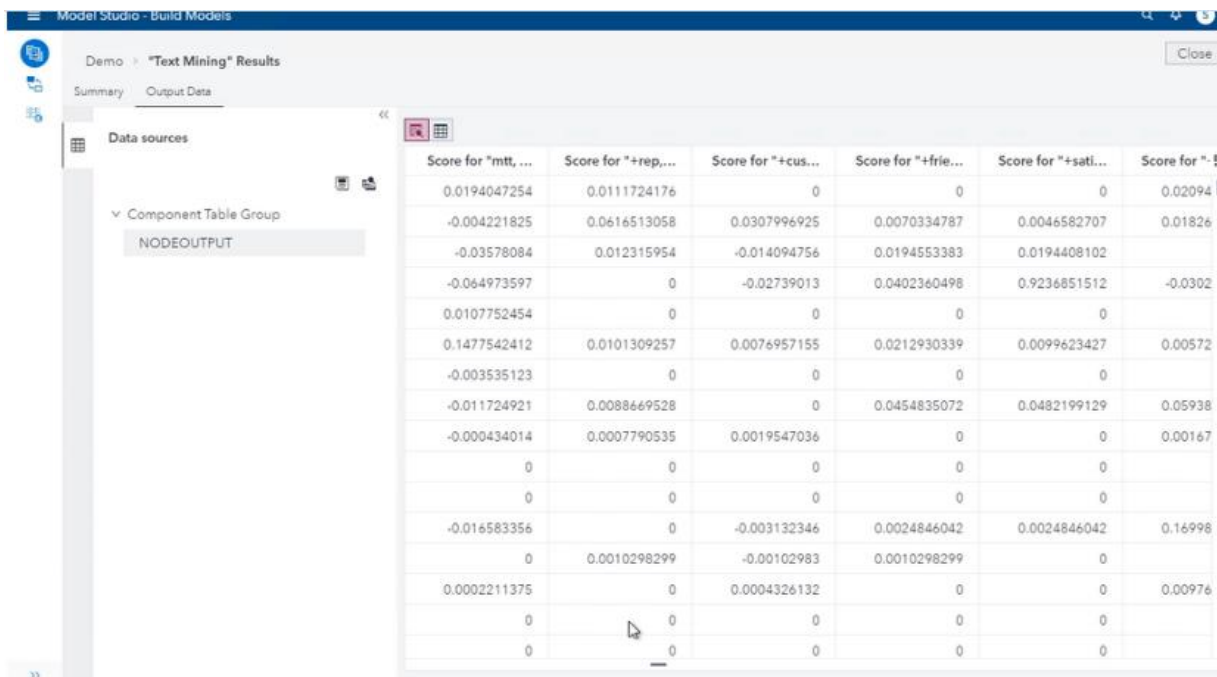
Topic ID	Topic	Term Cutoff
1	helpful,very,+great service,pleasant,+great	0.0120
2	very,professional,+happy,pleasant,+well	0.0120
3	ok,+thank	0.0120

Scroll down and expand the Topics table. These 15 topics were created based on groups of terms that occur together in several documents. Each term-document pair is assigned a score for every topic.



Topic ID	Topic	Term Cutoff
1	helpful,very,+great service,pleasant,+great	0.0120
2	very,professional,+happy,pleasant,+well	0.0120
3	ok,+thank	0.0120
4	+great,always,mtt	0.0120
5	+thank,no,+speak	0.0120
6	+good,+keep,+job,+good service,very	0.0120
7	+good,all	0.0120
8	mtt,+love,+happy,+love,+call	0.0120
9	+rep,pleasant,mtt,+understand,helpful	0.0120
10	+customer,+customer service,mtt,+service,+great	0.0120
11	+friendly,very,helpful,+customer,professional	0.0120
12	+satisfy,very,mtt,+customer,+speak	0.0120
13	+year,many,best,don,ever	0.0130
14	+speak,+understand,english,best,don	0.0130
15	+great service,+great,+job,+service,+great experience	0.0120

Because 15 topics were discovered, 15 new columns of inputs are created. The output columns contain SVD, or singular value decomposition, scores that can be used as inputs for the downstream nodes. I'll scroll to the right to see the column headings that begin with Score for. These columns are for new variables based on the topics created by the Text Mining node. For each topic, the SVD coefficients (or scores) are shown for each observation in the data set. Notice that the coefficients have an interval measurement level. The Text Mining node converts textual data into numeric variables, specifically interval variables. These columns will be passed along to subsequent nodes.



Score for "mtt, ...	Score for "+rep,...	Score for "+cus...	Score for "+frie...	Score for "+sati...	Score for "+...
0.0194047254	0.0111724176	0	0	0	0.02094
-0.004221825	0.0616513058	0.0307996925	0.0070334787	0.0046582707	0.01826
-0.03578084	0.012315954	-0.014094756	0.0194553383	0.0194408102	
-0.064973597	0	-0.02739013	0.0402360498	0.9236851512	-0.0302
0.0107752454	0	0	0	0	
0.1477542412	0.0101309257	0.0076957155	0.0212930339	0.0099623427	0.00572
-0.003535123	0	0	0	0	
-0.011724921	0.0088669528	0	0.0454835072	0.0482199129	0.05938
-0.000434014	0.0007790535	0.0019547036	0	0	0.00167
0	0	0	0	0	
0	0	0	0	0	
-0.016583356	0	-0.003132346	0.0024846042	0.0024846042	0.16998
0	0.0010298299	-0.00102983	0.0010298299	0	
0.0002211375	0	0.0004326132	0	0	0.00976
0	0	0	0	0	
0	0	0	0	0	

In the Incoming Variables table are the 15 new columns representing the dimensions of the SVD calculations based on the 15 topics discovered by the Text Mining node. These 15 columns, COL1 through COL15, serve as new interval inputs for subsequent models.

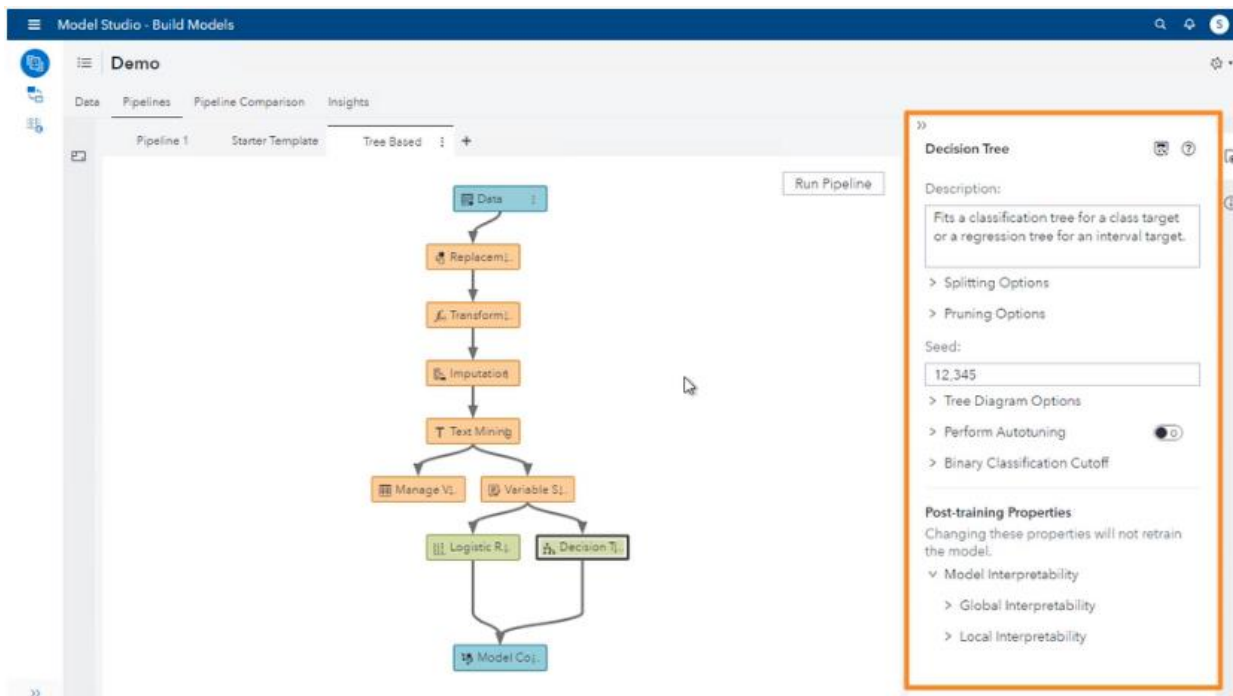
Adding these text features does not necessarily guarantee that the performance of the model will improve. We'll see if any of the new attributes made it into the final model by looking at the results of the Logistic Regression node, and if it does not actually, we see KS is the same as before, 0.57, there is no significant effect.



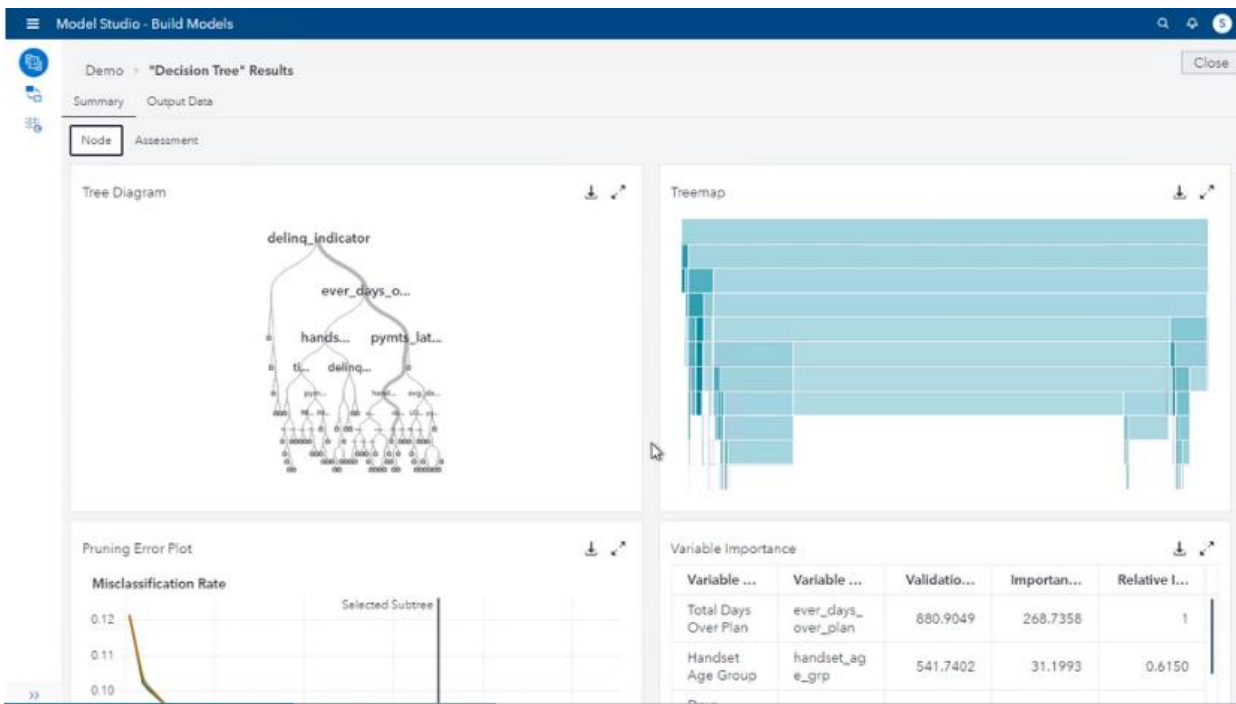
The screenshot shows a table titled 'Incoming Variables' with the following columns: Obs, Variable Name, Role, Measurement Level, Order, Label, Count, Number of Missing Values, Percentage Missing, Minimum, Maximum, and Measure. The table displays data for five variables: COL1, COL10, COL11, COL12, and COL13. Each variable is an INPUT with an INTERVAL measurement level, based on scores for specific sentiment topics. All variables have a count of 254, 0 missing values, and 0.0000 percentage missing.

Obs	Variable Name	Role	Measurement Level	Order	Label	Count	Number of Missing Values	Percentage Missing	Minimum	Maximum	Measure
1	COL1	INPUT	INTERVAL		Score for "helpful, very, +great service, pleasant, +great"	254	0	0.0000	-0.102815192	0.9711783205	0.04221535
2	COL10	INPUT	INTERVAL		Score for "+customer, +customer service, mtt, +service, +great"	254	0	0.0000	-0.043227404	0.992500277	0.0184347
3	COL11	INPUT	INTERVAL		Score for "+friendly, very, helpful, +customer, professional"	254	0	0.0000	-0.017	0.987	0.01986061
4	COL12	INPUT	INTERVAL		Score for "+satisfy, very, mtt, +customer, +speak"	254	0	0.0000	-0.034009412	0.952	0.02485545
5	COL13	INPUT	INTERVAL		Score for	254	0	0.0000	-0.037788973	0.956	0.01329111

Building a Decision Tree Model Using the Default Settings



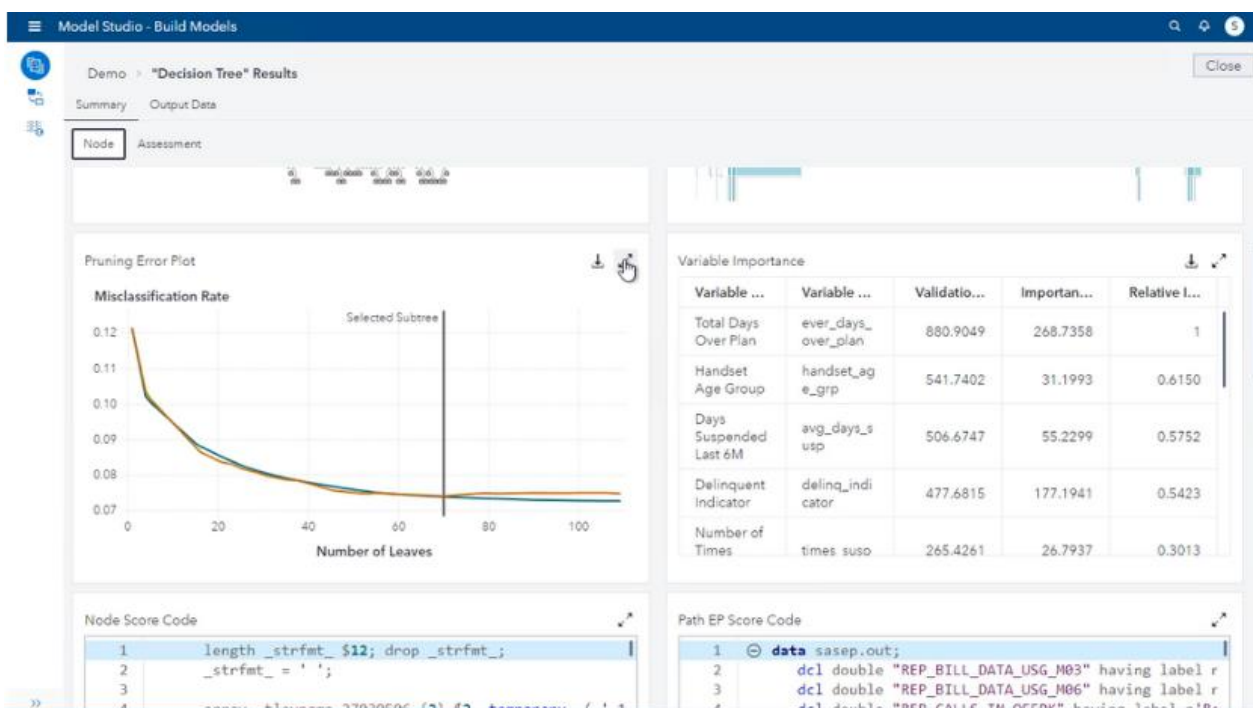
The tree diagram presents the final tree structure for this particular model, such as the depth of the tree and all end leaves.



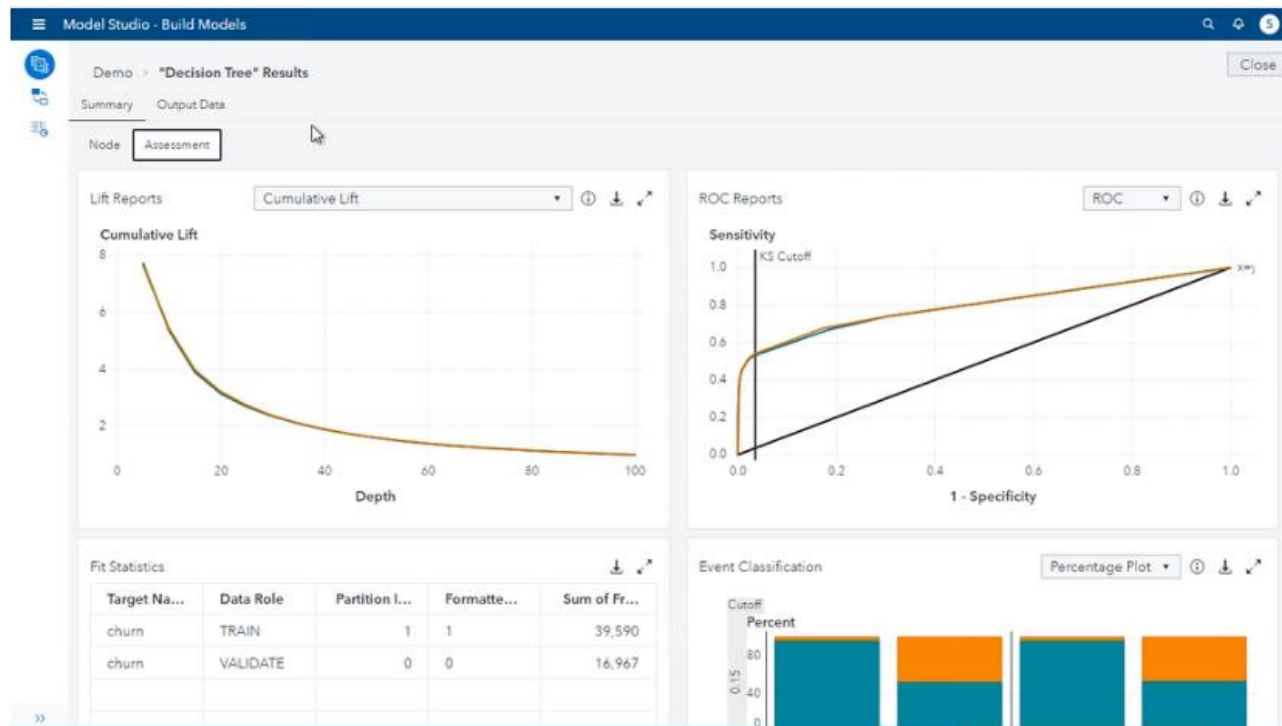
Farther down, I'm going to expand the Pruning Error plot. This plot is based on the misclassification rate because we have a binary target. The plot shows the change in misclassification rate on training and validation data as the tree grows or as more leaves are added to the tree. The green line represents the training data and the orange line represents the validation data.

Next, I'll expand the Variable Importance table. This table shows the final variables selected by the decision tree and their relative importance. The most important input variable has the relative importance 1. And all others are measured based on the most important input. In this case, notice that the most important variable selected by the decision tree was ever_days_over_plan. I'll close the Variable Importance table.

Farther down in the results are several score code windows, one for each type of score code. Supervised Learning nodes can generate up to four types of score code: node score code, path EP score code, DS2 package code, and training code.



A Cumulative Lift plot appears by default. We can interpret the plot as a comparison of the performance of the particular model at certain depths of the data ranked by the posterior probability of the event compared to a random model. Ideally, you want to see a lift greater than 1, which means that your model is outperforming a random model. Because our data set has a binary target, the ROC Reports plot is also available. ROC chart plots sensitivity against 1 minus specificity for varying cutoff values. Sensitivity is defined as the true positive rate. And 1 minus specificity is defined as the false positive rate. A useful fit statistic to consider is average squared error. Notice that ASE is 0.0648.



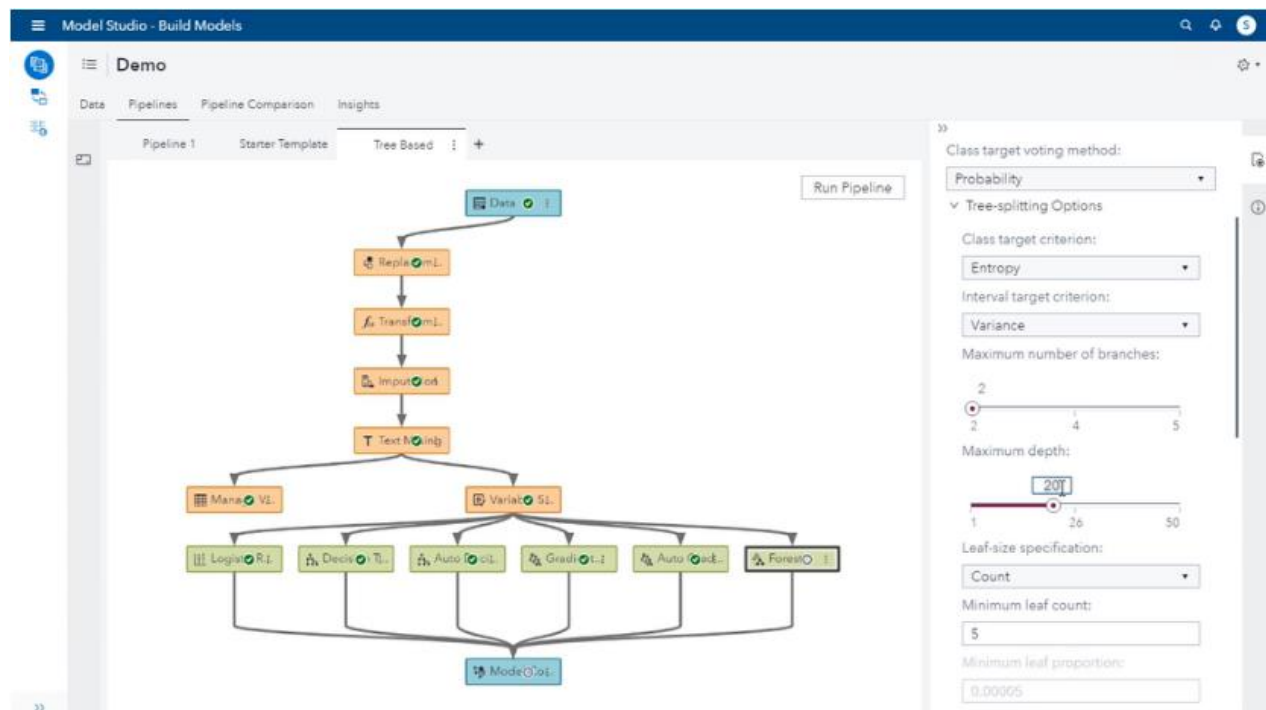
We see that the decision tree is currently selected as the champion model. Recall that, by default, the champion is selected based on the KS (0.5548) statistic value, where larger is better. Even if we compare the average squared error values for the two models, we get a smaller average squared error for the decision tree than the logistic regression model, which indicates that the decision tree performs better on that statistic.

Model Studio - Build Models

Model Comparison

Champi...	Name	Algorith...	KS (You...	Misclas...	Misclas...	Root Av...	Averag...	Sum of ...	Multi-Cl...	Gini Co...	Area
Decision Tree	Decision Tree	Decision Tree	0.5548	0.0683	0.0683	0.2465	0.0608	16,967	0.2391	0.6077	0.
Logistic Regression	Logistic Regression	Logistic Regression	0.5417	0.0821	0.0821	0.2627	0.0690	16,967	0.2604	0.6178	0.

Gradient Boosting, Auto Gradient B. and Forest Model nodes were also respectively applied with default settings



Fit Statistics table, we see that, on the validation data set, average squared error is 0.0572, which decreased a little bit comparing to first versions with Logistic Regression. Remember that smaller is better for the average squared error, so this model does outperform the prior models by just a bit.

[illegible]

Per model comparison table, the forest is the champion model from the pipeline based on KS, with 0.5867.

Demo > "Model Comparison" Results Close

Node Assessment

Model Comparison

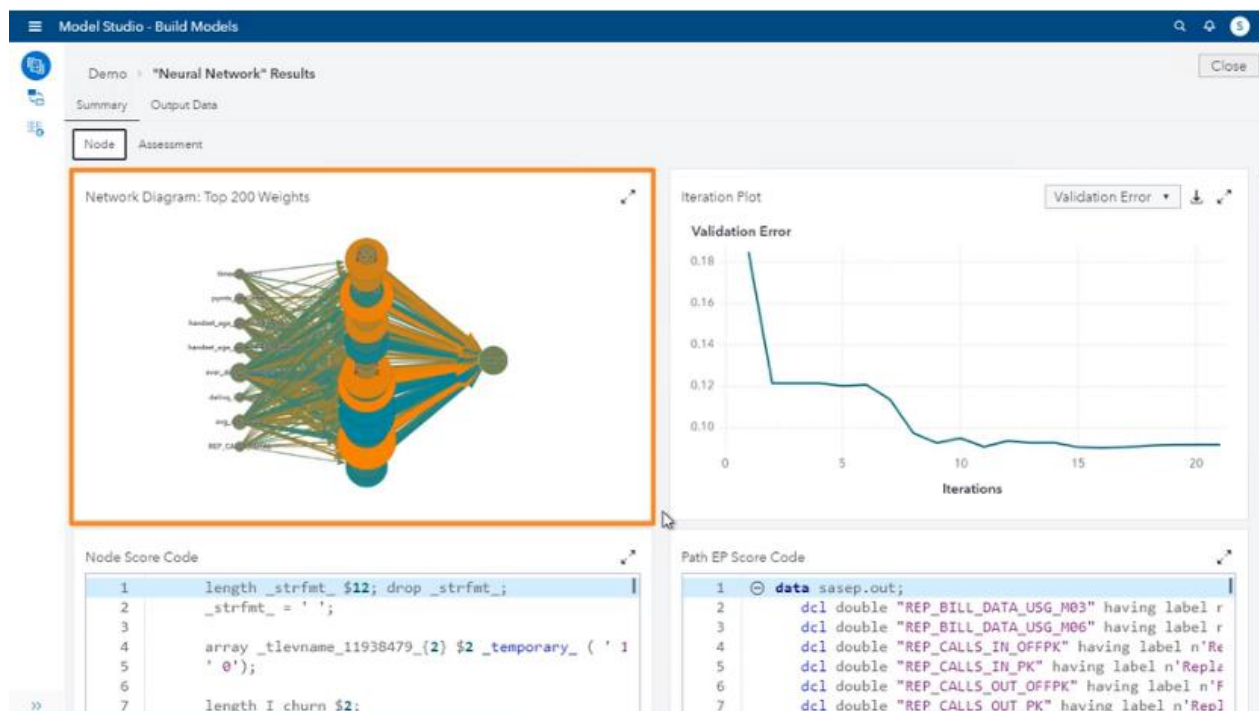
Champion	Name	Algorithm Name	KS (Youden)	Misclassification Rate
	Forest	Forest	0.5867	0.0650
	Auto Gradient Boosting	Gradient Boosting	0.5854	0.0647
	Gradient Boosting	Gradient Boosting	0.5825	0.0628
	Auto Decision Tree	Decision Tree	0.5724	0.0665
	Decision Tree	Decision Tree	0.5548	0.0683
	Logistic Regression	Logistic Regression	0.5417	0.0821

Properties

Property Name	Property Value
selectionCriteriaClass	Kolmogorov-Smirnov statistic (KS)
selectionCriteriaInterval	Average squared error
selectionTable	Validate
selectionDepth	10

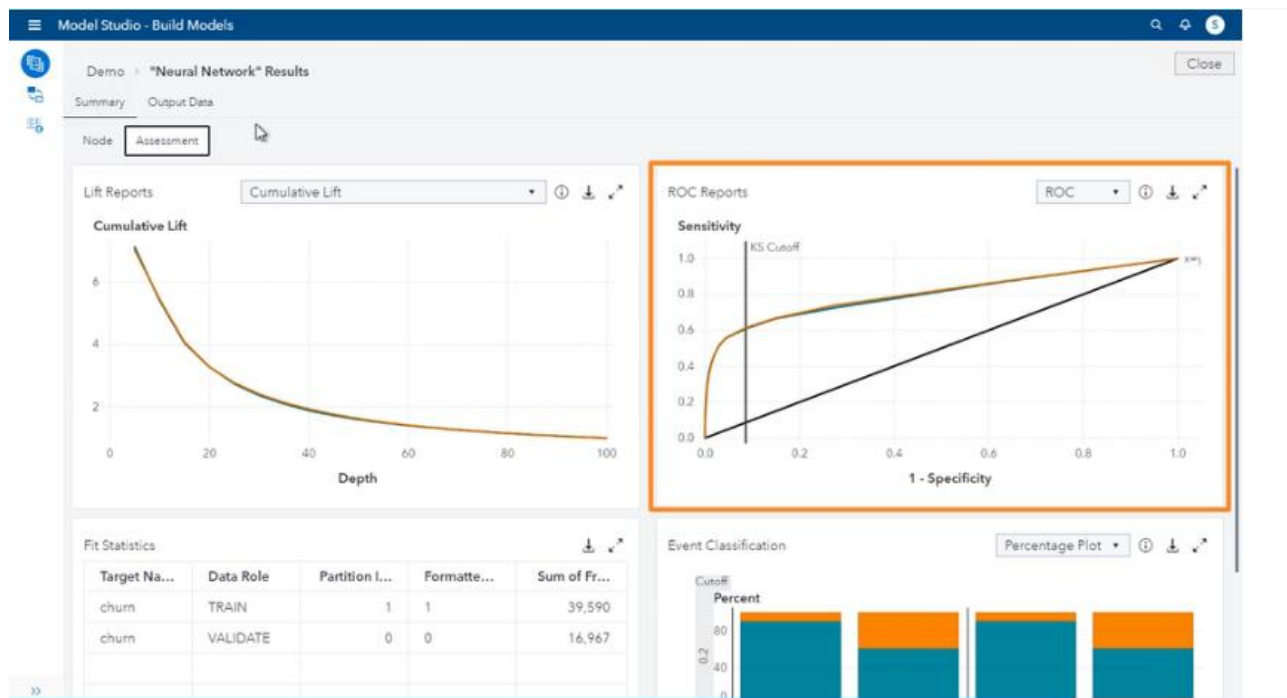
Building a Neural Network Using the Default Settings

The first plot in the upper left corner is the Network Diagram. This plot represents the final neural network structure for the model, including the hidden layer and the hidden units. The Iteration Plot shows the model's performance based on the validation error throughout the training process, when new iterations are added.



We can further assess the performance of the model from the Assessment tab. In the upper left corner, as usual, we see the cumulative lift plot showing the model's performance ordered by the percentage of the population. In the upper right corner, we see the ROC curve. This curve shows the model's performance by considering the true positive rate and the false positive rate.

Per the Fit Statistics table, we notice that the average squared error on validation data is 0.0734, which is higher to prior models. (KS figures are very similar too, so there is no need to make efforts to modify the average square error.)

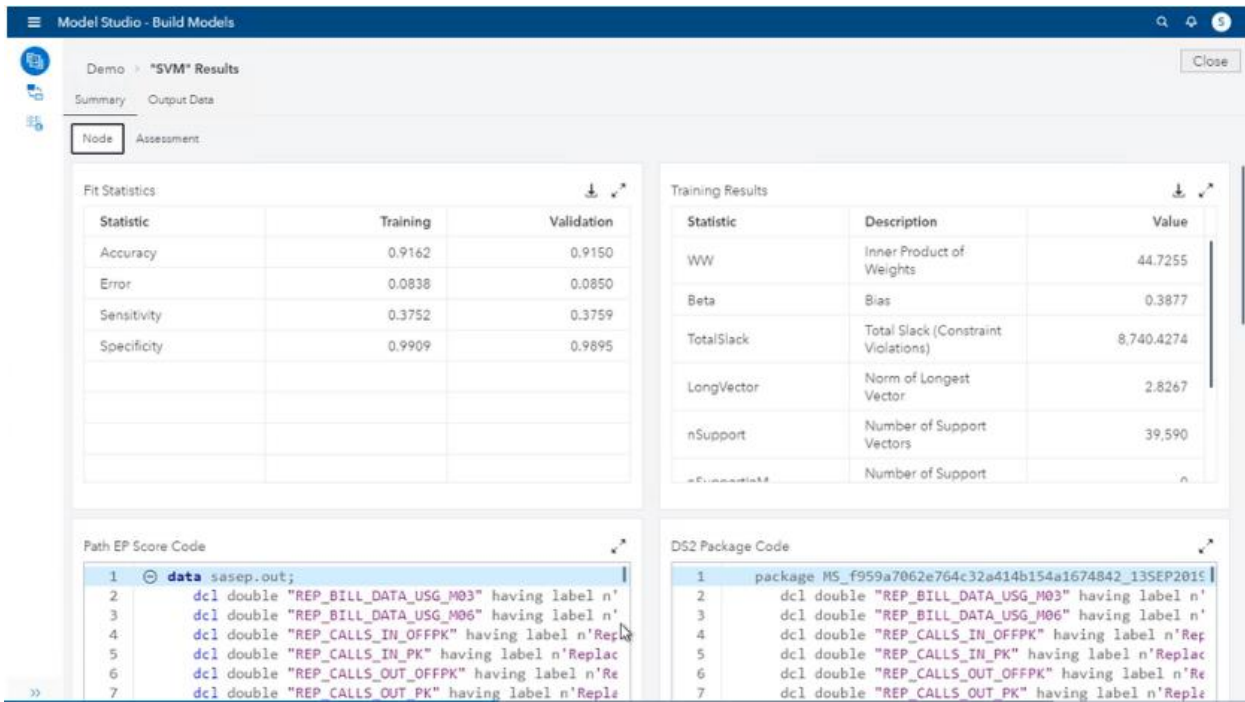


Building a Support Vector Machine Using the Default Settings

Per Node tab, the upper left corner shows the Fit Statistics table. This table shows the support vector machine's performance, based on several assessment measures.

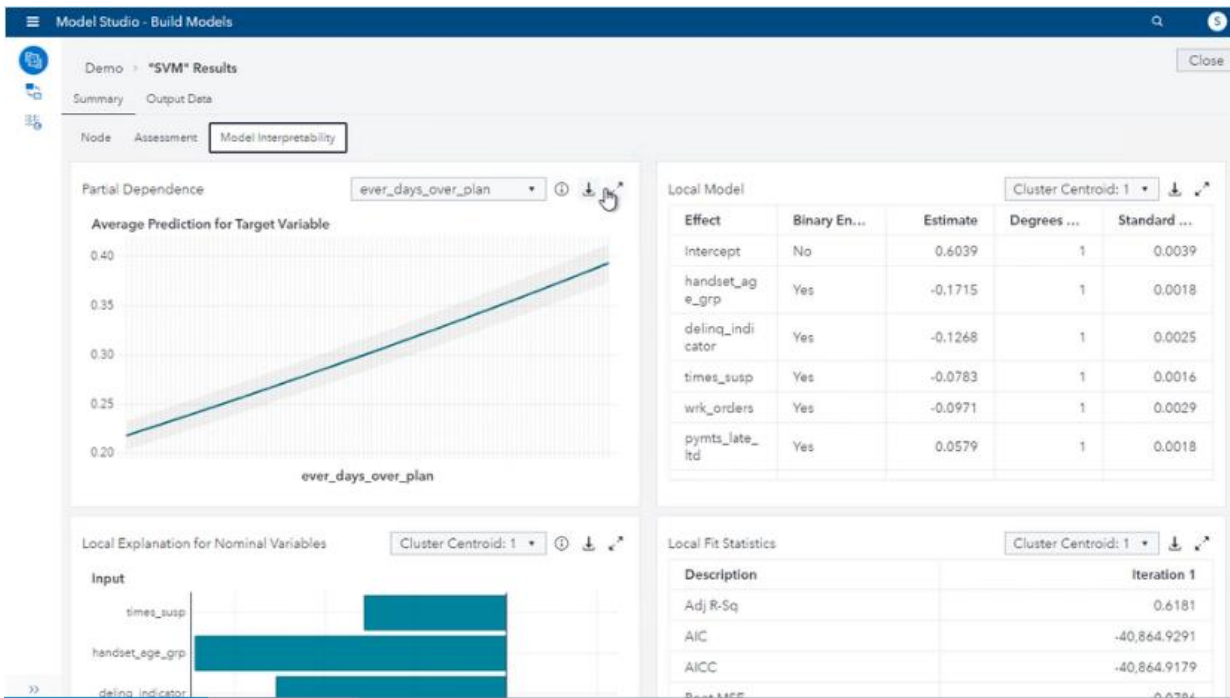
In the upper right corner, I'll maximize the Training Results window. This table shows the parameters of the final support vector machine model, such as the number of support vectors and the bias, which is the offset that defines the support vector machine. I'll close the Training Results window.

Per Assessment tab we see model performance results. As usual, we see the lift reports, the ROC reports, the Event Classification plot, and the Fit Statistics table. Notice that the average squared error on validation data is 0.1141, higher than prior models and KS is a bit lower comparing to others, so there is no need to modify the ASE at this moment.

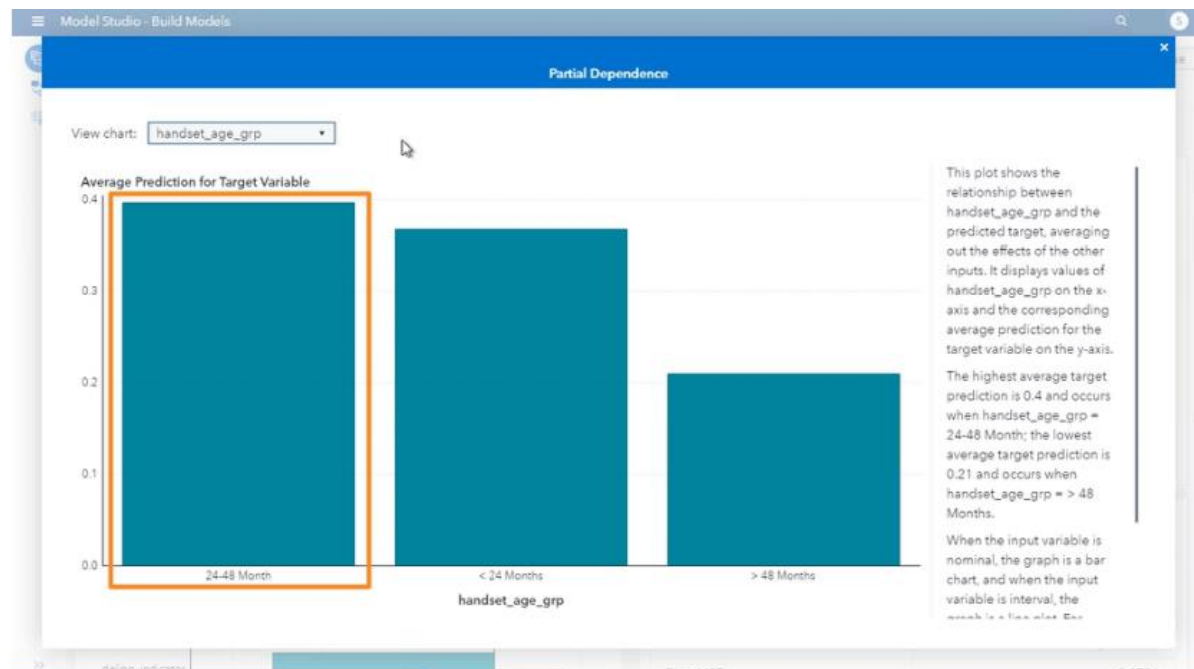


Adding Model Interpretability

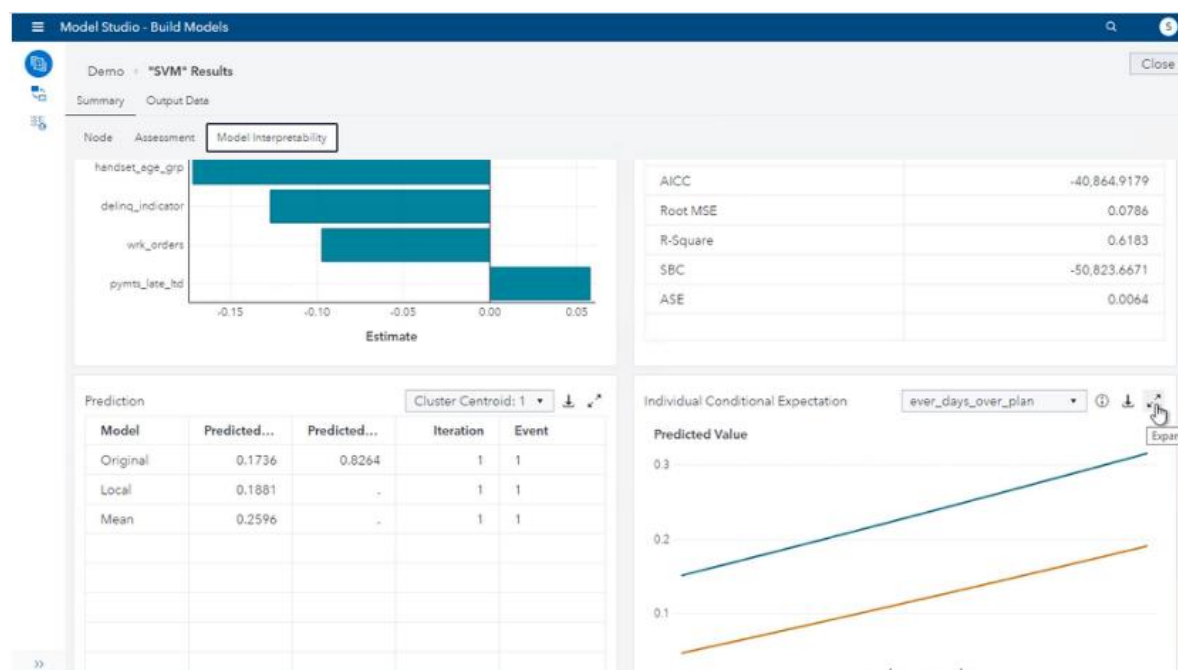
This plot can show whether the relationship between the target and the feature is linear, monotonic, or more complex. Here, there is a positive linear relationship. Basically, there are two clusters:



This plot indicates that the highest probability of churn is associated with the middle age group (that is, the middle level of the variable): handsets between 24 and 48 months old. The newest handsets (those less than 24 months old) have the next highest probability of churn. And the oldest handsets have the lowest probability of churn. Does this make business sense? Yes, it does. A new device has a lower probability of churn because the customer hasn't had time to test it out yet. At the other end, if a customer has had a handset for more than 4 years, they probably like it.



This ICE plot shows churn probability by ever_days_over_plan. Each line represents the conditional expectation for one customer cluster. The plot indicates that for both clusters, there is a consistent increase in the probability of churn as ever_days_over_plan increases, given that other features are constant, the longer that a customer's usage exceeds plan limits, the more likely the customer is to churn.




Based on Kolmogorov-Smirnov statistic, KS, Forest is the champion model.

Model Studio - Build Models

Demo > "Model Comparison" Results

Node Assessment

Model Comparison

Champion	Name	Algorithm Name	KS (Youden)	Misclassification Rate
	Forest	Forest	0.5867	0.0650
	Auto Gradient Boosting	Gradient Boosting	0.5854	0.0647
	Auto Forest	Forest	0.5852	0.0639
	Gradient Boosting	Gradient Boosting	0.5825	0.0628
	Auto Decision Tree	Decision Tree	0.5724	0.0665
	Decision Tree	Decision Tree	0.5548	0.0683
	Logistic Regression	Logistic Regression	0.5417	0.0821

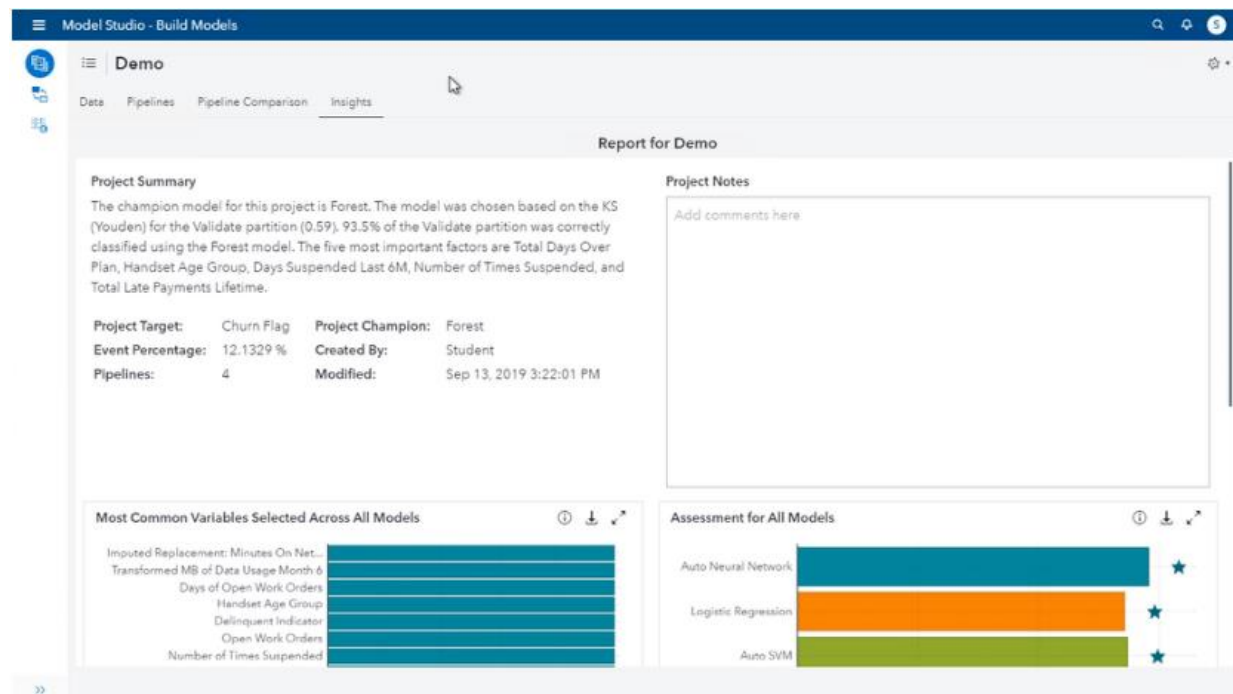
Properties

Property Name	Property Value
selectionCriteriaClass	Kolmogorov-Smirnov statistic (KS)
selectionCriteriaInterval	Average squared error
selectionTable	Validate
selectionDepth	10

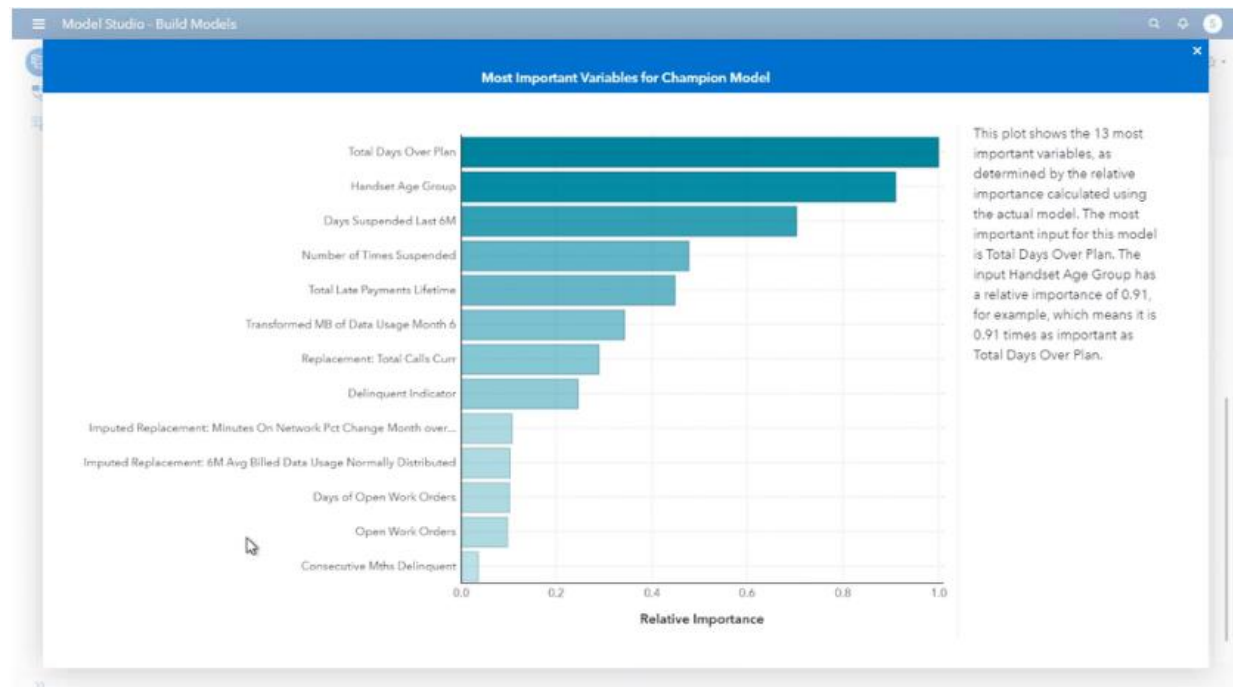
Scroll down to see the Fit Statistics table and maximize it. The Fit Statistics table shows how each model in the pipeline performs on the data partitions defined in the project settings for a series of fit statistics, such as area under the ROC curve, average squared error, Gini coefficient, and KS. Here we have 0.0511 and 0.0580, respectively for Training and Validation data, again for the Champion model, Forest.

[illegible]

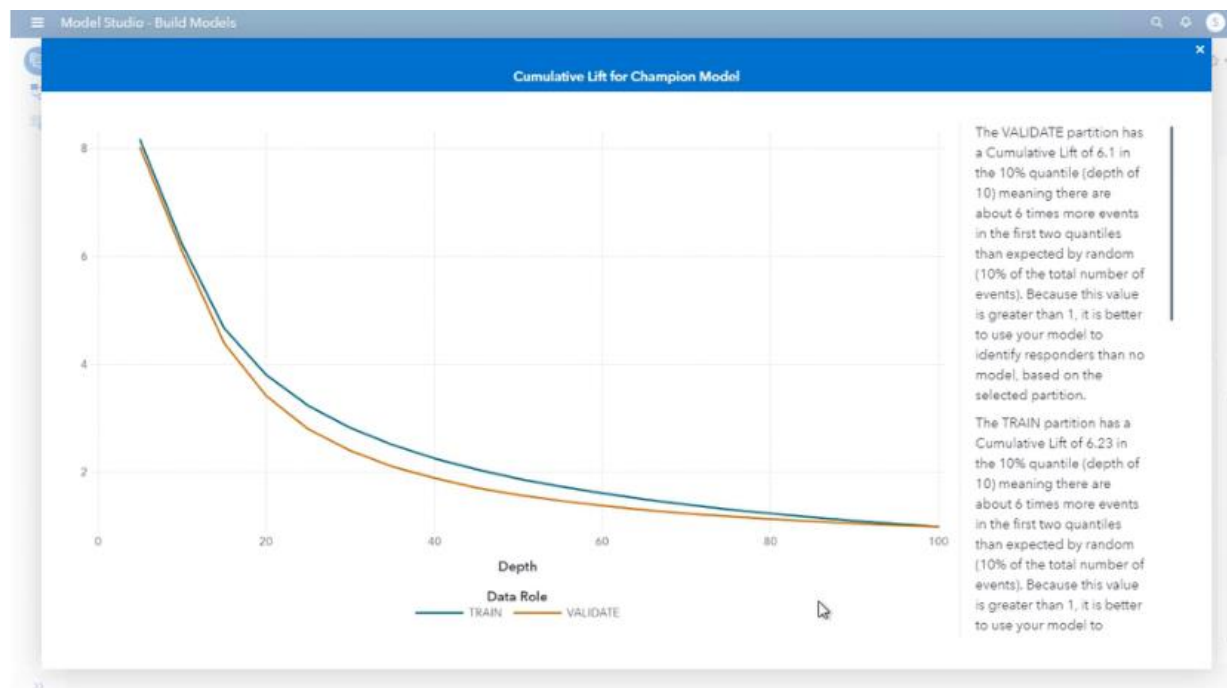
Reviewing a Project Summary Report on the Insights Tab



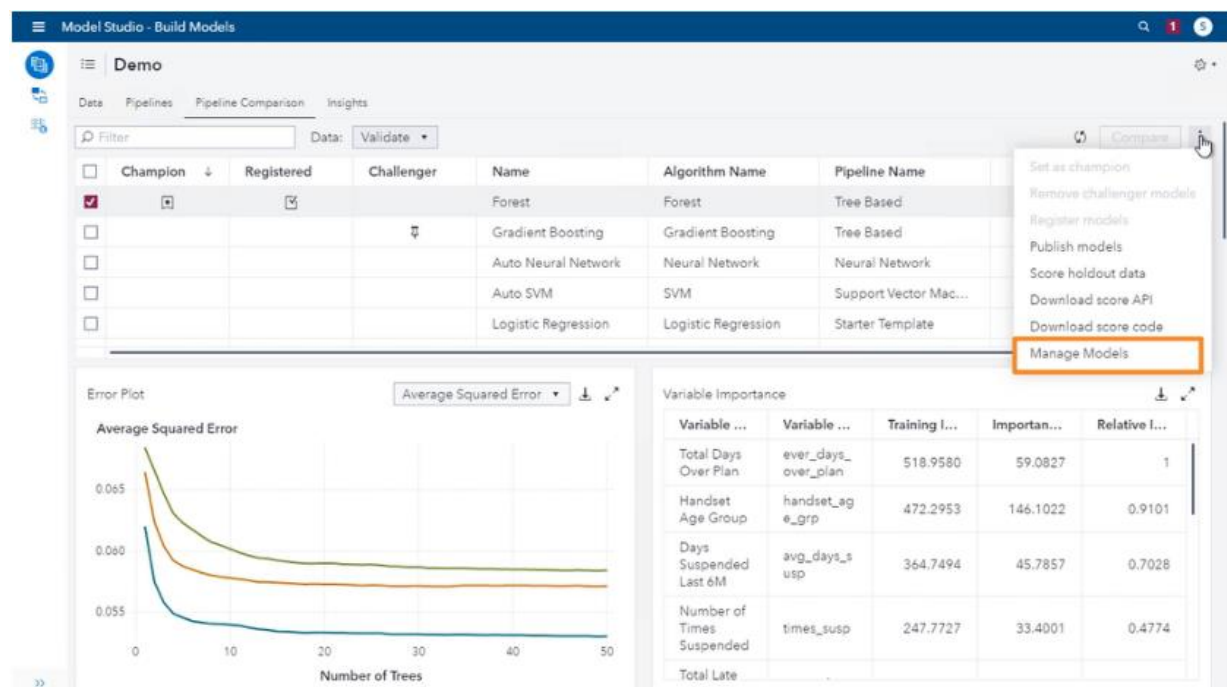
Most important variables



This plot displays the cumulative lift for the project champion model for both the training and validation partitions.



Registering the Champion Model



Output. By default, the score data table shows all the original variables; new variables created during data preparation; and the new variables created during the scoring process, which contain the predictions.

SAS® Model Manager - Manage Models

Demo > CPML_Forest

Search

Student

Test Results

- Output
- Code
- Log

Output Table

Warnings	Probability for churn = 1	Predicted for churn
	0.0597801318	0
	0.0387541818	0
	0.063000619	0
	0.0480084993	0
	0.9809884131	1
	0.0403340593	0
	0.8288900767	1
	0.0447800515	0
	0.0389200345	0
	0.0491627666	0
	0.0445688168	0