Modeling Customer Response (Auto Classifier) Model

The Auto Classifier node enables you to automatically create and compare a number of different models for flags (such as whether or not a given customer is likely to default on a loan or respond to a particular offer) targets.

In this example we'll search for a flag (yes or no) outcome. Within a relatively simple stream, the node generates and ranks a set of candidate models, chooses the ones that perform the best, and combines them into a single aggregated model. This approach combines the ease of automation with the benefits of combining multiple models, which often yield more accurate predictions than can be gained from any one model.

This example is based on a fictional company that wants to achieve more profitable results by matching the right offer to each customer.

This approach stresses the benefits of automation. For a similar example that uses a continuous (numeric range) target, see Property Values (Auto Numeric).

Figure 1. Auto Classifier sample stream



The file *pm_customer_train1.sav* has historical data tracking the offers made to specific customers in past campaigns, as indicated by the value of the *campaign* field. The largest number of records fall under the *Premium account* campaign.

The values of the *campaign* field are actually coded as integers in the data (for example $2 = Premium \ account$). Later, you'll define labels for these values that you can use to give more meaningful output.

The file also includes a *response* field that indicates whether the offer was accepted (0 = no), and 1 = yes). This will be the **target field**, or value, that you want to predict. A number of fields containing demographic and financial information about each customer are also included. These can be used to build or "train" a model that predicts response rates for individuals or groups based on characteristics such as income, age, or number of transactions per month.

Building the Stream

1. Add a Statistics File source node.

Figure 1. Reading in the data

😡 pm_customer_train1.sav	X
Preview) 2 Refresh	
\$CLEO_DEMOS/pm_customer_train1.sav	
Data Filter Types Annotations	
Import file: \$CLEO_DEMOS/pm_customer_train1.sav	
Variable names: 🔘 Read names and labels 🛛 Read labels as names	
Values: Read data and labels Read labels as data 	
☑ Use field format information to determine storage	
ОК Сапсе	Apply Reset

2. Add a Type node, and select *response* as the target field (Role = Target). Set the Measurement for this field to Flag.

Figure 2. Setting the measurement level and role

	eview				0-0
-35					
Types Format	Annotations				
×	Read Va	ilues Clear	Values	Clear All Va	alues
Field -	Measurement	Values	Missing	Check	Role
📿 customer_id 🖌	🔗 Continuous	[7,116993]		None	None ■
🔆 campaign 🛛	Nominal	1,2,3,4		None	🔪 Input
> response	🎖 Flag	1/0		None	O Target
response 🖌	Continuous	[2006-04		None	O None
🔿 purchase 🛛 🖌	Continuous	[0,1]		None	
purchase	Continuous	[2006-04		None	○ None
A product id	Continuous	[183,421]		None	None ■
🗸 product_id 🛛 👔	A Continuous	[1,19599]		None	○ None
Rowid	y conunuous				

- 3. Set the role to None for the following fields: *customer_id*, *campaign*, *response_date*, *purchase*, *purchase_date*, *product_id*, *Rowid*, and *X_random*. These fields will be ignored when you are building the model.
- 4. Click the Read Values button in the Type node to make sure that values are instantiated.

As we saw earlier, our source data includes information about four different campaigns, each targeted to a different type of customer account. These campaigns are coded as integers in the data, so to make it easier to remember which account type each integer represents, let's define labels for each one.

Figure 3. Choosing to specify values for a field

Type								
Format	Annotations	lues Clear	Values	Clear All Va	alues			
Field -	Measurement	Values	Missing	Check	Role	T		
Customer_id	🖉 Continuous	[7,116993]		None	S None	1		
campaign	💑 Nominal	<curr td="" 💌<=""><td></td><td>None</td><td>🔪 Input</td><td></td></curr>		None	🔪 Input			
> response	🎖 Flag	<read></read>		None	O Target			
response	🖉 Continuous	<read +=""></read>		None	O None			
purchase	🖉 Continuous	<pass></pass>		None	O None	1		
purchase	🖉 Continuous	«Current»		None	O None			
product_id	🔗 Continuous	Specify N		None	O None			
决 Rowid	🔗 Continuous	Veecer, IL		None	O None			
ana 🐔	A Continuous	110 061	2	None	N Input	Ľ		
OK Cancel	fields 🔘 View unu:	sed field settin	gs		Apply	ese		

5. On the row for the campaign field, click the entry in the Values column.

6. Choose Specify from the drop-down list.

Figure 4. Defining labels for the field values

😡 campaign	Values		×
Measurement:	💰 Nominal 🔄 Storage:	Onteger Model Field	
Values:	Read from data O Specify values and labels	Pass	
	Values	Labels	
	1	Standard account	<u> </u>
	2	Premium account	
	3	Gold account	
	4	Platinum account	
📃 Define blank	(5		1
	Missing values		
			×
	Range	to	
	Null White space		
Description:]
	ОК Саг	icel Help	

Although the data includes information about four different campaigns, you will focus the analysis on one campaign at a time. Since the largest number of records fall under the Premium account campaign (coded *campaign*=2 in the data), you can use a Select node to include only these records in the stream.

Figure 6. Selecting records for a single campaign

Select		×
<u>_</u> ?>	Preview	0
Settings	Annotations	
Mode:	🔘 Include 🔘 Discard	
Condition:	campaign = 2	
OK Can	cel	Apply Reset

Generating and Comparing Models

- 1. Attach an Auto Classifier node, and select Overall Accuracy as the metric used to rank models.
- 2. Set the Number of models to use to 3. This means that the three best models will be built when you execute the node.

😡 resp	🖸 response 🛛 🛛 🔀									
						0				
	Estim	ated num	per of mod	els to be e:	kecuted: 9					
Fields	Model	Expert	Discard	Settings	Annotations					
Model nar	ne:	0.	Auto 🔘 C	ustom						
👿 Use p	artition	ed data								
🔽 Build i	model f	or each s	plit		_93					
Rank mod	lels by:	Ove	erall accur	acy 🔻						
Rank mod	lels usir	ng: 🔘 🤇	Training pa	artition	Test partition					
Number o	f mode	ls to use:			3 ≑					
🛃 Calcu	late pre	dictor imp	ortance							
Profit Cri	teria (v	alid only f	or flag targ	gets)	4	-				
Costs:	٢	Fixed		5.0 🌲	© ∨ariable		-			
Revenu	e: 🔘	Fixed		10.0 ≑	© ∨ariable		-			
Weight:	0	Fixed		1.0 ≑	🔘 Variable					
Lift Crite	ria (vali	d only for	flag targe	ts)	2					
Percenti	le to us	e for lift c	alculation:	30	•					
ок		Run	Cancel				Reset			

Figure 1. Auto Classifier node Model tab

On the Expert tab you can choose from up to 11 different model algorithms.

3. Deselect the Discriminant and SVM model types. (These models take longer to train on these data, so deselecting them will speed up the example. If you don't mind waiting, feel free to leave them selected.)

Because you set Number of models to use to 3 on the Model tab, the node will calculate the accuracy of the remaining nine algorithms and build a single model nugget containing the three most accurate.

Model Expert Discard Settings Annotations odels used: Model type Model parameters No of models Image: Setting in the set in th	Experi Discard Settings Annotations Model type Model parameters No of models	Fields Model Expert Discard Settings Annotations									
Model type Model parameters No of models Image: Second state	Model type Model parameters No of models	ields Mode	Expert	Discard Se	ttings Annotations						
Image: Construction of the second	Children C5 Default 1 Logistic r Default 1 Decision Default 1 Bayesian Default 1 Discrimin Default 1 Logistic r Default 1	se?	Model	tvpe	Model parameters	No of models					
Image: Construction of the construc	Image: Logistic r Default 1 Image: Decision Default 1 Image: Decision Default 1 Image: Decision Default 1 Image: Decision Default 1 Image: Discrimin Default 1 Image: KNN Alg Default 1	C5 C5		C5	Default	1					
Image: Constraint of the second se	Decision Default 1 Bayesian Default 1 Discrimin Default 1 KNN Alg Default 1			Logistic r	Default	1					
Bayesian Default 1 Discrimin Default 1	Bayesian Default 1 Discrimin Default 1 Liscrimin Default 1		3	Decision	Default	1					
Discrimin Default	Discrimin Default 1	-	- 33-	Bayesian	. Default	1					
	KNN Alg Default 1			Discrimin	Default	1					
KNN Alg Detault		KNN Alg		Default	1						
E SVM Default 1	SVM Default 1		SVM		Default	1					
Sector Car Tree Default 1	ACRT C&R Tree Default 1		₹ A C&R Tree		Default	1					
Quest Tr Default 1	A control portant		A	Quest Tr	Default	1					
OUEET	Quest Tr Default	-	CHAID	CHAID Tree	Default	1 ,					
SVM Default 1 Image: SVM Default 1 Image: SVM Default 1 Image: SVM Default 1 Image: SVM Default 1	SVM Default 1		SVM		Default Default Default	1					
A KT C&R Tree Default	ALT C&R Tree Default 1		C&R Tree		Default	1					
🗹 🐣 Quest Tr Default 1	A control portain	-	A COUEST	Quest Tr	Default	1					
	A Quest Tr Default 1		WEST A	CHAID Tree	Default	1					

Figure 2. Auto Classifier node Expert tab

4. On the Settings tab, for the ensemble method, select Confidence-weighted voting. This determines how a single aggregated score is produced for each record.

With simple voting, if two out of three models predict *yes*, then *yes* wins by a vote of 2 to 1. In the case of confidence-weighted voting, the votes are weighted based on the confidence value for each prediction. Thus, if one model predicts *no* with a higher confidence than the two *yes* predictions combined, then *no* wins.

Figure 3. Auto Classifier node: Settings tab

rest	ponse					× •
Fields	Estim	nated nu	mber of r Discard	nodels to Settings	be executed:	9
Flag Ens If vo	ble Settir Target – emble m oting is ti 0 Randon 0 Raw pr	ngs ethod: ed, select n selection opensity	Confidence value usir n () Highe	e-weightec ng: est confider	I voting 💌	
ОК	🕨 Run	Cano	el			Apply Reset

5. Click Run.

After a few minutes, the generated model nugget is built and placed on the canvas, and on the Models palette in the upper right corner of the window. You can browse the model nugget, or save or deploy it in a number of other ways.

Figure 4. Generated model displayed in the palette

Streams	Outputs	Models	
response	2		

Open the model nugget; it lists details about each of the models created during the run. (In a real situation, in which hundreds of models may be created on a large dataset, this could take many hours.)

Figure 5. Auto Classifier stream with model nugget



If you want to explore any of the individual models further, you can double-click on a model nugget icon in the Model column to drill down and browse the individual model results; from there you can generate modeling nodes, model nuggets, or evaluation charts. In the Graph column, you can double-click on a thumbnail to generate a full-sized graph.

Figure 6. Auto Classifier results

💟 res	ponse								×
-	jie <u>File</u>	🏷 Generate 🛛 💰 View	Preview					0	
Model	Graph Summ	nary Settings Annotatio	ons						_
Sort by	Sort by: Overall accuracy 🔻 🛇 Ascending 🔘 Descending								
Use?	Graph	Model	Build Time (mins)	Max Profit	Max Profit Occurs in (%)	Lift{Top 30%}	Overall Accuracy ∇	No. Fields Used	Area Under Curve
		c5 1	< 1	4,906.667	8	2.203	92.861	10	0.777
		C&R Tree 1	3	4,602.692	9	2.778	92.365	8	0.924
		CHAID Tree 1	3	4,145.668	8	2.851	91.706	4	0.927
—									
ОК	Cancel							Ar	ply <u>R</u> eset

By default, models are sorted based on overall accuracy, because this was the measure you selected on the Auto Classifier node Model tab. The C51 model ranks best by this measure, but the C&R Tree and CHAID models are nearly as accurate.

You can sort on a different column by clicking the header for that column, or you can choose the desired measure from the Sort by drop-down list on the toolbar.

Based on these results, you decide to use all three of these most accurate models. By combining predictions from multiple models, limitations in individual models may be avoided, resulting in a higher overall accuracy.

In the Use? column, select the C51, C&R Tree, and CHAID models.

Attach an Analysis node (Output palette) after the model nugget. Right-click on the Analysis node and choose Run to run the stream.





The aggregated score generated by the ensembled model is shown in a field named *\$XF*-*response*. When measured against the training data, the predicted value matches the actual response (as recorded in the original *response* field) with an overall accuracy of 92.82%.

While not quite as accurate as the best of the three individual models in this case (92.86% for C51), the difference is too small to be meaningful. In general terms, an ensembled model will typically be more likely to perform well when applied to datasets other than the training data.

Figure 8. Analysis of the three ensembled models

