CIDM 6355 Data Mining Methods Week 6 Lab - Neural Nets and Model Evaluation

Purposes: Understand and apply the neural network algorithm to develop a data mining model to solve a business problem; interpret the neural network DM model; Evaluate DM model performance using confusion matrix and lift chart; compare different DM models.

Dataset: Please download the dataset titled "EastWestAirlinesNN.csv" from WT Class to use it with this lab session.

Requirements: Each student must practice this lab individually, and then participate in our online discussion board.

1 Business Understanding

An airline company is entering a partnership with a telecom company. The mutual goal of the companies is to identify which airline customers would be likely to purchase the services offered by the telecom firm. The dataset "EastWestAirlinesNN.csv" shows the results from a pilot study. Open the dataset using a text processor and observe the data and the attributes. Your goal is to build different classification models and test their performance to predict the behavior of future customers. Notice that all attributes values are numeric by default.

2 Data Understanding

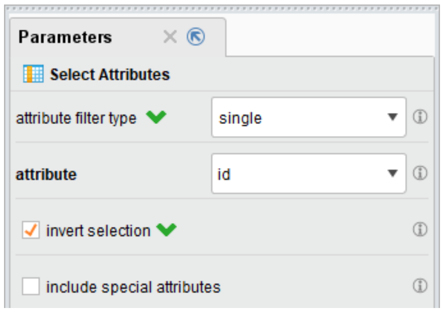
The target attribute is Phone_sale (a class label), where it is 1 if the customer purchased the cellular services and 0 if he/she has not. Remaining attributes represent the relationship of customers with the airline -- such as flight miles, online booking, club membership, etc. The following table describes each attribute.

Field Name	Description
ID#	Unique ID
Topflight	Indicates whether flyer has attained elite "Topflight" status, 1 = yes, 0 = no
Balance	Number of miles eligible for award travel
Qual_miles	Number of miles counted as qualifying for Topflight status
cc1_miles	Has member earned miles with airline freq. flyer credit card in the past 12 months (1=Yes/0=No)?
cc2_miles	Has member earned miles with Rewards credit card in the past 12 months (1=Yes/0=No)?
cc3_miles	Has member earned miles with Small Business credit card in the past 12 months (1=Yes/0=No)?
Bonus_miles	Number of miles earned from non-flight bonus transactions in the past 12 months
Bonus_trans	Number of non-flight bonus transactions in the past 12 months
Flight_miles_12mo	Number of flight miles in the past 12 months
Flight_trans_12	Number of flight transactions in the past 12 months
Online_12	Number of online purchases within the past 12 months
Email	E-mail address on file. 1= yes, 0 =no?
Club_member	Member of the airline's club (paid membership), 1=yes, 0=no
Any_cc_miles_12mo	Dummy variable indicating whether member added miles on any credit card type within the past 12 months (1='Y', 0='N')
PHONE SALE	Dummy variable indicating whether member purchased Telcom service as a result of the direct mail campaign (1=sale, 0=no sale)

3 Data Preparation

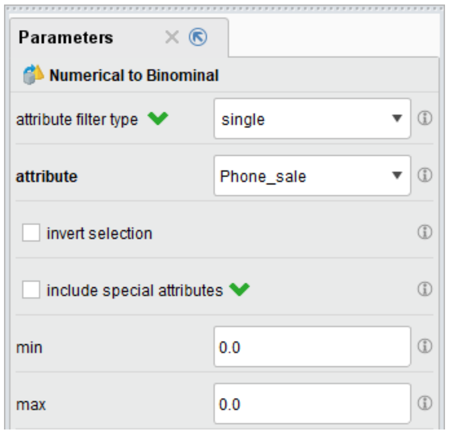
3.1 Read the data file using the Read CSV operation with all the default parameters. Run it and observe the results and statistics. In terms of data quality, the data set seems to be OK: No missing or extreme values. However, please notice three issues at this point: the attribute ID, the target attribute, and highly correlated attributes.

3.2 The first issue is that there is an attribute called ID. It would be better to remove this attribute before building our models (for reference, please read Feature/Variable Selection in Week 3). Use "Select Attributes" operator to select all attributes except ID.


Parameters
Select Attributes
attribute filter type: single
attribute: id
invert selection: ✓
include special attributes: ☐

Tip: Because we need to drop only one attribute, we have only one attribute to drop. In addition to the method used in Step 1.5 (we call direct selection) in Week 4 Lab, another method (reverse selection) can be used too (see the parameters of the Select Attributes in the right)

3.3 Since all attribute values are numeric by default, RM may automatically detect all attribute types to be integer -- even if some of them are certainly binomial (such as phone_sale, club_member, etc.). There is nothing wrong with this, in fact Neural Net implementations only **work on numeric values**. However, it is a better practice to change the class label (phone_sale) type to binomial at this point; otherwise certain algorithms may not run properly. To do this, use the Numerical to Binominal operator and configure it as attribute filter type = single and attribute= Phone_sale (See the screenshot in the right).

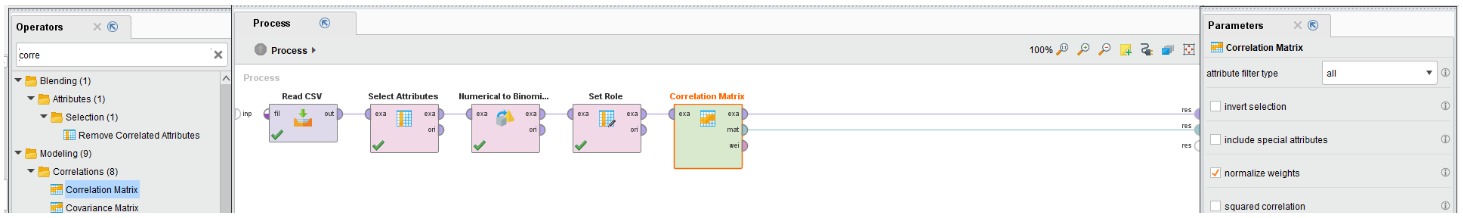

Parameters
Numerical to Binominal

attribute filter type	single
attribute	Phone_sale
invert selection	☐
include special attributes	☐
min	0.0
max	0.0

Tip: Of course, you can change the data type of this operator in the third step of the Import Configuration Wizard, just as we did in Week 3 Lab. However, if you do not want to import your data again, the operators under Blending->Attributes->Types can be used for such transformation.

3.4 Then, add the operator Set Role to set Phone_sale as the label in the target role.

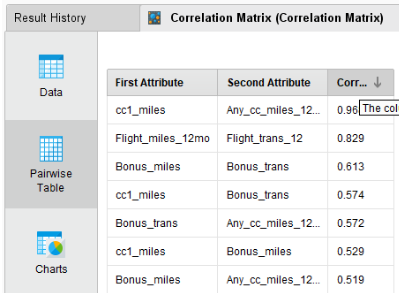
3.5 The third issue is that certain attributes may be highly correlated with each other (flight_miles, flight_trans, etc.). Using highly correlated attributes in the same model may affect the performance of the classification. We need to first to generate a correlation matrix to check if there is any high correlation coefficient. Add the operator Correlation Matrix (Under Modeling->Correlation or search it in the operator tab). This operator has three possible outputs (Example Set, Correlation Matrix, and Attribute weights vector). At this stage, we are going to need the first two outputs. Run this process and you will see the two outputs. Please take a look at the correlation matrix and you will find that those numbers are colored differently: **the darker the color is, the greater the correlation is**. Please ignore those numbers in the diagonal because all of them describe self-correlation, that is, 1.



Parameters
Correlation Matrix

attribute filter type	all
invert selection	☐
include special attributes	☐
normalize weights	✓
squared correlation	☐

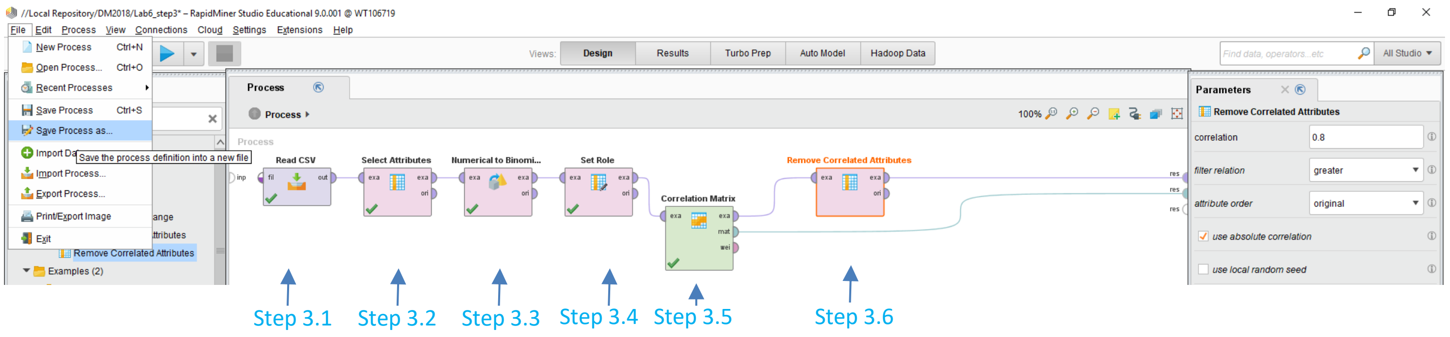
Tip: When you click Pairwise Table (as shown below), you will see the correlation coefficient between any two regular attributes. Sort them by the third column, Correlation, in the descending order, and you will see the highly-correlated attributes. **Attention**: Both 0.92 and -0.92 indicate a high correlation (Please the Description of the operator to know more about correlation). Because RM sorts correlation by the value of correlation coefficient, rather than absolute value, you need to sort this column twice: from largest to smallest (to find the highest positive correlation coefficients) and from the smallest to the largest (for the highest negative correlation coefficients). However, in this case, we notice that our negative correlations are quite small, so you can sort them in the descending order.



First Attribute	Second Attribute	Corr...
cc1_miles	Any_cc_miles_12...	0.96
Flight_miles_12mo	Flight_trans_12	0.829
Bonus_miles	Bonus_trans	0.613
cc1_miles	Bonus_trans	0.574
Bonus_trans	Any_cc_miles_12...	0.572
cc1_miles	Bonus_miles	0.529
Bonus_miles	Any_cc_miles_12...	0.519

3.6 To remove correlated attributes automatically, use the Remove Correlated Attributes operator. Set the correlation value to 0.8 (meaning that we will remove an attribute with a correlation coefficient of 0.8 or higher with another attribute).

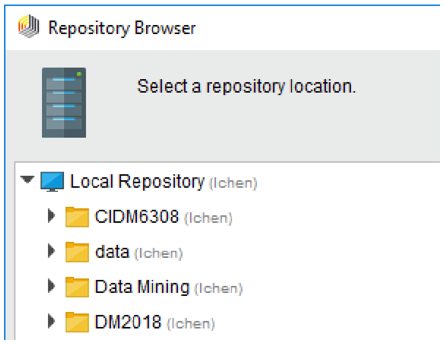
3.7 Below is the screenshot of the process to conduct these operations. Save this process as Week6Labstep3 via File -> Save Process or Save Process as and then you can select an appropriate repository to save this process so that you can retrieve it in the future.

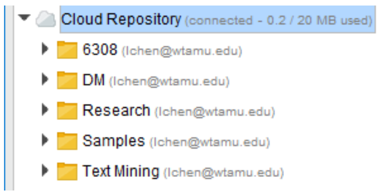

//Local Repository/DM2018/Lab6_step3* – RapidMiner Studio Educational 9.0.001 @ WT106719
File Edit Process View Connections Cloud Settings Extensions Help
New Process Ctrl+N
Open Process... Ctrl+O
Recent Processes
Save Process Ctrl+S
Save Process as...
Save the process definition into a new file
Import Process...
Export Process...
Print/Export Image
Exit
Remove Correlated Attributes
Examples (2)
Views: Design Results Turbo Prep Auto Model Hadoop Data
Process
Read CSV
Select Attributes
Numerical to Binomi...
Set Role
Correlation Matrix
Remove Correlated Attributes
Step 3.1
Step 3.2
Step 3.3
Step 3.4
Step 3.5
Step 3.6

Parameters — Remove Correlated Attributes

correlation: 0.8
filter relation: greater
attribute order: original
☑ use absolute correlation
☐ use local random seed

Tip: You can save the process in your local repository or cloud repository. For the cloud repository (20 MB free space), you can get access to them in any computer with RM as long as you log into your RM account. Therefore, please find a good way to organize your folder in your repository for future use.




Cloud Repository (connected - 0.2 / 20 MB used)
6308 (lchen@wtamu.edu)
DM (lchen@wtamu.edu)
Research (lchen@wtamu.edu)
Samples (lchen@wtamu.edu)
Text Mining (lchen@wtamu.edu)

3.8 Run this process and Take a look at your ExampleSet and Correlation Matrix to see which attributes were dropped in Step 3.6?

Tip: Here, you may wonder which attribute should be dropped if two attributes A and B are highly correlated: A or B? The parameter, attribute order, in the Remove Correlated Attributes operator is the algorithm takes the attribute order to calculate correlations and for filtering the attributes. This parameter will decide which attribute will be dropped based on their occurrence order in your dataset. It has three options:

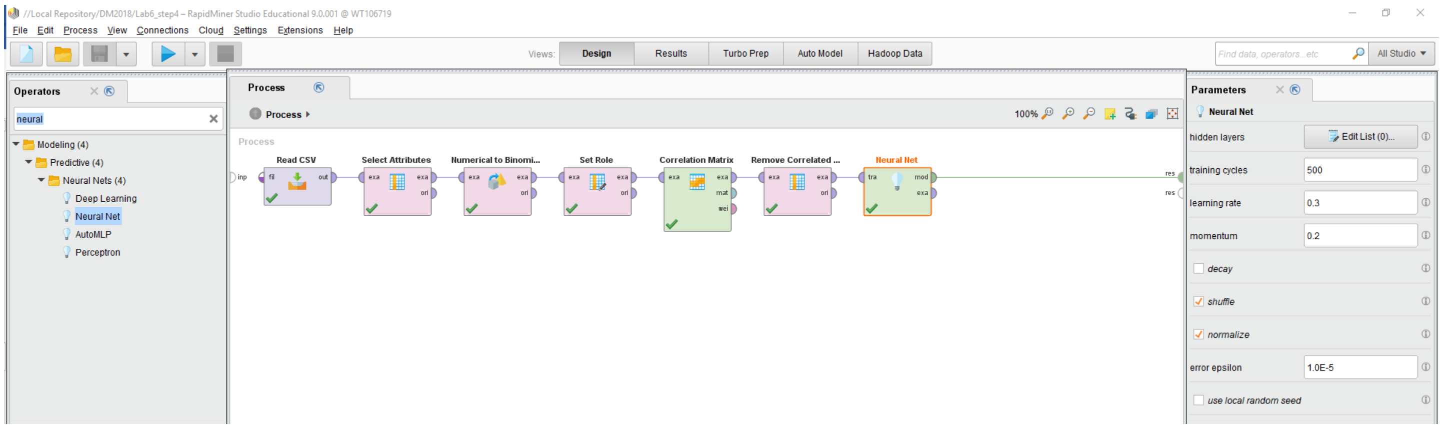
- Original (the default): keep the attribute which appears earlier in your dataset (see the table under Data Understanding) while drop the other.
- Random: drop and keep the attribute randomly.
- Reverse: the opposite selection of the original, dropping the attribute appearing earlier in your dataset.

[Optional] Change the parameter from original to reverse to see which attributes are dropped. **Once you finish this optional practice, please change this parameter back to original for the following practice**.

4 Modeling (Neural Network)

4.1 Add a new operator Neural Net (under Modeling -> Predictive->Neural Nets). Now, you do not need the output of correlation matrix, so you either drop the operator of Correlation Matrix or drop the connection between mat and res.

4.2 Use the following parameters of the Neural Net operator and then run it (note: as mentioned in our PPT slides, the calculation is quite complex, so this operator is computationally expensive, and you need to wait for a while to see your results. If you allow more than one hidden layer, it may take even longer to get the results).



Parameters — Neural Net

hidden layers	Edit List (0)...
training cycles	500
learning rate	0.3
momentum	0.2
decay	☐
shuffle	✓
normalize	✓
error epsilon	1.0E-5
use local random seed	☐

4.3 Take a look at the neural network model and then think about the following questions (You may refer to our PPT slides or assigned readings).

4.3.1 How many input nodes (excluding the bias node) in the model?

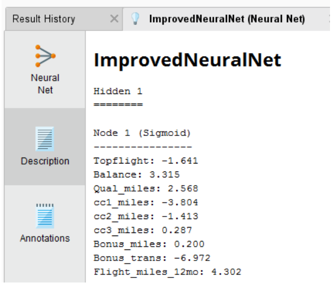
4.3.2 How many output nodes in the model?

4.3.3 How many hidden nodes (excluding the threshold node) in the model?

4.3.4 How many hidden layers in the model?

4.3.5 How many weights in total in the model?

Note: There is a bias node in the input layer and one threshold node in the hidden layer. Both of them are displayed in a different color with other nodes at the same layer.
When you click the description tab (see the right), you will see the weights between input nodes and hidden nodes, and the weights between hidden nodes and output nodes. Of course, you can also find the weights of bias/threshold nodes.
For question 4.3.5, you can use the formula mentioned in our PPT slide 13: the number of weights for a network with *s* input units, *h* hidden units, and *i* outputs is *h**(*s*+1)+(*h*+1)**i*. You can also count the number of weights in the description or the number of line in your neural network graph (it will take a while).
Each line in the neural network model represents a weight and the size of the line represents the size of its weight: the bolder a line is, the stronger a weight is (a stronger could be positive or negative).



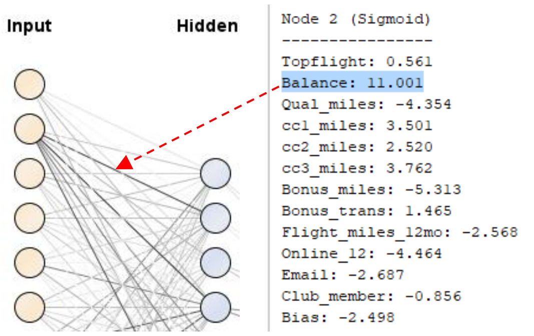
```
ImprovedNeuralNet

Hidden 1
========

Node 1 (Sigmoid)
----------------
Topflight: -1.641
Balance: 3.315
Qual_miles: 2.568
cc1_miles: -3.804
cc2_miles: -1.413
cc3_miles: 0.287
Bonus_miles: 0.200
Bonus_trans: -6.972
Flight_miles_12mo: 4.302
```

For example, the line between hidden note 2 and input node 2 is quite bold. Take a look at the weight in Description:
The weight of Balance (input node 2) on the hidden node 2 is -12.001, which makes the line between the two nodes quite bold.

As mentioned in our lecture, the process of neural network is like a black box and we do not know what exactly those hidden nodes, but it is a quite powerful method for making prediction.



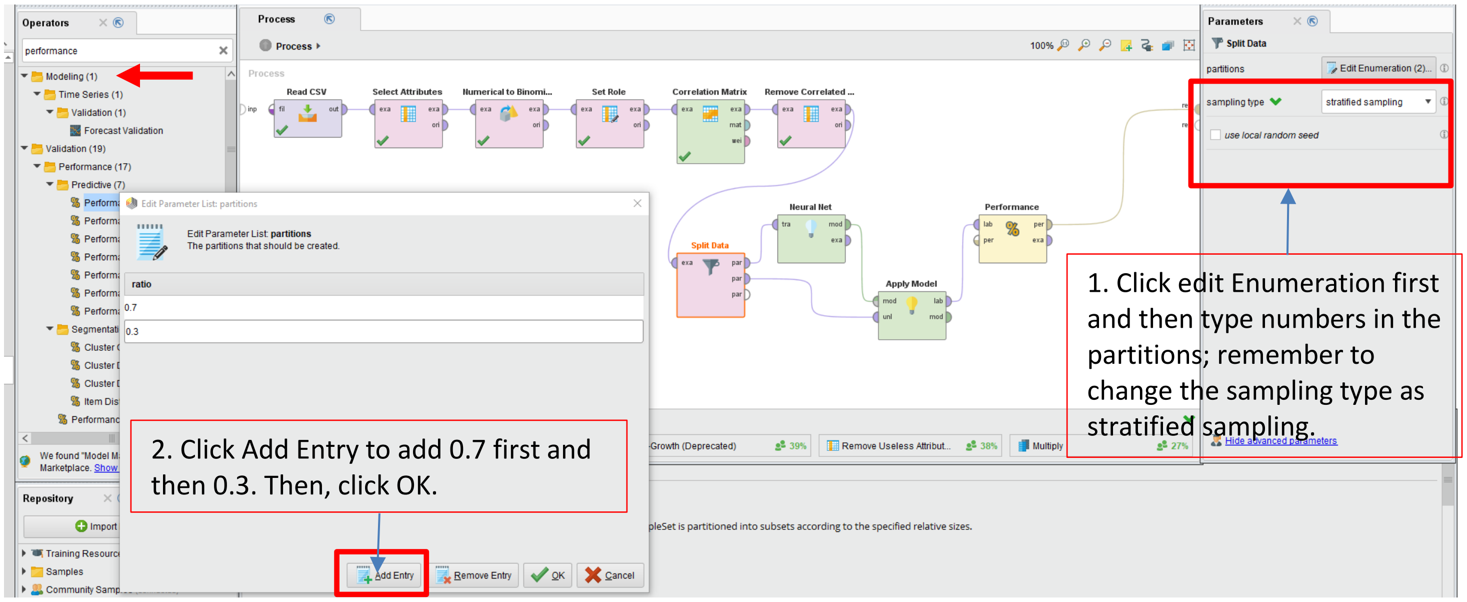
```
Node 2 (Sigmoid)
----------------
Topflight: 0.561
Balance: 11.001
Qual_miles: -4.354
cc1_miles: 3.501
cc2_miles: 2.520
cc3_miles: 3.762
Bonus_miles: -5.313
Bonus_trans: 1.465
Flight_miles_12mo: -2.568
Online_12: -4.464
Email: -2.687
Club_member: -0.856
Bias: -2.498
```

5 Model Evaluation

We are going to evaluate the model performance using the hold-out method:

5.1 Next, we will build a Neural Net model and assess its performance. We will use the Hold-out method for model assessment.

5.2 Build the RM process shown below using the hold-out method. Use the <u>Split Data</u> operator with 70/30 split to create two datasets -- a training one and a testing one. Use <u>stratified sampling</u> when splitting the data (see the screenshot below)



Edit Parameter List: partitions
The partitions that should be created.

ratio
0.7
0.3

partitions: Edit Enumeration (2)...
sampling type: stratified sampling
use local random seed

1. Click edit Enumeration first and then type numbers in the partitions; remember to change the sampling type as stratified sampling.

2. Click Add Entry to add 0.7 first and then 0.3. Then, click OK.

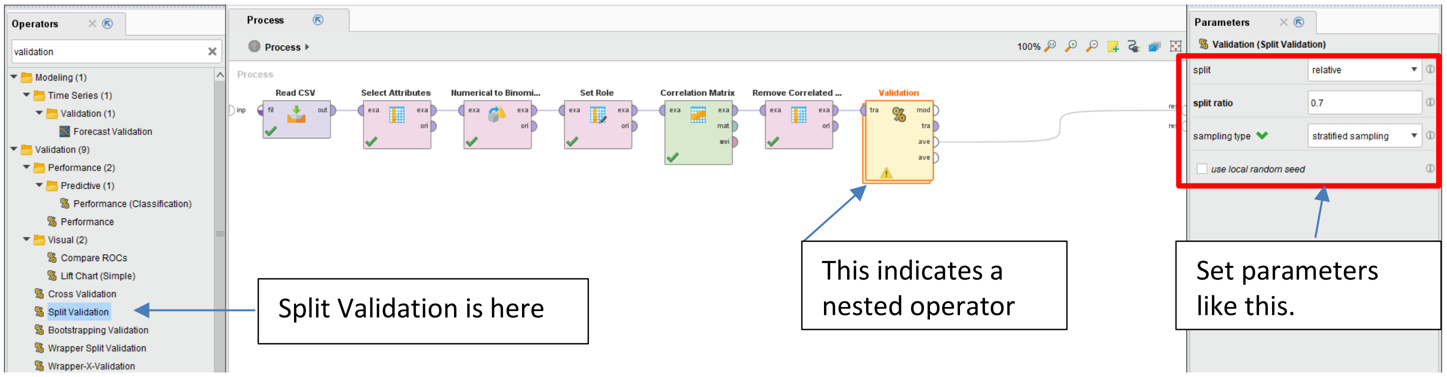
...pleSet is partitioned into subsets according to the specified relative sizes.

5.3 Then, organize your Neural Net operator and Apply Model operator as the process above. 70% of the data will be used for training the model and the other 30% will be used for testing the model.

5.4 A new operator that you will use for model evaluation is called Performance (Classification) (under Validation -> Performance -> Predictive or search performance in the search box, as shown in the screenshot above). This operator will create the Confusion Matrix on the test dataset as well as additional performance criteria.

5.5 Save this process as Week6Labstep5.5. Once you run the process, you will see the confusion matrix, including the accuracy, true class precision and true class recall metrics. Think about if this is a good model to predict who is a "true" person (True meaning that the person purchased the phone service). For more details about the confusion matrix, please refer to the Appendix at the end of this lab instruction.

5.6 A different way of building the hold-out method is using the Split Validation operator. Use the Split validation to create 2 different datasets from the existing data, train the model on one of them and test it on the other one. Note that Validation is a nested operator (i.e., it has a subprocess), so you need to double click on it and add additional operators as shown below.



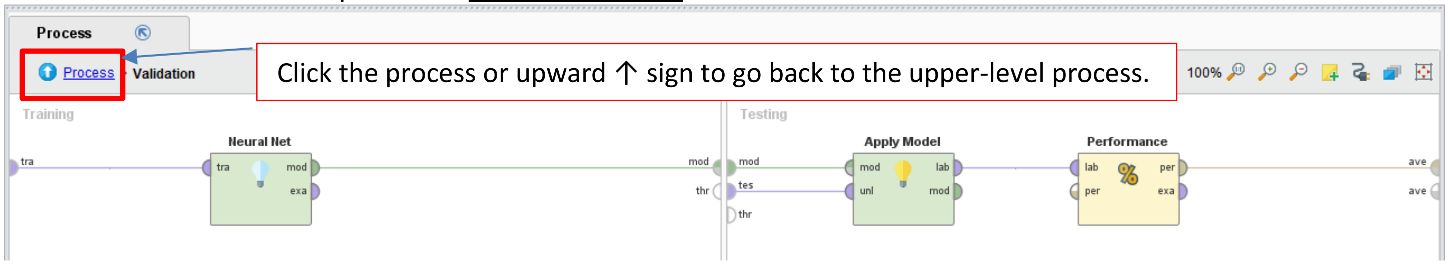
split: relative
split ratio: 0.7
sampling type: stratified sampling
use local random seed

Split Validation is here

This indicates a nested operator

Set parameters like this.

5.7 Double click the Split Validation operator and set the three operators inside as below. Make sure you have correct connections. Save this process as Week6Labstep5.7.



Click the process or upward ↑ sign to go back to the upper-level process.

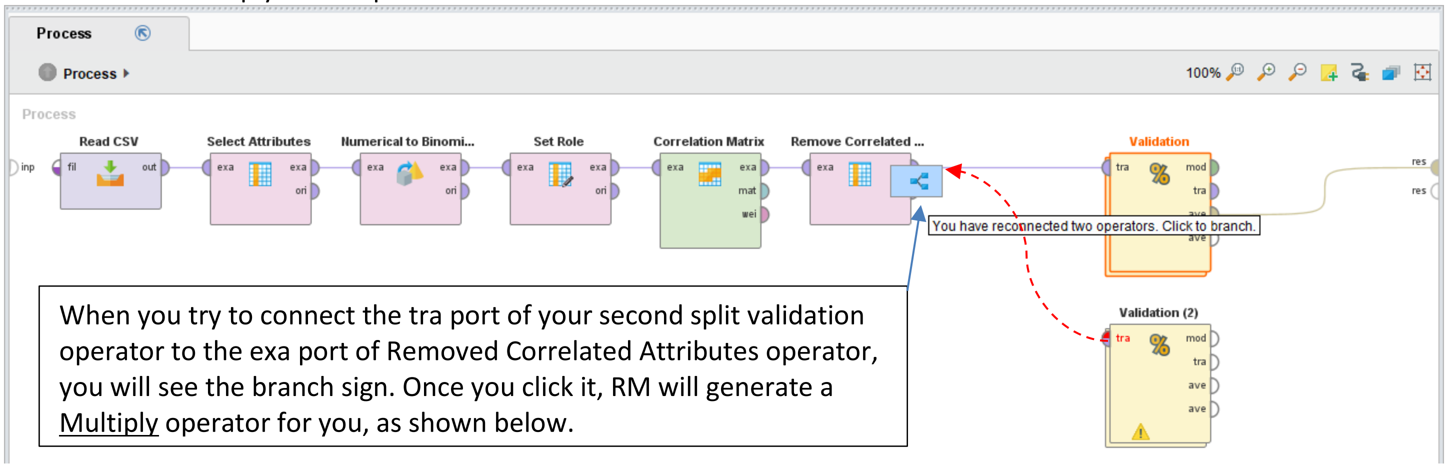
5.8 **Run this process.** Obtain the results after running the split validation and take a look at the confusion matrix. Notice that results should be identical to those in Step 5.5, because we are doing the same operation in two different ways (splitting the dataset 70-30).

Note: An interesting observation here is that even if you run the same process multiple times, RapidMiner always picks the same training and test datasets, even if you do random sampling. However, if you close and open the program again, your results may be different than the previous runs. Such a sampling issue can be overcome with cross validation (X-Validation).

6 Model Comparison

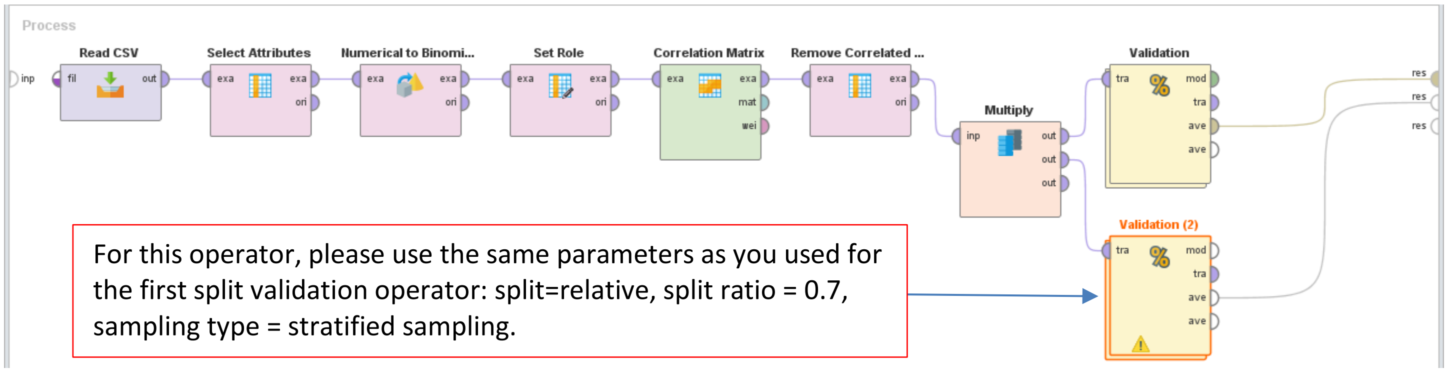
Please note that so far, we have only used the Neural Net algorithm and obtained its performance results. In a real world situation, you may want to try out different classification techniques for the same problem. Suppose that you would now like to use Naive Bayes and compare its performance with that of Neural Nets.

6.1 Add a second Split Validation operator. When you try to connect its tra port to the exa port of the Removed Correlated Attribute operator, the Multiply operator is generated automatically. The Multiply operator creates copies of a RapidMiner Object (e.g., a dataset). Alternatively, you may add the Multiply operator directly by search “multiply in the operator search box.



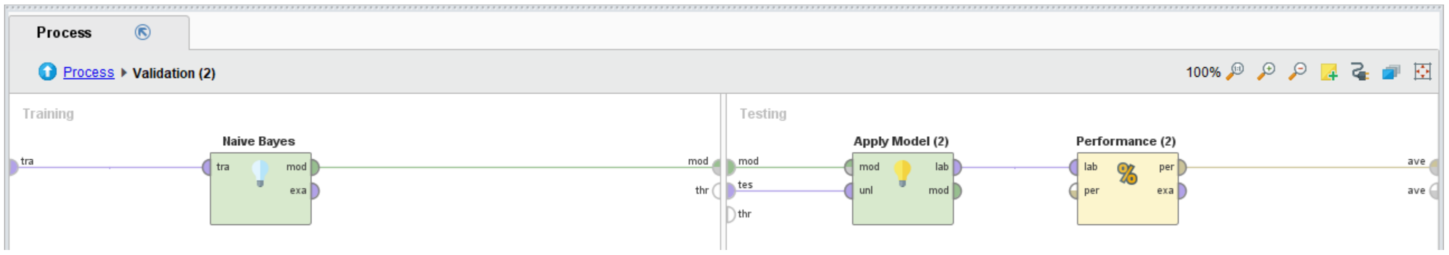
You have reconnected two operators. Click to branch.

When you try to connect the tra port of your second split validation operator to the exa port of Removed Correlated Attributes operator, you will see the branch sign. Once you click it, RM will generate a Multiply operator for you, as shown below.



For this operator, please use the same parameters as you used for the first split validation operator: split=relative, split ratio = 0.7, sampling type = stratified sampling.

6.2 Double click the second Split Validation operator, and then set the similar subprocess as you did for Neural Net model. The only difference is that it is using Naive Bayes as the training algorithm.



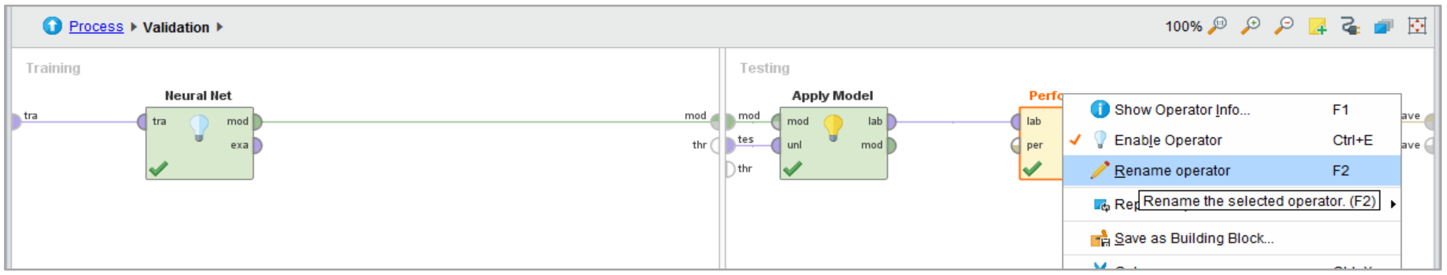
6.3 Save this process as Week6Labstep6. Run your process and check out your results. Take a look at the confusion matrix for the Naive Bayes model. Please think about the following questions:

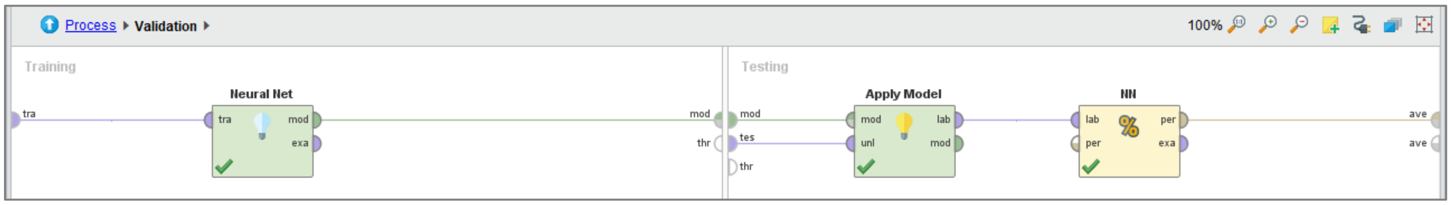
6.3.1 What is the total number of the cases in the confusion matrix is (the sum of all the four cells)?

6.3.2 If our goal were to only find out the True people, which algorithm seems to be better in each of the following measures?

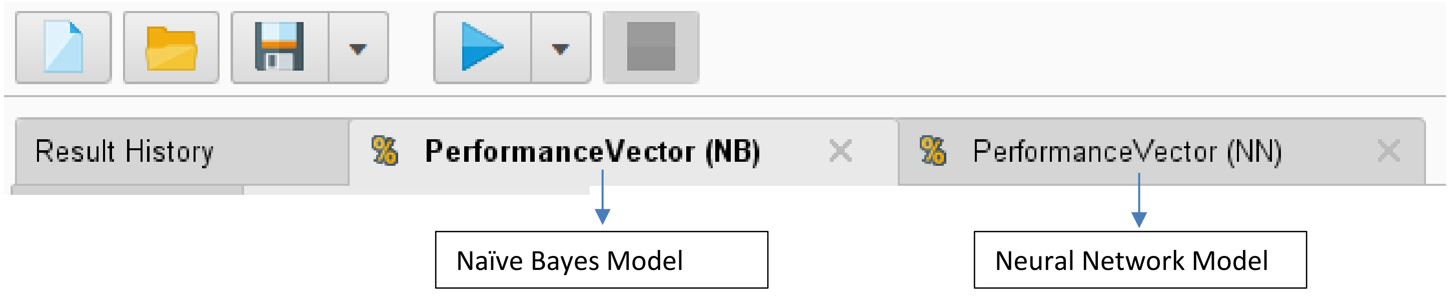
- Which model produce the higher accuracy?
- Which model produce the higher recall?
- Which model produce the higher precision?
- Which model produce the higher number of true cases which are also predicted as true cases?

Tip: Here, we are comparing multiple DM models, so we need to check multiple performance vectors, which are hard to distinguish which performance vector is for which model. In order to avoid such a confusion, you may change the name of Performance operator to indicate the model you used.





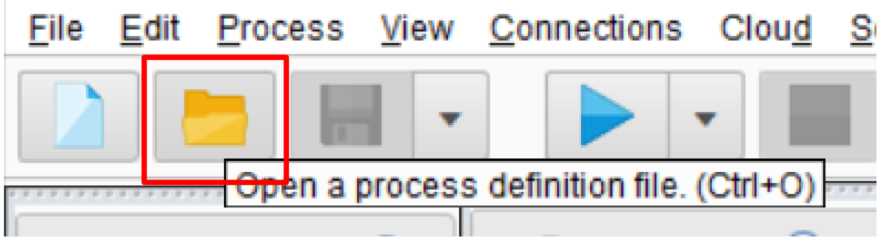
After run this process, you will know which confusion matrix belongs to which DM model.



7 Generate lift chart

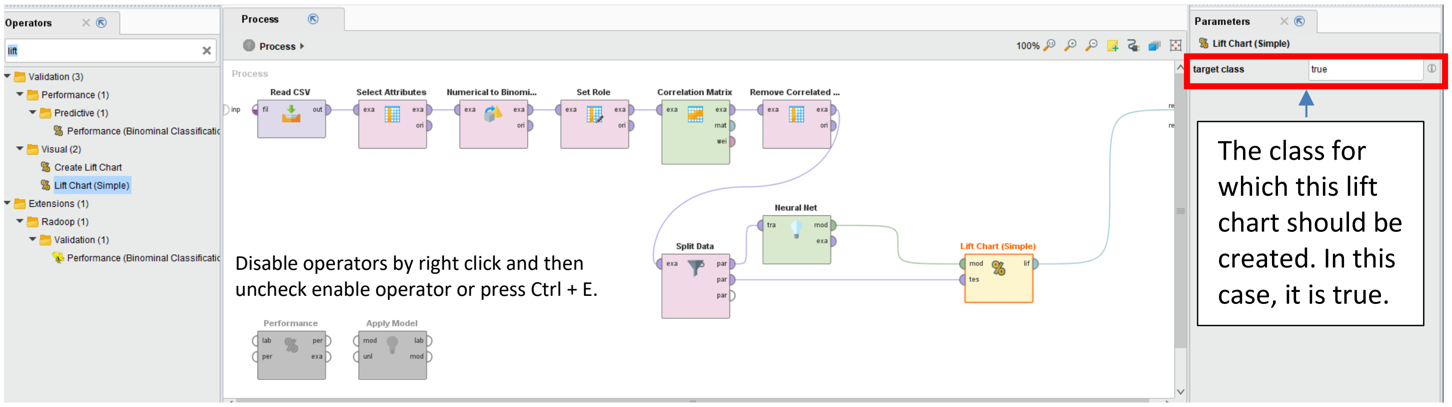
7.1 Create a lift chart (simple) for the Neural Net Model.

7.1.1 Open the process Week6Labstep5.5 which you saved earlier by clicking the following icon.



Open a process definition file. (Ctrl+O)

7.1.2 Add a new operator Lift Chart (Simple) and make the following connections (remove or disable any operators which are not used in this process).



Disable operators by right click and then uncheck enable operator or press Ctrl + E.

target class	true

The class for which this lift chart should be created. In this case, it is true.

As indicated in RM Help file, a lift chart shows how much better a machine learning model performs compared with a random guess. It also shows you the point at which the predictions become less useful. This is in particular useful if you can optimize for a cost-benefit ration as it often happens for marketing-related use cases. For example, a lift chart can show you that to reach 80% of your respondents you only need to reach out to 30% of your total address base.

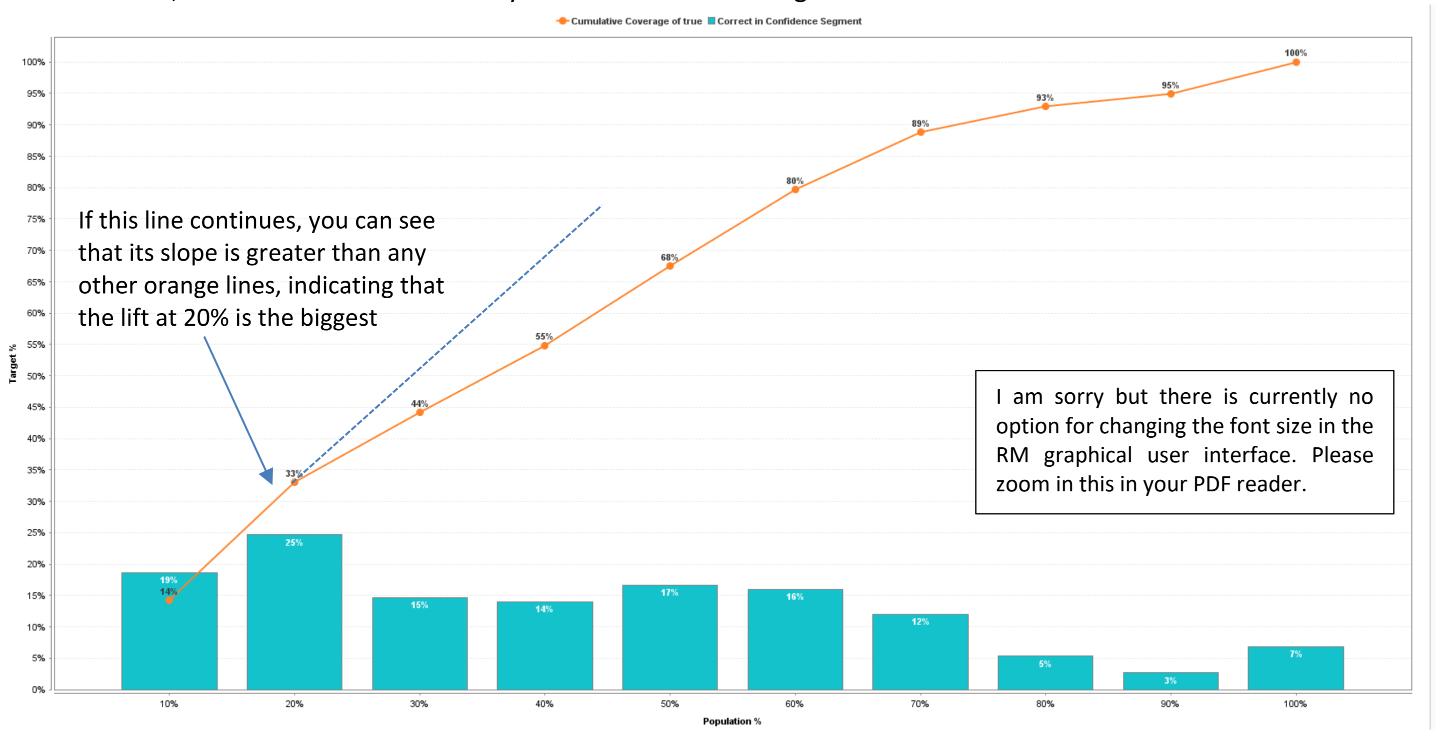
The input of a simple lift chart includes model (this input port expects a prediction model) and test data (the test data to create the lift chart for; it also needs a label attribute to compare with model predictions. It output is lift chart for the given testing data. The simple lift chart has only one parameter, target class. A target attribute has at least two classes (values), so you need to specify which one you are going to focus on. In this case, the target attribute has both true and false. We are interested in those who purchase cellular service, so we type true there.

7.1.3 Save this process as Week6Labstep7.1 and then run it. You will see a lift chart similar (yours may not be the same with mine due to sample selection) to the one in the next page.

How to understand and interpret the simple Lift Chart?

The simple lift chart shows you 10 bins for your test data (in total, the testing dataset has 4985*30%=1496 examples). Each bin is filled with decreasing confidence of the model for the target class. That means that the examples with highest confidence values are in the first bin, then in the second, and so on.

The chart consists of two parts. The bars show you the correct percentage for the target class. For example, if the first bar in the lift chart shows 19%, this means that 19% of all examples in this confidence bin are actually from the desired target class (customers purchasing cellular service). The second part of the chart is a line which shows you the cumulative coverage of the target class if you would consider only examples of at least the confidence of the corresponding bar or higher. A value of 44% above the third bar for example means that you covered 44% of the desired target class at that point. But the third bar only represents 30% of your total population. That means that this model would correctly identify 44% of the target with only using 30% of the total population (the 30% with the highest confidence for this class). In contrast to this, a random model would only achieve 30% of the target class.



Based on this lift chart, you can also compute the lift at each point. For example, the lift at 20% is 33%/20%=1.65, which is the highest lift in this case.

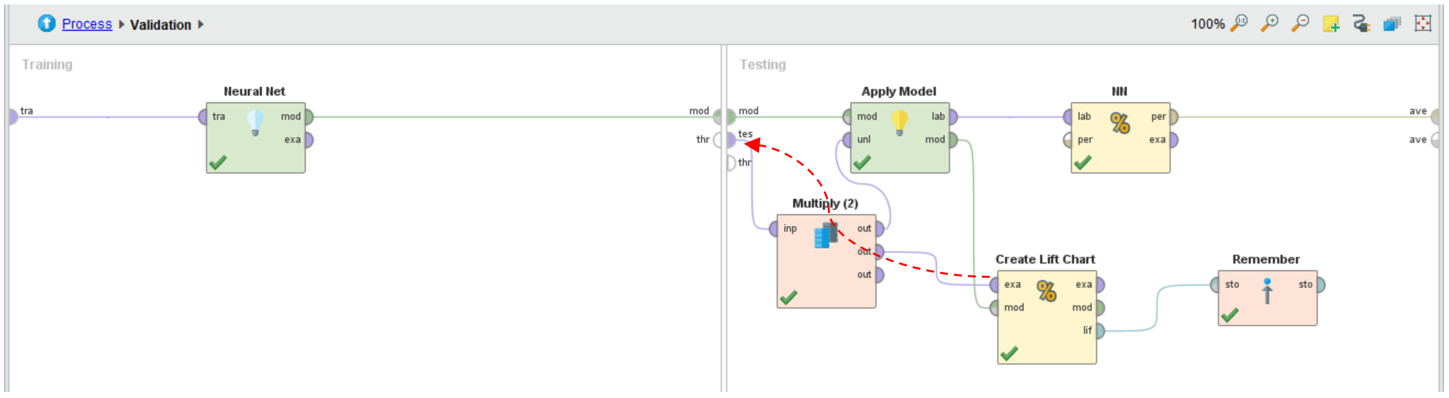
7.1.4 Follow the same procedure to create a simple lift chart for the Naïve Bayes model (you just need to replace the Nerual Net opertor by the Naive Bayes operator). Take a look at the simple lift chart for the Naïve Bayes model.

You may wonder how can we create a customized lift chart and deliver confusion matrix for multiple DM models at the same time. The following steps will help you with this issue.

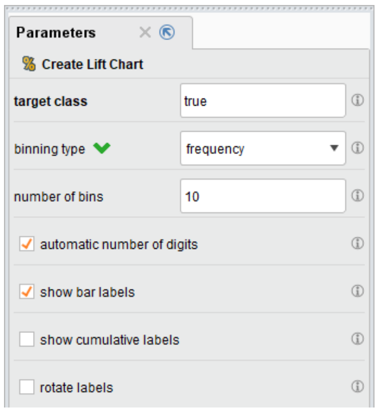
7.2 Create lift charts for multiple DM models

7.2.1 Open the process Week6Labstep6, which you saved ealier.

7.2.2 Double click the Validation operator (you will see what you had in Step 6.3) and then add the Create Lift Chart operator in the testing subprocess (again, when you try to connect the exa port of Create Lift Chart to the tes port, a new operator Multiply is generated.



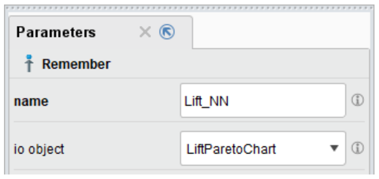
7.2.3 Set the paramters of the operator <u>Create Lift Chart</u> as the right screenshot shows. The <u>Create Lift Chart</u> operator creates a lift chart based on a Pareto plot for the discretized confidence values of the given ExampleSet and model. The model is applied on the ExampleSet and a lift chart is produced afterwards. Please note that any predicted label of the given ExampleSet will be removed during the application of this operator. In order to produce reliable results, this operator must be applied on **<u>data that has not been used to build the model</u>**; otherwise the resulting plot will be too optimistic. The lift chart measures the effectiveness of models by calculating the ratio between the result obtained with a model and the result obtained without a model. The result obtained without a model is based on randomly selected records.


Parameters

Create Lift Chart

target class	true
binning type	frequency
number of bins	10
✓ automatic number of digits	
✓ show bar labels	
☐ show cumulative labels	
☐ rotate labels	

7.2.4 Add a new operator <u>Remember</u> (see the process in Page 9). This operator can be used to store the input object into the object store of the process under the specified name. The name of the object is specified through the name parameter. The io object parameter specifies the class of the object. The stored object can later be restored by the Recall operator by using the same name and class (i.e. the name and class that was used to store it using the Remember operator). There is no scoping mechanism in RapidMiner processes therefore objects can be stored (using <u>Remember</u> operator) and retrieved (using <u>Recall</u> operator) at any nesting level. But care should be taken that the execution order of operators is such that the <u>Remember</u> operator for an object always executes before the <u>Recall</u> operator for that object. The combination of these two operators can be used to build complex processes where an input object is used in completely different parts or loops of the processes.

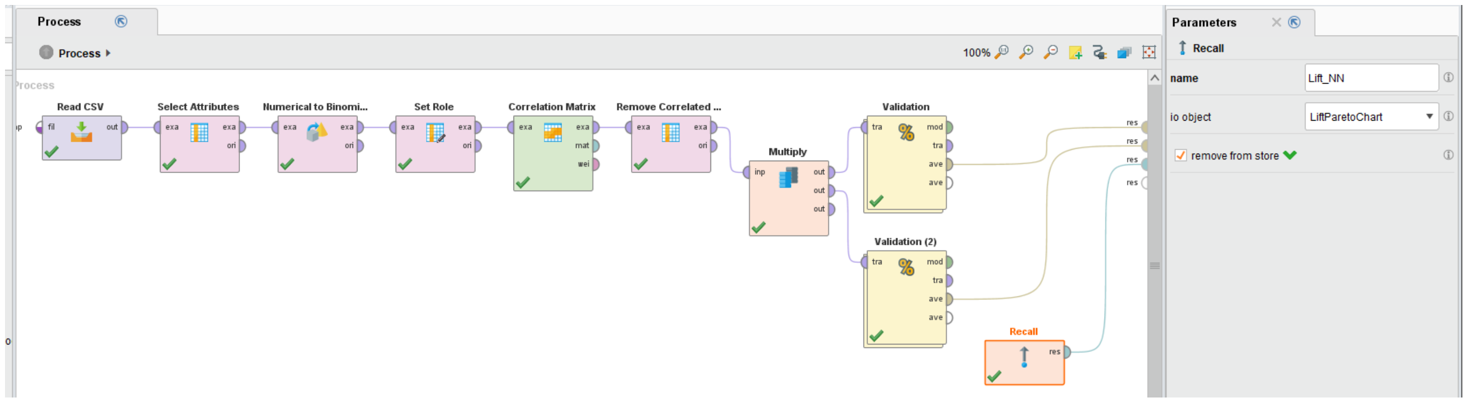

Parameters

Remember

name	Lift_NN
io object	LiftParetoChart

7.2.5 Set the paramters of the <u>Remember</u> operator as the right screenshot shows: name=Lift_NN and io object = LiftParetoChart.

7.2.6 Go back to your main process and add a <u>Recall</u> operator and set its paramters as below and connect its res port to the result. Doing so, you will retrieve the lift chart Lift_NN, which will be shown in your result. The Recall operator can be used for retrieving the specified object from the object store of the process. The name and class of the object are specified when the object is stored using the Remember operator. The same name (in *name* parameter) and class (in *io object* parameter) should be specified in the Recall operator to retrieve that object. The same stored object can be retrieved multiple number of times if the *remove from store* parameter of the Recall operator is not set to true.

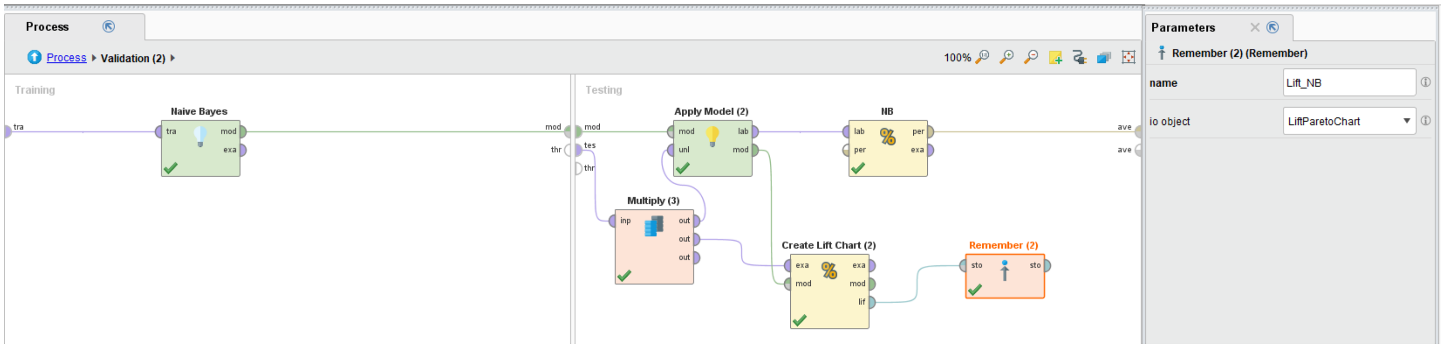


Parameters

Recall

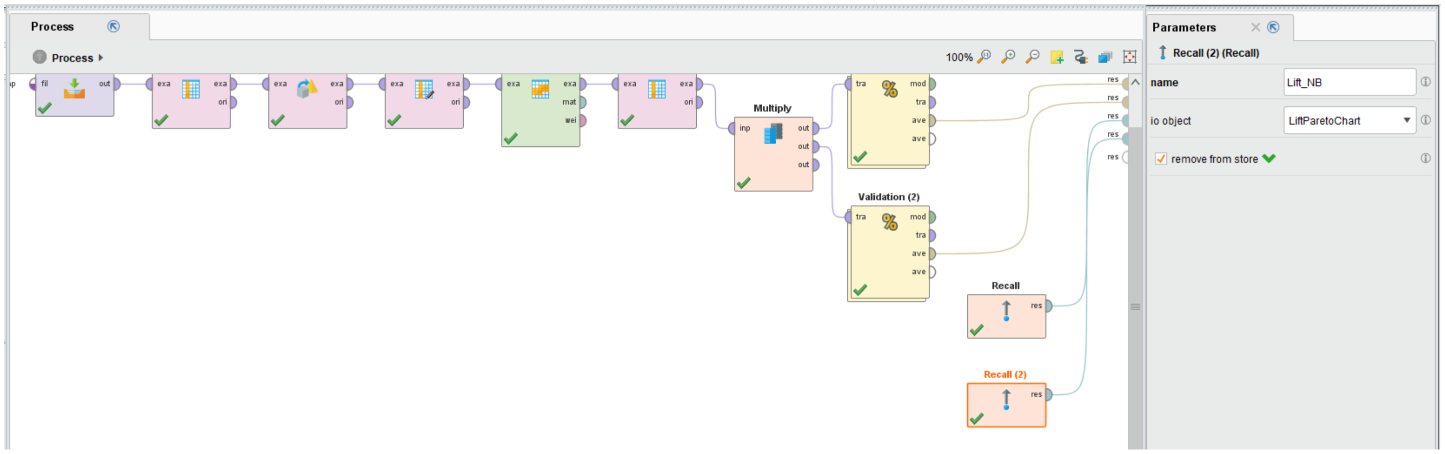
name	Lift_NN
io object	LiftParetoChart
✓ remove from store	

7.2.7 Repeat steps 7.2.2 to 7.2.6 for the second Validation operator (for the Naïve Bayes model). The following two screenshots may help you with this step. Make sure you have the correct connections and parameters.



name: Lift_NB

io object: LiftParetoChart



name: Lift_NB

io object: LiftParetoChart

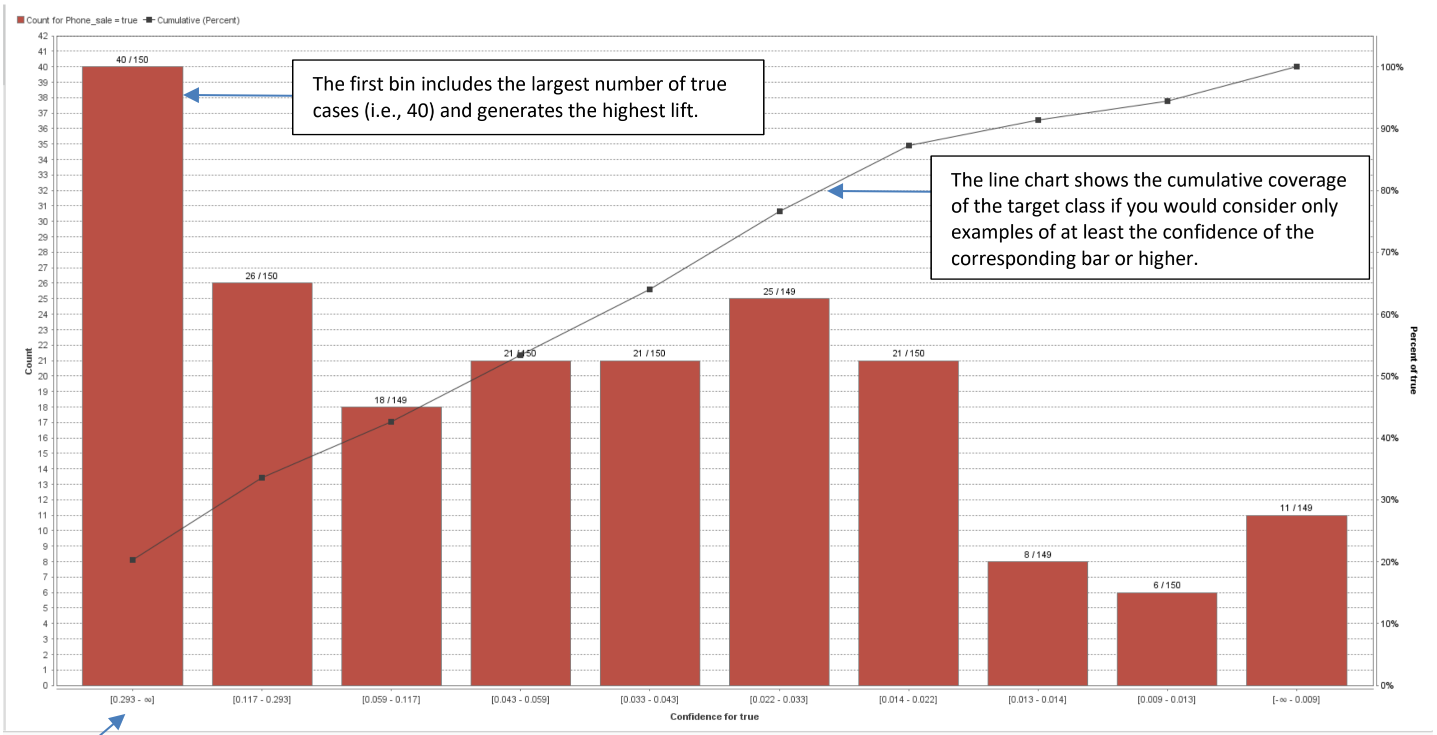
✓ remove from store

7.2.8 Save the process as <u>Week6Labstep7.2</u> and run it. You will receive two lift charts and two confusion matrices: one pair for Neurel Net and one pair for Naïve Bayes (Of course, you can rename the operators <u>Create Lift Chart</u> so that you can clearly know which lift chart belongs to which DM model).

7.2.9 Observe the two lift charts and you will find that they are different from the simple lift charts we created in Step 7.1. Take a look at the two lift charts and then think about which model (Naïve Bayes vs. Neural Network) performs better if our goal is to find out the people who purchase cellular service and why.

How to understand and interpret the lift chart?

Similar to the simple lift chart, the lift chart shows you 10 bins for your test data. Each bin is filled with decreasing confidence of the model for the target class. That means that the examples with highest confidence values are in the first bin, then in the second, and so on. The x-axis presents ten bins of confidence for true at the descending order, the primary y-axis (the bar chart) shows the count of true cases, and the secondary y-axis (the line chart) describes the accumulative percent of true cases. If you add the number of cases in each bin, you will get 1496. RM assigns each bin the almost same number of cases (149 or 150).



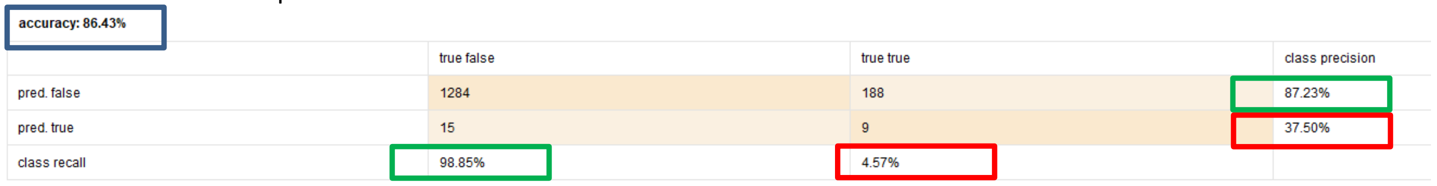
The first bin includes the largest number of true cases (i.e., 40) and generates the highest lift.

The line chart shows the cumulative coverage of the target class if you would consider only examples of at least the confidence of the corresponding bar or higher.

The minimum confidence in the first bin (top 10% confidence) is 0.293, indicating that the DM model did not perform very well in identifying true cases, even though it is slightly better than random guess. That is why the recall and precision are quite low.

Appendix: How to use the confusion matrix correctly?

This link (Simple guide to confusion matrix terminology) is very useful for you to understand the confusion matrix. One thing this post does not mention clearly is that we need to know our main interest. Are we interested in predicting Yes or No (if our target variable has only two values: Yes, No)? If Yes, the count of (Yes, Yes) will be True Positive; otherwise, (No, No) will be our true positive. The screenshot in RapidMiner displays the confusion matrix different from what we have in our lecture or the link I provide above.



accuracy: 86.43%

	true false	true true	class precision
pred. false	1284	188	87.23%
pred. true	15	9	37.50%
class recall	98.85%	4.57%	

The horizontal dimension is actual class (that is why it has true before true and false).
The vertical dimension is predicted class (that is why it has pred before true and false).
If you rotate it by 90 degrees counterclockwise, it will be same with the one in our lecture (suppose that you are interested in predicting True). The reason why RapidMiner displays two precisions and two recalls is that data scientists may have different research interests:

A. If you are interested in predicting True (purchase the cellular services):

- True positives (TP): These are cases in which we predicted True (purchase the cellular services), and they do purchase the cellular services.
- True negatives (TN): We predicted false, and they don't purchase the cellular services.
- False positives (FP): We predicted true, but they don't actually purchase the cellular services.
- False negatives (FN): We predicted false, but they actually do purchase the cellular services.

	True false	True true	Total
Pred false	1284 (TN)	188 (FN)	1472
Pred true	15 (FP)	9 (TP)	24
	1299	197	

Recall = TP/(FN+TP)=9/(188+9)=4.57%; Precision =TP/(TP+FP)=15/(15+9)=37.50% (see the numbers in red boxes in the confusion matrix above generated by RapidMiner)

B. If you are going to predict False (not purchase the cellular services):

- True positives (TP): These are cases in which we predicted False (not purchase the cellular services), and they do not purchase the cellular services.
- True negatives (TN): We predicted true, and they actually purchase the cellular services.
- False positives (FP): We predicted false, but they actually do purchase the cellular services.
- False negatives (FN): We predicted true, but they actually do not purchase the cellular services.

	True false	True true	Total
Pred false	1284 (TP)	188 (FP)	1472
Pred true	15 (FN)	9 (TN)	24
	1299	197	

Recall = TP/(FN+TP)=1294/(1294+15)=98.85%; Precision =TP/(TP+FP)=1284/(1294+188)=87.23% (see the numbers in green boxes in the confusion matrix above generated by RapidMiner)

The above indicates that the model performs better to predict “False” (not purchase) than to predict “True” (purchase). However, predicting “Yes” in this case is more valuable for the companies in this case. Take a look at the accuracy. No matter what you are going to predict (True or False), it is always the same. That’s why RapidMiner put accuracy at the top (see the number in the blue box in the confusion matrix above generated by RM).