Naïve Bayes Classification

Naïve Bayes is a simple classification technique wherein the Naïve assumption that the effect of the value of each variable is independent from all other variables is made. For each independent variable, the algorithm then calculates the conditional likelihood of each potential class given the particular value for that variable and then multiplies those effects together to determine the probability for each class. The class with the highest probability is returned as the predicted class.

Even though Naïve Bayes makes a naïve assumption, it has been observed to perform well in many real-world instances. It is also one of the most efficient classification algorithms with regard to performance.

For more detailed information on Naïve Bayes, please consult this site [3].

This R Script has two functional modes:

- Training creates a model which is persisted as an .Rdata file and returns predictions.
- Scoring uses the model created during training to return predictions..

How to Deploy to MicroStrategy:

Prerequisite: Please follow the instructions in the R Integration Pack User Guide [1] for configuring your MicroStrategy environment with R and that the R Script functions have been installed in your MicroStrategy project(s).

1) Download the NaiveBayes.R file from the R Script Shelf [2].
2) From the R console, run the NaiveBayes.R script to verify the script runs correctly. For details, see the "**Running from the R Console**" section below.
3) Cut-and-paste the appropriate metric expression below in any MicroStrategy metric editor. Map the arguments to the appropriate MicroStrategy metrics. A description of each output is shown below.
4) Use the new metric in reports, dashboards and documents.

Metric Expressions:

1) **Class:** Returns the predicted class as a string:

   For training, use this metric expression:

```
RScript<_RScriptFile="NaiveBayes.r",_InputNames="Target,Vars",StringParam9="Nai
veBayes",BooleanParam9=TRUE, NumericParam1=1>(Target, Vars)
```

   For scoring, use this metric expression:

```
RScript<_RScriptFile="NaiveBayes.r",_InputNames="Target,Vars",StringParam9="Nai
veBayes",BooleanParam9=FALSE, NumericParam1=1>(Target, Vars)
```

2) **ClassID:** Returns the predicted class as a number:

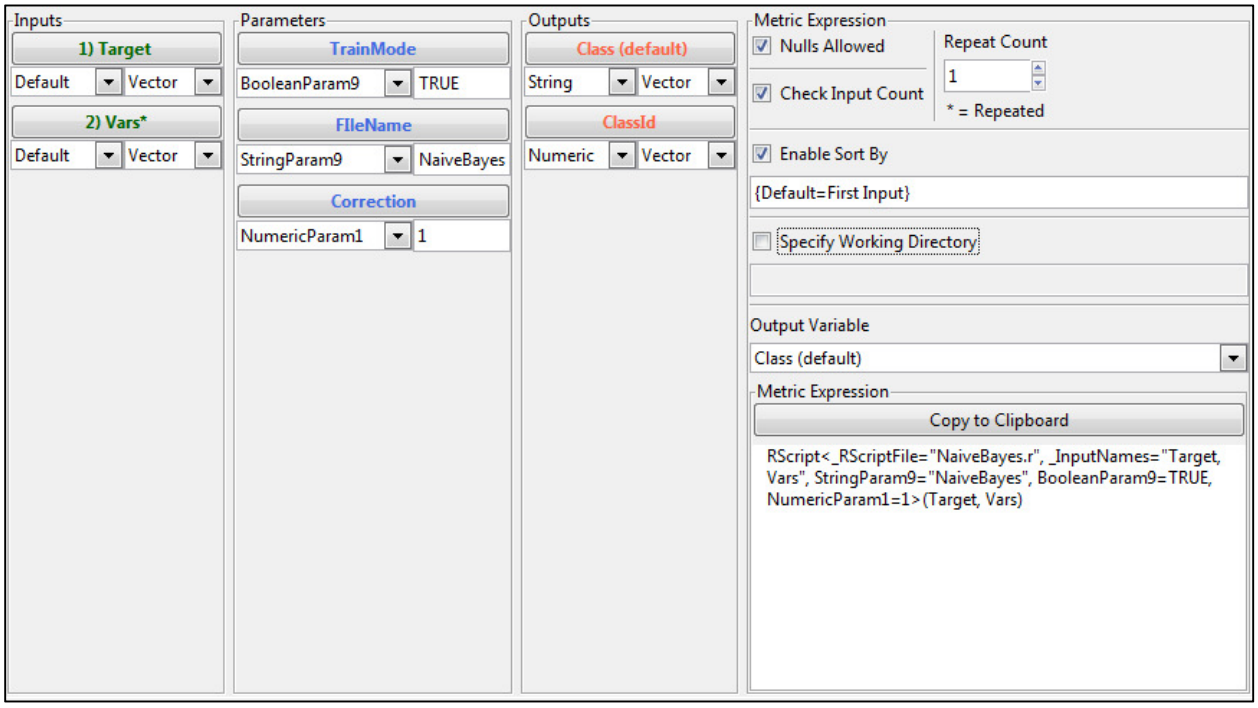
   For training, use this metric expression:

```
RScript<_RScriptFile="NaiveBayes.r",_InputNames="Target,Vars",[_OutputVar]="Cla
ssId",StringParam9="NaiveBayes",BooleanParam9=TRUE,NumericParam1=1>(Target,
Vars)
```

   For scoring, use this metric expression:

```
RScript<_RScriptFile="NaiveBayes.r",_InputNames="Target,Vars",[_OutputVar]="Cla
ssId",StringParam9="NaiveBayes",BooleanParam9=FALSE,NumericParam1=1>(Target,
Vars)
```

Analytic Signature:



Inputs:

Target: The observed class of each record. When training, this input represents the value we are trying to predict. When scoring, this value is generally not known so the script does not rely on the metric passed in. Nevertheless, this input is required to maintain the structure of the expression so map target to any metric with non-null values (i.e. revenue) when scoring.

Vars: The independent variables that are used as the basis for generating predictions. Since the Vars argument is a repeated input, it can be mapped to any number of MicroStrategy metrics. In this way, we enable Naïve Bayes to consider any number of variables when generating predictions.

Note: **Repeated inputs must all be the same type (i.e. all variables must be numeric or all variables must be strings)**

Parameters:

TrainMode: Uses BooleanParam9 with a default of TRUE. When set to TRUE, the R script will train (i.e. create) and save a NaiveBayes model using the dataset provided. When set to FALSE, the R script will score the records on the report against a pre-existing model. In practice, one first trains a model and once that

model has been validated, scores unknown records against that model. For more information on training/scoring, please see this document.

FileName: Uses StringParam9 with a default of "NaiveBayes". This parameter specifies the file name where the trained model is stored in an Rdata file . Please note the R Script automatically appends the ".Rdata" file extension to this file name. This parameter is only used when TrainMode=TRUE; when TrainMode=FALSE, no Rdata file is saved.

Correction: Uses NumericParam1 with a default of 1. This parameter controls the correction that is used when the conditional probability of a class given a particular value for a variable is zero. For instance, if the class of interest was whether a person is a fan of the Cowboys or Redskins and no person in our dataset is a Cowboys fan living in Washington D.C., then the probability assigned to the class "Cowboys" for an individual living in Washington D.C. would be zero, regardless of the values for all other variables we are considering. That can be corrected using the Laplace correction which assigns the conditional probability m/(n+m) rather than 0. So, if we use a correction of 1 and there are 100 people from Washington D.C. in our dataset, we would assign the conditional probability 1/101 for a person being a Cowboys fan given a residence in Washington D.C.

Outputs:

Class: A string value representing the predicted class for that record. Used when classes are represented by strings.

ClassID: A numeric value representing the predicted class for that record. Used when classes are represented by numbers.

Additional Results Generated by the R Script:

One file is stored in the working directory:

Rdata File: This file persists the state of several objects from the R environment for later inspection, analysis, and reuse, including df (a data frame containing the data read in from MicroStrategy), model (the Naïve Bayes model object), and Class (the predicted class of each record). This file is loaded when scoring new records to reference the model that was trained based on past history.

Running from the R Console:

In addition to processing data from MicroStrategy during execution of a report or dashboard, the R script is also configured to run from the R console. Running the script for the R Console verifies that the script is functioning as expected, a good practice when initially deploying this analytic to a new system (for more details, see "Configuring dual execution modes" in [1]).

When run from the R Console, if the script is executing properly, a "Success!" message will appear in the console. If a "Success!" message does not appear, then please note the error in order to take appropriate action. For common pitfalls, please consult the **Troubleshooting** section below.

Troubleshooting:

This section covers certain situations you might encounter but it's not intended as a comprehensive list of possible errors.

If an error occurs, the report may fail with an error message, or nulls returned as the output. In these cases, please refer to the RScriptErrors.log file generated for further guidance and the DSSErrors.log. Please consult the User Guide [1] and the R documentation for additional guidance.

The script will attempt to install the required R package. If the package is not successfully installed, you can install using the R console using the command:

```
install.pacakges("e1071", repos="http://cran.rstudio.com/")
```

- **e1071**: This R package contains functions that support the creation of Naïve Bayes classification models.

If a mix of string and numeric variables is passed in to the Vars argument, the report will fail with an error message indicating that a variable with unexpected type was passed in. This can be remedied by using all strings or all numeric variables corresponding to the Vars argument.

If, when scoring, the error message "cannot open the connection" is returned, that means that the R script cannot load the Rdata file from the specified location. Please make sure that the value passed in to StringParam9 is an existing file. For instance, if StringParam9 is set to NaiveBayes, then there must be a file NaiveBayes.Rdata in the working directory. Please consult the User Guide [1] for more information.

Example (using MicroStrategy Tutorial Project):

In order to combat an escalation in the number of customers who are churning, or leaving the company in the midst of their contracts, a large Telco organization wants to create and deploy a NaiveBayes model that predicts which customers are most likely to churn. By proactively identifying at-risk customers, the organization can implement different measures aimed at assuaging these disgruntled customers.

Using historical information about the characteristics of customers and whether they've churned or not, we'll create a metric NaiveBayes Trainer with the following definition:

```
RScript<[BooleanParam9]=True,[NumericParam1]="1",[StringParam9]="NaiveBayes",[_OutputVar]="ClassId",[_RScriptFile]="NaiveBayes.r">(TelcoChurn,[Household Count ID], [Income Bracket ID], HelpdeskCalls, DroppedCalls)
```

Where Target is mapped to TelcoChurn and Vars consists of the metrics Household_Count_ID, Income_Bracket_ID, HelpdeskCalls, and DroppedCalls.

Then, open up report "1—Training Report for Telco Churn Predictor (Tree)" and add the NaiveBayes Trainer metric to the report. By adding this metric to the report, we train the model and store it in a file called NaiveBayes.Rdata. Here is how the report should look after all metrics not included in our model are removed from the grid:

Metrics / Customer	Household Count	Income Bracket	DroppedCalls	HelpdeskCalls	TelcoChurn	Training metric for 'Telco Churn Predictor'	Naive Bayes Trainer	DT Correct?	NB Correct
Total	--	--	**858**	**829**	**374**	**185**	**184**	**1,235**	**1,072**
7	3	20K-30K	1	1	0	0	0	Yes	Yes
14	3	30K-40K	0	0	0	0	0	Yes	Yes
21	2	80K-90K	0	0	1	0	0	No	No
28	2	60K-70K	0	1	0	0	0	Yes	Yes
35	2	80K-90K	1	0	0	0	0	Yes	Yes
42	2	60K-70K	1	2	0	0	0	Yes	Yes
49	6	50K-60K	1	0	1	1	0	Yes	No
56	1	40K-50K	1	1	0	0	0	Yes	Yes
63	Over 6	50K-60K	0	1	0	0	0	Yes	Yes
70	4	50K-60K	1	0	0	0	0	Yes	Yes
77	2	20K-30K	0	0	0	0	0	Yes	Yes
84	5	20K-30K	2	2	1	1	1	Yes	Yes
91	6	30K-40K	1	1	0	0	0	Yes	Yes
98	2	Over 100K	0	1	0	0	0	Yes	Yes
105	3	30K-40K	0	0	0	0	0	Yes	Yes
112	3	60K-70K	1	1	0	0	0	Yes	Yes
119	6	50K-60K	1	1	1	1	0	Yes	No
126	2	60K-70K	0	0	1	0	0	No	No
133	6	50K-60K	2	2	1	1	1	Yes	Yes
140	3	60K-70K	0	1	0	0	0	Yes	Yes
147	5	40K-50K	0	1	1	1	0	Yes	No
154	3	80K-90K	1	1	0	0	0	Yes	Yes
161	2	70K-80K	0	1	0	0	0	Yes	Yes
168	1	40K-50K	0	0	0	0	0	Yes	Yes
175	5	40K-50K	1	2	1	1	1	Yes	Yes
182	2	50K-60K	0	0	0	0	0	Yes	Yes
189	2	40K-50K	0	1	0	0	0	Yes	Yes
196	1	50K-60K	0	0	0	0	0	Yes	Yes
203	5	20K and Under	0	0	0	0	0	Yes	Yes
210	3	50K-60K	0	0	1	0	0	No	No

As you can see, the Naïve Bayes algorithm does a significantly worse job than the native decision tree of classifying customers in this dataset by correctly classifying only 1,072 out of 1,428 customers. The native decision tree correctly classified 1,235 customers. However, since this model appears adequate, we can score the model against the active customers that we have. In this case, we utilize the metric expression for scoring to create a metric called Naïve Bayes Predictor:

```
RScript<[BooleanParam9]=False,[NumericParam1]="1",[StringParam9]="NaiveBayes",[_Output
Var]="ClassId",[_RScriptFile]="NaiveBayes.r">(Revenue,[Household Count ID], [Income
Bracket ID], HelpdeskCalls, DroppedCalls)
```

We then place the Naïve Bayes Predictor on the report "4—Telco Churn Predictor (Tree)". Here is the scoring report:

Patient	Metrics	Telco Churn Predictor (Tree)	Telco Churn Propensity (Tree)	Naive Bayes Predictor
Total		**28**	**15%**	**569**
Count		**7350**	**7350**	**7350**
Aadland	Miko	0	15%	1
Aadland	Constant	0	15%	0
Aafedt	Wendy	0	15%	0
Aalgaard	Kenney	0	15%	0
Aarestad	Benjamine	0	15%	0
Aarnink	Marlan	0	15%	0
Aaron	Ferrell	0	15%	0
Aaronson	Maxwell	0	15%	0
Aasen	Beatrice	0	15%	1
Aba	Blain	0	15%	0
Aba-Bulgu	Leslie	0	15%	1
Abad	Geoffrey	0	15%	0
Abad	Bekir	0	15%	0
Abadilla	Lennie	0	15%	0
Abajian	Lorin	0	15%	0
Abarbanel	Hassam	0	15%	0
Abarca	Hugh	0	15%	0
Abatemarco	May	0	15%	0
Abbasi	Dwayne	0	15%	0
Abbasi	Donnis	0	15%	0
Abbruscato	Shaun	0	15%	0
Abdala	Stacy	0	15%	0
Abdala	Vivian	0	15%	0
Abdallah	Erling	0	15%	0
Abdullah	Candice	0	15%	0
Abdullah	Clarkelle	0	15%	0
Abdyusheva	Dominique	0	15%	0
Abel	Edward	0	15%	0
Abeleda	Devon	0	15%	0
Abeles	Rolland	0	15%	0

As can be seen, the Naïve Bayes algorithm identifies 569 customers who are likely to churn, as opposed to the decision tree which only identifies 28. With knowledge of the customers who are at-risk, the Telco company can now intelligently target and assuage the correct customers with promotions, combating the escalation in customer churn that is posing a threat to the organization.

References:

1) MicroStrategy R Integration Pack User Guide: https://rintegrationpack.codeplex.com/documentation
2) R Script Shelf: http://rintegrationpack.codeplex.com/wikipage?title=R%20Script%20%22Shelf%22&referringTitle=Home
3) http://en.wikipedia.org/wiki/Naive_Bayes_classifier

