

# DATA MINING

## ASSIGNMENT 4

Ali Gholami

Department of Computer Engineering & Information Technology  
Amirkabir University of Technology

<https://aligholamee.github.io>

[aligholami7596@gmail.com](mailto:aligholami7596@gmail.com)

### Abstract

*RapidMiner* provides data mining and machine learning procedures including: data loading and transformation (Extract, transform, load (ETL)), data preprocessing and visualization, predictive analytics and statistical modeling, evaluation, and deployment. In this report, we'll apply some useful tools of *RapidMiner* on real world data mining problems.

**Keywords.** *RapidMiner, Data Visualization, Machine Learning, Statistical Modeling, Predictive Analytics.*

## 1 Preprocessing

### Impute Missing Values

From operators, import an *Input Missing Values* operator and change the *attribute filter type* to *subset*. Check the *other-social-network* and *online gaming* from the attribute types. Enter operator design section and import an *Lazy K-NN* predictive model. Run the process and compare your results.

### Results

Figure 1.1 illustrates the results before and after performing the missing value imputation. As can be seen, the *online-gaming* column's missing values is filled.

Read_News	Online_Sho...	Online_Gam...	Facebook
Y	N	N	Y
Y	N	N	Y
Y	Y	?	Y
N	Y	N	N
Y	Y	N	Y
Y	N	Y	N
?	Y	?	Y
Y	?	?	Y
N	Y	N	N
Y	?	Y	Y
Y	N	N	Y

Read_News	Online_Sho...	Online_Gam...	Facebook
Y	N	N	Y
Y	N	N	Y
Y	Y	Y	Y
N	Y	N	N
Y	Y	N	Y
Y	N	Y	N
?	Y	Y	Y
Y	?	Y	Y
N	Y	N	N
Y	?	Y	Y
Y	N	N	Y

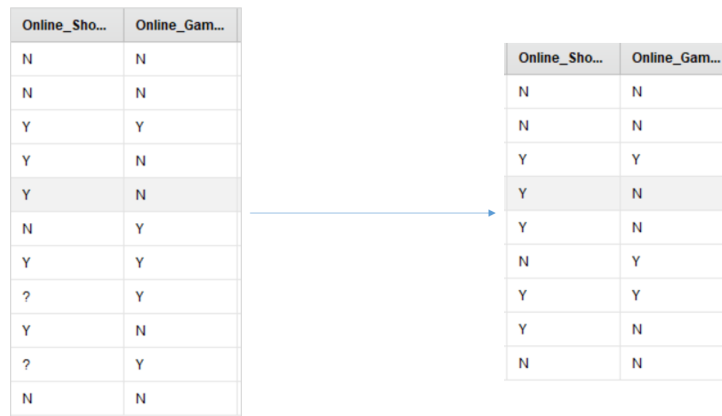
Figure 1.1: Illustration of how missing values are filled.

## Example Filter

Using *Blending Filters* from the *operators* section, we can apply an attribute *type* filtering. Under operators, blending, examples, filters and then filter examples, right click on the operator and change the condition class parameter to *attribute-value-filter*. Change the parameter string to *Online\_Shopping = .* and analyze the results.

## Results

Examples which their value of *Online\_Shopping* is missing will be removed. Number of samples will be reduced from 11 to 9 examples. The missing values under *Online\_Shopping* are gone.



Online_Sho...	Online_Gam...
N	N
N	N
Y	Y
Y	N
Y	N
N	Y
Y	Y
?	Y
Y	N
?	Y
N	N

Online_Sho...	Online_Gam...
N	N
N	N
Y	Y
Y	N
Y	N
N	Y
Y	Y
Y	N
N	N

Figure 1.2: Illustration of how samples with missing values are filtered.

## Sampling

Use sampling to choose a subset of the examples. Initialize this operator to choose half of the dataset. Explain the parameters you choose and justify the results.

## Results

Out of 9 examples from the previous section, we'll set the sample size to 5.

## Store Results

Choose the appropriate operator to store the result in *.csv* or *.xlsx* format.

## Results

Saved file is under *preprocess* directory under *src* folder. Figure 1.3 illustrates the final model.

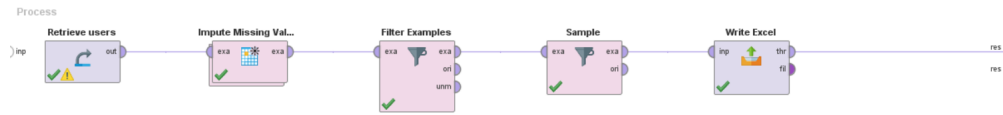


Figure 1.3: Illustration of final preprocessing model on *users* dataset.

## 2 Modeling a Decision Tree

Use *dt\_train* and *dt\_score* datasets for this section.

### Attributes Blending

Under blending, attributes, names and roles, add *set role* to the process. In both of the operators change *attribute name* to *User\_ID* and *target role* to *id*. Then connect training input to one and score input to another. Explain the purpose of this task?

#### Results

The role of an Attribute describes how other Operators handle this Attribute. The default role is regular, other roles are classified as special. An ExampleSet can have many special Attributes, but each special role can only appear once. If a special role is assigned to more than one Attribute, all roles will be changed to regular except for the last Attribute. The different types of roles are explained below in the parameter section. An Attribute with the **id** role acts as an identifier for the Examples. Different Blending Operators (Join, Union, Transpose, Pivot, ...) uses the id Attribute to perform their tasks.

### Attributes Blending 2

Add another *set role* to the process. Change the attribute name to *eReader\_Adoption*. Change the attribute type to *label*. Explain the task.

#### Results

This task provides a new column (which will be used for the test data) as a prediction column. Predicted labels will be filled in this column. This is also called the *class* or *target attribute*.

### Model Training

Under modeling, predictive, choose decision tree model. Run the model and explain the results.

#### Results

Models starts to be trained after hitting the start process button. The learned decision tree can be found under **Results, Graph** section. A part of this tree is given in figure 2.1. Final model is provided in figure 2.2. Prediction is classified into 4 classes. For each column (category), the confidence of the

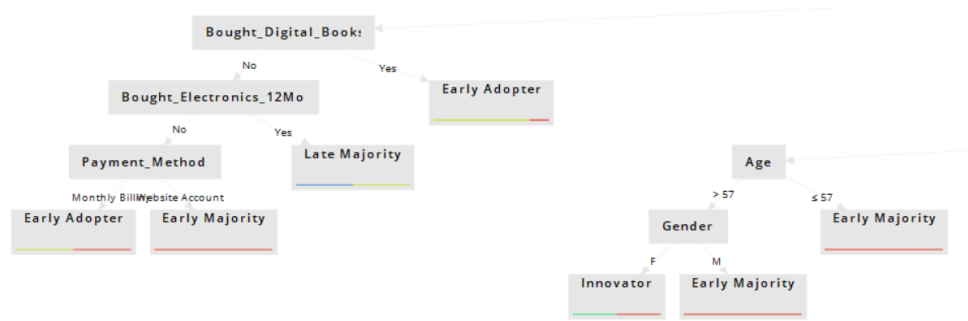


Figure 2.1: Illustration of a part of the Decision Tree model trained on the *DT\_train* data.

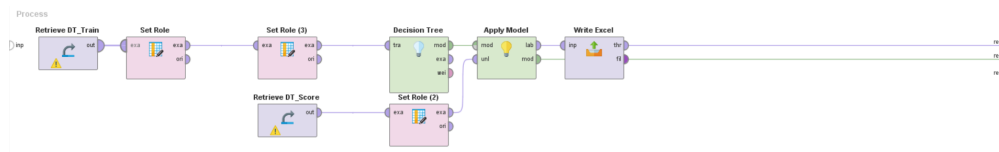


Figure 2.2: Illustration of the Decision Tree model trained on the *DT\_train* data.

prediction. The label assigned to each sample will be the category with highest value of confidence. Figure 2.3 demonstrates a part of examples with their confidence results after applying the Decision Tree model. Process data can be found under *dt-results*.

confidence(Late Majority)	confidence(Innovator)	confidence(Early Adopter) ↑	confidence(Early Majority)
0.250	0	0	0.750
0.250	0	0	0.750
0.500	0.500	0	0
0	0	0	1
0	0	0	1
0	0	0	1
0.083	0.028	0	0.889
0	0	0	1

Figure 2.3: Demonstration of the Decision Tree prediction on the *DT\_score* data.

## Gini Indexing

Change the decision tree criterion to *gini-index*. Analyze the results.

## Results

Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

### 3 Naive Bayes Modeling

Reform a model based on the Naive Bayes rule using the steps provided in the previous section. The goal attribute is *Second Heart Attack*. Perform training and scoring the *NB\_train* and *NB\_score* data.

#### Results

Prediction results is stored under *src* directory in *nb-results* folder.

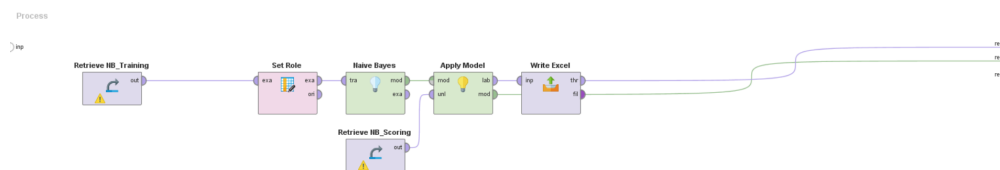


Figure 3.1: Illustration of Naive Bayes model trained on the *NB-train* data.

### 4 ROC Curve

We'll use the *NB-training* data for this section. From *operations* section, find the *Compare ROCs* operation. Add Decision Tree & Naive Bayes (plus another random model for classification) in the ROC operator and complete the connections.

- How are **ROC** curves plotted?

To draw a ROC curve, only the true positive rate (TPR) and false positive rate (FPR) are needed (as functions of some classifier parameter). The TPR defines how many correct positive results occur among all positive samples available during the test. FPR, on the other hand, defines how many incorrect positive results occur among all negative samples available during the test. A ROC space is defined by FPR and TPR as x and y axes respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). Since TPR is equivalent to sensitivity and FPR is equal to  $1 - \text{specificity}$ , the ROC graph is sometimes called the sensitivity vs  $(1 - \text{specificity})$  plot. Each prediction result or instance of a confusion matrix represents one point in the ROC space. The best possible prediction method would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing 100% sensitivity (no false negatives) and 100% specificity (no false positives). The (0,1) point is also called a perfect classification. A completely random guess would give a point along a diagonal line (the so-called line of no-discrimination) from the left bottom to the top right corners (regardless of the positive and negative base rates). An intuitive example of random guessing is a decision by flipping coins (heads or tails). As the size of the sample increases, a random classifier's ROC point migrates towards (0.5,0.5). The diagonal divides the ROC space. Points above the diagonal represent good classification results (better than random), points below the line poor results (worse than random). The ROC comparison results is given in the figure 4.1.

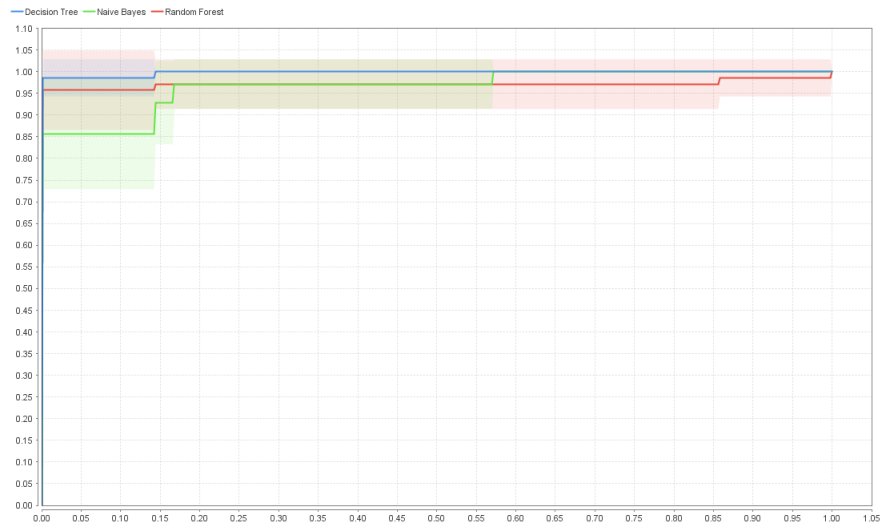


Figure 4.1: ROC Curve comparison for 3 models trained on *NB-training* dataset.

- What is the relationship between *confidence* column and the ROC curve?

A confidence interval gives an estimated range of values which is likely to include an unknown population parameter, the estimated range being calculated from a given set of sample data. (Definition taken from Valerie J. Easton and John H. McColl's Statistics Glossary v1.1). The higher AUC is (sometimes AUC is referred to as AUROC), the better model will be. Higher AUCs yields in more TPR and less FPR. Thus, each model with higher TPR and less FPR (practically smaller AUCs) will predict the TRUE labels as positive and FALSE labels as negative more often. This, simply means that the confidence (mathematical assurance for being correct) is higher for the models with higher AUCs.

- What is **AUC** and how is that analyzed?

AUC is an abbreviation for area under the curve. It is used in classification analysis in order to determine which of the used models predicts the classes best. Please refer to the previous section for more information on **AUC**.