Chapter 5

Naïve Bayes Classification I

M. Fareed Akhtar

Fastonish, Australia

5.1 Introduction

This chapter explains the Naïve Bayes classification algorithm and its operator in RapidMiner. The use case of this chapter applies the Naïve Bayes operator on the Credit Approval dataset. The operators explained in this chapter are: Rename by Replacing, Filter Examples, Discretize by Binning, X-Validation, and Performance (Binominal Classification) operator.

The Naïve Bayes algorithm is a simple probabilistic classifier based on applying Bayes' theorem with strong independence assumptions. In simple words, the Naïve Bayes algorithm assumes that the presence of a particular value of an attribute is unrelated to the presence of any other attribute value. For example, a ball may be classified as a tennis ball if it is green, round, and about 2 and half inches in diameter. Even if these properties depend on each other, the Naïve Bayes algorithm considers all of these features to independently contribute to the probability that this ball is a tennis ball.

Let **X** be an example that we want to classify. **X** is described by measurements made on a set of **n** attributes. Let **H** be some hypothesis, such as that the example **X** belongs to a specified class **C**. For classification problems, we want to determine **P(H|X)**, the probability that the hypothesis **H** holds given the example **X**. In other words, we are looking for the probability that example **X** belongs to class **C**, given that we know the attribute description of **X**.

P(H|X) is the posterior probability of **H** conditioned on **X**. For example, suppose our dataset is about whether golf will be played or not in the given weather conditions like temperature and outlook. Thus, temperature and outlook are our attributes. The example **X** represents a particular scenario of weather conditions, for example, a weather condition with 80 degrees temperature and sunny outlook. Suppose that **H** is the hypothesis that golf will be played in the given weather conditions. Then **P(H|X)** reflects the probability that golf will be played in **X** weather condition given that we know the temperature and outlook of weather.

In contrast, **P(H)** is the prior probability, or a priori probability, of **H**. For our example, this is the probability that golf will be played in any given weather, regardless of temperature, outlook, or any other information. The posterior probability, **P(H|X)**, is based on more information, i.e., weather information, than the prior probability, **P(H)**, which is independent of **X**.

Similarly, **P(X|H)** is the posterior probability of **X** conditioned on **H**. That is, it is the probability that the weather, **X**, is sunny and 80 degrees hot, given that we know that golf will be played.

P(X) is the prior probability of **X**. Using our example, it is the probability that a weather condition will have temperature of 80 degrees and sunny outlook in our complete dataset.

P(H), **P(X|H)**, and **P(X)** can be estimated from the given data. Bayes' theorem is useful in that it provides a way of calculating the posterior probability, **P(H|X)**, from **P(H)**, **P(X|H)**, and **P(X)**. Bayes' theorem states that:

P(H|X) = P(X|H) . P(H) / P(X).

How Bayes' theorem is used in the Naïve Bayes classifier is not discussed in this chapter. The use case of the next chapter explains, in detail, the working of the Naïve Bayes classifier.

The Naïve Bayes operator in RapidMiner has only one parameter, i.e., Laplace correction. This expert parameter indicates if the Laplace correction should be used to prevent high influence of zero probabilities. To avoid zero probabilities, it can be assumed that our training dataset is so large that the addition of one to each count that we need would only make a negligible difference in the estimated probabilities. This technique is known as the Laplace correction.

5.2 Dataset

5.2.1 Credit Approval Dataset

This dataset has been taken from the UCI repositories. This dataset can be accessed through this link: `http://archive.ics.uci.edu/ml/datasets/Credit+Approval`.

This dataset concerns credit card applications, therefore, all attribute names and values have been changed to meaningless symbols to protect confidentiality of the data.

5.2.2 Examples

This dataset has 690 examples and there are some missing values in this dataset.

5.2.3 Attributes

This dataset has 16 attributes (including the label attribute). The data set comes with some basic information but the type and role of attributes is set by the user of the dataset. The name, type, and range of attributes are explained below in this short format: '**name** (type): {range}'

1. **A1** (binominal): {b, a}
2. **A2** (real): {real values}
3. **A3** (real): {real values}
4. **A4** (polynominal): {u, y, l, t }
5. **A5** (polynominal): {g, p, gg }
6. **A6** (polynominal): {c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff }
7. **A7** (polynominal): {v, h, bb, j, n, z, dd, ff, o }
8. **A8** (real): {real values}
9. **A9** (binominal): {t, f }
10. **A10** (binominal): {t, f }
11. **A11** (integer): {integer values}
12. **A12** (binominal): {t, f }
13. **A13** (polynominal): {g, p, s }
14. **A14** (integer): {integer values}
15. **A15** (integer): {integer values}
16. **class** (label attribute): This is the class attribute. It has only two possible values: {+, −}. As this attribute has only two possible values, its type should be set to binominal in RapidMiner. The role of this attribute should be set to *label* because this is the target attribute or the attribute whose value will be predicted by the classification algorithms. The role of all other attributes should be set to regular.

As the label attribute of this dataset is of binominal type, the classification of this dataset will be binominal classification. More information about this dataset can be obtained from UCI repositories.

5.3 Operators in This Use Case

5.3.1 Rename by Replacing Operator

The Rename by Replacing operator replaces parts of the attribute names by the specified replacement. This operator is mostly used for removing unwanted parts of attribute names like whitespaces, parentheses, or other unwanted characters. This operator is very useful when many attribute names have some similar unwanted portion. The *replace what* parameter specifies the part of the attribute name that should be replaced. It can be defined as a regular expression which is a very powerful tool. The *replace by* parameter specifies the replacement.

5.3.2 Filter Examples Operator

Filtering can be defined as selecting a particular subset of examples from the dataset. The subset of examples can have some common properties, like having missing values. Filtering can be useful in a number of ways. It can be used to discard certain types of examples or it can be used to filter only the required type of examples. The Filter Examples operator in RapidMiner filters examples from the given dataset on the basis of some properties of the subset. The Filter Example Range operator filters examples within the specified range, for example, the first 100 examples.

The Filter Examples operator filters out only those examples that satisfy the specified condition. Several pre-defined conditions are provided in the *condition class* parameter. The users can also define their own conditions to filter examples. The *invert filter* parameter can be used to invert the condition specified in the *condition class* parameter. This operator can be used to filter examples with (or without) correct prediction, with (or without) missing values in attributes, and with (or without) missing values in the label attribute. The Filter Examples operator is frequently used to filter out examples that have missing values.

5.3.3 Discretize by Binning Operator

Dividing numerical values into different groups is known as discretization. This can be considered as a form of numerical-to-nominal type conversion. It is often required to group numerical values into certain number of groups. Many operators require all attributes to be of nominal form or many operators produce better results when applied on nominal attributes. In such cases, it is necessary to discretize numerical attributes to nominal type. The Discretization operators are located at 'Data Transformation / Type Conversion / Discretization'.

The Discretize by Binning operator discretizes the selected numerical attributes into a specified number of groups (or bins). The Values falling in the range of a group are named according to the name of that group. The range of all groups is equal. The number of the values in different groups may vary, i.e., the frequency of groups may vary. The resultant attribute is of nominal type. The groups are named automatically and the naming format can be changed by the *range name* parameter. Only a selected range of values can also be discretized by using the *define boundaries* parameter. The *min value* and *max value* parameters are used for defining the boundaries of the range. If there are values that are less than the *min value* parameter, a separate range is created for them. Similarly, if there are values that are greater than the *max value* parameter, a separate range is created for

them. Then, the discretization by binning is performed only on the values that are within the specified boundaries.

5.3.4 X-Validation Operator

The X-Validation operator (also written as Cross-Validation) is a nested operator like the Split Validation operator. The major difference between Split Validation and X-Validation is that the Split Validation operator iterates just once. On the other hand, the X-Validation operator has multiple iterations. The given dataset is partitioned into k subsets of equal size. Of the k subsets, a single subset is retained as the testing data set, i.e., the input of the testing sub-process, and the remaining $k-1$ subsets are used as the training dataset, i.e., the input of the training sub-process. The cross-validation process is then repeated k times, with each of the k subsets used exactly once as the testing data. The value k can be adjusted using the *number of validations* parameter.

5.3.5 Performance (Binominal Classification) Operator

This operator is used for statistical performance evaluation of binominal classification tasks, i.e., classification tasks where the label attribute is of binominal type. This operator delivers a list of performance criteria values of the binominal classification task. The performance vector delivered by this operator has numerous binominal classification-specific performance criteria, e.g., lift, fallout, f measure, false positive, false negative, true positive, true negative, sensitivity, specificity, positive predictive value, negative predictive value, etc.

5.4 Use Case

The Credit Approval dataset is loaded using the Read CSV operator. The Rename by Replacing, Filter Examples, and Discretize by Binning operators are used for pre-processing purposes. The X-Validation and Performance (Binominal Classification) operators are used during the testing phase. The effect of discretization and filtering on the accuracy of the model is also observed.

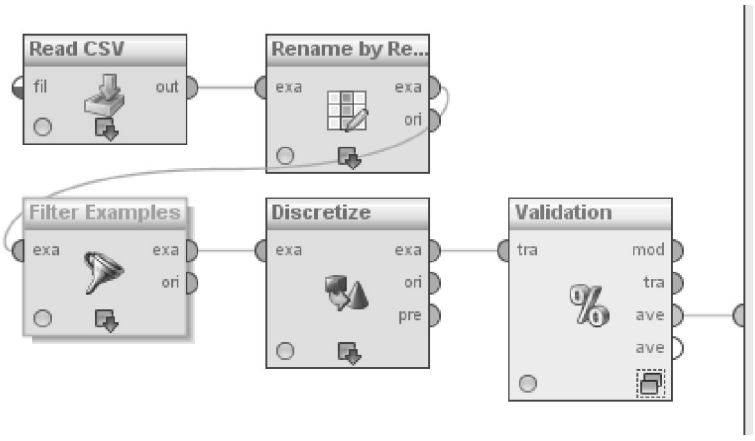

FIGURE 5.1: Workflow of the process.

5.4.1 Data Import

The Read CSV operator is used for importing data in this process. The data at the following url is stored in the form of a text file.

`http://archive.ics.uci.edu/ml/machine-learning-databases/credit-screening/crx.data`

That text file is used as data source in RapidMiner. The Read CSV operator is used for loading data from a csv file. The *Import Configuration Wizard* is used for loading the data from the file in this process. The steps of this configuration wizard are explained below.

Step 1: Select file.

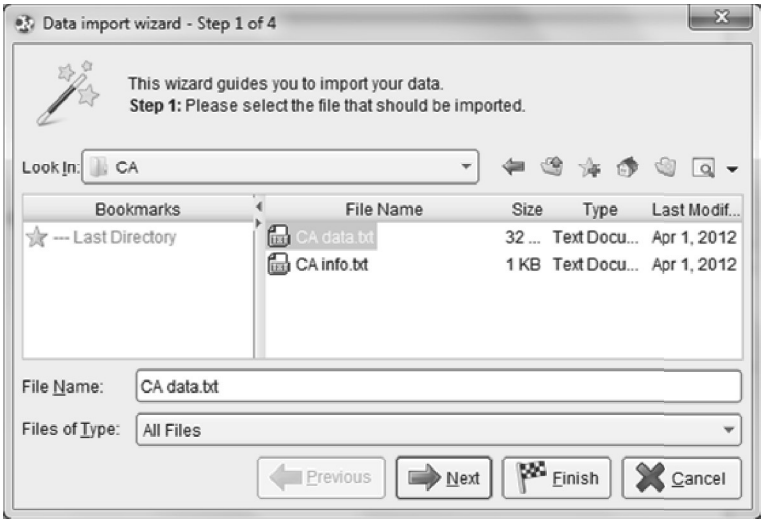

FIGURE 5.2: Step 1 of Import Configuration Wizard of the Read CSV operator.

Step 2: File reading and column separation.

RapidMiner fetches the data and displays it, in this step. The most important decision is selecting the column separator at this step. As the data in the Credit Approval dataset is comma separated, 'Comma' is selected as column separator.

Step 3: Annotations.

This step allows specifying the annotations. By default, the annotation of the first row is set to 'name'. If the first row of data does not contain the names of attributes, its annotation should be unchecked (as shown in Figure 5.4).

Step 4: Set the name, type, and role of attributes.

This is the most crucial step of the import wizard. The name, type, and role of attributes are specified in this step. Figure 5.5 shows the name, type, and role of attributes set for the Credit Approval dataset. A breakpoint is inserted after this operator so that the output of the Read CSV operator can be seen in the Results Workspace.

5.4.2 Pre-processing

Three pre-processing steps are discussed in this use case; renaming attributes, filtering missing values, and discretization.

Renaming attributes The name format of all the attributes of this dataset is of the form att1, att2 etc. This is the default naming format of the Read CSV operator. However, this naming format does not match the naming format that was mentioned in

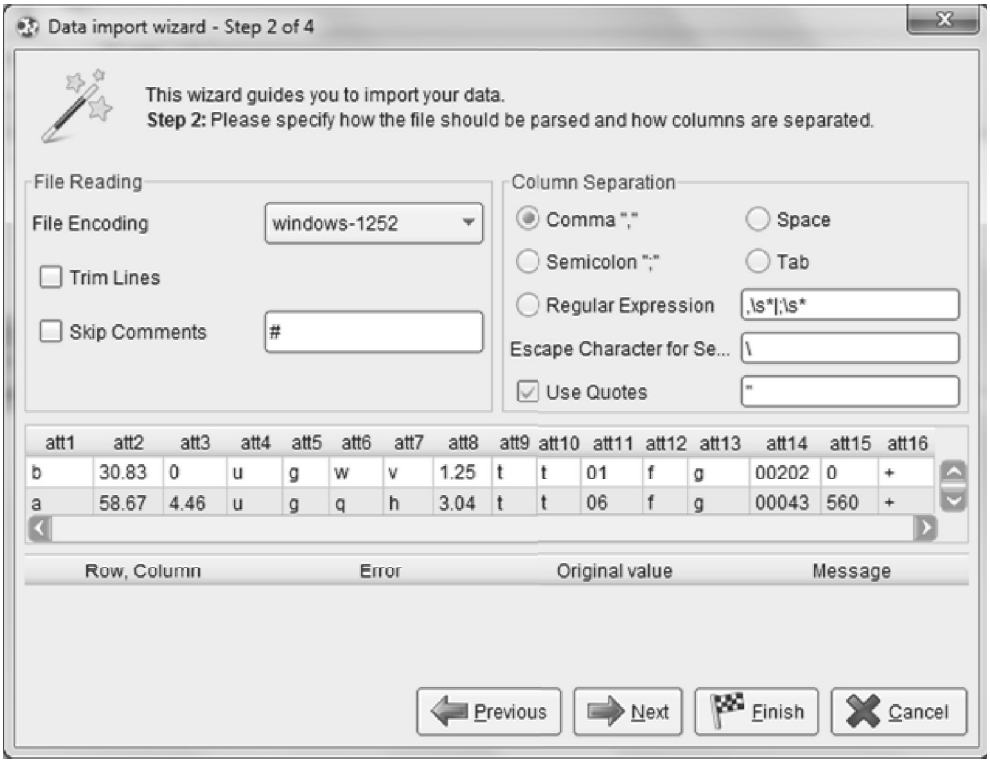

FIGURE 5.3: Step 2 of Import Configuration Wizard of the Read CSV operator.

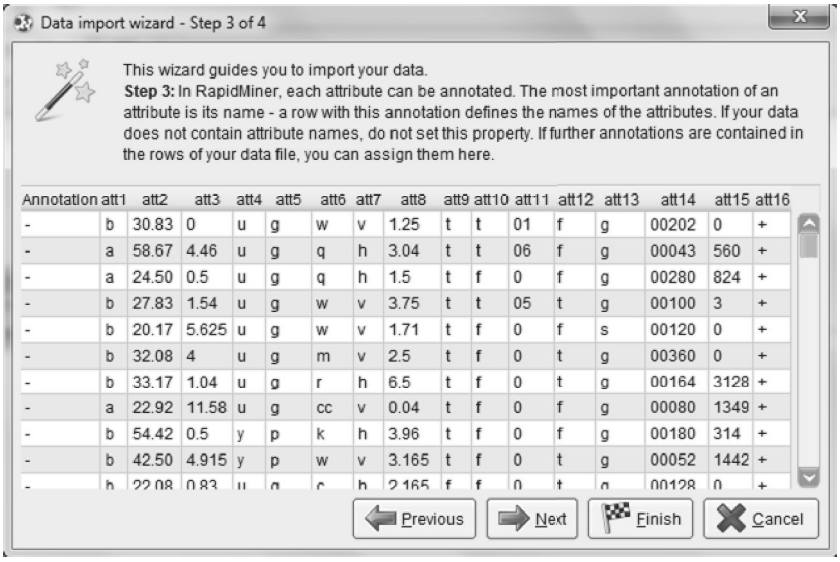

FIGURE 5.4: Step 3 of Import Configuration Wizard of the Read CSV operator.

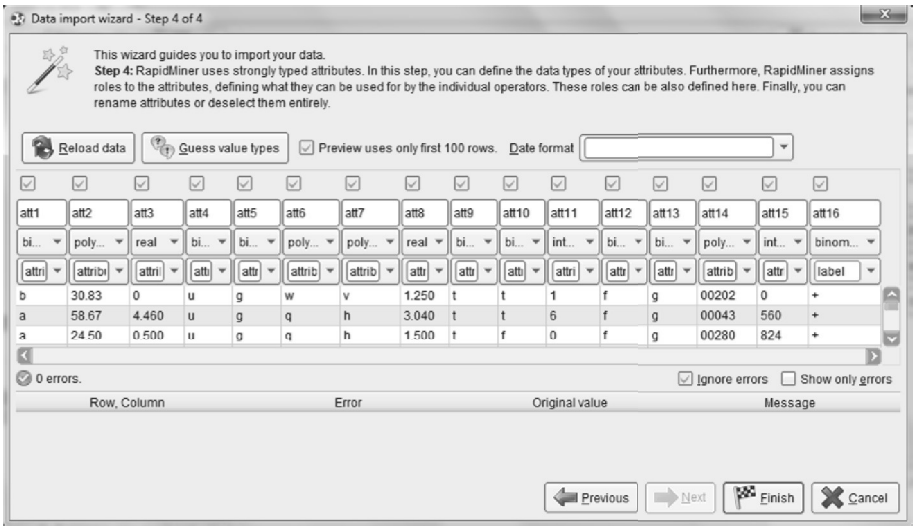

FIGURE 5.5: Step 4 of Import Configuration Wizard of the Read CSV operator.

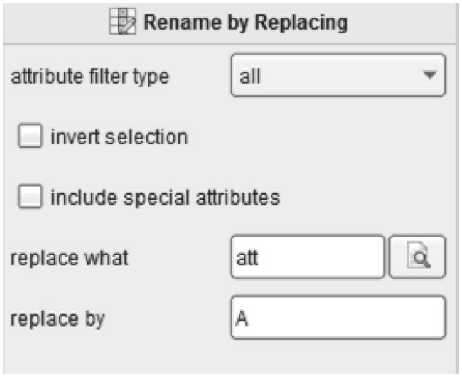

FIGURE 5.6: Parameters of the Rename by Replacing operator.

the explanation of the Credit Approval dataset. The naming format should be of the form A1, A2, etc. To change the naming format of all the attributes the Rename by Replacing operator is used.

Figure 5.6 shows the parameters of the Rename by Replacing operator. The *attribute filter type* parameter is set to 'all' because all the attributes are to be renamed. The common pattern to replace is 'att' and it is to be replaced by 'A'; this explains the values of *replace what* and *replace by* parameters. A breakpoint is inserted after this operator so that the effect of this operator can be seen in the Results Workspace.

Filtering missing values The Credit Approval dataset has missing values in some attributes (A1, A2, and A14). This can be seen in the outputs of the previous two operators. The Filter Examples operator is used in this process to simply remove the examples with missing values. The *condition class* parameter is set to 'no missing values'. A breakpoint is inserted after this operator so that the results of this operator can be seen in the Results Workspace.

Discretize By Binning The Discretize By Binning operator is introduced in this process. The Discretize By Binning operator is used for converting the type of all the numeric attributes (A2, A3, A8, A11, A14, A15) to nominal. The parameters of this operator are shown in the Figure 5.7. A breakpoint is inserted after this operator so that the output of this operator can be seen in the Results Workspace. After application of this operator, all numeric attributes are changed to nominal type. All these attributes have three possible values, i.e., range1, range2, and range3. The *number of bins* parameter is set to 3, therefore, the ranges of all the attributes are divided into three equal partitions. All the values in a partition are named according to that partition. For example, the attribute A11 had range of values from 0 to 67. Three equal ranges of A11 are:

- range1: from 0 to 22.3
- range2: from 22.3 to 44.7
- range3: from 44.7 to 67

Most values fall in range1. Only the examples on Row No 45 and 47 fall in range2 (values 40 and 23, respectively). Only the example on Row No 117 falls in range3 (value 67).

5.4.3 Model Training, Testing, and Performance Evaluation

The X-Validation operator is used in this process for facilitating in model training, testing, and performance evaluation. All parameters are used with default values (default value of the number of validations parameter is 10). Here is an explanation of what happens inside the X-Validation operator.

1. The X-Validation operator provides the training dataset (composed of 9 out of 10 subsets) through the *training* port of the Training sub-process. This training dataset is used as input for the Naïve Bayes operator. Thus, the Naïve Bayes classification model is trained on this training dataset.

2. The Naïve Bayes operator provides the Naïve Bayes classification model as its output. This model is connected to the *model* port of the Training sub-process.

3. The Naïve Bayes classification model that was provided at the *model* port of the Training sub-process is provided by the X-Validation operator at the *model* port of the Testing sub-process. This model is provided as input to the Apply model operator (at *model* port of the Apply Model operator).

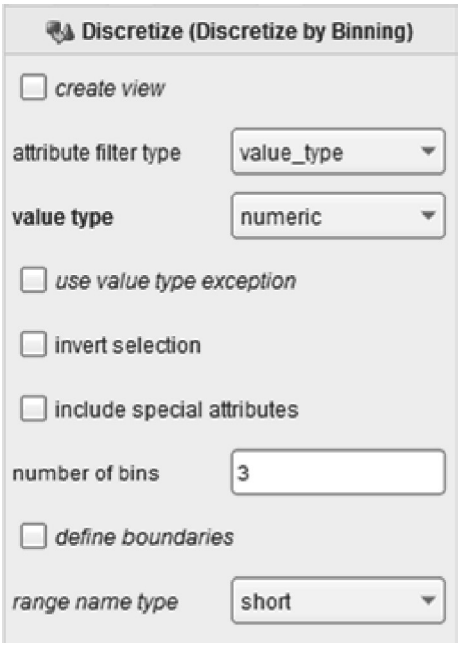

FIGURE 5.7: The parameters of the Discretize By Binning operator.

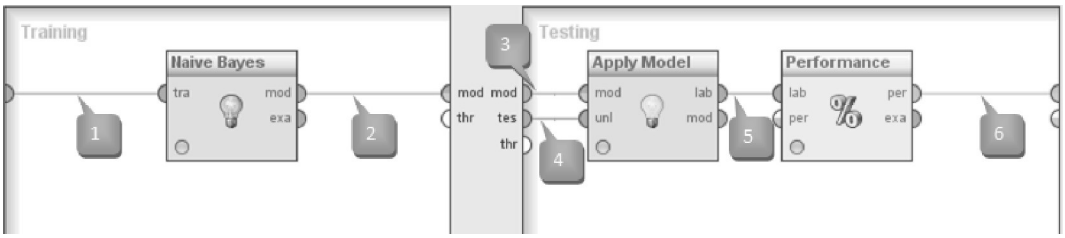

FIGURE 5.8: Training and Testing subprocesses of the X-Validation operator.

4. The X-Validation operator provides the testing dataset through the *test set* port of the Testing sub-process. The testing dataset is composed of the last remaining subset because 9 of the 10 subsets were part of the training dataset. This testing dataset is provided as input to the Apply Model operator (at *unlabeled data* port).

5. The Apply model operator applies the Naïve Bayes classification model (that was provided at its *model* input port) on the testing dataset (that was provided at its *unlabeled data* input port). The resultant labeled dataset is delivered as output by the Apply Model operator. This labeled dataset is provided as input to the Performance (Binominal Classification) operator.

6. The Performance (Binominal Classification) operator is used in this process instead of other performance operators because this operator is used specifically for evaluating the performance of binominal classification tasks. It has many binominal classification-related performance measures that are not present in other performance operators. The Performance (Binominal Classification) operator evaluates the statistical performance of the model through the given labeled dataset and generates a performance vector which holds information about various performance criteria.

7. All the steps from step-1 to step-6 are repeated 10 times; every time a different subset (of total 10 subsets) is used as the testing subset and the remaining 9 subsets as the training subset. Thus, in every iteration, a new model is trained and it is tested on a new testing subset, following which its performance is measured. Finally, at the end of all iterations, the X-Validation operator delivers the model with the best performance, together with its performance vector.

The accuracy of the model turns out to be 85.17%. Figure 5.9 shows how the accuracy of the model varies with discretization and filtering.

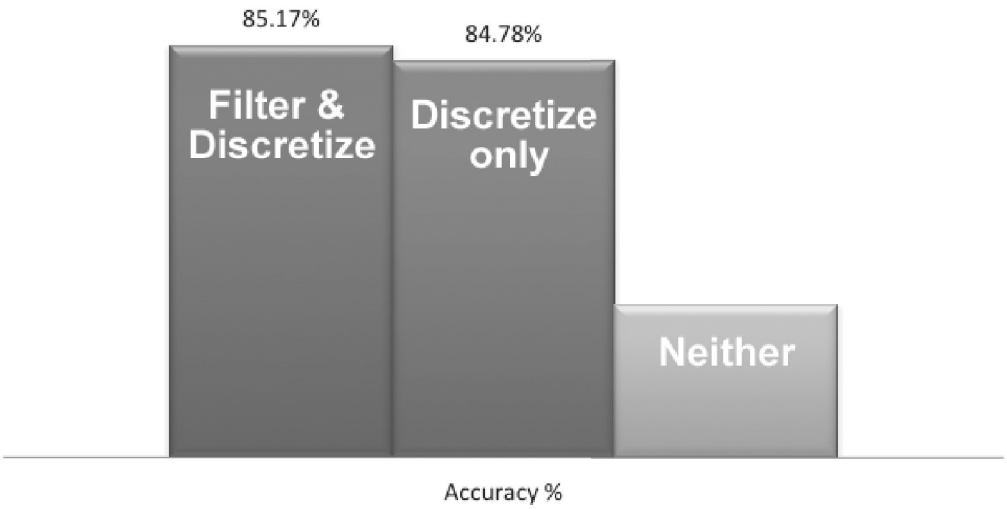

FIGURE 5.9: Effect of discretization and filtering on the accuracy of the Naïve Bayes model.

The figure shows that in this use case the most accurate model is obtained when both filtering and discretization are used.