CS36110: Machine Learning – Assignment 1

By Christian Holland

Contents

[3.1 Evaluate the results of classifiers 2](#_Toc466295402)

[3.1.1 Naive Bayes 2](#_Toc466295403)

[3.1.2 J48 2](#_Toc466295404)

[3.1.3 Baseline for comparison 2](#_Toc466295405)

[3.2 Evaluate ‘Replace Missing Values’ filter 3](#_Toc466295406)

[3.2.1 Method 3](#_Toc466295407)

[3.2.2 Is it appropriate? 3](#_Toc466295408)

[3.3 Evaluate the performance after missing values replaced 3](#_Toc466295409)

[3.3.1 Naïve Bayes after filter 3](#_Toc466295410)

[3.4 Compressing all classes into a single class 4](#_Toc466295411)

# 3.1 Evaluate the results of classifiers

## 3.1.1 Naive Bayes

In a very broad view, Naïve Bayes will split the data set up into separate classes and evaluate each one independently. This means that the independently evaluated features are analysed more in-depth to show relationships between attributes that may not have been initially obvious. As it uses conditional independence, this makes it significantly stronger than many other classifiers such as J48. This is proven by its correctly classified instances; 55.4455%

## 3.1.2 J48

Decision trees can be useful with medical diagnosis, as they are commonly effective with attribute-value pairs. In this case, our attribute-value pair would be, for example, ‘age’ and its respective value. However, our Cleveland Heart Disease dataset has been prepared without careful checking – meaning there are potential anomalies in the dataset. Due to this, it would be common for the decision tree to fall behind Naïve Bayes as sparse data can cause ‘overfitting’; the random error or noise in a data set. An example of why this might be an issue in the decision tree would be if we arrive at a branch, but neither of the leaf nodes apply to the situation – as a previous anomaly has caused misdirection in our decision. This would explain why the correctly classified instances is less than Naïve Bayes – being 51.8152%

## 3.1.3 Baseline for comparison

In order to establish classification which would be the best baseline for performance, it would be effective to simply assume that everyone in the data set has a specific value. In this case – assuming that everyone doesn’t have heart disease would be a very basic outcome for the scenario. ZeroR is this baseline classification – as it does exactly this. It will obviously have a large error rate; however, this can be a good average as we know that if the percentage of incorrectly classified instances is any lower than this, it is proof that the classification performance is not accurate enough to give a reliable outcome.

# 3.2 Evaluate ‘Replace Missing Values’ filter

## 3.2.1 Method

In order to establish what to replace the missing values with, the filter will check through the training data and establish a mode and mean for each nominal and numeric attribute. This will ensure that the replaced value is realistic and is not out of the ordinary in comparison to the other data. An extra option is available in order to remove the class index temporarily before the filter is applied, meaning that if the option is disabled, it will only use the values within the class as a reference to get the mean and mode, however if enabled, the whole data set is used to find said mean and mode. This results in a broader training data set to obtain a more accurate mode and mean.

## 3.2.2 Is it appropriate?

This method of replacing the missing values is what I believe to be the most accurate and efficient way considering the difficulty of getting values that were never in the data set, as it uses all of the data so far to establish a realistic number to replace it with. The only way this method could produce erratic values would be if our training set was not large enough to get a mean or mode that correlates with the rest of the data set.

# 3.3 Evaluate the performance after missing values replaced

## 3.3.1 Naïve Bayes after filter

Without the ignoreClass option set to true, the correctly classified instances are now 56.1056% - almost a percent more than with the missing values.

## 3.3.2 J48 after filter

With the same options as above, correct classified instances are now 52.4752% - again almost a percent more than when there were missing values.

## 3.3.3 Overall comparison

We can see that the new data set can be more accurately classified – albeit not by much. The accuracy has grown by about the same amount for both of the classifiers; when were are not removing the class index before the filter, there is a small improvement. When we remove the class index before the filter, it helps with overfitting, which is more prominent in J48.

# 3.4 Compressing all classes into a single class