# Abstract

Predicting medical data is becoming more and more vital towards helping reduce death rates, or reducing risks in medical procedures. With the use of tools such as WEKA \cite{weka}, the ability to build machine learning classifiers has become easier, and has helped produced more reliable results. This paper looks at death rate prediction following thoracic surgery, and compares different means of prediction and configurations on a given test set of data.

# Introduction

Thoracic surgery is a means of lung resection, and is performed in order to remove part or all of a lung from a patient who is suffering or has suffered from lung cancer \cite{thoracic}. The data provided for training a machine learning approach to predicting the death rate one year after surgery contains 300 patients, each of which hold several attributes, including age and ‘Risk1Yr’. The most latter of these states is the example patient survived (indicated by a 0) or did not (indicate by a 1). The aim of the classifier trained at the end of this report is to be able to predict the survival rate in conjunction with an identical test set of data. \par

For easier statistical reading, and the ability to easily try a range of classifiers on the training data, Weka was chosen to perform experiments and training. Weka allows pre-processing of data, classifiers and filters to be applied, and also provides good visualisation of training results, such as ROC curves. Throughout this paper, both graphs generated within Weka, and ones created from comparing different classifiers will help to reinforce result conclusions. \par

To visualise data quickly during the fine tuning and selection of classifiers during the start of this project, the Weka Explorer application will be used, and then will be moved onto the experimenter during latter parts of the project in order to quickly run multiple experiments at once. Finally, the Weka Knowledge flow application will allow comparisons of the ROC curve of multiple solutions in order to choose the most suitable set up for predicting the test data results. \par

As stated in the assignment, more than one classifier will be selected and compared to see which one best suits predicting this type of data. Following that, more comparisons will be made looking at changing a range of hyper-parameters for chosen classifiers.

## Data pre-processing

In Weka, and in any machine learning approach, some data pre-processing should be performed. Because the data set is real-life based, data is not always complete, or follow a linear pattern. Due to this, an issue known as class imbalance can hinder the performance of learning systems, and means that one instance of an attribute (in this example, surviving patients) is the majority sample causing a bias when training the machine learning algorithm. On first inspection of the data in the training set, it can be seen in Figure 1 that the ‘Risk1Yr’ attribute holds far more surviving patients that ones who did not.

[IMAGE FOR DATA HERE]

There are two means of settling imbalances in data however \cite{imbalance}:

\begin{enumerate}

\item Under-sampling – where the size of the samples are reduced proportionally

\item Over-sampling – where the size of samples are increased proportionally

\end{enumerate}

In the case of Weka, there are filters available within the pre-processing stage of experiments allowing the addition of under or over-sampling. In this case, the SMOTE (Synthetic Minority Over-sampling Technique) \cite{smote} plugin for Weka offered an easy means of resampling data based on a given set of parameters. In an accompanying paper \cite{over-sampling}, SMOTE is described as creating ‘synthetic examples’ rather than replacements that are generated in feature space rather than data space. In the Weka library, synthetic samples are created by using a range of nearest neighbour minority samples, and placing new samples between them. By default, the SMOTE filter uses five nearest neighbours to create new samples, and as this is recommended for this size dataset, this will remain the selected value. \par

[IMAGE FOR FILTER DATA HERE]

Once the SMOTE filter was applied, there was a clear view of change in the proportions of the survival and death rates. As Figure 2 shows, over-sampling has been applied to 200%, which has meant the minority of samples showing death rates only has 50 samples less than the majority as opposed to the 138 sample difference previously. \par

In addition to balancing data, viewing the data in Weka explorer on scatter gave good insight into any data trends, and also gave hints into which attributes carried more weighting towards the risk attribute than others. At first, both age and risk were plotted against each other to see if there were any trends towards being younger meant better survival rates (as shown in Figure 3). In order to view the scatter plot well enough, Jitter was added in Weka to plot points further away from each other.

[IMAGE FOR AGE PLOT]

Although the plot looks fairly evenly spread on age for both survival and mortality rates, there is a slight higher number of patients who survived aged around 50 than those who were older. \par

By looking at the scatter plots for various combinations of data, dimensionality reduction is something that can be used to both increase the accuracy and speed of the resulting algorithm. There are various means of dimensionality reduction, such as PCA (Principle Component Analysis) and SOM (Self Organising Maps) /cite{dimensionality}, whilst in Weka, attributes can be ranked, in order to see which ones carry the most important in a data set. \par

For this experiment, the InfoGainAttributeEvaluator was used to rank attributes in Weka. This algorithm was selected because of the simple output of ‘worth of an attribute’ that allows the least useful attributes or features to be omitted from any classification later on. The attribute evaluator configuration run can be found in Appendix A From first run, and shown in Figure 3 is the results from the ranking. It appears that three attributes listed have no effect on the prediction of one year risk, so these will be omitted when testing classifiers in the following section.

[IMAGE OF RANK]

Although only the three at the bottom were removed from this process, it is possible to remove more features. Three was chosen because it appeared to be a global minimum feature set before overfitting occurred. This is when the learning curve appears to go back above its minimum due to not enough descriptive features.

# Classifying data

For the set of experiments following, multiple classifiers have been selected to find the best prediction possible. In a notable paper by D. Wolpert (The Lack of A Priori Distinctions Between Learning Algorithms) \cite{algorithms}, this points out that there is not one classifier that will work for every problem in machine learning. Therefore, by comparing a range of classifiers, the best traits will be found, and will contribute towards the final predictor provided in the appendices. This section takes a look at what classifiers have been selected, how each should work, and what fine tuning was done to each.

## Classifier selection

Three classifiers have been chosen for this part of the paper, and will be compared directly and indirectly in the results section of this report. The three classifiers selected are as follows:

\begin{itemize}

\item Decision tree – J48 for generating C4 trees

\item Naïve Bayes – A Naive Bayes classifier

\item Neural network – Multilayer perception classifier using backpropagation

\end{itemize}

The first of which was selected because of both its practicality and popularity, which has been proved useful for data mining. Decision trees are used in a range of medical diagnoses, and because decision trees allow human readable rules of classification and are easy to interpret, they are very useful in this field of research. \par

With J48, similarly to most decision trees, the structure created appears like a flow chart, with each node denoting a test, each branch denoting a an outcome of a test, and each leaf node holding a class label. This classifier has two parts \cite{decision} to its use: the growth phase and the pruning phase. Trees split training sets based on the criteria (in this case mortality rate), and perform this until all records belonging to each split hold the same class label (see example in Figure 4). Overfitting is the outcome of creating trees because of this, therefore pruning is added to the scenario. Outliers and fuzzy data from the tree to ensure it holds only useful information. Construction of flow chart is quick, therefore the overall prediction rate in this paper should be quicker than other algorithms. \par

[IMAGE OF TREE]

The second classifier listed that will be compared in order to find out which algorithm will be most suited to this problem is a using a Naïve Bayes approach. The simplest of the three, Weka offers an implementation of this which uses Bayes rule as shown in Figure 6. From first observation, it does appear that Bayes may work better with more evidence, though this may not be the case, as overfitting may occur.

[IMAGE OF BAYES]

Bayes has always been another popular machine learning approach to medical data, and many papers have justified the use of it. One such paper which looked at predicting heart disease \cite{bayes}, gives good reasons for the choice of the algorithm. When data amounts are high, this way of training and predicting does not exponentially increase in terms of time. Where attributes are independent of each other, Naïve Bayes can handle it better than other algorithms (as shown in pre-processing, there are attributes in this paper’s example for the need of this), and output is more efficient compared to other methods. \par

The final classifier that will be used in the prediction of thoracic surgery mortality risk will be with neural networks. In Weka, by default one neural network library comes pre-installed, Multilayer perception. Artificial neural networks (ANN) are a more recent development in medical diagnosis, and come from more of an artificial intelligence background. These are used widely in science, and are popular where attribute relationships may be unknown or very complex \cite{neural}. \par

Looking at neural networks from a statistical point of view, these in essence are formed by an input layer, hidden layers and an output layer. The more complex the system studied, the more layers are added. Neurons in the input layer receive data to be trained on, and move them to the first hidden layer through weighted links. Once all data is passed through hidden layers, data is mathematically processed and reach the network’s output. Figure 7 illustrates this.

[IMAGE OF NEURAL]

## Configuration

Each classifier has a set of hyper-parameters within Weka that can be adjusted according to the problem they face. By default, these are set in a way to be useful to the widest range of data sets, therefore may need to be changed to increase the effectiveness for the data tested in this paper. For each configuration change following, the configuration string can be found in Appendix B. Before each classifier is configured, the classifiers were tested using default options, and the statistic of how correct they were at predicting can be found in Figure 8. \par

[IMAGE OF CORRECTNESS]

As shown in the Figure 8, Naive Bayes is the classifier with the weakest result as of yet. In Weka, without the use of filters or boosting, this classifier is limited in the number of configuration options available. Of the parameters that were tested, one known as ‘supervised discretization’ had some positive effect to the amount of correctly classed instances \cite{discretization}. This is a method of variable selection, and transforms continuous values of variables to discrete ones. This option has been shown to improve classifiers (in this case Naïve Bayes) that are sensitive to high dimensional data. Table 1 shows the difference in the performance of this classifier with the addition of this method. \par

[TABLE]

The second of the classifiers, J48 in Weka carries more configuration options than that of Bayes, with most relating to the pruning of the tree. As mentioned earlier, pruning is performed after the tree is generated in order to remove outliers, and helps prevent under-fitting. Initially, the decision tree classifier was quite successful in classifying correctly, but with changes to the way it pruned, this could be refined. \par

Starting with the confidence factor, this was increased, which would mean that instances of predictions that had less confidence would be pruned less often. Although this meant the tree may then be more biased, the accuracy did in fact increase. Figure 9 shows the prediction correctness based on the confidence factors tested.

[IMAGE OF CONFIDENCE]

As shown in the previous figure, a confidence factor of 0.6 was where the accuracy peaked. Whilst testing the different confidence factor values, it was noted that the time taken to run the classifier increased as confidence factors reached above 0.5. As for further configurations, none were found that could increase both the accuracy and the ROC curve values. \par

The final classifier (Multilayer perception) also had a strong selection of parameters available, but in this case none could help increase the accuracy further than the default state. By increasing the learning rate of the neural network, it was hoped that the classifier would learn faster, and therefore be more accurate towards the end of training. As shown in Figure 10 though, the increase in learning rate actually decreased the accuracy, and the same happened when decreasing the learning rate.

[IMAGE LEARNING RATE]

For this reason, this classifier remains unchanged, including the learning rate of 0.3, which already seems to be at its peak for accuracy. Although the neural network classifier could not be configured directly, further adaptions to the algorithm could be performed either through boosting or further filtering. To test this and the other classifiers, an Ada-boost filter was wrapped around each and compared with the previous results (Shown in Table 2). Note: AdaBoost has been applied to all current configuration changes mentioned previously in an attempt to increase accuracy further.

[TABLE ADA BOOST]

As shown in the above Table 2, using AdaBoost has increased the effectiveness of two of the classifiers, though two of the classifiers also suffered heavily in terms of speed. With this taken into account, it was decided that only Naïve Bayes benefitted from boosting, so was the only one to keep this as part of the algorithm.

# The final configuration change made was to try each classifier with a filter applied. Because the data had been pre-processed already, using a filtered classifier would be beneficial. In Weka, applying a filter is simple, and any classifier can be wrapped by it. The tests using this with each classifier (again, any configuration completed up to this stage has been taken into account) can be found below in Table 3.

[TABLE FILTER]

The table shows that only the neural network classifier benefits from the filtered wrapper. Therefore, in Appendix E, the configuration for this algorithm is available. The other classifier algorithms will remain the same due to no need to add the filter. It can also be noted that unlike AdaBoost, the filter wrapper did not add any significant amount of time to compute predictions.

# Results discussion

Following pre-processing, classification configuration, and additions of filters/ boosting, the three can now be compared and evaluated. When using Weka, there are many statistics available following a test run. One of these is the root mean squared error. This statistic

It should be noted that the results shown in this section of the report were run in the Weka experimenter for efficiency of running multiple tests at once.

Root mean squared error

Test classifiers before and after SMOTE

Test classifiers with filter for balancing

Compare speed of classifiers

Number of cross validations

Explain what knowledge flow is

ROC area – use curve from knowledge flow

Show other graphs from knowledge flow?

Summarise best classifier from the three, why it is better

Create classifier for solution based on findings

# Conclusion

Cancer could have spread before, therefore data may not be completely true, though should be fairly accurate.

Thoracic

<http://www.bmc.org/thoraciconcology/treatments/lung-resection.htm>

imbalance

<https://hydra.hull.ac.uk/resources/hull:10103>

smote

<https://msdn.microsoft.com/en-us/library/azure/dn913076.aspx>

over-sampling

<https://www.jair.org/media/953/live-953-2037-jair.pdf>

dimensionality

<http://www.turingfinance.com/artificial-intelligence-and-statistics-principal-component-analysis-and-self-organizing-maps/>

algorithms

Wolpert DH. The Lack of A Priori Distinctions Between Learning Algorithms. Neural Computation. 1996;8:1341-1390.

decision

<http://www.ijcaonline.org/volume26/number4/pxc3874247.pdf>

bayes

<http://ijarcce.com/upload/2014/may/IJARCCE9E%20%20a%20rupali%20%20Heart%20Disease%20Prediction.pdf>

neural

<http://jab.zsf.jcu.cz//11_2/havel.pdf>

discretization

<http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2656082/>