Predicting Diabetes

Authors: Hugh Purnell, Ashwin Ramesh

Table of Contents

Aim

The aim of Assignment 1 is to use statistical Machine Learning to predict diabetes, using a training set representing a population subset of patients with diabetes.

This study investigates the classification performance of a number of Machine Learning (ML) algorithms. The study is to use a real-world data set with eight known attributes and class variable, per sample. K-Nearest Neighbour and Naive Bayes classifiers are to be implemented and their performance compared to Machine Learning classifiers from the Weka Software package[1] using 10-fold stratified cross validation. The effects of feature selection - a pruning (or noise elimination technique), will also be explored.

Data

Data Set

The data-set represents a subset of samples from a study originally done by the National Institute of Diabetes, Digestive and Kidney Diseases based in the United States. It studied the diabetes rates of Americans with Pima Indian heritage living near Phoenix, Arizona. The subset of data donated to the UCI Machine Learning Repository is focused on females over 21 years of age and contains sample attributes that may have correlation with diabetes incidence; such as the World Health Organisation's criteria for diabetes diagnosis: 2 hour post load plasma glucose levels. Wherein levels above 200mb/dl (11.1 mm/l) being indicative of the disease.[3]

Data Attributes

The data-set is stored in text-readable CSV (comma separated values) format. Each row represents a sample, and each element of the row represents attributes 1 through to 8 in order, followed by a binary class variable {0,1}.

1. Number of times pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test

3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/(height in m)^2)
7. Diabetes pedigree function
8. Age (years)
9. Class variable (0 or 1)

Implementation Language

The pre-processing and implementation components of this Assignment were programmed in Python 2.7.

Data preparation (Pre-processing)

To pre-process the data, the sample set was read from a file in CSV format. With the exception of Class variable, each attribute - column wise - was normalised using:

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

where *i* is the attribute type.

This results in each attribute taking a value within the range [0, 1]. The Class attribute for each sample is then changed from a binary 0 or 1 to "class0" or "class1". The data having been now normalised is then written back into a new CSV file named "pima.csv", and a file header for the attribute types inserted at the top, per Assignment requirements. The CSV file header is as follows:

"num_pregnant,plasma_glucose_concentration,diastolic_blood_pressure,tricepts_skin_fold_thickness,2hour_serum_insulin,bmi,diabetes_pedigree,age,class"

Attribute Selection

Feature selection is the process of selecting a subset of correlating "features" or "attributes" to be used to create a better machine learning model. One of the most common feature selection algorithm is known as the Correlation feature Selection (CFS).

As expressed in Mark Hall's paper on Correlation-based Feature Selection for Machine Learning[4]: "The underlying principle behind CFS is 'Good feature subsets contain features highly correlated with the classification, yet uncorrelated to each other'". In the case of PIMA's diabetes data, the classification is whether each patient is diabetic or not, while the features are:

- Number Of Pregnancies
- Plasma Glucose Concentration
- Diastolic Blood Pressure
- Triceps Skinfold Thickness
- 2 Hour Serum Insulin
- Bmi
- Diabetes Pedigree
- Age

With the use of WEKA, CFS was calculated on the PIMA data. The subset of features is as follows:

- Plasma Glucose Concentration
- BMI
- Diabetes Pedigree
- Age

This subset reflects that together, these four attributes most affect the diabetic/non-diabetic nature of a patient, while simultaneously do not have a strong correlation between each other. Through the use of CFS, the PIMA data can be reduced and a better prediction can hopefully be made. The following sections will discuss this rationale.

Results

Two sets of results were calculated to identify the success rate of predicting diabetes. The first involved running various classifiers in Weka against the data, while the second involved running a personal implementation of the k-nearest and Naïve Bayes classifiers against the data.

Each percentage represents the result of a ten-fold stratified cross validation between the data set and the classifier algorithm. This simply involves evenly distributing the two classes ("class0" vs. "class1") and created data sets to represent training data and test data. The average of ten runs for each classifier is the final result. The Weka and personal results are tabled below respectively.

WEKA Classifier Percentages

	ZeroR	OneR	1-NN	5-NN	NB	DT	MLP	SVM
Full Data Set	72.14%	72.14%	70.18%	73.18%	76.30%	73.83%	75.13%	77.34%
CFS Data Set	72.14%	72.14%	68.36%	73.83%	77.47%	74.87%	75.26%	76.95%

Personal Implementation Percentages

	1-NN	5-NN	Naive Bayes
Full Data Set	69.27%	73.70%	75.91%
CFS Data Set	69.14%	73.18%	77.86%

Discussion

Diabetes is a group of metabolic diseases with which the human body is unable to produce insulin (type 1) or unable to use insulin (type 2). Currently there is no known cure, and as of 2010 it affects the lives of over 285 million people worldwide.[5] This assignment showcases a relevant application of Machine Learning - prediction of disease such as diabetes in an individual based on only a handful of abstract attributes. Such predictive power - being able to identify risk

in an individual increases the chance of preventive intervention in the form of treatment or lifestyle changes.

As the results above show, the accuracy of predicting diabetes using the ten-fold cross validation ranged between 65% and 80%, identifying that there is a high probability that diabetes can be predicted using any one of these classification methods.

Although the various classifiers used correctly predicted diabetes over 70% of the time, on further inspection, we found that they were not all as efficient as each other or provided the same level of integrity when trying to predict diabetes.

Both ZeroR (Zero Rule) and OneR (One Rule) are very simplistic algorithms that do not completely take predictors into account when trying to classify a new data set. Zero Rule takes no predictors into account, but rather simply creates a frequency table for the target and selects the most frequent outcome[6]. One Rule on the other hand takes the best predictor (one with the least error) to classify a new data set.[7] Although both algorithms are extremely fast and produce a 72% chance of successfully predicting diabetes, our research suggests that by not taking all data into account, mistakes will be made during the classification process.

On the other hand, the Naive Bayes (NB) classifier does take all predictors into account. By using a probability density function to predict diabetes for each attribute, a more correct classification will be made. Both Weka and our personal implementation of the naïve bayes algorithm resulted in an accuracy rate over 75%. The small difference between the two naïve bayes results can be attributed to the randomness of the ten-fold cross validation arrays. However, the one disadvantage of this algorithm compared to ZeroR/OneR is that it's running time is slighly higher.

Multilayer Perceptron (MLP) is similar to NB in terms of results. It too averages at about 75% correct classification rate. This method of classification creates an artificial neural network containing multiple layers of nodes connected to each other. It uses back propagation to train the

network and classify a new data item. However, unlike NB, MLP does not make any assumptions regarding the probability density functions, but rather creates decision functions through training. MLP however is also very slow, taking around 40 times the time it took Naïve Bayes to classify a data set.

By far the most correct classifier used was the Support Vector Machine (SVM) at over 76.5% accuracy. By being supervised learning models that analyse training data, SVMs create their own learning algorithm that classifies data input into one of two classes. This methodology, in tandem with its performance of non-linear calculations using kernel vector optimisations, makes it not only the most accurate, but also one of the more faster algorithms to classify diabetes.

The final two classifiers, Nearest neighbour (NN) and Decision Tree (DT) were worst in the group chosen. The Nearest-Neighbor algorithm works well as a classifier for datasets with low dimensionality. Higher dimensionality increases the apparent distance between sample instances, weakening correctness exponentially. Formally this problem has been referred to as "the curse of dimensionality". One solution to this problem is attribute culling - lowering the dimensionality of the training set. This data culling results in a loss of information that other classifiers - such as Naive Bays, are able to otherwise utilise. As such, the average accuracy for NN ranged between 70%-73%.

Decision Tree (DT) similarly averaged just over 73.5%, which was well below to overall average for all classifiers. The reason for this is due to the complexity created by the decision tree, possible incorrect relationships and problems in “applying regression and predicting continuous values”[8]. Additionally, decision tree algorithms use a greedy approach, making it far slower than some of the prior classifiers described.

Using the Correlation Feature Selection (CFS) did not remarkably improve classification rate, giving a classification performance within ±2% of the full data set. In some cases it acted to marginally degrade performance, as in 1-NN. CFS feature selection improved most the performance of the implemented Naive Bayes algorithm with a +2% advantage. That said, data

culling does improve both time and space complexity of all the algorithms used here. For example, linearly halving the number of attributes would half both runtime and memory requirements for the Nearest Neighbour algorithm. In datasets far greater in scope than the 768 sample set used in this study, such runtime and space complexity constraints become an concern and it is advisable to optimise using CFS.

Conclusions for Future Work

One aspect this study did not explore is the classification performance on an individual sample by sample basis. While all of the classifiers had performance around the 66-76% percent range, no information was drawn from any unique outlaying cases that were classified correctly by certain types of classifiers above others - for example, statistical classifiers over linear classifiers or visa versa. It may be worthwhile study to discover why these are difficult outliers in one algorithm but correctly classified in another.

A real world study exploring correlating features, that applies Machine Learning as a tool in fields besides computer science might choose to employ at most only two or three Learning Machine algorithms. Naive Bayes giving a statistical/ probabilistic analysis, while Support Vector Machines or Multilayer Perceptrons giving a non-probabilistic linear classification. Support Vector Machines over Perceptrons if runtime speed is a concern, with no loss in accuracy. Alternatively a study could form ensembles of classifiers.

Reflection

Both team members found the project to be a worthwhile educational experience, with Git version control and shared online google documents used collaboratively. Research skills strengthened, utilising Machine Learning as a real world tool for data analysis.

The implementation task reveals a difference in programming style - one opting for array and index based access, another opting for dictionary lookup based access. This task was split between Preprocessing and K-Nearest algorithms , and Ten-Fold Cross Validation and Naive

Bayes algorithms. One unexpected challenge was having the different modules interoperate despite the difference in technique. For example, the implementation of Feature Selection used attribute names as dictionary keys to reference the subset, but the K-Nearest was originally programmed using indexes. Small chunks of code were written to translate between the two, instead of rewriting the system.

Program Instructions

The assignment is designed to be run from a unix bash console environment with Python 2.7.

The assignment archive can be extracted using the unix command:
unzip <assignment archive.zip>
cd <assignment archive>

To then execute the Data Preprocessing module:
python data_preprocessing.py

The preprocessing module assumes the unprocessed file "pima-indians-diabetes.data" is in the assignment directory. If it cannot be found the program quits with an error. By default the module writes the processed data to "pima.py". Alternatively custom input and output files can be used with the command:
python data_preprocessing.py <input file> <output file>

To execute the main analysis:
python tenFold.py

The tenFolds module assumes that the "pima.cvs" file is in the same directory. It writes the folded data out to "pima-folds.cvs".

References

[1] Weka - Waikato Environment for Knowledge Analysis. University of Waikato, New Zealand.
Available From: http://www.cs.waikato.ac.nz/ml/weka/ [accessed 2013 April 30]

[2] Pima Indians Diabetes Data Set, National Institute of Diabetes and Digestive and Kidney Diseases.
Available From: UCI Machine Learning Repository
URL: http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes [accessed 2013 April 30]

[3] WHO technical report series (1985). Technical Report 727, WHO Study Group.

[4] M. Hall 1999, Correlation-based Feature Selection for Machine Learning.

[5] Williams textbook of endocrinology (12th ed.). Philadelphia: Elsevier/Saunders. pp. 1371–1435. ISBN 978-1-4377-0324-5.

[6] Saedsayad.com (n.d.) OneR. [online] Available at: http://www.saedsayad.com/oner.htm.

[7] Saedsayad.com (n.d.) ZeroR. [online] Available at: http://www.saedsayad.com/zeror.htm.

[8]Brighthubpm.com (n.d.) Untitled. [online] Available at:http://www.brighthubpm.com/project-planning/106005-disadvantages-to-using-decision-trees/.