COMP3308 Introduction to Artificial Intelligence

Assignment Two

Pima Dataset Classification

Table of Contents

1. Introduction 3

1.1 Aim 3

1.2 Importance of the study 3

2. Data 3

2.1 Data Description 3

2.2 Attributes 4

2.3 CFS Attributes: 4

3. Results and Discussion 4

3.1 Stratified Cross Validation Accuracy (10-fold Average) 4

3.2 Classifier Performance Ranked 5

3.3 Overview of Classifier Performance 5

3.4 Python Implementation and Weka Comparison 7

3.5 Effect of CFS 7

4. Conclusion 8

4.1 Summary 8

4.2 Future Work Suggestions 8

5. Reflection 8

6. References 8

1. Introduction

1.1 Aim

In contemporary times, quality of life can be significantly affected by disease. According to the World Health Organisation, in 2019, an estimated 1.5 million deaths were directly caused by diabetes and approximately 90-95% of people affected by diabetes have type 2 *(WHO, 2021)*. Simultaneously, artificial intelligence (AI) and machine learning (ML) have seen rapid and widespread utilization in the hopes to provide useful information and to also solve real world problems such as that of diabetes.

This study will use several machine learning classifiers on the Pima Indians Diabetes Dataset and the aims of the study include the following:

- Python Implementation of Naive Bayes and K-Nearest Neighbour to classify if a person is likely to have diabetes or not.
- The implementation of eight different classifiers on the same dataset using software called Weka.
- Evaluating and comparing the performance of classifier accuracy using both Correlated-based feature selection (CFS) and no feature selection.

1.2 Importance of the study

There is an abundance of idle data that has the potential to save lives, solve complex problems and boost the knowledge of scientists, engineers, scholars, etc. The assistance of machine learning and classifiers can transform this data into useful information. In the context of this study, ML provides the potential to understand the attributes that contribute to type 2 diabetes which may see real world adaption.

Since there is great importance on the classifiers to provide information, it is crucial that they perform accurately, the performance of classifiers holds incredible importance in the future of decision-based computing. Evaluation methods such as cross fold validation are essential because an understanding of when classifiers are successful and unsuccessful allows for the integration of decision-based ML in the real world.

2. Data

2.1 Data Description

The Pima Indians Diabetes Data set is sourced from the National Institute of Diabetes and Digestive and Kidney Diseases which was donated on the 9th of May 1990. The data has also been modified in March 2015 by COMP3308 as there is a replacement of missing values with averages, and the class has been changed to nominal values.

The data includes 8 attributes with 768 instances and a class variable. If the class variable's value is "yes" it is interpreted as "tested positive for diabetes". There are 500 instances for the value “no” and 268 instances for the value “yes”.

There are several constraints on the data:

- All patients are females
- All patients are at least 21 years old
- All patients are of Pima Indian Heritage

2.2 Attributes

All attributes are numeric except class:

1. Number of times pregnant
2. Plasma glucose concentration 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 ("yes" or "no")

2.3 CFS Attributes

Correlation based feature selection (CFS) ranks attributes according to a heuristic evaluation function based on the correlations between the features and the classification. Weka has CFS built-in as a function and by applying the best-first search approach, the following features were selected:

1. Plasma glucose concentration 2 hours in an oral glucose tolerance test
2. 2-Hour serum insulin (mu U/ml)
3. Body mass index
4. Diabetes pedigree function
5. Age (years)

3. Results and Discussion

3.1 Stratified Cross Validation Accuracy (10-fold Average)

	ZeroR	1R	1NN	5NN	NB	DT	MLP	SVM	RF
No feature selection (%)	65.104	70.833	67.839	74.479	75.130	72.005	75.391	76.302	74.870
CFS (%)	65.104	70.833	69.010	74.479	75.130	73.307	75.781	76.693	75.912
Change (%)	0.000	0.000	+1.171	0.000	0.000	+1.302	+0.390	+0.391	+1.042

Figure 3.1.1 Weka Classifier Accuracy (3 d.p)

	My1NN	My5NN	MyNB
No feature selection (%)	69.265	76.299	75.120
CFS (%)	67.845	74.862	76.685
Change (%)	-1.420	-1.437	+1.565

Figure 3.1.2 Python Implementation Classifier Accuracy (3 d.p)

3.2 Classifier Performance Ranked

	ZeroR	1R	1NN	5NN	NB	DT	MLP	SVM	RF
No feature selection Rank	9	7	8	5	3	6	2	1	4
CFS Rank	9	7	8	5	4	6	3	1	2

Figure 3.1.3 Weka Classifier Accuracy Rank (1 Highest - 9 Lowest)

3.3 Overview of Classifier Performance

The classifier's performance can be assessed with many different methods, in this study 10-fold stratified cross validation has been performed in order to obtain an accuracy result for each predictive model. Cross validation is a technique that partitions the original data into a training set to train the model and a test set to evaluate it. Accuracy is simply calculated by dividing the correct classifications over the total number of classifications. However, in 10-fold cross validation the original sample is partitioned into 10 equal subsamples, where one of the subsamples serves as the testing data and the rest as training. This process is then repeated 10 times and each of the subsamples are used for testing exactly once. The folds are stratified so that each fold contains approximately the same proportion of class labels. Therefore, the performance result for a classifier in this study is the average accuracy of its 10 folds. The advantage of this method is that all observations are used for both training and testing, which can assist in flagging problems such as overfitting or selection bias for predicting new data.

For the purpose of this evaluation a criteria for performance is defined:

- If a classifier's accuracy is between 60% and 70%, it is considered a poor model.
- If the accuracy falls in between 70% and 80%, it is considered as a good model.
- If the accuracy falls in between 80% and 95%, it is considered as a great model.
- Accuracies above 95% can be dangerous with overfitting and need to be reviewed.

In general, most of the classifiers performed good, however, some performed poorly. Results for the Weka models are presented in *Figure 3.1.1* and the rank for each classifier is displayed in *Figure 3.1.3*, where rank 9 has the worst performance and rank 1 has the best.

ZeroR is the simplest classifier that has been used because it selects the majority class. ZeroR performed poorly with an accuracy of approximately 65.1%, and with CFS applied to the data, it still performed the exact same. The low accuracy can be explained by the simplicity of the classifier, however, even though the performance is not ideal, ZeroR can be used as a benchmark to compare the other models against.

The next classifier to evaluate is 1R which differs from ZeroR because it creates one rule that is used for each predictor in the data. 1R then selects the rule with the lowest total error and uses that as the one rule. Despite the classifier being relatively simpler than the other classifiers, it provides a decent accuracy of 70.8% for both with CFS and without. 1R performed approximately 5.6% better than ZeroR, the performance of 1R can be accredited to the fact that 1R selects the rule with the smallest total error.

The K-nearest neighbour classifier is an instance based lazy classifier. This means that it chooses the k nearest instances based on a distance measure such as in this context the Euclidean distance. The 1NN classifier provided an accuracy of 67.8%, which can be considered a fairly poor performance.

Interestingly, the 1NN classifier performed worse than 1R but better than ZeroR, with an overall rank of 7th. These results can be explained due to the fact that the classifier is extremely simple since the model is fit only to the 1-nearest point and this can mean that the model is very close to the training data, making it harder to predict.

The Naive Bayes (NB) classifier with the Probability Density Function is based on the Bayesian Theory of Probability and the assumption that the data is in a normal curve, and unrelated to each other. The results show that the effects of 75.13% accuracy with or without feature selection. It is the 3rd best without feature selection and 4th best with CFS, performing drastically better than the prior tests of ZeroR, and 1NN. The great accuracy can be explained by the data having a distribution somewhat close to normal distribution, so the accuracy of the probability density function is high. The size of the data is also not immense, so the fact that high sample size makes Naive Bayes less effective versus other more complex classifiers does not matter to an extent in this case.

DT is the decision tree classifier that takes rules (iteratively chosen to find the purest subtrees) at each point to create a list of subtrees that are closer to one or the other class. The DT algorithm ranked 6th regardless of feature selection. The accuracy of DT was approximately 72% and is noticeably lower than that of Naive Bayes. This lower accuracy could be due to overfitting, where it matches the folds that it trains using so well that it becomes closer to memorisation than to prediction. The results may be due to overfitting because it may become a problem as you scale the sample size.

The MLP classifier is the multi-layer perceptron classifier. It works by sending data from the input through multiple layers of hidden perceptron's, until it reaches the output layer. The hidden layers, and the output layer uses a differentiation function applied to a weighted sum of the previous layer's output as its output. Errors are calculated based on expected output vs actual output, and back propagated throughout the network to update weights. The result is approximately 75.39% without feature selection, and 75.78% with CFS. The noticeable increase makes sense, although should not be significant since weights can correct for the same sorts of correlation that feature selection does. It ranks 2nd without feature selection and 3rd with CFS. This makes sense as it allows for the data to self-correct, which is a lot better than methods such as Naive Bayes, which is one and done.

The SVM classifier involves the use of vectors to create a maximum margin hyperplane which is formed by training vectors resistant to overfitting. The results gave an accuracy of approximately 76% for both with and without CFS. This makes it the most accurate in both categories. The success of SVM does make sense since it has built-in ways of training itself, as well as being far more efficient to train than the backpropagation of perceptron's.

The Random Forest classifier uses bagging to modify the training data, and randomly selects features to limit the overlap, before creating multiple decision trees based on different choices in the bagging and random feature selection. The results were 74.87% and 75.9% for without and with feature selection. This makes it one of the classifiers with the highest change in accuracy. This can be explained because by getting rid of unnecessary features, the chances of getting affected by correlation is lesser. In addition to that, the amount changed is less than that of the decision tree. This also makes sense since it already limited the effects of correlation with the bagging, and random feature selection.

3.4 Python Implementation and Weka Comparison

Weka provides a collection of machine learning algorithms for data mining tasks. The python implementation of 1NN, 5NN and Naive Bayes provided similar results to ones obtained from Weka's classifiers. The best performing out of the three was 5NN for the python implementation and NB for the Weka classification. However, surprisingly, with CFS applied Naive Bayes performed the best for both the python implementation and Weka models. When CFS is applied the python implementation of NB performed with 76.7% compared to Weka's NB with 75.1% accuracy. Even though the python implementation with CFS performed slightly better, the majority of both CFS and non-CFS data provided similar accuracy results usually differing between 1-2%.

The slight difference in the python implementation and Weka results may be explained by the method within the 10-fold stratified cross validation. The method of partitioning the data is randomised from in Weka and therefore when starting a new session each time the data will be randomly partitioned into different sub-samples and therefore can provide different results to the python implementation. It is important to note that the difference in performance of both the python implementation and Weka classifiers are somewhat insignificant as the accuracy percentage change is on a small scale.

3.5 Effect of CFS

The goal of using Correlation-based Feature Selection (CFS) is to improve classifier accuracy. Since CFS filters the attributes to keep only those with high correlation to class, removing the potentially unnecessary features should improve classifier performance. The attributes selected by CFS seen in Section 2.3 make intuitive sense. Plasma glucose concentration, serum insulin levels, BMI, diabetes pedigree and age are known by health professionals and the general public to be important variables that indicate or influence the onset of type 2 diabetes.

Overall, the use of CFS on the Weka classifiers made a positive impact on accuracy. There are some cases where there was no change in accuracy such as ZeroR, 1R, 5NN and Naive Bayes. Perhaps this is the consequence of the classifiers' simplicity rather than CFS not working to improve accuracy.

Despite this, the DT, MLP, SVM and RF classifiers saw a slight improvement in the accuracy and performance of the classifier. The change can be seen in *Figure 3.3.1,* and it is evident that in general accuracy improved between 0-2%. The most notable differences are the DT classifier with +1.3% and RF classifier with +1.0% accuracy.

Since there are no cases where CFS had a negative impact on the performance of the classifier, the CFS method did have a positive and somewhat significant impact on the performance of the classifiers. However, it is important to consider that the data's dimensions are quite small and therefore the effect of CFS is lessened. It would be interesting to analyse the effect of CFS on a much larger pima dataset in the future.

Despite the success of CFS on the Weka classifier's performance, the python implementation classifiers, specifically 1NN and 5NN saw a decrease in accuracy as seen in *Figure 3.1.2.* Both classifiers' accuracies decreased by 1.4%. However, the python implementation of Naive Bayes saw an increase of 1.6%. The reasoning for this is unknown and to gain further insight into the effects of CFS more data is needed for testing. Overall, the effect of CFS was advantageous to the performance of the classifiers.

4. Conclusion

4.1 Summary

Overall, the findings of this study are the following:

- Some classifiers such as SVM, MLP and RF performed relatively better when tested against the pima data set.
- CFS does improve the accuracy and performance of classifiers.
- The python implementation of NB and 5NN can predict diabetes with decent percentage of accuracy (approximately 75%) with and without CFS.
- The choice of classifier is important because the results of this study showed some variation in the performance of each classifier.

Through this study and many others, Machine Learning has shown its potential to provide beneficial information and predictions for real world applications. Medical data similar to the one of the pima dataset can be used to calculate the severity or risk for diseases and ultimately save lives. This is just one of the many examples, and one of the countless applications of ML we will see in the future.

4.2 Future Work Suggestions

There are several recommendations for work:

- To ensure a more precise accuracy more methods can be implemented such as repeated k-fold stratified cross validation.
- It is important to beware and take caution with AI fairness and ethics in studies such as in this study and others.
- Different datasets can be used to test against the accuracy and performance of classifiers used in this study.
- Perhaps using classifiers together as an ensemble would perform better.
- Future work may include testing CFS on a much larger dataset.

Therefore, for there are many suggestions for future work in this study, and these can be applied to other machine learning projects.

5. Reflection

This study has built upon my understanding that there is incredible opportunity for machine learning and decision-making algorithms in the future. I learnt that building a classifier from scratch is not as hard as I initially thought. This assignment has reinforced my understanding of the 8 classifiers that were used, since I needed to re-visit lecture slides and research reasons to explain the results in the discussion.

I took the different range of classifiers and their differing results based on data as the most important thing learnt. It was really eye-opening to see the actual accuracy of different classifiers on the same dataset and compare it to personally made ones. It also showed me one could use similar techniques in the future to help evaluate for accuracy.

6. References

Who.int. 2021. *Diabetes*. [online] Available at: https://www.who.int/news-room/fact-sheets/detail/diabetes [Accessed 14 May 2021].