THE UNIVERSITY OF SYDNEY

Assignment 2 Report

COMP3608: Introduction to Artificial Intelligence (Advanced)

2024 Semester 1

SIDs: 510415022, 520422935

Contents

1 Aim

The primary aim of this investigation is to deepen our understanding of machine learning classification techniques through hands-on experience with building, evaluating, and comparing classifiers.

Developing two foundational classifiers from scratch – specifically, Naive Bayes and Decision Tree – and applying these to the Pima Indians Diabetes dataset and the Room Occupancy Estimation dataset helped us understand the foundational mechanics of these machine learning algorithms and the factors which influence their performance.

We further aim to understand and evaluate what makes one classifier more effective than another through comparing the performance of our own classifiers with open-source classifiers available through Weka. Throughout this comparative analysis, we are not only enhancing our practical skills in implementing machine learning algorithms but also improving our theoretical understanding of various classification methodologies and their suitability to specific types of data.

2 Data

This section describes the two datasets used in this investigation including their provenance and structure. It ends with a brief comparison of the two datasets.

2.1 Pima Indians Diabetes dataset

Our first dataset, *Pima Indians Diabetes Dataset* (hereinafter referred to as `pima`), aims to classify whether or not a patient has diabetes given a series of personal characteristics and diagnostic measurements.

2.1.1 Provenance and structure

The data was originally collected by the United States National Institute of Diabetes and Digestive and Kidney Diseases in 1988 for the purposes of predicting whether or not an individual would present with diabetes given certain diagnostic measures. The 768 patients in the dataset are all females of at least 21 years of age of Pima Indian heritage. 268 of the 768 tested positive for diabetes and 500 tested negative for diabetes. The population was sampled from near Phoenix, Arizona, and was chosen as people of Pima Indian heritage were found to have a high incidence rate of diabetes.

It can be said that the `pima` dataset is canonical in the world of machine learning: it has been used extensively in academic and research settings since its inception. Relevant to our purposes, the dataset was modified for COMP3308/3608 in 2015 and 2024 by filling in missing values and nominalising the class attribute. The dataset was also normalised using 0-1 normalisation prior to conducting any analysis.

There are a total of 8 independent predictor variables in the dataset, all some form of medical predictor (e.g. diastolic blood pressure), and one class variable. See Table 1 for the full metadata of each column.

The dataset has been provided to us in two formats: one numerical and one ordinal. The ordinal version has converted the 8 numerical attributes into classes ranging from “low” to “very high”. The motivation for this was to allow students to explore various classification techniques, both those which work with continuous data and discretised data.

The dataset has no null values and no missing values.

Column	Description	Data type	Range
tp	Number of times pregnant	Numeric	$[0, 17]$
gc	Plasma glucose concentration	Numeric	$[44, 199]$
bp	Diastolic blood pressure (mm Hg)	Numeric	$[24, 122]$
sft	Triceps skin fold thickness (mm)	Numeric	$[7, 99]$
si	2-Hour serum insulin (μU/ml)	Numeric	$[14, 846]$
bmi	Body mass index (weight in kg/(height in m)^2)	Numeric	$[18.2, 67.1]$
dpf	Diabetes pedigree function	Numeric	$[0.078, 2.42]$
age	Age (years)	Numeric	$[21, 81]$
class	Tested positive for diabetes or not	Factor	$[0, 1]$

Table 1: Data dictionary for the original Pima Indians Diabetes dataset

2.1.2 Rights for use

The dataset is under the CC0: Public Domain License. It is properly anonymised and does not contain any identifiable features of the patient subjects. We would like to acknowledge the creators and original source of the data:

> Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). *Using the ADAP learning algorithm to forecast the onset of diabetes mellitus.* In Proceedings of the Symposium on Computer Applications and Medical Care (pp. 261–265). IEEE Computer Society Press.

2.2 Room Occupancy Estimation dataset

Our second dataset, the *Room Occupancy Estimation* dataset (referred to as `occupancy`), aims to estimate a room's occupancy state (whether or not someone is in a room) based on environmental sensor values.

2.2.1 Provenance and structure

The `occupancy` dataset is much more recent than `pima`: the dataset was donated to the University of California Irvine's Machine Learning Repository in August of 2023, but was originally used for an academic paper in 2018. Data was collected from sensors in an office room measuring environmental variables over a period of time, including temperature, humidity, light, CO2 concentration, and humidity ratio. A webcam was also used to record room occupancy, providing a binary label for whether the room was occupied.

The original dataset contained 10129 observations with 16 columns tracking 5 variables (multiple sensors were used to track the same attribute). For the purposes of COMP3308/3608, the dataset was trimmed down to 2025 observations tracking 4 independent variables: temperature, light, sound, and CO2 content. See Table 2 for the complete metadata of each column in this modified dataset.

Similar to the `pima` dataset, two versions of the data were distributed: one ordinal and one numerical. There are no null or missing values in either of the versions.

2.2.2 Rights for use

This dataset is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license. We would like to acknowledge the creators and original source of the data:

> Adarsh Pal Singh, Vivek Jain, Sachin Chaudhari, Frank Alexander Kraemer, Stefan Werner and Vishal Garg, *"Machine Learning-Based Occupancy Estimation Using Multivariate Sensor Nodes,"* 2018 IEEE Globecom Workshops (GC Wkshps), 2018.

Column	Description	Data type	Range
temp	Temperature reading (°C)	Numeric	$[24.94, 26.38]$
light	Light reading (Lux)	Numeric	$[0, 165]$
sound	Output of an amplifier attached to a microphone (Volts)	Numeric	$[0.06, 3.83]$
CO2	Carbon dioxide reading (PPM)	Numeric	$[345, 1270]$
class	Room occupancy (no = no people, yes = 1-3 people)	Factor	[“no”, “yes”]

Table 2: Data dictionary for the modified COMP3608 Room Occupancy Estimation dataset

2.3 Comparison of Datasets

There are many similarities between the modified `pima` and modified `occupancy` datasets. Both `pima` and `occupancy` display a similar structure in their original datasets, containing a set of numeric columns with a single factor column. In the nominal versions of the datasets, both `pima` and `occupancy` also discretised their numeric data into categories from low to high/very high. Finally, all of the modified datasets contain no null or missing values.

Despite the similarities, there are also many key differences between the two datasets. One such difference is the time period in which the data has been collected. While `occupancy` was collected in 2018, `pima` had been collected two decades prior in 1988. Another key difference is the size of the two datasets: `pima` has 768 rows, while `occupancy` originally had 10129 rows, before being trimmed down to 2025. The number of features in the two datasets is also notably different, with `pima` having 8 input features while `occupancy` has only 4 input features.

Lastly, let's briefly touch on two incredibly influential factors in machine learning: class imbalance and the correlation between variables in a dataset.

- **Class imbalance:** This is the phenomenon where the number of instances of one class in a dataset significantly outnumbers the instances of other classes. As shown in Figure 1, both of our datasets exhibit this data quality issue, with `pima` having 268 `Yes` entries and 500 `No` classifications, and `occupancy` having 379 `Yes` entries and 1648 `No` classifications. This imbalance can lead to models which are biased towards the majority class, as ML algorithms are usually designed to maximise overall accuracy, even if this is at the cost of minority class performance. A potential solution to this is to resample the data, e.g. oversample the minority class.

- **Correlation between variables:** In the ideal case, each variable in a dataset used for classification purposes would be independent – that is, changing one variable has no impact on the others.

However, in practice this is not always the case. For example, Figures 2 and 3 show the correlation matrices for the `occupancy` and `pima` datasets. It is clear that there are high correlations between several features in both datasets, e.g. `bmi` (body mass index) and `sft` (triceps skin fold thickness) show a correlation of 0.54. The impact of this is *multicollinearity* between our variables, where one feature can be linearly predicted from another. This redundancy can degrade the performance of some algorithms.

We will explore the impact of these factors in section 3.3.

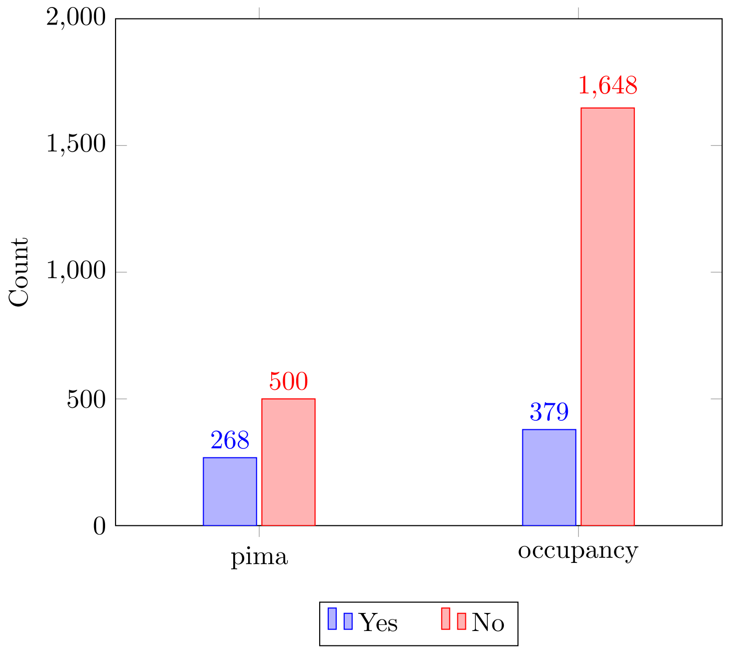

Figure 1: Comparing the class distribution of the `pima` and `occupancy` datasets

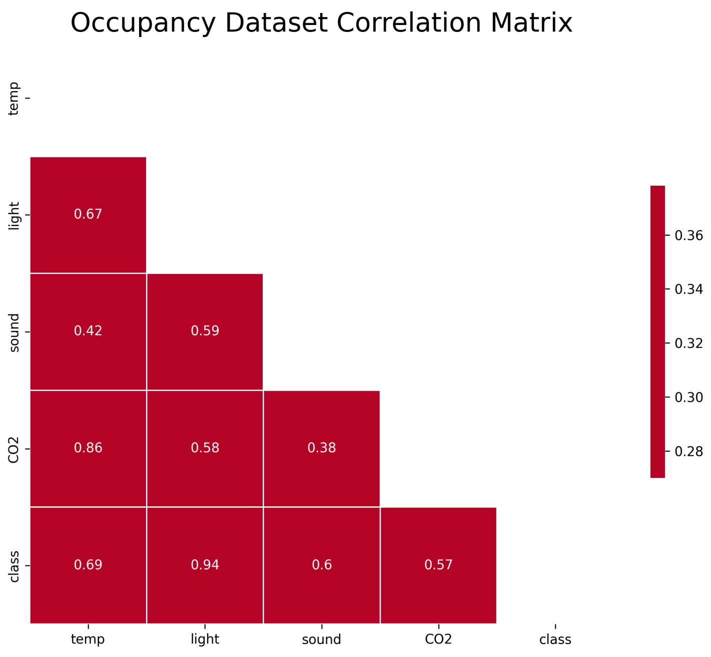

Figure 2: Correlation matrix for the `occupancy` dataset

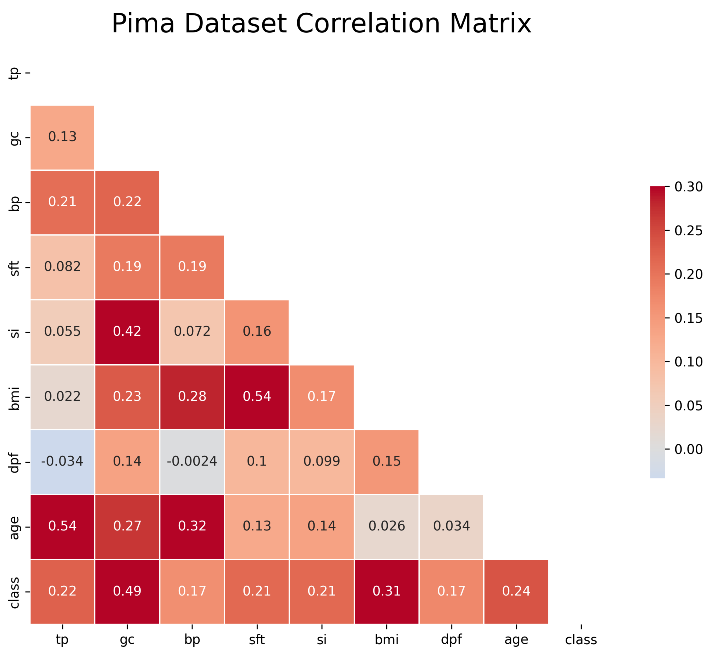

Figure 3: Correlation matrix for the `pima` dataset

3 Results and discussion

After building our own classifier for Naive Bayes and Decision Tree, we evaluated the accuracy of these along with other popular classification algorithms on our two datasets using stratified 10-fold cross validation. The results can be seen in Tables 3 and 4.

3.1 Continuous classifiers

In the `pima` dataset, the Support Vector Machine classifier showed the highest accuracy (76.30%) and F1-statistic (0.75) whilst maintaining a fast training time of 0.02 seconds. Comparatively, in the `occupancy` dataset, the 1-Nearest-Neighbour algorithm performed incredibly well with an accuracy of 99.41% and an F1-measure of 0.99. As expected, the ZeroR classifier performed the worst across both the `pima` and `occupancy` datasets with an accuracy of 65.10% and 81.28% respectively. The ZeroR classifier simply just predicts the majority class of the dataset (which explains the absence of F1 scores for it) and is used as a baseline to compare the performance of other classifiers.

Training times for all classifiers were generally low across both datasets. The one exception to this was the Multi-Layer Perceptron classifier which took 0.56s and 0.71s to train on `pima` and `occupancy` respectively, reflecting the higher computation cost of neural networks. Though MLP performed decently well on both datasets, the greater training time and resource load was not justified in its rankings amongst the suite of classifiers tested here – you could get comparable results from running a much simpler 5-Nearest-Neighbour algorithm.

Comparing our implementation of Naive Bayes against Weka's, MyNB displayed slightly lower performance than NB when factoring in both accuracy and F1-measures. For instance, in the `pima` dataset, MyNB's accuracy was 75.26% compared to NB's 75.13%, but its F1-statistic was notably lower (0.63 vs 0.75). This suggests that our implementation of Naive Bayes fails to balance precision and recall as effectively as Weka's NB. We discuss why this might be occurring ahead in section 3.3.

3.2 Nominal classifiers

When taking into account all three measures recorded from the classifiers seen in Table 4 (accuracy, training time, and the F1-statistic), the overall best performing model for classifying our discrete datasets was Boost with an accuracy of 78.91% on `pima` and 98.47% on `occupancy`. The F1-measure for Boost was one of the greatest at 0.79 for `pima` and 0.98 for `occupancy`, reflecting a balanced performance in terms of precision and recall. The model was also quick to train, averaging a training time of 0.025 seconds. In practice, boosting generally tends to outperform other tree-based classifiers such as Bagging and Random Forests as it is a compound classification model – it works by sequentially training models, each focused on fixing the errors of the previous ones. This approach allows the algorithm to adaptively improve itself.

The worst performing model for `pima` was our own implementation of Decision Tree, which correctly identified the class attribute just 73.18% of times and obtained an F1 score of 0.63. It also took the longest to train with a training time of 0.28 seconds – four times the duration of the Random Forest classifier which had the second-longest training time at 0.07 seconds. This is perhaps reflective of the lack of optimisation in our Decision Tree algorithm.

Dataset	Classifier	Accuracy (%)	Training time (s)	F1-Measure
pima	ZeroR	65.10	0.00	N/A
	1R	70.83	0.01	0.70
	1NN	67.84	0.00	0.68
	5NN	74.48	0.00	0.74
	NB	75.13	0.01	0.75
	MLP	75.39	0.56	0.75
	SVM	76.30	0.02	0.75
	MyNB	75.26	0.00	0.63
occupancy	ZeroR	81.28	0.00	N/A
	1R	98.47	0.00	0.98
	1NN	99.41	0.00	0.99
	5NN	99.21	0.00	0.99
	NB	96.74	0.00	0.97
	MLP	99.26	0.71	0.99
	SVM	98.42	0.01	0.98
	MyNB	96.84	0.00	0.92

Table 3: Accuracy, training time, and F-Measure for numerical classifiers

Dataset	Classifier	Accuracy (%)	Training time (s)	F1-Measure
pima	DT unpruned	75.00	0.00	0.75
	DT pruned	75.39	0.02	0.75
	MyDT	73.97	0.28	0.63
	Bagg	76.17	0.04	0.76
	Boost	78.91	0.03	0.79
	RF	73.18	0.07	0.73
occupancy	DT unpruned	98.42	0.00	0.98
	DT pruned	98.47	0.00	0.98
	MyDT	98.52	0.02	0.96
	Bagg	98.47	0.02	0.98
	Boost	98.47	0.02	0.98
	RF	98.52	0.03	0.99

Table 4: Accuracy, training time, and F1-Measure for nominal classifiers

In saying this, MyDT did not perform *incredibly* differently to Weka's own DT implementation. There is only a slight difference of 1.03% in the accuracy values when you compare our unpruned MyDT model to Weka's unpruned model. The difference is largely in the F1-statistic, which shows that our Decision Tree implementation may not be generalising well to new data. This may be due to an overly complex tree structure formed as a result of our splitting criteria; different methods of splitting, such as Gini impurity or in our case information gain, can lead to variations in the extent of the overfitting of the model to the training data. The extremely large DT formed through running our algorithm (see Listing 6) is perhaps indicative of this.

There is only a slight improvement when you compare Weka's unpruned and pruned versions of Decision Tree, as demonstrated by the meagre 0.39% and 0.05% improvements in accuracy across the two datasets. Unusually, the F1 scores of both the unpruned and pruned versions of Weka's DTs are equal. This is odd, as pruned DTs typically work to resolve overfitting issues and therefore should have a higher F1 score than their unpruned counterpart.

There was very little change between the different variations of DT in the `occupancy` dataset. Both our implementation and Weka's pruned and unpruned variations achieved accuracy values within 0.1% of one another.

3.3 Comparing datasets

There is a common trend across all of the classifiers that shows a significantly higher accuracy when run on `occupancy` than on `pima`.

When considering the size of the two datasets, the higher accuracy value of `occupancy` makes intuitive sense as `occupancy`'s dataset is much larger than `pima`'s, thus giving the classifier more instances to learn from and potentially improving its ability to generalize to unseen data. This pattern is further seen when looking at results for Bagg, Boost, as well as RF. All of the models perform remarkably similar on `occupancy`, but vary greatly when looking at `pima`.

However, when considering the number of input features, the much higher accuracy in `occupancy` compared with `pima` seems counter-intuitive as `pima` has many more input features and thus much more information. The issue with this line-of-thought has to do with the class imbalance and independence of variables present in the datasets, as discussed in section 2.3.

- **Class imbalance:** Both our datasets show a significant class imbalance. This is detrimental to classification algorithms as the classifier often develops a bias towards the majority class (in our case "no"). This leads to a poor recall for the minority class, which in turn impacts the F1 score. As we have not directly treated the class imbalance in our MyNB and MyDT algorithms, this could be part of the reason our F1 scores are lower than Weka's implementations.

- **Feature dependency:** As seen in Figures 2 and 3, the attributes in our datasets are not all completely independent of one another. This is particularly influential on the Naive Bayes classifier, which inherently assumes that all features are independent of each other (hence the "naive" name). This is strong evidence to explain why our Naive Bayes classifier particularly struggles in the `pima` dataset. To resolve this, we could use dimensionality reduction on attributes which are highly correlated.

4 Conclusion

This study has produced interesting insights into the difficulty of coding machine learning models from scratch. For both Naive Bayes and Decision Trees, our version of the individual classifiers performed marginally worse when compared with existing implementations of these classifiers in Weka. Nuances in the datasets also played a significant role in shaping the performance of classification algorithms, as oftentimes certain classifiers would perform slightly better on `pima` and slightly worse on `occupancy`.

Our results also show that more advanced machine learning models - such as Support Vector Machines and Boost - perform better on datasets where there is much more room for improvement in terms of accuracy (i.e. `pima`), whereas on datasets where the relationships are fairly obvious (such as `occupancy`), simpler models such as 1NN and our own implementation of decision trees have just as good if not better results than more complicated models.

Moving forward, further work for this study points towards the importance of understanding dataset structures and nuances, including issues like class imbalance, biased data, and the impact of feature engineering, as well as exploring other machine learning models that could be used to improve our results. Further investigations could delve into how feature selection affects model accuracy, and could potentially employ advanced techniques like deep learning neural networks to be able to locate more complex patterns in datasets such as `pima`.

By delving deeper into dataset analysis and leveraging more complex machine learning models, future studies can offer a comprehensive and nuanced perspective on challenges and solutions when developing classification algorithms.

5 Reflection

Throughout this research project we have gained an appreciation towards the art of choosing an appropriate machine learning pipeline for the relevant task at hand. Right from the beginning, the methods and techniques that one chooses to apply to a dataset – whether it is balancing an unequal distribution of classes, or performing some form of dimensionality reduction – can have a major impact on your final accuracy and F-statistic values. Choosing the appropriate machine learning classifier for the data is itself not an exact science; it requires some amount of trial-and-error to decide which algorithm finds a good balance between accuracy and training time.

What added to this appreciation was programming two canonical classification algorithms – Naive Bayes and Decision Tree – from scratch. This activity made us realise how important and influential each step of the classification algorithm truly is e.g. deciding how to split on an attribute in a Decision Tree can completely change your entire final result.

Finally, working on this project as a team helped us learn so much more than we ever could alone. Bouncing ideas off one another, especially when running into errors whilst programming the classification algorithms, and having insightful discussions not only enhanced our technical skills, but also honed our communication and collaboration abilities. We look forward to working on more data science projects in the future.

References

[1] Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). *Using the ADAP learning algorithm to forecast the onset of diabetes mellitus.* In Proceedings of the Symposium on Computer Applications and Medical Care (pp. 261–265). IEEE Computer Society Press.

[2] Singh, A.P., Jain, V., Chaudhari, S., Kraemer, F.A., Werner, S., & Garg, V. (2018). *Machine Learning-Based Occupancy Estimation Using Multivariate Sensor Nodes.* 2018 IEEE Globecom Workshops (GC Wkshps).

A Output of Decision Tree classifiers

A.1 Room Occupancy Estimation data

Listing 1: Unpruned Weka DT for Occupancy Estimation data

```
light = low
| temp = low: no (1126.0)
| temp = medium: no (446.0/25.0)
| temp = high
| | sound = low: no (100.0/3.0)
| | sound = high
| | | CO2 = low: yes (0.0)
| | | CO2 = medium: yes (2.0)
| | | CO2 = high: no (3.0/1.0)
light = high: yes (348.0)

// Accuracy 98.42%
```

Listing 2: Pruned Weka DT for Occupancy Estimation data

```
light = low: no (1677.0/31.0)
light = high: yes (348.0)

// Accuracy 98.47%
```

Listing 3: Our own DT classifier for Occupancy Estimation data

```
light = low
| temp = low: no
| temp = medium
| | sound = medium: no
| | sound = high: yes
| temp = high
| | sound = low: no
| | sound = high: yes
light = high: yes

// Accuracy 98.52%
```

A.2 Pima Indian Diabetes data

Listing 4: Unpruned Weka DT for Pima Diabetes data

```
gc = high
| bmi = high
| | sft = high
| | | tp = low
| | | | dpf = high
| | | | | age = high: yes (16.0/5.0)
| | | | | age = low
| | | | | | bp = high: yes (11.0/5.0)
| | | | | | bp = low: no (5.0/2.0)
| | | | dpf = low
| | | | | bp = high: no (43.0/19.0)
| | | | | bp = low: yes (10.0/4.0)
| | | tp = high
| | | | bp = high: yes (29.0/8.0)
| | | | bp = low
| | | | | dpf = high: no (2.0)
| | | | | dpf = low: yes (3.0)
| | sft = low: no (13.0/4.0)
| bmi = low: no (29.0/4.0)
gc = low
| bmi = high
| | si = high
| | | age = high
| | | | dpf = high: yes (7.0/3.0)
| | | | dpf = low: no (28.0/4.0)
| | | age = low: no (43.0/4.0)
| | si = low: no (48.0/2.0)
| bmi = low: no (66.0)
gc = very high
| si = high
| | bmi = high: yes (103.0/16.0)
| | bmi = low
| | | age = high: yes (12.0/3.0)
| | | age = low: no (4.0/1.0)
| si = low: no (3.0/1.0)
gc = medium
| age = high
| | si = high
| | | bmi = high
| | | | dpf = high: yes (37.0/10.0)
| | | | dpf = low
| | | | | bp = high: no (57.0/24.0)
| | | | | bp = low
| | | | | | sft = high: yes (15.0/7.0)
| | | | | | sft = low: no (3.0/1.0)
| | | bmi = low: no (27.0/3.0)
| | si = low: no (8.0)
| age = low
| | bmi = high
| | | tp = low
| | | | sft = high
| | | | | dpf = high
| | | | | | bp = high: no (17.0/2.0)
| | | | | | bp = low: yes (7.0/3.0)
| | | | | dpf = low: no (54.0/8.0)
| | | | sft = low: no (24.0/1.0)
| | | tp = high: yes (2.0/1.0)
| | bmi = low: no (42.0/1.0)

// Accuracy 75.00%
```

Listing 5: Pruned Weka DT for Pima Diabetes data

```
gc = high
| bmi = high
| | sft = high: yes (119.0/51.0)
| | sft = low: no (13.0/4.0)
| bmi = low: no (29.0/4.0)
gc = low: no (192.0/14.0)
gc = very high: yes (122.0/24.0)
gc = medium
| age = high
| | bmi = high
| | | dpf = high: yes (37.0/10.0)
| | | dpf = low: no (80.0/33.0)
| | bmi = low: no (30.0/3.0)
| age = low: no (146.0/17.0)

// Accuracy 75.39%
```

Listing 6: Our own DT classifier for Pima Diabetes data

```
b = high
| f = high
| | h = high
| | | g = high
| | | | c = high
| | | | | a = low: yes
| | | | | a = high: yes
| | | | c = low
| | | | | a = low: yes
| | | | | a = high: no
| | | g = low
| | | | d = high
| | | | | a = high
| | | | | | c = high: yes
| | | | | | c = low: yes
| | | | | a = low: yes
| | | | d = low
| | | | | a = low
| | | | | | c = high: yes
| | | | | | c = low: no
| | | | | a = high: no
| | h = low
| | | d = high
| | | | c = high
| | | | | g = high
| | | | | | a = low: no
| | | | | | a = high: yes
| | | | | g = low
| | | | | | a = low: yes
| | | | | | a = high: no
| | | | c = low
| | | | | a = high: no
| | | | | e = low
| | | | | | g = high
| | | | | | | a = high: yes
| | | | | | | a = low: no
| | | | | | a = low: no
| f = low
| | d = high
| | | h = high
| | | | e = high: no
| | | | g = low
| | | | | a = high: yes
| | | | | a = low: no
| | | e = low
| | | | c = high
| | | | | g = high
| | | | | | d = low
| | | | | | | a = high: no
| | | | | | | a = low: no
| | | | | d = high: no
b = low
| f = high
| | e = high
| | | h = high
| | | | c = low
| | | | | d = low
| | | | | | g = high
| | | | | | | a = low: no
| | | | | | | a = high: no
| | | | | | a = high: no
| | | | c = high
| | | | | g = low
| | | | | | c = high
| | | | | | | a = high: no
| | | | | | | a = low: no
| | | | | | c = low
| | | | | | | d = high
| | | | | | | | a = high: yes
| | | | | | | | a = low: no
| | | | | | | a = low: no
| | | c = low: no
b = very high
| e = high
| | f = low
| | | a = high
| | | | g = high
| | | | | c = high
| | | | | | a = low: yes
| | | | | | a = high: yes
| | | | | a = low: yes
| | | | c = low
| | | | | g = high
| | | | | | a = high: yes
| | | | | | a = low: yes
| | | | | a = low: no
| | f = high
| | | h = high
| | | | a = high
| | | | | g = high: yes
| | | | | c = low
| | | | | | d = high
| | | | | | | a = high: yes
| | | | | | | a = low: yes
| | | | | g = low
| | | | | | c = high: yes
| | | | | | a = low: yes
| | | g = low
| | | | a = high: yes
| | | | c = low
| | | | | d = high
| | | | | | a = high: yes
| | | | | | a = low: yes
| | | | | a = low: yes
| f = low
| | g = low: no
| | g = high: yes
b = medium
| h = high
| | f = low
| | | c = high
| | | | a = low
| | | | | g = low: no
| | | | | g = high: no
| | | | a = high: no
| | | c = low
| | | | a = low
| | | | | d = high: no
| | | | | d = low: no
| | | | a = high: yes
| | f = high
| | | g = low
| | | | e = high
| | | | | c = high
| | | | | | a = high
| | | | | | | a = low: no
| | | | | | | a = high: no
| | | | | c = low: yes
| | | | e = low: no
| | | g = high
| | | | a = low
| | | | | d = high
| | | | | | c = high: yes
| | | | | | c = low: yes
| | | | | d = low: yes
| | | | a = high: yes
| h = low
| | f = high
| | | d = high
| | | | g = high
| | | | | c = high
| | | | | | e = high
| | | | | | | e = low: no
| | | | | | | e = high: no
| | | | | c = low
| | | | | | e = low
| | | | | | | e = high: yes
| | | | | | | e = low: yes
| | | d = low
| | | | g = low: no
| | f = low
| | | d = high
| | | | g = high
| | | | | c = high
| | | | | | e = high
| | | | | | | a = high: no
| | | | | | | a = low: no
| | | | | | e = low
| | | | | | | a = high: yes
| | | | | | | a = low: yes
| | | | | c = low
| | | | | | d = high
| | | | | | | g = high
| | | | | | | | a = high: yes
| | | | | | | | a = low: yes
| | | | | | d = low
| | | | | | | g = high
| | | | | | | | a = high: no
| | | | | | | | a = low: no
| | | | | | a = low: no
| | | | g = low
| | | | | e = high
| | | | | | a = high: no
| | | | | | a = low: no
| | | | | e = low
| | | | | | a = high: yes
| | | | | | a = low: yes
| | | d = low
| | | | g = low: no
| | | | e = high: no
| | | | c = low
| | | | | d = high: yes
| | | | | a = low: yes

// Accuracy 73.97%
```