COMP3308 Assignment 1
Predicting Diabetes

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

1 Aim

The aim of our study is to predict whether a new patient will test positive for diabetes (class 1). The study is important because it can help to determine what sort of factors can lead to having diabetes, and as a result, what can be adjusted in a patient's lifestyle to lessen the chance of diabetes.

2 Data

2.1 Dataset

The data set we used was the Pima Indians Diabetes Database found at `http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/`.
There are nine attributes:

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)

and two classes:

1. class1 (testing positive for diabetes)
2. class0 (testing negative for diabetes)

2.2 Data preparation

2.2.1 Manual preprocessing

The raw `.data` file was preprocessed manually in multiple ways. Firstly we had to come up with simple names for the nine attributes:

1. num_pregnant
2. plasma_glucose_concentration
3. diastolic_blood_pressure
4. tricep_skin_fold_thickness
5. 2h_serum_insulin
6. bmi
7. diabetes_pedigree_function

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

8. age
9. class

Using these names, we added a header row the `.data` file. We then changed the values of the class column to class0 and class1 instead of 0 and 1. This file was then saved a `.csv` file for later convenience.

2.2.2 Missing Values

We wrote a script `csv_scripts\missing_values.py` that dealt with the missing values in the following attributes:

1. plasma_glucose_concentration
2. diastolic_blood_pressure
3. tricep_skin_fold_thickness
4. 2h_serum_insulin
5. bmi
6. diabetes_pedigree_function

We determined that these attributes consisted of missing values because 0 is an impossible or unrealistic value.

This script outputs two `.csv` files, one with missing values taking the calculated average of its respective attribute(`pima-indians-diabetes-avg.csv`) and the other with any instances with missing values omitted(`pima-indians-diabetes-omit.csv`).

We noticed that if we omit the instances with missing values we would be left with 392 instances out of the original 768 (only 51%!), so we decided to use the averaged output file for the rest of the assignment.

2.2.3 Normalisation

Once this new file was loaded into Weka, each of the attributes were normalised in the range $[0, 1]$. The final output file was saved as `pima.csv`

2.3 Attribute selection

With the `pima.csv` loaded into Weka, we used the `CfsSubsetEval` attribute evaluator with the `BestFirst` search method available on the `Select attributes` tab to determine the best subset of features based on how good the individual feature are at predicting the class and how much they correlate with the other features. The selected attributes were:

1. plasma_glucose_concentration
2. 2h_serum_insulin
3. bmi
4. diabetes_pedigree_function
5. age

We then proceed to use Weka's preprocessing features to remove the attributes that were not selected and saved the file as `pima-CFS.csv`.

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

3 Results and Discussion

3.1 Results from Weka

	ZeroR	1R	1-NN	5-NN	NB	DT	MLP
No feature selection	65.1042	70.8333	67.7083	74.7396	75.0000	72.2656	75.3906
Correlation-based feature selection	65.1042	70.8333	69.0104	74.6094	76.4323	73.9583	75.7813

Table 1: Results from Weka (all values in %)

ZeroR predicts class0 with the same percentage in both sets of results as it just counts which instance is more popular. In our datasets, we had 500 instances of class0 and 268 of class1, giving us the same result of 65.1042%. Similarly, 1R picked the plasma_glucose_concentration attribute and counted instances, giving us the same result of 70.8333%.
For all the other algorithms, the performance was generally better with the feature selection. The 5-NN was much more accurate than the 1-NN in both sets, as would be expected by the selection and comparison of more neighbours. NB and MLP were the most accurate algorithms, possibly due to their effectiveness on such simple datasets.
MLP displayed a training time of 0.47s on the complete data (0.24s on CFS data), while all other algorithms ran instantly (displayed 0s). We believe that these low running times are due to the small size of the datasets along with the relative simplicity of the algorithms.

need to talk more about accuracy of each algo?

3.2 Results from own implementation

	My1-NN	My5-NN	MyNB
No feature selection	68.7645	75.2546	75.3828
Correlation-based feature selection	68.4979	76.0509	76.6883

Table 2: Results of implementations of kNN and NB (all values in %)

Once again, feature selection generally produced more accurate results from our algorithms. Our implementations had very similar results to the Weka output, which was pleasant to see.

3.3 General discussion

Weka's CFS provided us with a set of five attributes as seen in section 2.3 above. The selection was somewhat intuitive; the excluded attributes were num_pregnant, diastolic_blood_pressure and tricep_skin_fold_thickness. We could not reason why the number of pregnancies or minimum blood pressure would contribute to or result from diabetes. Tricep skin fold thickness, however, is a measure of body fat percentage, which is related to BMI (a feature that was selected) and may have been excluded due to its similarity and possibly redundant nature in the dataset. In both sets of results, it was apparent that correlation-based feature selection was beneficial - using the CFS results in both Weka and our own algorithms provided more accurate results.

4 Conclusions

Firstly, it should be noted that all the attributes tested were relevant in predicting diabetes (although some may have not been intuitively so for us). This is due to the fact that the CFS results were only slightly more

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

accurate than the results with no feature selection (up to around 1.8% in both result sets). However, the use of CFS was beneficial as it allowed us to narrow down the data sets to the most relevant attributes and get a better performance.

For future experiments and research in this topic, it would be ideal to have less fragmented data and missing values. This will allow us to create a more accurate and realistic classifier where it does not need to take an average value of an attribute.

5 Reflection

The most important thing we learned from this assignment was the aspect of teamwork. We split the tasks we had to do and took turns doing various parts of coding or writing. It was especially beneficial when we could have each other double check code or results which led to us both having a more correct understanding of the implementation or results at hand. Also, splitting the work allowed us to have a more fun and easier experience with the assignment.

Another thing we learnt was that artificial intelligence can be very interesting and quite fun to code. It has given us ideas to build other classifiers for a variety of different research topics. It has ultimately given us a tool for analysis and prediction for future research and development.

6 Instructions

Our implementation of K Nearest Neighbour, Naive Bayes and 10fold stratified cross validation was written in Python 2.7.5.

The `ai_main.py` file runs 10-fold stratified cross validation on K Nearest Neighbour ($k = 1$ and $k = 5$) and Naive Bayes on a given `.csv` file.

To run our implementation, you must `cd` into the `ai_code` directory and ensure that `pima.csv` and `pima-CFS.csv` are present in the directory. Then run:

```
python ai_main.py pima.csv
python ai_main.py pima-CFS.csv
```

This should output `pima-folds.csv` and `pima-CFS-folds.csv` respectively, which are the stratified folds.

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865