CS8803 BDS / CS4365

Homework Assignment 4

(Programming Category)

Student Name:___________Enmao Diao______________
Student Session: cs8803 or CS4365 (circle one)

In this 4th programming assignment, you are given two types of programming problems to gain hand-on experience with ensemble learning. You are required to choose only one of the two problems.

Post Date: Monday of Week 9 (Oct. 19)
Due Date: Midnight on Saturday of Week 11 (Oct. 31 with no penalty grace period until midnight on Sunday of Nov. 1)

Problem 2. Hand-on Experience with Classification Ensemble
Your task for this assignment is to select 4~5 classification algorithms and run all of them on a chosen dataset and then use a classification ensemble method, such as majority voting, to show how much an ensemble method can improve the learning accuracy.

1. Select a dataset from the UCI repository (http://archive.ics.uci.edu/ml/datasets.html) or use a dataset of your own choice.
2. Determine how you will measure the quality of the clusters produced. Some reference on clustering evaluation can be found at http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html.
3. Choose 4~5 different implementations of one classifier, such as the C4.5 decision tree classifier or Naïve Bayesian classifier or SVM or any other classifier that you are familiar with. You can find them from Weka (http://www.cs.waikato.ac.nz/ml/weka/), Mahout (http://mahout.apache.org/users/classification/) or R (http://www.rdatamining.com).
4. Evaluate the individual classifiers using the chosen dataset from UCI repository. One example is the mushroom dataset from UCI repository. The training dataset contains 7423 records and the test dataset 701 records. The first attribute is the class of each record and the rest 21 attributes are categorical attributes.
5. Write a brief report that include the following: Present and discuss the results of your experiments on the chosen dataset with each of the chosen individual classifiers and the classification ensemble; and discuss the experiences and lessons you have learned from the experimentation.

6. Deliverable:
    - One tar or zip file that contains your source files (if available), the executable, a readme file explaining how to compile/run your program.
    - The output file for the test dataset screen shots of your execution process.
    - Runtime statistics in excel plots or tabular format.
    - Report in pdf/word/ppt.

1) I use the full mushroom dataset downloaded from UCI repository. I train and test all classifiers on Weka since it provides the largest number of classifiers and it is very easy to use.
   I choose following classifiers to elaborate including
   - Naïve Bayes
   - C4.5 (J48)
   - Random Forest
   - SVM (SMO)
   - AdaBoost

2) I will measure my classifiers based on following standards:
   - Correctly Classified ratio (Naive metric)
   - ROC Area (Golden metric, how good this model is)
   - Kappa statistic (Agreement of prediction with true class, to measure how good this model is better than chance)
   - Root mean squared error (average magnitude of the error, large errors are not desirable)
   - Root relative squared error (what it would have been if a simple predictor had been used)

3)
Preprocess:
First I download the dataset and Weka from the web.
After I try some examples like
http://machinelearningmastery.com/how-to-run-your-first-classifier-in-weka/
http://www.ibm.com/developerworks/library/os-weka2/
I understand the general process of running Weka. It is so simple that I don't even want to touch Mahout.
Then I try to input mushroom dataset and I realize I would better turn the raw data into .arff format which is used by Weka default examples.
Next I read the raw file like example data file iris.arff. At first I decided to write a program to transform mushroom dataset tinto .arff format. After think a while, I believe that there is no need to do a write a parse program for a non-standard dataset. Then I spend a few minutes manually adding following head information to the mushroom dataset. I also get error about using the dataset with one character representing a word like 'e' for EDIBLE. Weka reports that I could not have same value for two different attributes. Then I switch to expanded version of mushroom dataset and works fine.

```
@RELATION mushroom

@ATTRIBUTE class        {EDIBLE,POISONOUS}
@ATTRIBUTE cap-shape string
```

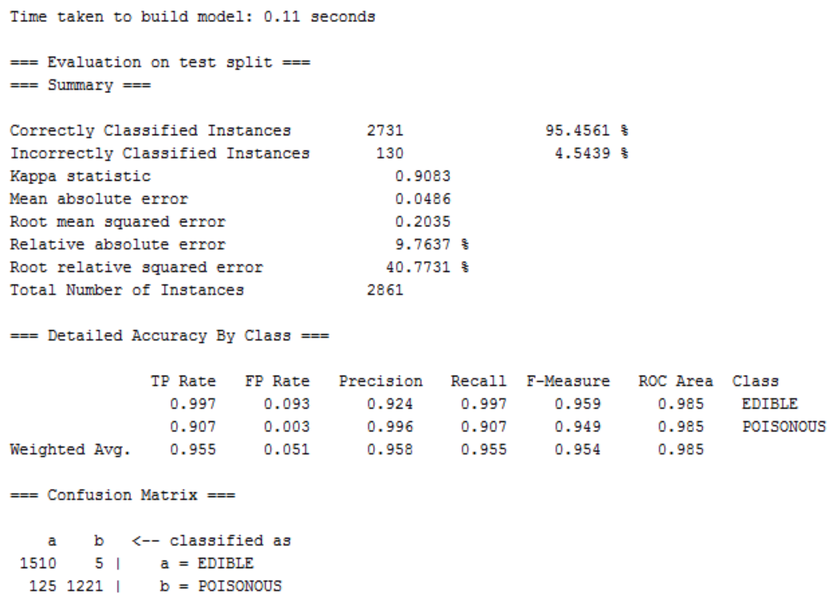
```
@ATTRIBUTE cap-surface string
@ATTRIBUTE cap-color string
@ATTRIBUTE bruises? string
@ATTRIBUTE odor string
@ATTRIBUTE gill-attachment string
@ATTRIBUTE gill-spacing string
@ATTRIBUTE gill-size string
@ATTRIBUTE gill-color string
@ATTRIBUTE stalk-shape string
@ATTRIBUTE stalk-root string
@ATTRIBUTE stalk-surface-above-ring string
@ATTRIBUTE stalk-surface-below-ring string
@ATTRIBUTE stalk-color-above-ring string
@ATTRIBUTE stalk-color-below-ring string
@ATTRIBUTE veil-type string
@ATTRIBUTE veil-color string
@ATTRIBUTE ring-number string
@ATTRIBUTE ring-type string
@ATTRIBUTE spore-print-color string
@ATTRIBUTE population string
@ATTRIBUTE habitat string

@DATA
```

Then, I have problem with training the classifiers. No classifiers allow me to start on this dataset because they do not allow string attributes. After search on the web, I find out that I can transform the string attributes by using **Filter** like StringToWordVector, StringToNominal under preprocess tab. After finish this, I can run many classification algorithms on Weka now.

Algorithms:
I include all outputs into ~/output folder. These outputs come from training on 66% of data and test on the remainder.

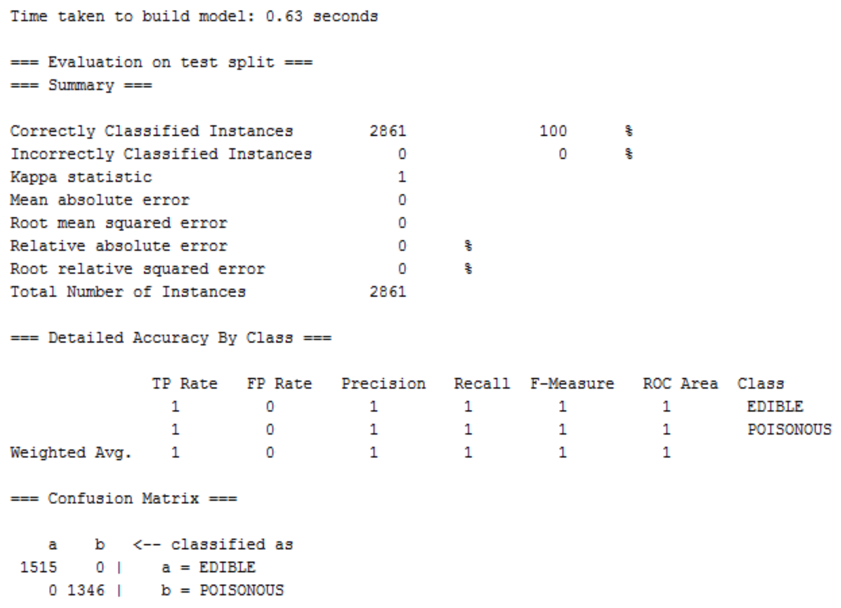
Sample Screenshots

Naïve Bayes



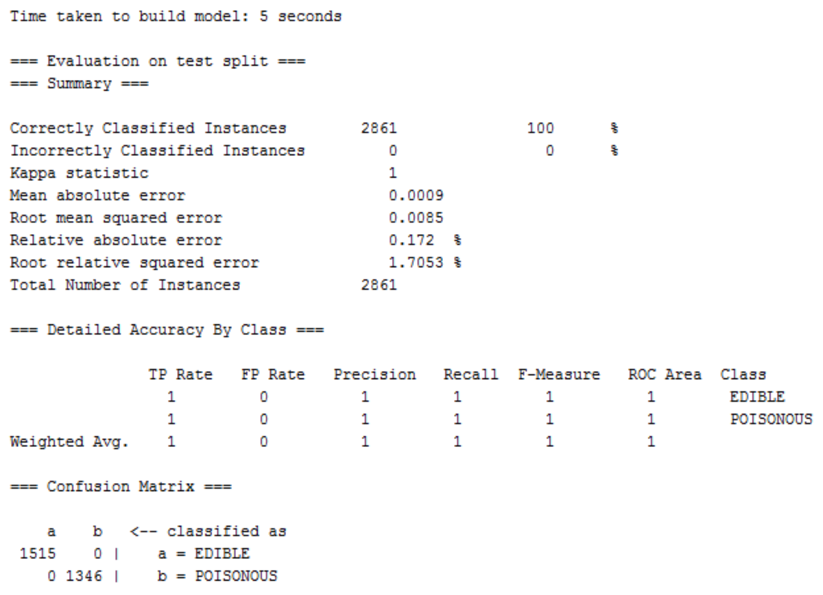
```
Time taken to build model: 0.11 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances        2731               95.4561 %
Incorrectly Classified Instances       130                4.5439 %
Kappa statistic                          0.9083
Mean absolute error                      0.0486
Root mean squared error                  0.2035
Relative absolute error                  9.7637 %
Root relative squared error             40.7731 %
Total Number of Instances             2861

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.997     0.093      0.924     0.997     0.959      0.985    EDIBLE
                 0.907     0.003      0.996     0.907     0.949      0.985    POISONOUS
Weighted Avg.    0.955     0.051      0.958     0.955     0.954      0.985

=== Confusion Matrix ===

    a    b   <-- classified as
 1510    5 |    a = EDIBLE
  125 1221 |    b = POISONOUS
```

C4.5 (J48)



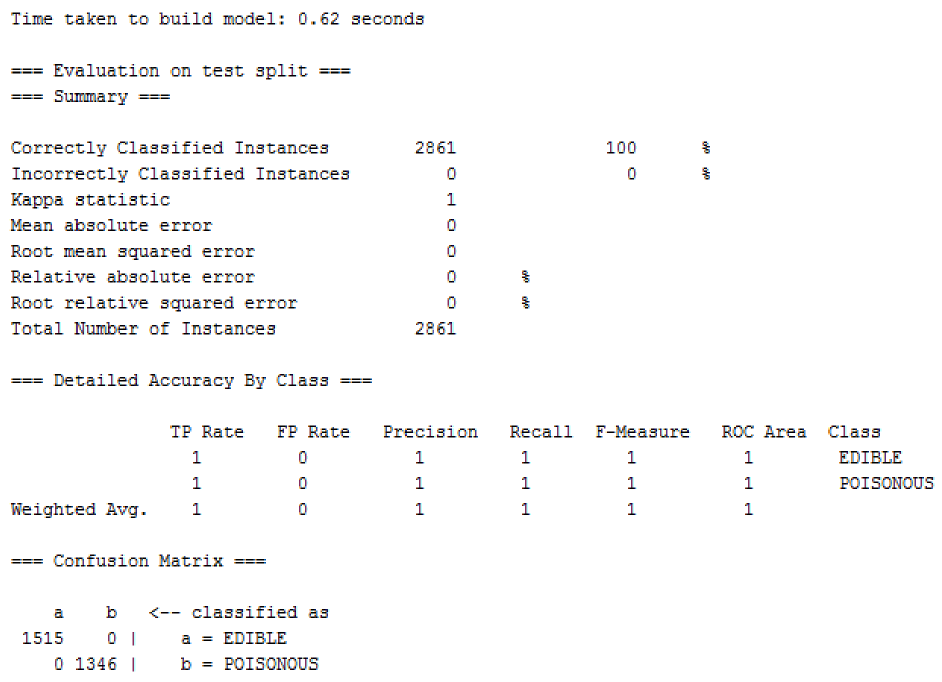
```
Time taken to build model: 0.63 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances        2861              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0      %
Root relative squared error              0      %
Total Number of Instances             2861

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        EDIBLE
                 1         0          1         1         1          1        POISONOUS
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

    a    b   <-- classified as
 1515    0 |    a = EDIBLE
    0 1346 |    b = POISONOUS
```

Random Forest



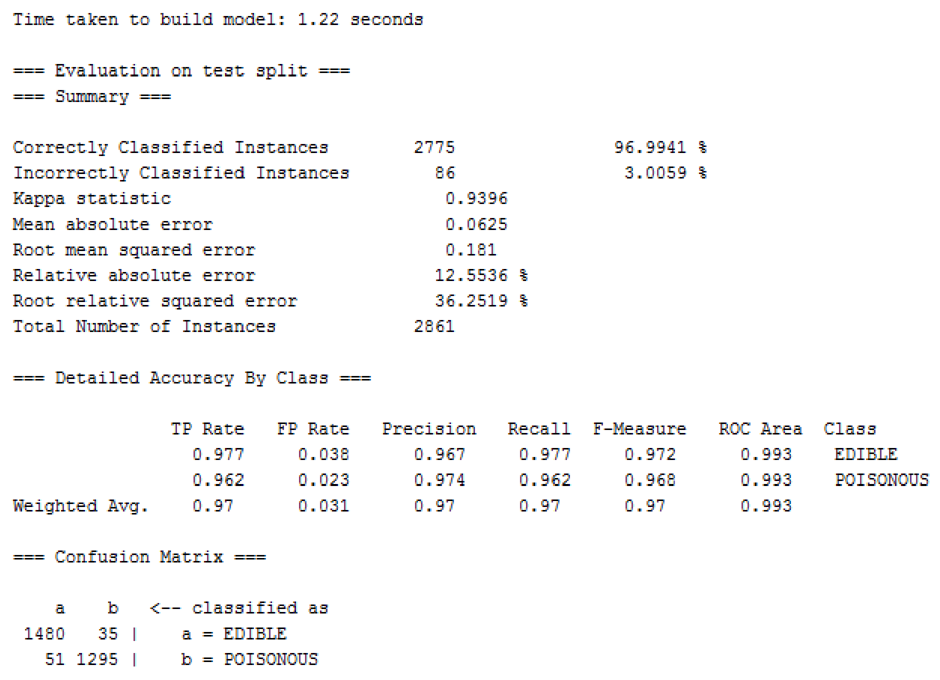
```
Time taken to build model: 5 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances        2861              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0.0009
Root mean squared error                  0.0085
Relative absolute error                  0.172  %
Root relative squared error              1.7053 %
Total Number of Instances             2861

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        EDIBLE
                 1         0          1         1         1          1        POISONOUS
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

    a    b   <-- classified as
 1515    0 |    a = EDIBLE
    0 1346 |    b = POISONOUS
```

SVM (SMO)



```
Time taken to build model: 0.62 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances        2861              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0      %
Root relative squared error              0      %
Total Number of Instances             2861

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        EDIBLE
                 1         0          1         1         1          1        POISONOUS
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

    a    b   <-- classified as
 1515    0 |    a = EDIBLE
    0 1346 |    b = POISONOUS
```

AdaBoost



```
Time taken to build model: 1.22 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances        2775               96.9941 %
Incorrectly Classified Instances        86                3.0059 %
Kappa statistic                          0.9396
Mean absolute error                      0.0625
Root mean squared error                  0.181
Relative absolute error                 12.5536 %
Root relative squared error             36.2519 %
Total Number of Instances             2861

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.977     0.038      0.967     0.977     0.972      0.993    EDIBLE
                 0.962     0.023      0.974     0.962     0.968      0.993    POISONOUS
Weighted Avg.    0.97      0.031      0.97      0.97      0.97       0.993

=== Confusion Matrix ===

    a    b   <-- classified as
 1480   35 |    a = EDIBLE
   51 1295 |    b = POISONOUS
```

Summary

First I want to determine which one is the best classifier when splitting from 66%. Classifier with higher ROC Area and Correctly Classified ratio should be better. If all equal I use other criterions like Kappa statistic, Root mean squared error and Root relative squared error.

Table I Classifier Performance with fixed training set ratio 66%

	ROC Area	Correctly Classified ratio	Kappa statistic	Root mean squared error	Root relative squared error
C4.5(J48)	1	100%	1	0	0
SVM (SMO)	1	100%	1	0	0
Random Forest	1	100%	1	0.0085	1.7053%
AdaBoost	0.993	96.9941%	0.9396	0.181	36.2519%
Naïve Bayes	0.985	95.4561%	0.9083	0.2035	40.7713%

Based on the result, clearly Naïve Bayes and AdaBoost performs worse than the other three classifiers. However, it is not quite clear which one is better because all of them have done a good job when splitting the dataset by 66%. Therefore, I decided to make a plot on their performance when given less training set.

The raw data I get is showing below

Table II Classifier Correctly Classified ratio with varying training set instances

Train set Instances	0	1	5	10	50	100	200	500	1000	2000	4000	5000	5555	6500	7000	8416
Naive Bayes (%)	53.3	62.7	80.4	89.8	90.2	92.4	95.8	96.4	96.2	96.1	95.8	95.6	95.5	95.4	96.0	95.4
C4.5 (J48) (%)	0	46.7	76.0	58.9	81.9	95.3	98.0	98.4	99.3	99.6	99.8	99.9	100	100	100	100
Random Forest (%)	NA	46.7	79.6	88.3	93.9	97.1	97.8	99.9	99.9	100	100	100	100	100	100	100
SVM (SMO) (%)	NA	46.7	83.3	88.4	93.9	96.9	97.7	99.5	99.8	100	100	100	100	100	100	100
AdaBoost (%)	NA	46.7	76.0	89.3	93.0	91.6	94.9	97.1	97.2	97.2	94.6	94.5	97.0	96.9	96.8	97.2

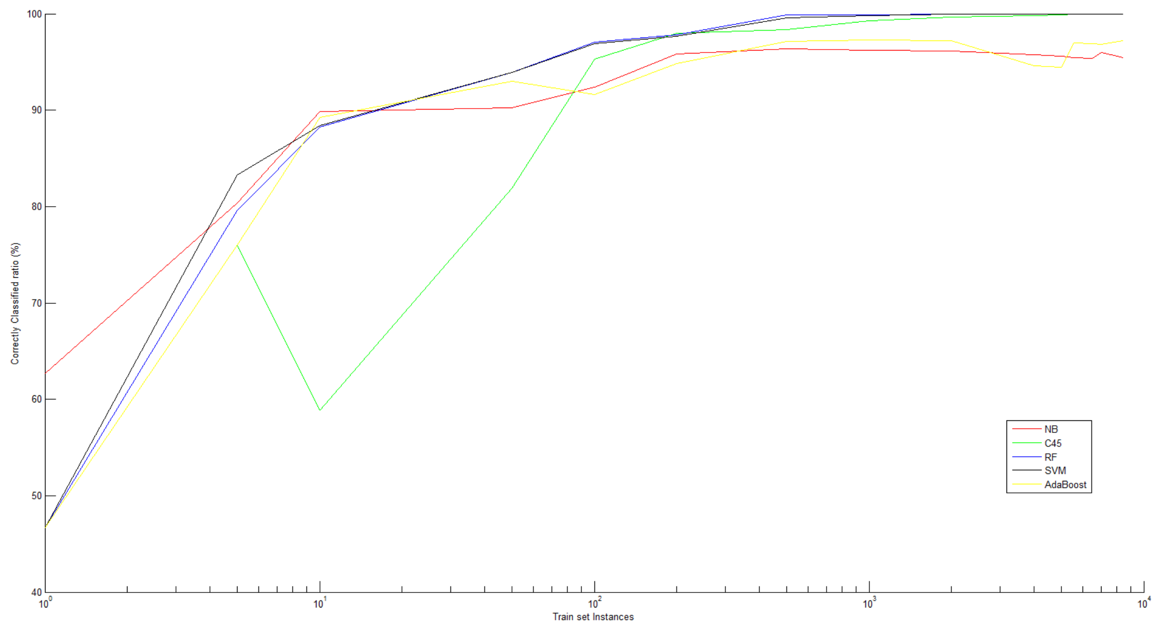


Figure 1. Correctly classified ratio vs training set instances in log scale.

I learn following things from this plot.

1. For these four classifiers, more training data does not necessarily mean it always classify better. While the ratio is not monotonically increasing, large training set generally yield good performance.
2. For mushroom dataset, the best classifier should be Random Forest and SVM because their correctly classified ratio is smoothly increasing and also have good performance when given very limited amount of training data, especially SVM. Moreover, their performance can reach up to 100% when have enough training data. Although C4.5 also can achieve that, it has unstable performance when given limited amount of training data.
3. Naïve Bayes and AdaBoost performs very likely. They do not reach 100% accuracy even train and test on the whole dataset. The ratio is also not monotonically

increasing and sometimes less training data can yield result better than train on the whole dataset.

Thought:

1. One thing I notice is even given only 5 or 10 training data, most classifiers can give out acceptable result. I guess it is because mushroom dataset has so many attributes and actually these attributes determine classification effectively in nature. If we have attributes with less influences on classification, we might result in random guesses. One thing for sure is that data mining is powerful, but right now I think the data we gather and how we gather the data is also crucial since it fundamentally determines the prediction accuracy so that we can avoid random guesses.
2. Without doubt, data mining algorithm is powerful. For this mushroom dataset, can I build an application which helps people identifying poisonous mushrooms? Therefore are some limitations like users might only be able to provide small amount of attributes value since they are not experts. Whether a mushroom is poisonous or not is very sensitive, since people may lose their lives because they eat poisonous mushrooms. Even if we prove we can achieve 100% accuracy, can we really trust data mining on this? This ethical issue is very hard to resolve because users have no idea how the algorithm works and they might not want to take any risks. Another thing about data mining is that data although comes from real world, actually does not directly reveal the natural laws or the secrets behind the data. We can use data mining to help us understand the secrets behind scenes. Moreover, we can also use other methods, in this mushroom case, image processing and consulting, to complement the data mining result. We do ensemble with different data mining algorithms, why not do ensemble over a large scale.
3. Another thing worth mentioning is that in HW 2 I also use mushroom dataset in mahout with Naïve Bayes. However, the result is extremely poor. Training on 7423 instances and testing on 701 datasets yield only 80.13%. I guess this is because I do not use the expanded version of mushroom dataset. Weka has reminded me that I could not use the same values for different attributes.