Assignment #1

Sprint on One Dataset Each

COMP 309 Machine Learning Tools and Techniques

Qiangqiang Li(Aaron)

ID: 300422249

Part 1: Core: Investigate Basic Use of The Different Tribes of AI

1. Choose one classification method from WEKA for each of the (non-EC) AI-tribes. Perform classification using your chosen method on the assigned dataset (initially, use the whole dataset without splitting, which will be implemented in the later parts of this assignment). Present and describe in the report what you consider to be the important aspects of the results of each technique on the one dataset (approximately two paragraphs of text plus figures).

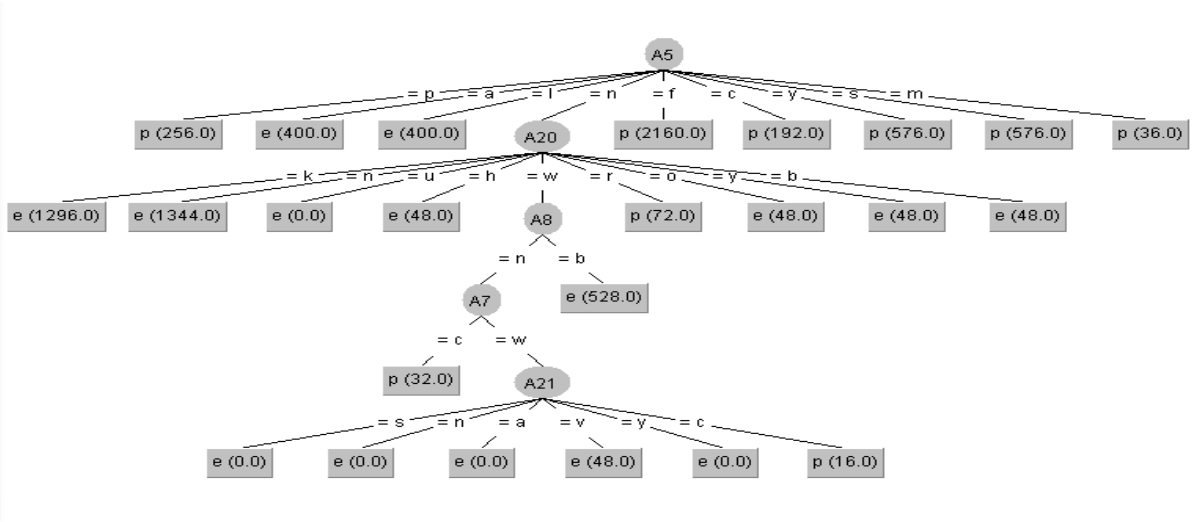
Reply:

Dataset: Mushroom(https://archive.ics.uci.edu/ml/datasets/mushroom)

5 Tribes of AI : Symbolists, Connectionists, Evolutionaries, Bayesians and Analogizers.

(1) Symbolists: work on the premise of inverse deduction. Instead of starting with the premise and looking for the conclusions, inverse deduction starts with some premises and conclusions, and essentially works backward to fill in the gaps.

The classification method: Decision tree (J48)




```
Time taken to build model: 0.08 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        8124               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             8124

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     p
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     e
Weighted Avg.  1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

    a    b   <-- classified as
 3916    0 |    a = p
    0 4208 |    b = e
```

(2) Connectionists: "Connectionists" want to reverse engineer the brain. Create artificial neurons and connect them in a neural network.

The classification method: Multilayer Perceptron (Neural Network)

```
Time taken to build model: 340.42 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        8124               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0002
Root mean squared error                  0.0005
Relative absolute error                  0.032  %
Root relative squared error              0.1025 %
Total Number of Instances             8124

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     p
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     e
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

    a    b   <-- classified as
 3916    0 |    a = p
    0 4208 |    b = e
```

(3) Bayesians: deal in uncertainty and solutions. Their master algorithm solution is called probabilistic inference.

The classification method: Naive Bayes. It is a very simple probability- based technique.

```
=== Evaluation on training set ===

Time taken to test model on training data: 0.03 seconds

=== Summary ===

Correctly Classified Instances        7790               95.8887 %
Incorrectly Classified Instances       334                4.1113 %
Kappa statistic                          0.9175
Mean absolute error                      0.0405
Root mean squared error                  0.1717
Relative absolute error                  8.1085 %
Root relative squared error             34.3721 %
Total Number of Instances             8124

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area
                 0.923    0.008    0.991      0.923    0.956      0.920    0.998     0.998
                 0.992    0.077    0.933      0.992    0.962      0.920    0.998     0.998
Weighted Avg.    0.959    0.044    0.961      0.959    0.959      0.920    0.998     0.998

=== Confusion Matrix ===

    a    b   <-- classified as
 3614  302 |    a = p
   32 4176 |    b = e
```

(4) Analogizers: The analogizers, or pioneers in the field of matching particular bits of data to each other. The master algorithm is the "nearest neighbor" principle.

The classification method: K-Nearest Neighbour(IBK)

```
Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        8124               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0.0029 %
Root relative squared error              0.003  %
Total Number of Instances             8124

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     p
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     e
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

    a    b   <-- classified as
 3916    0 |    a = p
    0 4208 |    b = e
```

Tribes	Symbolists	Connectionists	Bayesians	Analogizers
Method	**Decision tree**	**Multilayer Perceptron**	**Naive Bayes**	**K-Nearest Neighbour**
Correctly Classified Instances	**100%**	**100%**	**95.8887%**	**100%**
Incorrectly Classified Instances	0 %	0%	4.1113%	0%
Mean absolute error	0	0.0002	0.0405	0
Root mean squared error	0	0.0005	0.1717	0
Relative absolute error	0%	0.032%	8.1085%	0.0029%

From the result, we can see that the correctly classified instances are very high value. Naive Bayes provided the lowest correctly classified instances are 95.8887%. the others methods have the high correctly classified instances is 100%. Looking at this table, Decision tree is a good classified method, because MES (mean squared error), MAE(Mean absolute error), and relative squared error have the lowest value in this dataset. On the contrast, Naïve Bayes has the high value in MES,MAE and relative absolute error, but using 10-folds cross-validation, correctly classified instances was lower, at 95.8887%.So only to see this results, Decision tree is the best classified method in this dataset.

2. The report should detail why each selected technique from the stated tool belongs to a given tribe.

Reply:

(1) Decision tree(J48):

- **Description**: Decision tree is constructed with a root node and decision nodes, at the bottom is leaf nodes. And the leaf nodes are the class nodes. It is a classifier belongs to **Symbolists**.

- **Representation:** Decision Tree algorithms are referred to as **CART** or Classification and Regression Trees. Inner nodes are logical criterias. For example a node "if(a) then b." has two outcomes "0" or "1".

- **Evaluation method:** **Gini Impurity** could be used to measure how many times a randomly selected piece of data is incorrectly labeled. It aims to average the impurity of child nodes. In order to have better performance, it adds weighting the impurities by the probability of nodes.

- **Optimization driver:** It is using **Pruning** to optimize the performance. Pruning is the inverse of splitting, and avoid overfitting problem in DT.

(2) Multilayer Perceptron:

- **Description**: A multilayer perceptron (MLP) is a deep, artificial neural network. It is composed of more than one perceptron. It is using backpropagation for construction of the network. It is a classifier belongs to **Connectionists**.

- **Representation:** It is using **backpropagation for construction** of the network.

- **Evaluation method:** MLP trains using **Stochastic Gradient Descent.** Stochastic Gradient Descent (SGD) updates parameters using the gradient of the loss function with respect to a parameter that needs adaptation.

- **Optimization driver:** A first-order iterative optimization algorithm for finding the minimum of a function. It uses **validation control** to avoid overfitting.

(3) Naive Bayes:

- **Description**: Naive Bayes is a simple technique for constructing classifiers, models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. Naïve Bayes is a conditional probability model. It is a classifier belongs to **Bayesians**.

- **Representation:** **Graphical Models**. It is very important that naive bayes is requires features are independent given class.

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

- **Evaluation method:** Bayesian can be evaluated by **posterior probability**, the higher the posterior probability we get the better performance it is. Because the function is unknown, for bayesian it will generate a random function.
- **Optimization driver:** As we import the training set it will take the evaluations, which are treated as data, the initial function is updated to form the posterior distribution over the objective distribution.

(4) KNN(K-Nearest Neighbour):

- **Description**: KNN is in the supervised learning family of algorithms. It is represented by support vectors. The KNN algorithm is a robust and versatile classifier that is often used as a benchmark for more complex classifiers such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM). It belongs to **Analogizes**.
- **Representation: Support Vectors**. The vectors(cases) that define the hyperplane are the support vectors.
- **Evaluation method: This distance** from the decision surface to the closest data point determines the margin of the classifier.
- **Optimization driver: Constrained Optimization** It uses validation control to avoid overfitting. It is important to **pick a suitable value for K**.

3. These important aspects may be different between each technique. illustrate any important differences using the dataset. For example, use an instance in the dataset to show the differences in representation of each technique.

Reply: There are two types of mushrooms in the dataset, one is poisonous, other is edible.

Decision tree (J48): Among these four techniques, DT has the best accuracy. J48 is suited with nominal type data and binary data. So **The dataset is very suitable for DT.**

Naive Bayes: it assumes all the attributes are independent of each other. The main idea is a family of simple **probabilistic classifiers,** based on applying Bayes' theorem with strong independence assumptions between the features. So **Naive Bayes is a good classifier technique in mushroom dataset.**

Multilayer Perceptron: it can be dealing with binary input and numeric inputs. If the data without doing data preparation, the input nominal type attributes will become meaningless. while using PCA to do data preparation, the attributes become more number at 60, the Correctly Classified Instances decrease. So **the mushroom dataset is not suitable for Multilayer Perceptron.**

K-nearest neighbor (lazy IBK): Although it showed a high accuracy with **mushroom** data, I think it is not suitable. Comparing to binary sets, numerical suits KNN much better. So **the mushroom dataset is not suitable for K-nearest neighbor.**

(optional) Insight into 'why one or more aspect is suited to a given dataset' will be needed to achieve high grades.

Reply:

J48 is suited with nominal type data and binary data (eg 1 or 0). Because it can split the attributes into several branches, which adapted to the tree structure.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem.

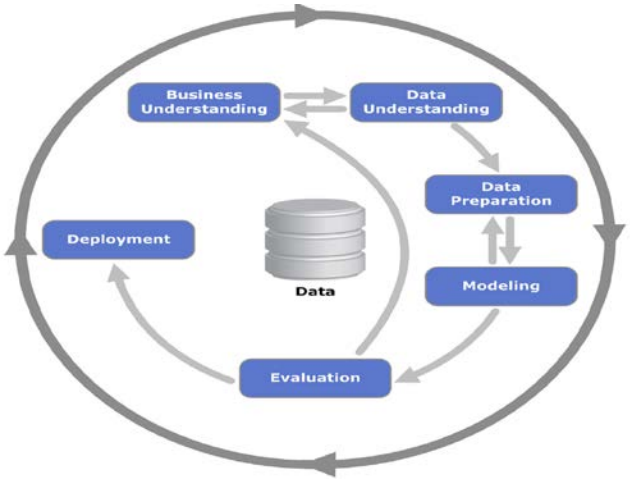
Multilayer Perceptron is suited to binary and numeric data. It referent the backpropagation algorithm to change weights. The adjust weights algorithm did use arithmetic operations.

K-nearest neighbor is suited with numeric data. Because in K-nearest neighbor method, the mean idea is to calculate the distance between testing instance to any other instance in the training dataset to find out the nearest neighbors.

So a given dataset show different data type, one or more techniques using different model, and it will generate different algorithm or different way to classify. These results after well training may all can well handle the problems. To get a accurate classifier, can use serval method for well training, and choose a suitable technique for the given dataset.

Part 2: Completion 1: Consider a Pipeline for Dataset Processing

CRISP-DM (Cross-Industry Standard Process for Data Mining) provides a structured approach to planning a data mining project. It is a robust and well-proven analytic model.



1. Business understanding - consider the business aspects of the dataset, e.g. why was the data gathered? what did the acquisition hope to achieve? Note, that this may be mere obvious in some

datasets than others.

Reply: Mushroom records drawn from the Audubon Society Field Guide to North American Mushrooms, and **described** in terms of **Mushroom's physical characteristics**. **The goal is** to see whether there is **Mushroom's physical characteristics** combinations to **predict the mushroom's type** (edible or poisonous).

2. Data understanding - not only should the metadata be described (which is readily available in the UCI repository), but any interesting factors should be noted, e.g. mixed attribute type, high epistasis, out- lier/noisy/missing data instances.

Reply: The Data Set **Characteristics** are **Multivariate**, t**he numbe**r of instances is **8124**, and the **number of attributes** is **22**. **The associated tasks** are **classification**, and the number of Missing Attribute Values is 2480 (**denoted by "?", all for attribute #11**). **Class Distribution** is edible **(4208,51.8%)** and poisonous**(3916,48.2%).**

3. Data preparation - state how the pipeline could assist in the preparation of the data prior to the technique being applied.

Reply: In this dataset, the missing values is yes, So the dataset need to do data preprocess by pipeline. For the dataset have a good balanced, class distribution is edible**(4208 ,51.8%)** and poisonous**(3916,48.2%)** .**So** Prior to the technique being applied, pipelining can be used to **filter out outliers or deal with missing data** using appropriate methods and could convert raw data into **more useful format** that better suits a algorithm..

4. Modelling - state whether this pipeline suits one or more of the five tribes of AI.

Reply: Mushroom dataset is small and data is categorical, and two types of mushroom were classified in the dataset, one is poisonous, other is edible. Compare the result of methods in four tribes of AI, it is clear that this pipeline not suits all of the four tribes of AI. There were three of accuracies **decreased** after using this pipeline. While Naïve Bayes have better accuracy in Mushroom dataset, So **this pipeline suits one or more of the five tribes of AI.**

5. Evaluation - similarly, state whether this pipeline supports one or more methods to evaluate a solution.

Reply: Using this pipeline, the result of methods in four tribes of AI shows that **Naïve Bayes** have better accuracy in Mushroom dataset, while other methods decreased accuracy. **So this pipeline supports one or more methods to evaluate a solution.**

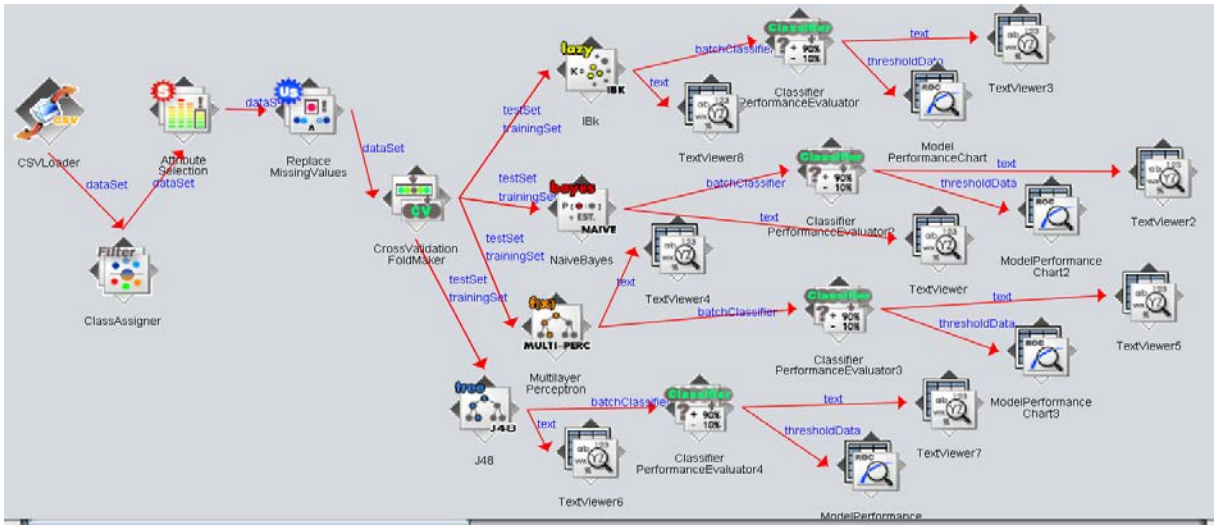
6. Deployment - explain whether the model produced can easily be deployed or whether additional effort is required.

Reply: In the aspects of Deployment, the Weka was gave GUI to build the model and easy to built a pipeline. Because the dataset includes some missing value, before using the model, need to remove the noisy, outlier and missing data instances. On the other hand, the method can use 10 folds cross-validation for the training set is a good way to get fair accuracy. When the model has been deployed, it needs to be monitored regularly and maintained if necessary. the result of report can be implementing a repeatable data mining process. It was easy to analyze the data mining result.

Part 3: Completion 2: Use the pipeline to reevaluate the selected techniques in Part 2.1 used to classify the dataset

1. ***Program code for your classifiers (both set-up details (e.g. code and scripts) and executable program capable of running on the ECS School machines). The program should print out the classifiers in as human readable form (text form is fine) as possible. Compare and contrast the results between the different tools.***

Tribes	**Symbolists**	**Connectionists**	**Bayesians**	**Analogizers**
Method	**Decision tree**	**Multilayer Perceptron**	**Naive Bayes**	**K-Nearest Neighbour**
Correctly Classified Instances	**99.0153%**	**99.0153%**	**98.5229%**	**99.0153%**
Incorrectly Classified Instances	0.9847%	0.9847%	1.4771%	0.9847%
Mean absolute error	0.0191	0.0195	0.0219	0.0191
Root mean squared error	0.0978	0.0978	0.1092	0.0977
Relative absolute error	3.825%	3.838%	4.3852%	3.8219%



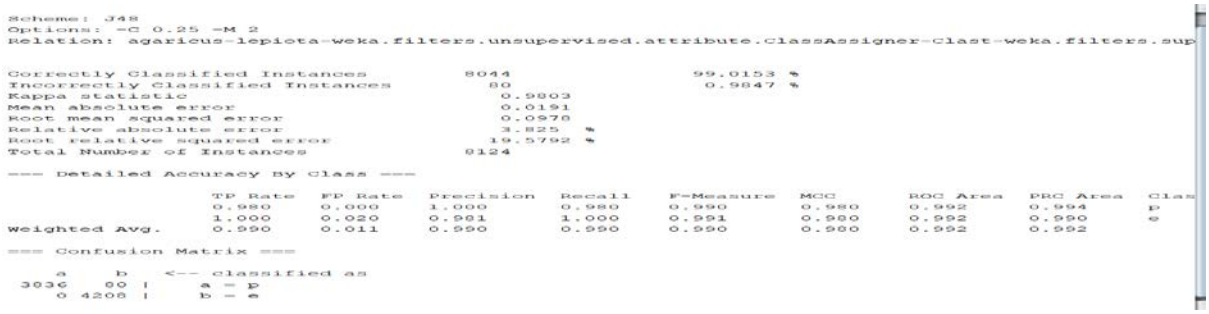
Through the pipeline, the class balance not to use and replace all the missing values with the means from the training data. The process time became long and the result will use the textView to show. In

this dataset, the accuracy of **Naive Bayes** improved about 3%, while the other methods decreased about 0.9%. Using the pipeline, the Attribute Selection and Replace Missing Value deleted the missing value and clean the unrelated attributes, this may be result to decrease the accuracy in the Symbolists, Connectionists, and Analogizers *tribes.*

2. The report should include:

(1) Accuracy in terms of the fraction of the test instances that it classified correctly.

Reply :J48 (Correctly Classified Instances 99.0153%)

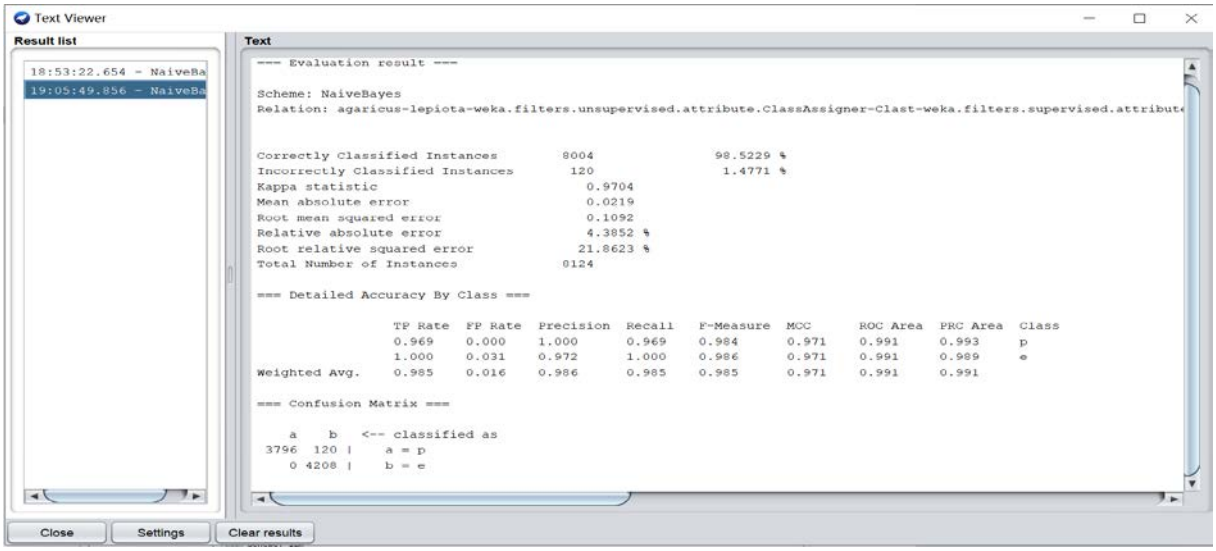


```
Scheme:  J48
Options:  -C 0.25 -M 2
Relation:  agaricus-lepiota-weka.filters.unsupervised.attribute.classAssigner-class-weka.filters.sup

Correctly Classified Instances        8044               99.0153 %
Incorrectly Classified Instances        80                0.9847 %
Kappa statistic                          0.9803
Mean absolute error                      0.0191
Root mean squared error                  0.0970
Relative absolute error                  3.825  %
Root relative squared error             19.5792 %
Total Number of Instances             8124

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Clas
                 0.980     0.000     1.000      0.980     0.990      0.980     0.992     0.994     p
                 1.000     0.020     0.981      1.000     0.991      0.980     0.992     0.990     e
Weighted Avg.    0.990     0.011     0.990      0.990     0.990      0.980     0.992     0.992

=== Confusion Matrix ===

    a    b   <-- classified as
 3836   80 |    a = p
    0 4208 |    b = e
```

MultilayerPerceptron **(Correctly Classified Instances 99.0153%)**

```
Scheme:  MultilayerPerceptron
Options:  -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a
Relation:  agaricus-lepiota-weka.filters.unsupervised.attribute.classAssigner-class-weka.filters.supervised.

Correctly Classified Instances        8044               99.0153 %
Incorrectly Classified Instances        80                0.9847 %
Kappa statistic                          0.9803
Mean absolute error                      0.0195
Root mean squared error                  0.0978
Relative absolute error                  3.8989 %
Root relative squared error             19.5746 %
Total Number of Instances             8124

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.980     0.000     1.000      0.980     0.990      0.980     0.994     0.995     p
                 1.000     0.020     0.981      1.000     0.991      0.980     0.994     0.992     e
Weighted Avg.    0.990     0.011     0.990      0.990     0.990      0.980     0.994     0.993

=== Confusion Matrix ===

    a    b   <-- classified as
 3836   80 |    a = p
    0 4208 |    b = e
```

NaiveBayes (Correctly Classified Instances 98.5229%)




```
=== Evaluation result ===

Scheme: NaiveBayes
Relation: agaricus-lepiota-weka.filters.unsupervised.attribute.ClassAssigner-Clast-weka.filters.supervised.attribute

Correctly Classified Instances        8004               98.5229 %
Incorrectly Classified Instances       120                1.4771 %
Kappa statistic                          0.9704
Mean absolute error                      0.0219
Root mean squared error                  0.1092
Relative absolute error                  4.3852 %
Root relative squared error             21.8623 %
Total Number of Instances             0124

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.969     0.000     1.000      0.969     0.984      0.971     0.991     0.993     p
                 1.000     0.031     0.972      1.000     0.986      0.971     0.991     0.989     e
Weighted Avg.    0.985     0.016     0.986      0.985     0.985      0.971     0.991     0.991

=== Confusion Matrix ===

    a    b   <-- classified as
 3796  120 |    a = p
    0 4208 |    b = e
```

IBK: **(Correctly Classified Instances 99.0153%)**

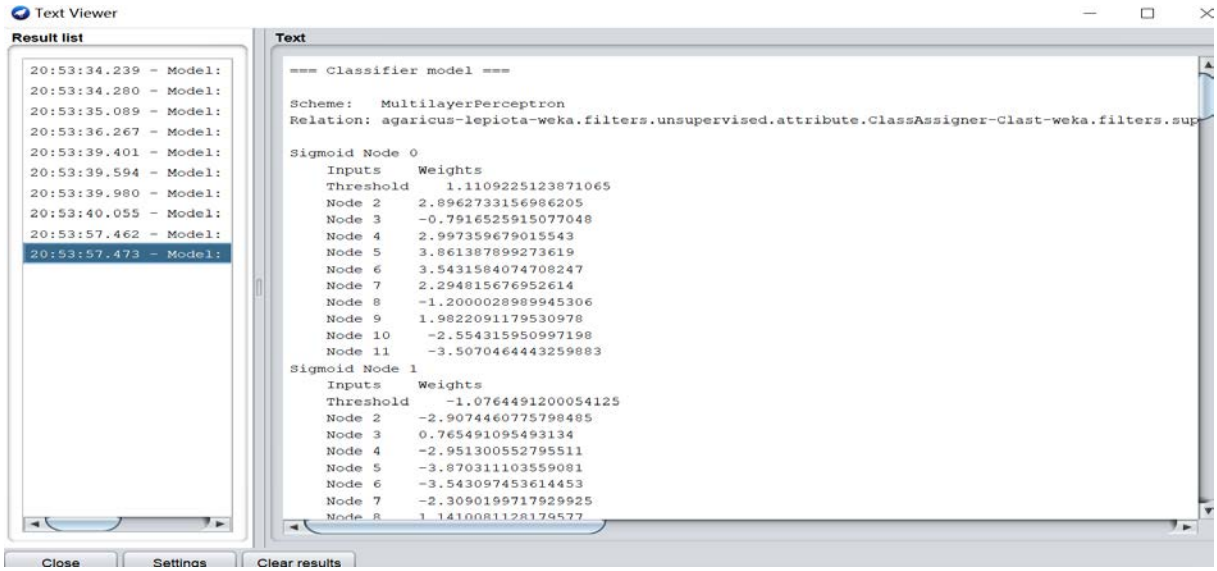
```
Scheme: IBk
Options: -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation: agaricus-lepiota-weka.filters.unsupervised.attribute.ClassAssigner-Clast-weka.filters.supervised.attribu


Correctly Classified Instances        8044               99.0153 %
Incorrectly Classified Instances        80                0.9847 %
Kappa statistic                          0.9803
Mean absolute error                      0.0191
Root mean squared error                  0.0977
Relative absolute error                  3.8219 %
Root relative squared error             19.5564 %
Total Number of Instances             8124

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.980     0.000     1.000      0.980     0.990      0.980     0.993     0.995     p
                 1.000     0.020     0.981      1.000     0.991      0.980     0.993     0.992     e
Weighted Avg.    0.990     0.011     0.990      0.990     0.990      0.980     0.993     0.993

=== Confusion Matrix ===

    a    b   <-- classified as
 3836   80 |    a = p
    0 4208 |    b = e
```

(2) Report a snapshot of the learned classifiers discovered by your program.

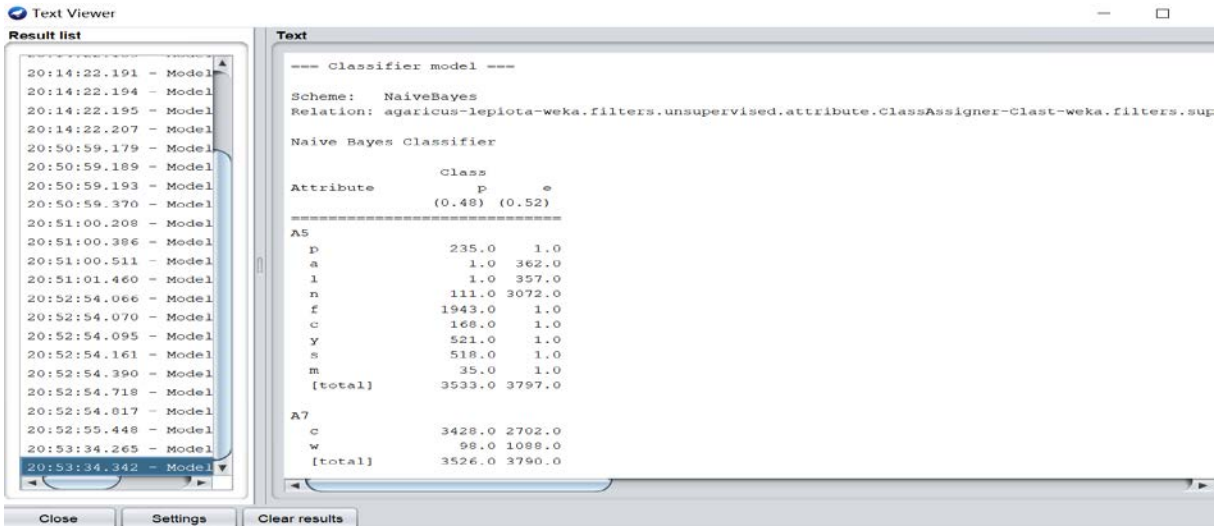
Reply :Decision tree

```
J48 pruned tree
------------------

A5 = p: p (234.0)
A5 = a: e (361.0)
A5 = l: e (356.0)
A5 = n
|   A17 = w
|   |   A12 = s: e (2471.0/73.0)
|   |   A12 = f: e (359.0)
|   |   A12 = k
|   |   |   A7 = c: p (30.0)
|   |   |   A7 = w: e (134.0)
|   |   A12 = y: e (14.0)
|   A17 = n: e (87.0)
|   A17 = o: e (79.0)
|   A17 = y: p (7.0)
A5 = f: p (1942.0)
A5 = c: p (167.0)
A5 = y: p (520.0)
A5 = s: p (517.0)
A5 = m: p (34.0)

Number of Leaves  :     16

Size of the tree :      20
```

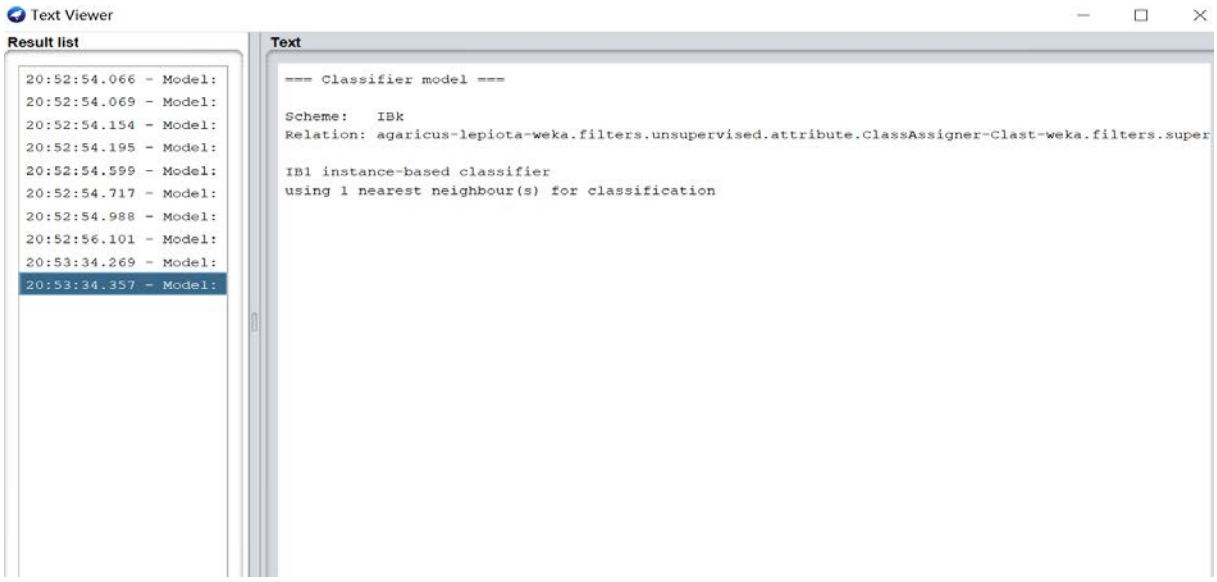
MultilayerPerceptron



NaiveBayes



IBK:



(3) Compare the accuracy of your techniques before and after using a data pipeline approach.

Please comment on any differences, suggesting reasons.

Before a data pipeline approach

Tribes	**Symbolists**	**Connectionists**	**Bayesians**	**Analogizers**
Method	**Decision tree**	**Multilayer Perceptron**	**Naive Bayes**	**K-Nearest Neighbour**
Correctly Classified Instances	**100%**	**100%**	**95.8887%**	**100%**
Incorrectly Classified Instances	0 %	0%	4.1113%	0%

Using a data pipeline approach

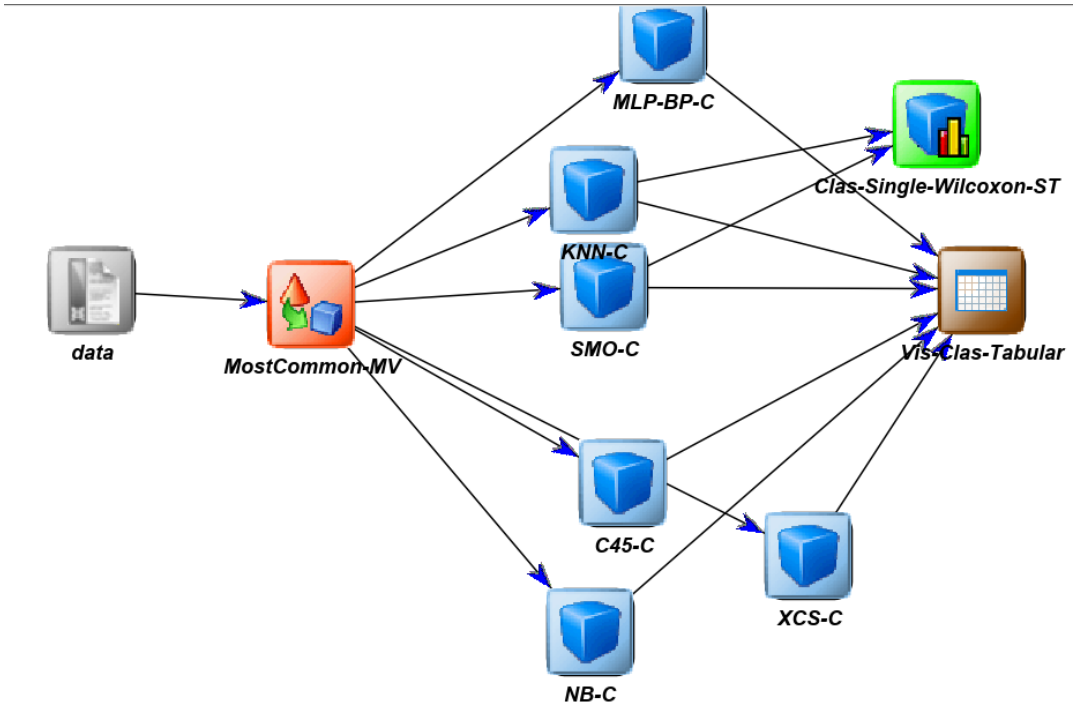
Tribes	**Symbolists**	**Connectionists**	**Bayesians**	**Analogizers**
Method	**Decision tree**	**Multilayer Perceptron**	**Naive Bayes**	**K-Nearest Neighbour**
Correctly Classified Instances	**99.0153%**	**99.0153%**	**98.5229%**	**99.0153%**
Incorrectly	0.9847%	0.9847%	1.4771%	0.9847%

Classified Instances				

Reply : From the table above, the **Correctly Classified Instances** of **NaiveBayes** become better, increased 3% , while the other three techniques have a little decrease. On the other hand, the dataset became well balanced but the data meaning were missing after this pipeline in the other methods. Using the pipeline, the **Attribute Selection** and **Replace Missing Value** deleted the missing value and clean the unrelated attributes, this may be result to decrease the accuracy in the **Symbolists, Connectionists, and Analogizers** *tribes. So* **NaiveBay**e model is suitable for *using the* pipeline.

Part 3: Challenge: Use the KEEL to evaluate the Evolutionary Computation tribe on the dataset in Part 2.1 to classify the dataset

1. ***Program code for your classifiers (both set-up details (e.g. code and scripts) and executable program (capable of running on the ECS School machines). The program should print out the classifiers in as human readable form (text form is fine) as possible.***
   1. ***import mushroom data***
   2. ***parttion(10 folds cross validation)and deal with the missing data (MostCommon-MV).***
   3. ***select Algorithms(Decision tree , Multilayer Perceptron , Naive Bayes ,KNN,*** **Genetic Algorithm*)***
   4. ***show results.***



1. ***The report should include:***

(1) Accuracy in terms of the fraction of the test instances that it classified correctly.

TEST RESULTS

TEST RESULTS

Dataset,MostCommon-MV.KNN-C,MostCommon-MV.KNN-C,MostCommon-MV.KNN-C,MostCommon-MV.SMO-C,MostCommon-MV.SMO-C,MostCommon-MV.SMO-C,MostCommon-MV.C45-C,MostCommon-MV.C45-C,MostCommon-MV.C45-C,MostCommon-MV.NB-C,MostCommon-MV.NB-C,MostCommon-MV.NB-C,MostCommon-MV.XCS-C,MostCommon-MV.XCS-C,MostCommon-MV.XCS-C,MostCommon-MV.MLP-BP-C,MostCommon-MV.MLP-BP-C,MostCommon-MV.MLP-BP-C
,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified
agaricus-lepiota-new relation agaricus-lepiota-new
,1.0000000000,0.0000000000,0.0000000000,1.0000000000,0.0000000000,0.0000000000,1.0000000000,0.0000000000,0.0000000000,0.9545783118,0.0000768652,0.0000000000,0.9955669569,0.0001183622,0.0000000000,0.8262403129,0.0509274813,0.0000000000

TRAIN RESULTS

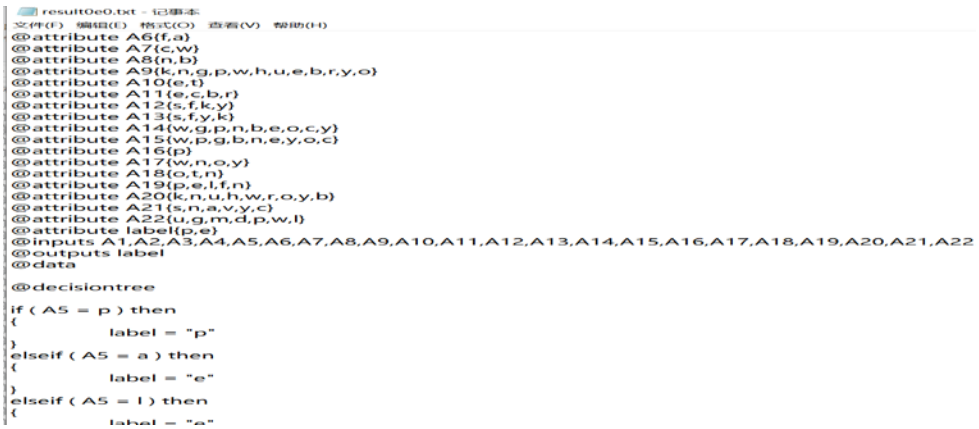
Dataset,MostCommon-MV.KNN-C,MostCommon-MV.KNN-C,MostCommon-MV.KNN-C,MostCommon-MV.SMO-C,MostCommon-MV.SMO-C,MostCommon-MV.SMO-C,MostCommon-MV.C45-C,MostCommon-MV.C45-C,MostCommon-MV.C45-C,MostCommon-MV.NB-C,MostCommon-MV.NB-C,MostCommon-MV.NB-C,MostCommon-MV.XCS-C,MostCommon-MV.XCS-C,MostCommon-MV.XCS-C,MostCommon-MV.MLP-BP-C,MostCommon-MV.MLP-BP-C,MostCommon-MV.MLP-BP-C
,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified,Average (Correctly Classified),Variance (Correctly Classified),Not Classified
agaricus-lepiota-new relation agaricus-lepiota-new
,1.0000000000,0.0000000000,0.0000000000,1.0000000000,0.0000000000,0.0000000000,1.0000000000,0.0000000000,0.0000000000,0.9556731849,0.0000006070,0.0000000000,0.9963210692,0.0000833863,0.0000000000,0.8256691623,0.0507259012,0.0000000000

Tribes	Symbolists	Connectionists	Bayesians	Analogizers	Evolutionaries
Method	**Decision tree**	**Multilayer Perceptron**	**Naive Bayes**	**K-Nearest Neighbour**	**Genetic Algorithm**
Correctly Classified Instances	**100%**	**82.57%**	**96.7054%**	**100%**	**99.63%**

(2) Report a snapshot of the learned classifiers discovered by your program.

Show the results in the keelcope.zip (keelcope/ results)

Like this : Decision tree


result0e0.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
@attribute A6{f,a}
@attribute A7{c,w}
@attribute A8{n,b}
@attribute A9{k,n,g,p,w,h,u,e,b,r,y,o}
@attribute A10{e,t}
@attribute A11{e,c,b,r}
@attribute A12{s,f,k,y}
@attribute A13{s,f,y,k}
@attribute A14{w,g,p,n,b,e,o,c,y}
@attribute A15{w,p,g,b,n,e,y,o,c}
@attribute A16{p}
@attribute A17{w,n,o,y}
@attribute A18{o,t,n}
@attribute A19{p,e,l,f,n}
@attribute A20{k,n,u,h,w,r,o,y,b}
@attribute A21{s,n,a,v,y,c}
@attribute A22{u,g,m,d,p,w,l}
@attribute label{p,e}
@inputs A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21,A22
@outputs label
@data

@decisiontree

if ( A5 = p ) then
{
        label = "p"
}
elseif ( A5 = a ) then
{
        label = "e"
}
elseif ( A5 = l ) then
{
        label = "e"
```

(3) Compare the Evolutionary Computation approach with the other AI tribes from earlier parts.

Tribes	Symbolists	Connectionists	Bayesians	Analogizers	Evolutionaries
Method	**Decision tree**	**Multilayer Perceptron**	**Naive Bayes**	**K-Nearest Neighbour**	**Genetic Algorithm**
Correctly Classified	**100%**	**82.57%**	**96.7054%**	**100%**	**99.63%**

Instances					

Evolutionaries is an algorithms that will constantly evolve and adapt to unknown conditions and processes. The technique I used above is **Genetic Algorithm** it represents Genetic programming, is to simulated the process of biological evolution to generate a algorithms that can find out the solutions. From the result, we can see that the correctly classified instances are very high value. **Multilayer Perceptron** provided the lowest correctly classified instances are 82.57%. **K-Nearest Neighbour and Decision tree** methods have the highest correctly classified instances is 100%. **Naive Bayes** and **Genetic Algorithm** are the middle level. Looking at this table, **Decision tree and K-Nearest Neighbour** are good classified methods in this dataset.

(4) Compare the WEKA tool with the KEEL tool commenting on ease-of-use, performance and any other aspects you consider important (e.g. data format, documentation, online tutorials and so forth).

Reply: Ease-of-use:

- ***WEKA: both Command Line and GUI approach to design experiments with different datasets and computational intelligence algorithms.***
- ***Keel: has a simple GUI based on the data flow to design experiments with different datasets and computational intelligence algorithms.***

Performance

- ***WEKA: both Command Line and GUI approach. Using the tool to save the results.***
- ***Keel: using the java -jar RunKeel.jar to run the code, and auto create the results files.***

Data format

- ***WEKA: supports various file formats, not to transform the same file format.***
- ***Keel: all the dataset need to transform to keel file formats.***

Documentation

- ***WEKA: https://www.cs.waikato.ac.nz/ml/weka/courses.html and show the page.***
- ***Keel: website(http://www.keel.es/) and show by pdf file.***

Online tutorials:

- ***WEKA: ease to find the resource online, and give Youtube online resources. https://www.cs.waikato.ac.nz/ml/weka/courses.html***
- ***Keel: website (http://www.keel.es/) and show by pdf files.***

Compare using two tools in the mushroom data, **weka tool gave the result is better than keel**, but using the pipeline by weka to run program need more time than keel. On the other hand, when **using the Evolutionaries tribe and only using the keel tool**. So, **using the datamining tool is depends on the datasets and algorithms you want to use**.