PART A

(PART A: TO BE REFFERED BY STUDENTS)

Experiment No.04

Aim: Implementation of naïve Bayesian Classifier using Weka Tool.

Outcome:

After successful completion of this experiment students will be able to

1. Demonstrate an understanding of the importance of data mining
2. Organize and Prepare the data needed for data mining using pre preprocessing techniques
3. Perform exploratory analysis of the data to be used for mining.
4. Implement the appropriate data mining methods like classification

Theory:

Naïve Bayes:

Naïve Bayes methods are aset of supervised learning algorithms based on applying Bayes' theorem with the "naïve" assumption of conditional independence between every pair of features given the value of the class variable. Bayes' theorem states the following relationship, given class variable y and dependent feature vector x1 through xn, :

$$P(y|x1,\ldots,xn) = P(y)P(x1,\ldots,xn|y)/P(x1,\ldots,xn)$$ Using

the naive conditional independence assumption that

$$P(xi|y,x1,\ldots,xi-1,xi+1,\ldots,xn) = P(xi|y),$$

for all i, this relationship is simplified to

$$P(y|x1,\ldots,xn)=P(y)\prod i=1nP(xi|y)/P(x1,\ldots,xn)$$

SinceP(x1,…,xn)is constant given the input, we can use the following classification rule:

$$P(y|x1,\ldots,xn)\propto P(y)\prod i=1nP(xi|y)Uy\hat{}$$
$$=\text{argmax}(y)(P(y)\prod i=1nP(xi|y)),$$

and we can use Maximum A Posteriori (MAP) estimation to estimate P(y) and P(xi|y); the former is then the relative frequency of class y in the training set.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of P(xi|y).

In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters. (For theoretical reasons why naive Bayes works well, and on which types of data it does, see the references below.)

Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions means that each distribution can be independently

estimated as a one-dimensional distribution. This in turn helps to alleviate problems stemming from the course of dimensionality.

Gaussian Naive Bayes:
When working with continuous data,an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be-

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Sometimes assume variance

- Is independent of Y(i.e.,σi),
- Or independent of Xi (i.e.,σk)
- Or both(i.e., σ)

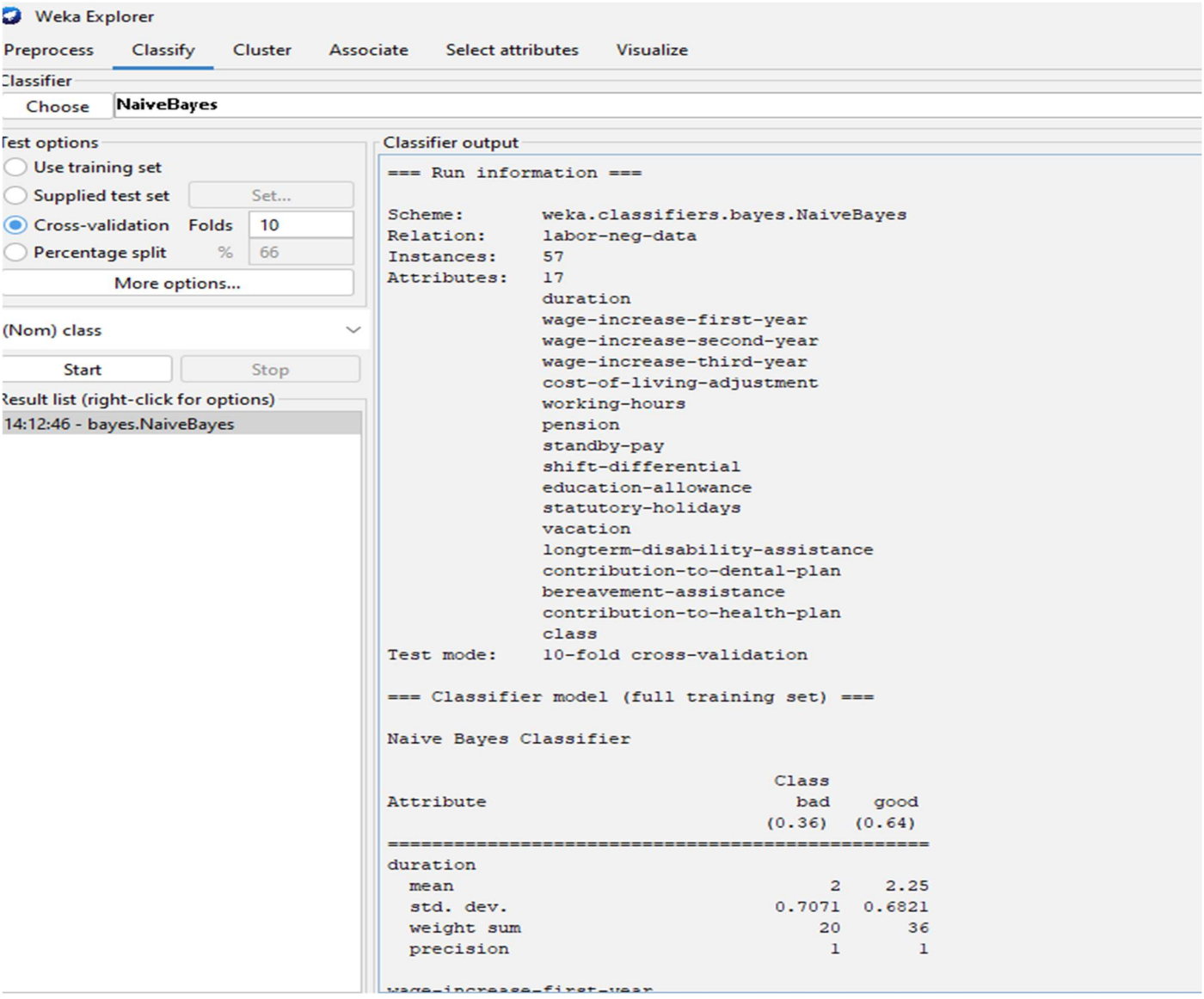
Gaussian NaiveBayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution.

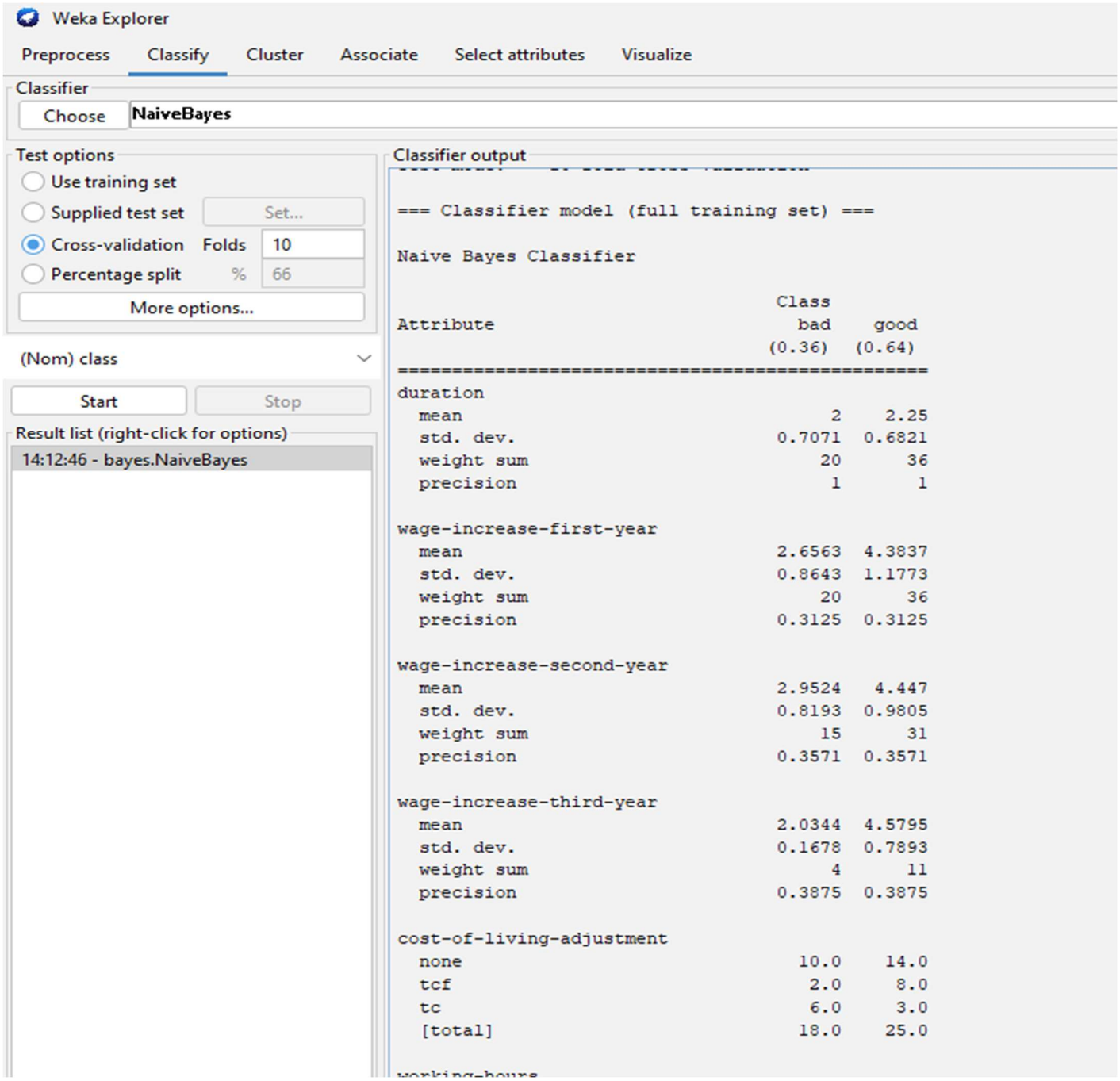
PART B

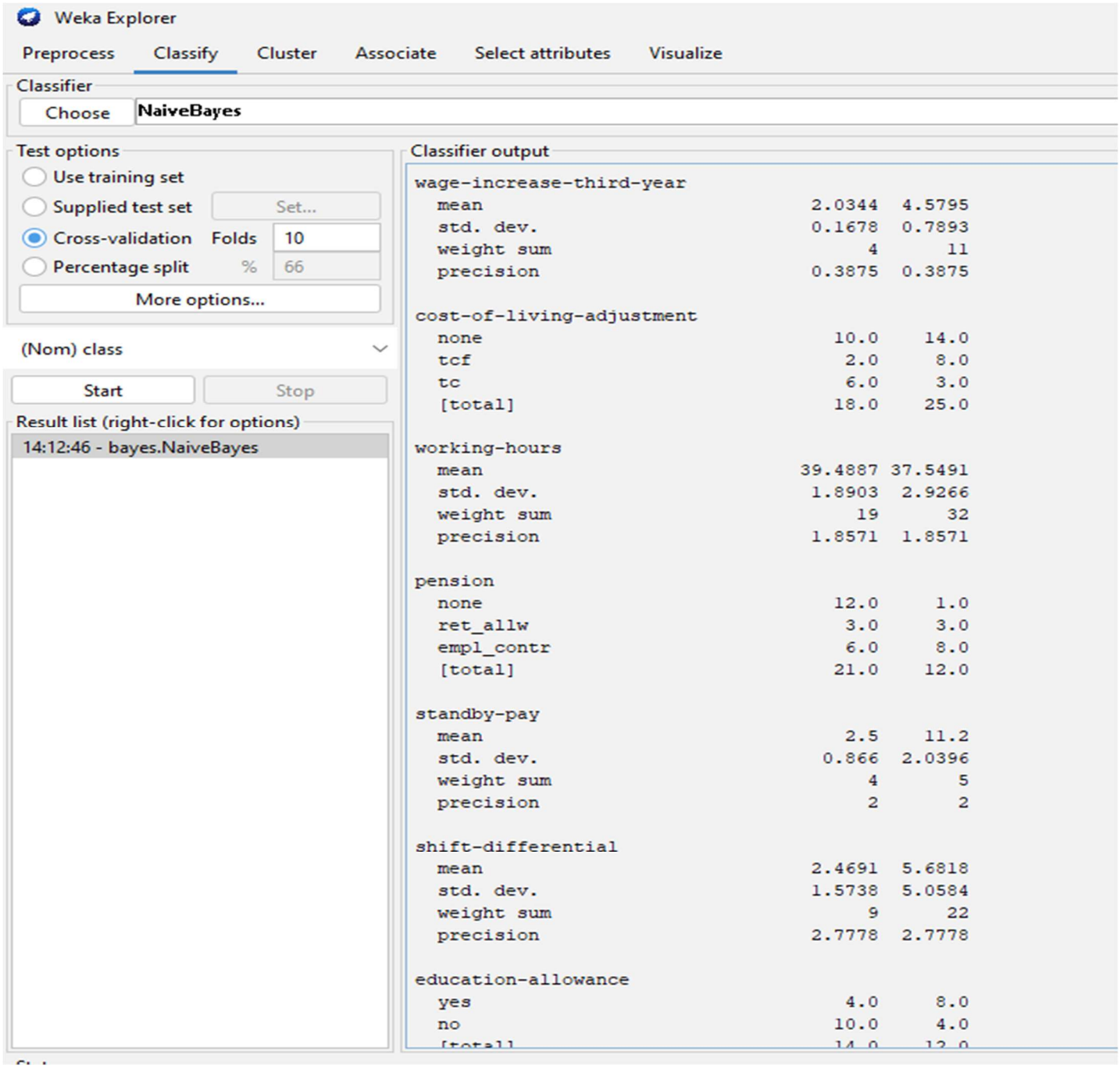
(PART B: TO BE COMPLETED BY STUDENTS)

Roll. No: A07	Name: Niharika Seth
Class: TE AI & DS	Batch: A1
Date of Experiment: 16/08/2024	Date of Submission: 30/08/2024
Grade:	

Input and Output:




Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayes
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
14:12:46 - bayes.NaiveBayes
Classifier output
=== Classifier model (full training set) ===
Naive Bayes Classifier
Class
Attribute
bad
good
(0.36)
(0.64)
duration
mean 2 2.25
std. dev. 0.7071 0.6821
weight sum 20 36
precision 1 1
wage-increase-first-year
mean 2.6563 4.3837
std. dev. 0.8643 1.1773
weight sum 20 36
precision 0.3125 0.3125
wage-increase-second-year
mean 2.9524 4.447
std. dev. 0.8193 0.9805
weight sum 15 31
precision 0.3571 0.3571
wage-increase-third-year
mean 2.0344 4.5795
std. dev. 0.1678 0.7893
weight sum 4 11
precision 0.3875 0.3875
cost-of-living-adjustment
none 10.0 14.0
tcf 2.0 8.0
tc 6.0 3.0
[total] 18.0 25.0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayes
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
14:12:46 - bayes.NaiveBayes
Classifier output
wage-increase-third-year
mean 2.0344 4.5795
std. dev. 0.1678 0.7893
weight sum 4 11
precision 0.3875 0.3875
cost-of-living-adjustment
none 10.0 14.0
tcf 2.0 8.0
tc 6.0 3.0
[total] 18.0 25.0
working-hours
mean 39.4887 37.5491
std. dev. 1.8903 2.9266
weight sum 19 32
precision 1.8571 1.8571
pension
none 12.0 1.0
ret_allw 3.0 3.0
empl_contr 6.0 8.0
[total] 21.0 12.0
standby-pay
mean 2.5 11.2
std. dev. 0.866 2.0396
weight sum 4 5
precision 2 2
shift-differential
mean 2.4691 5.6818
std. dev. 1.5738 5.0584
weight sum 9 22
precision 2.7778 2.7778
education-allowance
yes 4.0 8.0
no 10.0 4.0

Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Classifier

Choose **NaiveBayes**

Test options

- Use training set
- Supplied test set — Set...
- Cross-validation — Folds 10
- Percentage split — % 66

More options...

(Nom) class

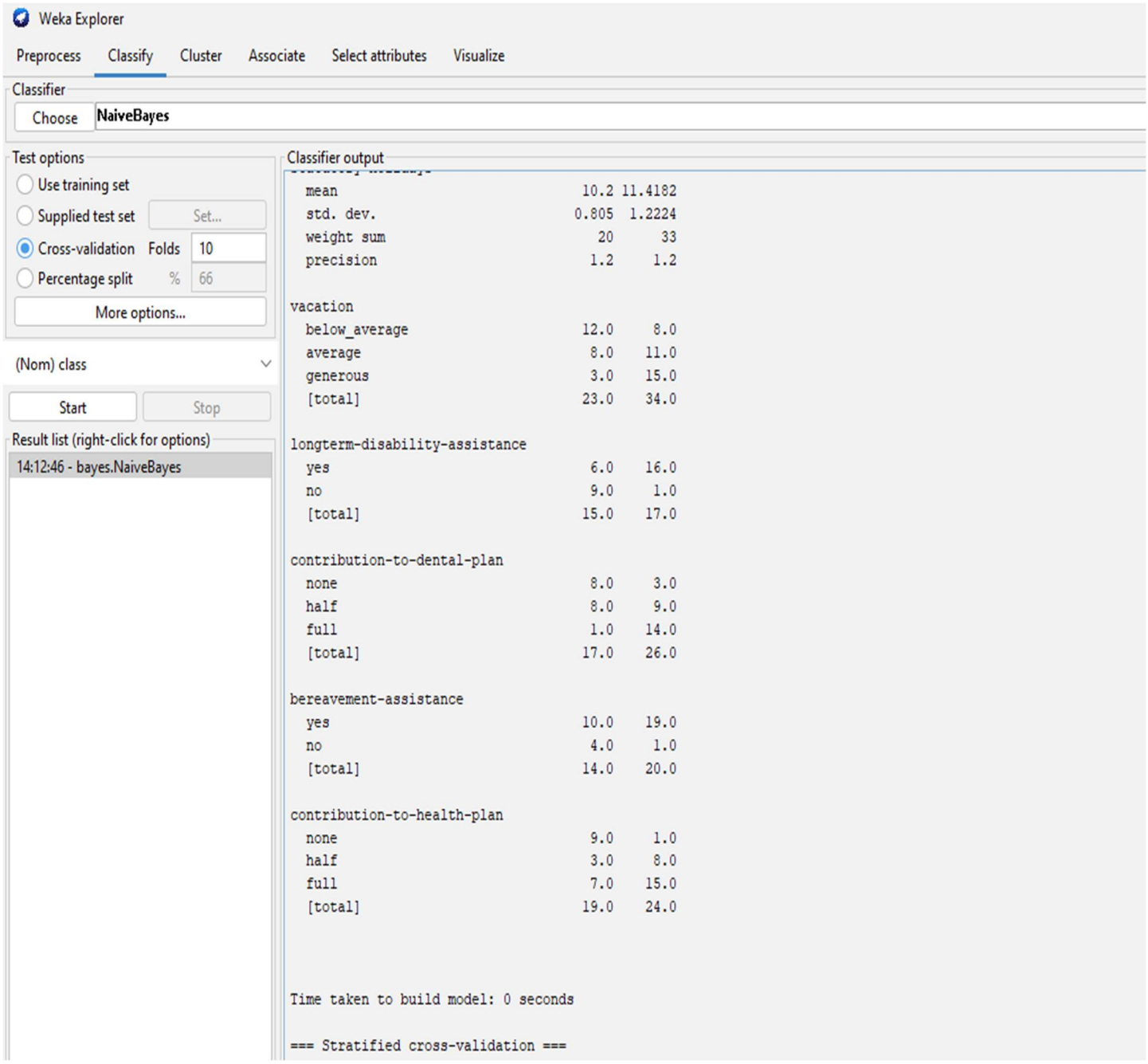
Start | Stop

Result list (right-click for options)

14:12:46 - bayes.NaiveBayes

Classifier output

```
shift-differential
  mean                          2.4691  5.6818
  std. dev.                     1.5738  5.0584
  weight sum                         9      22
  precision                     2.7778  2.7778

education-allowance
  yes                              4.0     8.0
  no                              10.0     4.0
  [total]                         14.0    12.0

statutory-holidays
  mean                            10.2 11.4182
  std. dev.                      0.805  1.2224
  weight sum                        20      33
  precision                        1.2     1.2

vacation
  below_average                   12.0     8.0
  average                          8.0    11.0
  generous                         3.0    15.0
  [total]                         23.0    34.0

longterm-disability-assistance
  yes                              6.0    16.0
  no                               9.0     1.0
  [total]                         15.0    17.0

contribution-to-dental-plan
  none                             8.0     3.0
  half                             8.0     9.0
  full                             1.0    14.0
  [total]                         17.0    26.0

bereavement-assistance
  yes                             10.0    19.0
  no                               4.0     1.0
  [total]                         14.0    20.0

contribution-to-health-plan
```


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayes
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
14:12:46 - bayes.NaiveBayes
Classifier output
mean 10.2 11.4182
std. dev. 0.805 1.2224
weight sum 20 33
precision 1.2 1.2
vacation
below_average 12.0 8.0
average 8.0 11.0
generous 3.0 15.0
[total] 23.0 34.0
longterm-disability-assistance
yes 6.0 16.0
no 9.0 1.0
[total] 15.0 17.0
contribution-to-dental-plan
none 8.0 3.0
half 8.0 9.0
full 1.0 14.0
[total] 17.0 26.0
bereavement-assistance
yes 10.0 19.0
no 4.0 1.0
[total] 14.0 20.0
contribution-to-health-plan
none 9.0 1.0
half 3.0 8.0
full 7.0 15.0
[total] 19.0 24.0
Time taken to build model: 0 seconds
=== Stratified cross-validation ===

Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Classifier

Choose **NaiveBayes**

Test options

- Use training set
- Supplied test set — Set...
- Cross-validation — Folds 10
- Percentage split — % 66

More options...

(Nom) class

Start | Stop

Result list (right-click for options)

14:12:46 - bayes.NaiveBayes

Classifier output

```
  yes                             10.0    19.0
  no                               4.0     1.0
  [total]                         14.0    20.0

contribution-to-health-plan
  none                             9.0     1.0
  half                             3.0     8.0
  full                             7.0    15.0
  [total]                         19.0    24.0



Time taken to build model: 0 seconds

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

Correctly Classified Instances          51               89.4737 %
Incorrectly Classified Instances         6               10.5263 %
Kappa statistic                          0.7741
Mean absolute error                      0.1042
Root mean squared error                  0.2637
Relative absolute error                 22.7763 %
Root relative squared error             55.2266 %
Total Number of Instances               57

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.900    0.108    0.818      0.900    0.857      0.776    0.965     0.926     bad
               0.892    0.100    0.943      0.892    0.917      0.776    0.965     0.983     good
Weighted Avg.  0.895    0.103    0.899      0.895    0.896      0.776    0.965     0.963

=== Confusion Matrix ===

  a  b   <-- classified as
 18  2 |  a = bad
  4 33 |  b = good
```

Status

Observations and learning:
The Naïve Bayes classifier is a probabilistic classifier based on Bayes' theorem with the assumption of independence between features. Weka's user-friendly interface simplified data import, preprocessing, and model training, making it straightforward to handle various datasets. The Naïve Bayes classifier, known for its simplicity and efficiency, performed well even with the assumption of feature independence.

Conclusion:
In this experiment, we implemented a Naïve Bayesian Classifier using Weka, successfully training it to classify data with good accuracy. The results demonstrated the model's efficiency and scalability for classification tasks, with insights into areas for potential improvement.

Question of Curiosity:

Q.1] What type of datasets are suitable for the Naive Bayesian Classifier in Weka and why?

Ans: Naive Bayes is particularly well-suited for datasets that:

1. Categorical Data: Naive Bayes works exceptionally well with categorical data because it estimates the probability of each category independently. In Weka, datasets with categorical features allow the Naive Bayes algorithm to directly calculate the probability of a given class based on the frequency of attribute values.
2. Text Data (Bag-of-Words Representation): Naive Bayes is effective in text classification tasks, such as spam detection or sentiment analysis, where data is often represented as a bag-of-words. Each word (feature) is treated as independent, making the algorithm simple and fast for text data.
3. Low Dimensional Datasets: Naive Bayes performs well on datasets with a relatively small number of features. In cases where the dimensionality is high but the features are sparse (like text data), the independence assumption of Naive Bayes simplifies computation and often leads to good results.
4. Moderately Sized Datasets: It is particularly effective on small to medium-sized datasets where other, more complex models might overfit. The simplicity of Naive Bayes allows it to generalize well even with limited data.

The Naive Bayesian Classifier assumes that all features are independent given the class label (the so-called "naive" assumption). This simplification makes the algorithm computationally efficient and easy to implement, even with relatively simple data. Despite the independence assumption, Naive Bayes often yields good results, especially when the independence assumption is approximately true or when the model's simplicity outweighs the impact of any correlations between features.

Q.2] How do you preprocess a dataset in Weka before applying the Naive Bayesian Classifier?

Ans: Before applying the Naive Bayesian Classifier, the following preprocessing steps are recommended in Weka:

1. Data Cleaning: Remove or impute missing values using Weka's "Filter" option under the "Preprocess" tab.
2. Discretization: If the dataset has continuous numerical attributes, consider discretizing them into categorical intervals using filters like unsupervised.attribute.Discretize.
3. Attribute Selection: Use Weka's attribute selection filters to remove irrelevant or redundant features, which can improve the model's performance.
4. Normalization: Though Naive Bayes generally handles raw data well, normalization can be applied to ensure that attributes are on a similar scale, especially when dealing with continuous data.

Q.3] How can you interpret the confusion matrix generated by the Naive Bayesian Classifier in Weka ?

Ans: The confusion matrix generated by the Naive Bayesian Classifier in Weka is a key tool for evaluating the performance of your model. The following are the ways to interpret the confusion matrix:

1. True Positives (TP): This value represents the number of instances that were correctly predicted as belonging to the positive class. For example, if you are classifying emails as "spam" or "not spam," TP would be the number of emails correctly identified as "spam."
2. True Negatives (TN): This value indicates the number of instances that were correctly predicted as belonging to the negative class. Continuing with the spam example, TN would be the number of emails correctly identified as "not spam.".
3. False Positives (FP): These are the instances where the classifier incorrectly predicted the positive class when it should have predicted the negative class. In the spam example, FP represents the number of "not spam" emails that were incorrectly classified as "spam." This is also known as a Type I error.
4. False Negatives (FN): These are the instances where the classifier incorrectly predicted the negative class when it should have predicted the positive class. In our example, FN would be the number of "spam" emails that were incorrectly classified as "not spam." This is also known as a Type II error.
5. Interpretation:
    - A high number of True Positives (TP) and True Negatives (TN) indicates that the classifier is performing well.
    - A high number of False Positives (FP) or False Negatives (FN) suggests areas where the model is misclassifying data, potentially requiring further tuning or alternative modeling approaches.
    - By analyzing the balance between Precision and Recall (via the F1-score), you can assess whether the model is biased towards one class, which is particularly important in imbalanced datasets.