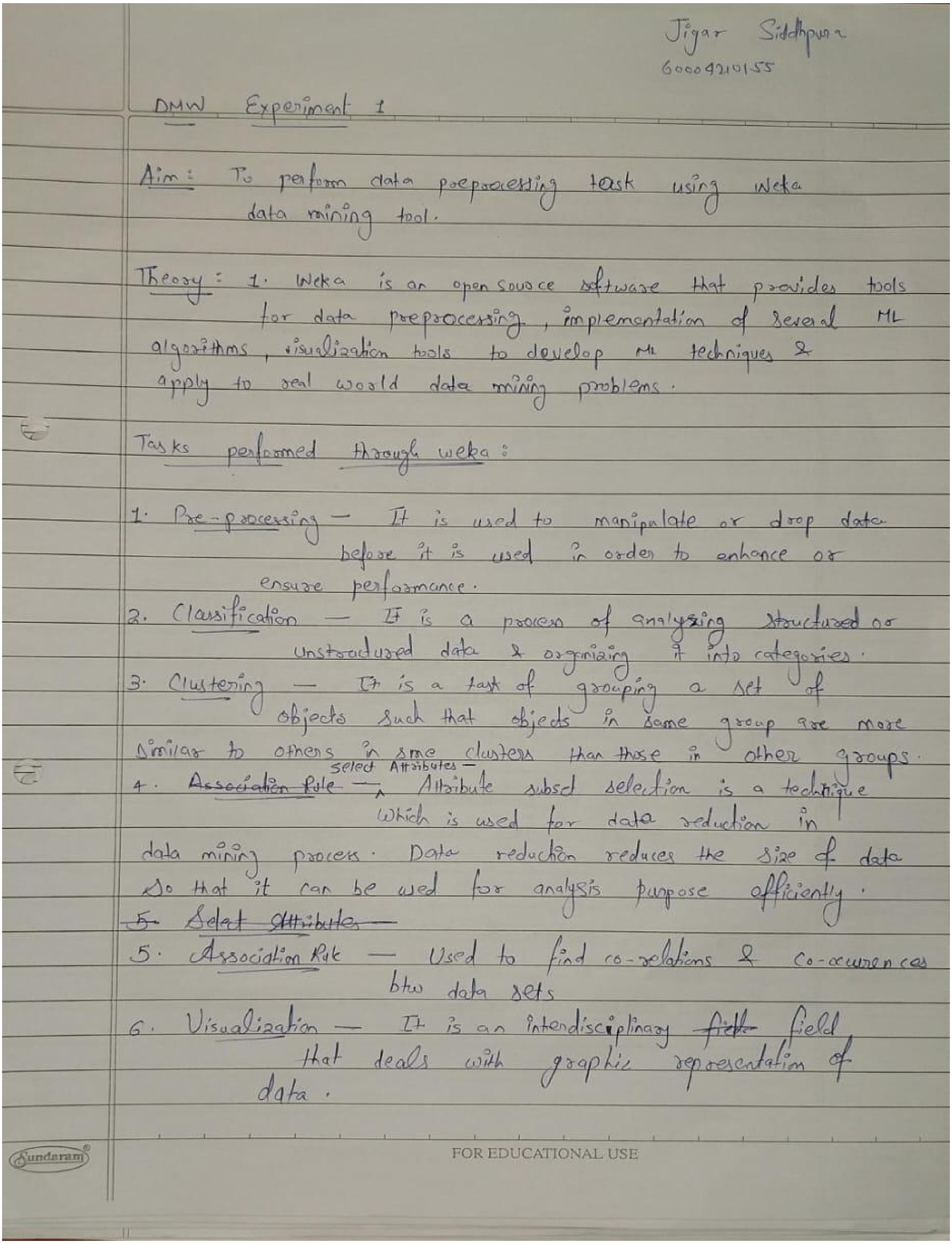
DATA MINING AND WAREHOUSE

PRACTICAL 1

Name: Jigar Siddhpura
SAP: 60004210155
Division/Batch: B2
Branch: Computer Engineering



Jigar Siddhpura
60004210155

DMW Experiment 1

Aim: To perform data preprocessing task using weka data mining tool.

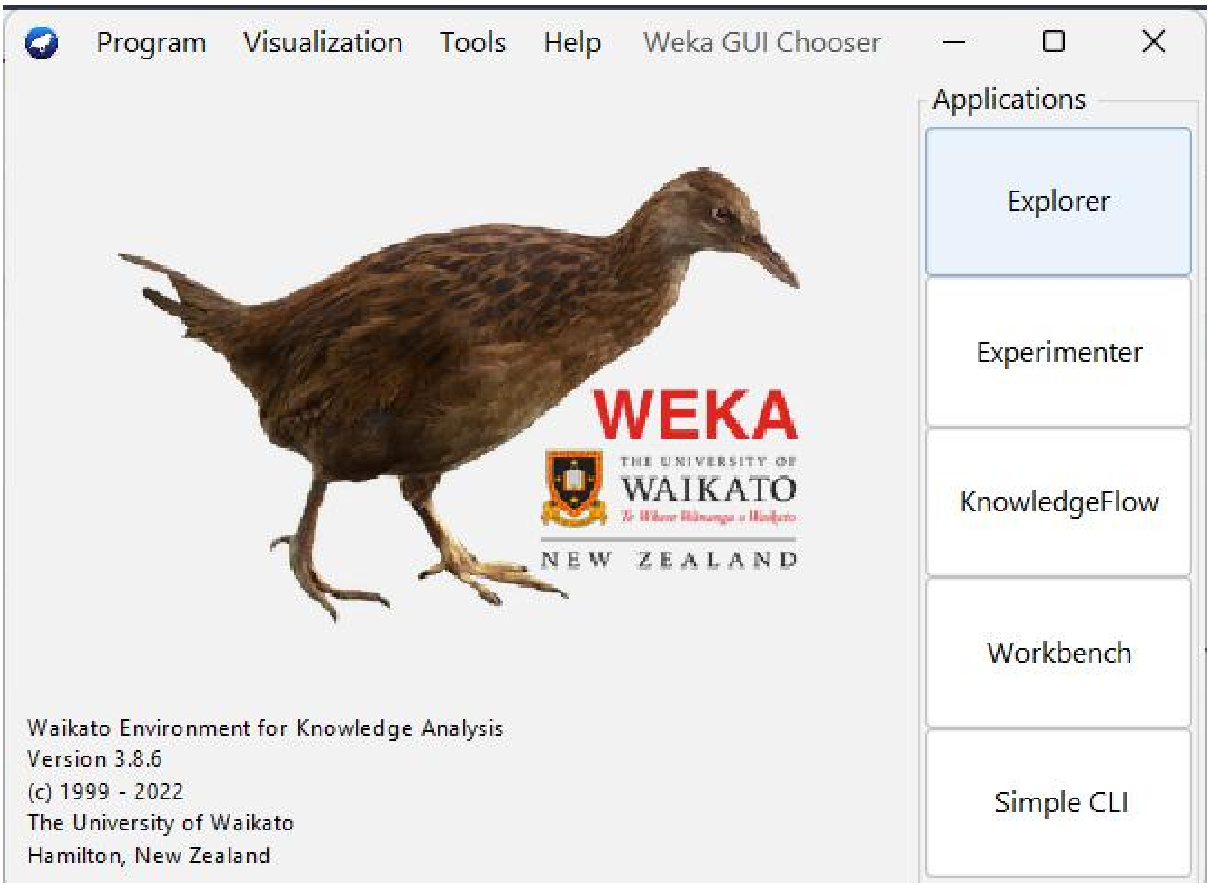
Theory: 1. Weka is an open source software that provides tools for data preprocessing, implementation of several ML algorithms, visualization tools to develop ML techniques & apply to real world data mining problems.

Tasks performed through weka:

1. Pre-processing — It is used to manipulate or drop data before it is used in order to enhance or ensure performance.
2. Classification — It is a process of analysing structured or unstructured data & organizing it into categories.
3. Clustering — It is a task of grouping a set of objects such that objects in same group are more similar to others in sme clusters than those in other groups.
4. ~~Association Rule~~ Select Attributes — Attribute subset selection is a technique which is used for data reduction in data mining process. Data reduction reduces the size of data so that it can be used for analysis purpose efficiently.

~~5. Select Attributes —~~

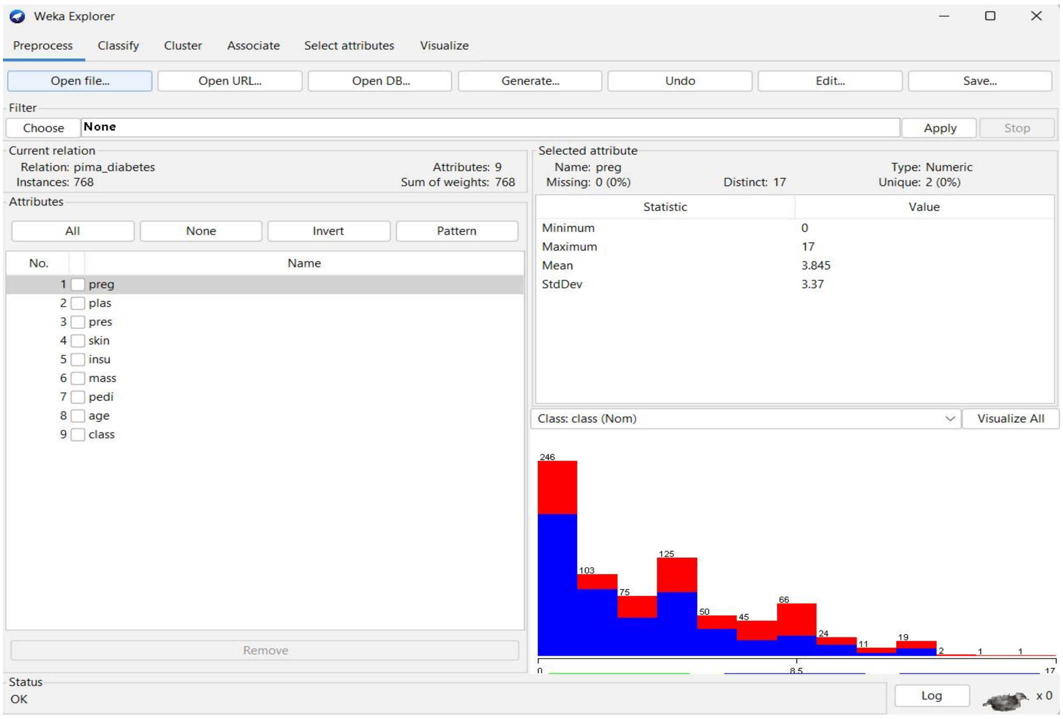
5. Association Rule — Used to find co-relations & co-occurences btw data sets
6. Visualization — It is an interdisciplinary ~~field~~ field that deals with graphic representation of data.



Waikato Environment for Knowledge Analysis
Version 3.8.6
(c) 1999 - 2022
The University of Waikato
Hamilton, New Zealand

Pre-processing activities to be observed in Weka

1. **Visualization:** Visualize scatter plot for all the attributes from the dataset selected from Weka. Determine correlation if any using these plots for different datasets



Filter: None

Current relation
Relation: pima_diabetes
Instances: 768
Attributes: 9
Sum of weights: 768

Selected attribute
Name: preg
Missing: 0 (0%)
Distinct: 17
Type: Numeric
Unique: 2 (0%)

Statistic	Value
Minimum	0
Maximum	17
Mean	3.845
StdDev	3.37

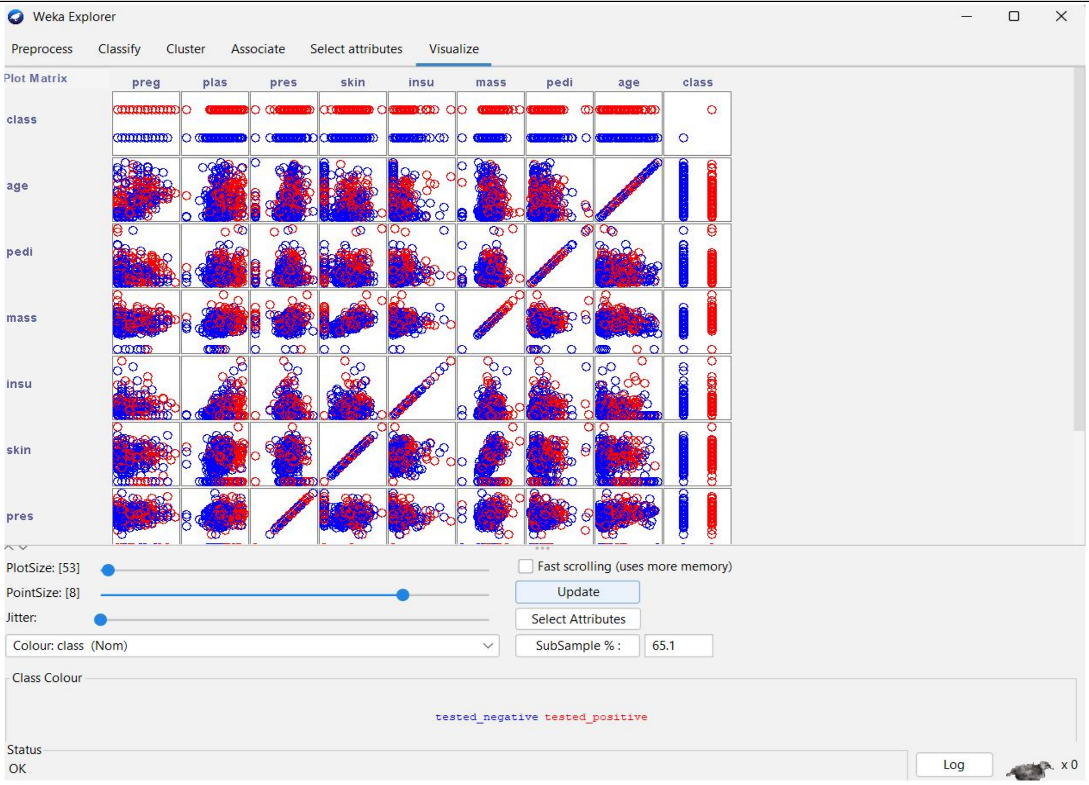
Attributes

No.	Name
1	☐ preg
2	☐ plas
3	☐ pres
4	☐ skin
5	☐ insu
6	☐ mass
7	☐ pedi
8	☐ age
9	☐ class

Class: class (Nom)

Status
OK

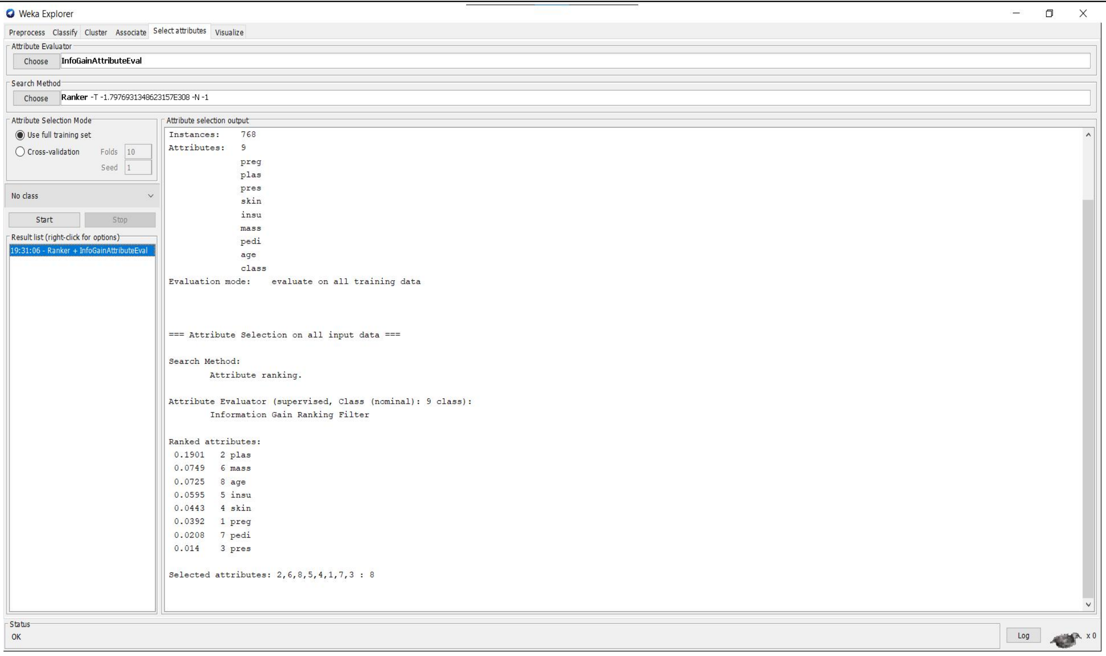
We have loaded the ***Diabetes*** dataset into Weka and have visualized the various attributes. The histograms for each attribute display the variation in values within that attribute and can be viewed for each attribute.



Here we can visualise the scatter plots for each of the attributes of the dataset taken in pairs. By observing the scatter plot, we can infer the following:

a. Body mass index (weight in kg/ (height in m) ^2) vs Triceps skin fold thickness (mm) – There is a positive correlation between these attributes
b. 2-Hour serum insulin (mu U/ml) vs Plasma glucose concentration 2 hours in an oral glucose tolerance test – There is a positive correlation between these attributes
c. Diabetes pedigree function vs Triceps skin fold thickness (mm) – There is no correlation between these attributes and the plot is sparsely distributed

2. **Select Attributes**: Apply suitable feature selection filters like Gain Ratio etc to choose relevant attributes from the list of attributes. Observe the ranks / priority provided by the filter.



Attribute Evaluator: Choose InfoGainAttributeEval

Search Method: Choose Ranker -T -1.7976931348623157E308 -N -1

● Use full training set
○ Cross-validation Folds 10 Seed 1

No class

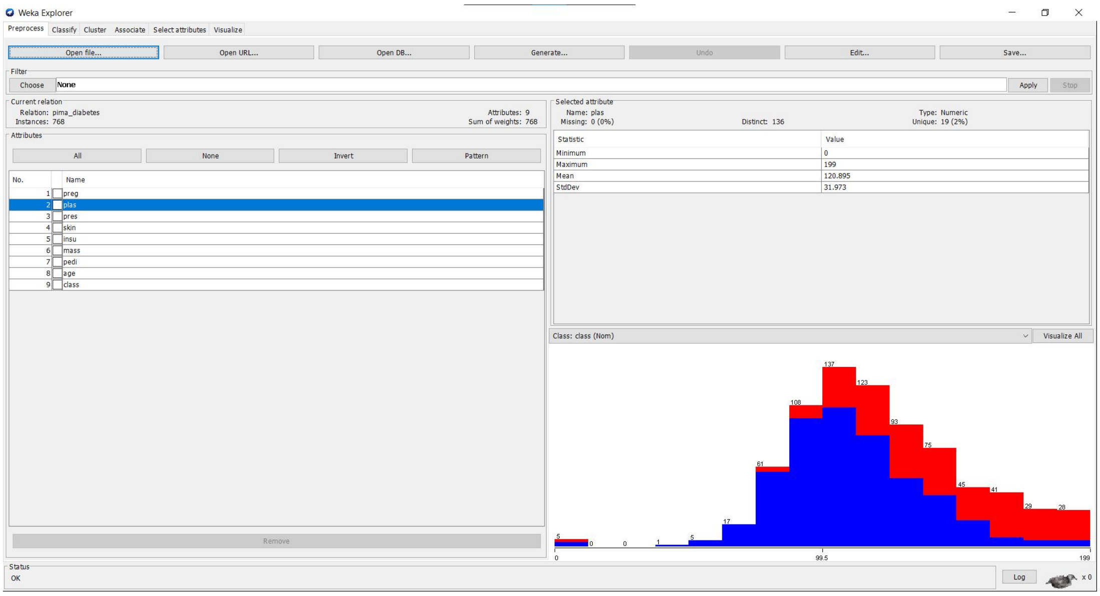
19:31:06 - Ranker + InfoGainAttributeEval

Attribute selection output

```
Instances:    768
Attributes:   9
              preg
              plas
              pres
              skin
              insu
              mass
              pedi
              age
              class
Evaluation mode:    evaluate on all training data



=== Attribute Selection on all input data ===

Search Method:
	Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 9 class):
	Information Gain Ranking Filter

Ranked attributes:
 0.1901   2 plas
 0.0749   6 mass
 0.0725   8 age
 0.0595   5 insu
 0.0443   4 skin
 0.0392   1 preg
 0.0208   7 pedi
 0.014    3 pres

Selected attributes: 2,6,8,5,4,1,7,3 : 8
```

Status
OK

We now switch to the Select Attribute tab and have used the InfoGainAttributeEval method with the attribute Ranker. From the above results, the attribute ‘plas’ which stands for ‘Plasma glucose concentration 2 hours in an oral glucose tolerance test’ is the highest ranked with a score of 0.1901. This shows that the plasma attribute is the most important attribute for further processing.



Filter: Choose None

Current relation
Relation: pima_diabetes
Instances: 768
Attributes: 9
Sum of weights: 768

Attributes

No.		Name
1	□	preg
2	□	plas
3	□	pres
4	□	skin
5	□	insu
6	□	mass
7	□	pedi
8	□	age
9	□	class

Selected attribute
Name: plas
Missing: 0 (0%)
Distinct: 136
Type: Numeric
Unique: 19 (2%)

Statistic	Value
Minimum	0
Maximum	199
Mean	120.895
StdDev	31.973

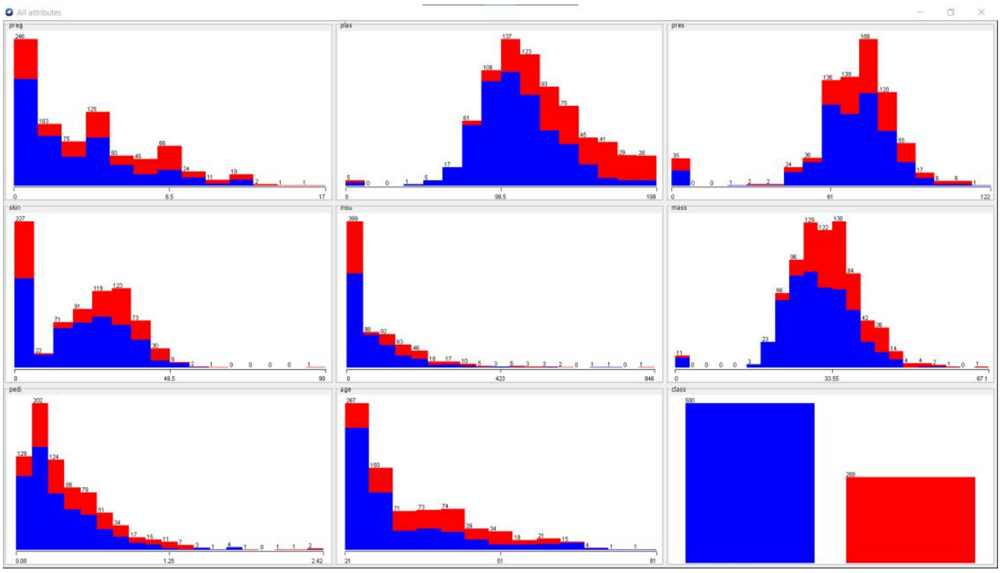
Class: class (Nom)

Status
OK

The data can be seen influenced by the attribute ‘plas’ and a comparably clear separation in classes correlating with its values.

3. **Pre-processing**:

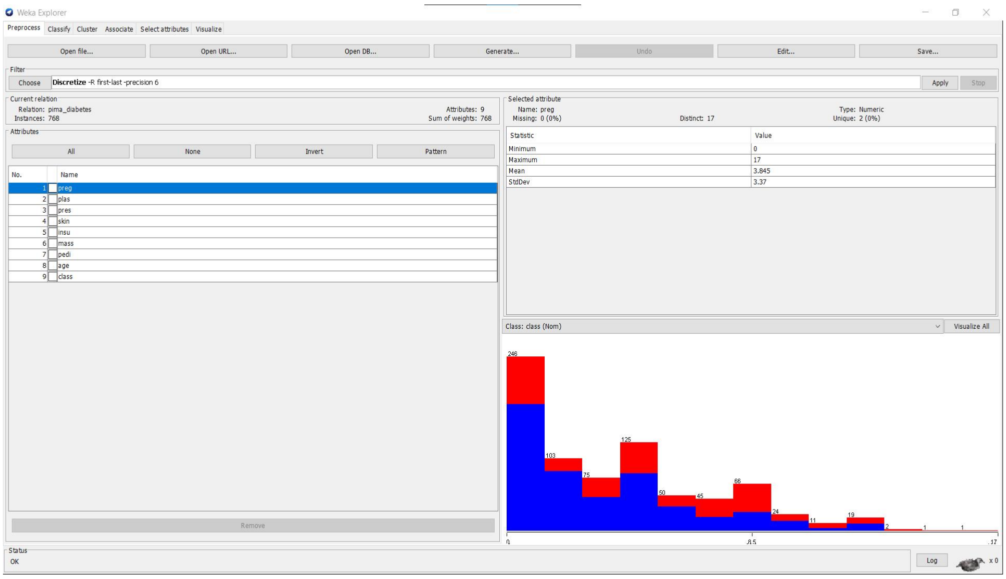
   a. **Visualize All**: Select this button to visualize histograms of all attributes.



Here we have visualized all of the attributes and their histograms. We can infer that some attributes like skin, pedi, preg, insu, age are left skewed while pres, plas, mass are normally distributed.

b. **Filter**: Choose Discretization under Unsupervised and Supervised methods. Observe the discretization and the outliers.

Before:



Filter: Discretize -R first-last -precision 6

Current relation
Relation: pima_diabetes
Instances: 768
Attributes: 9
Sum of weights: 768

Selected attribute
Name: preg
Missing: 0 (0%)
Distinct: 17
Type: Numeric
Unique: 2 (0%)

Statistic	Value
Minimum	0
Maximum	17
Mean	3.845
StdDev	3.37

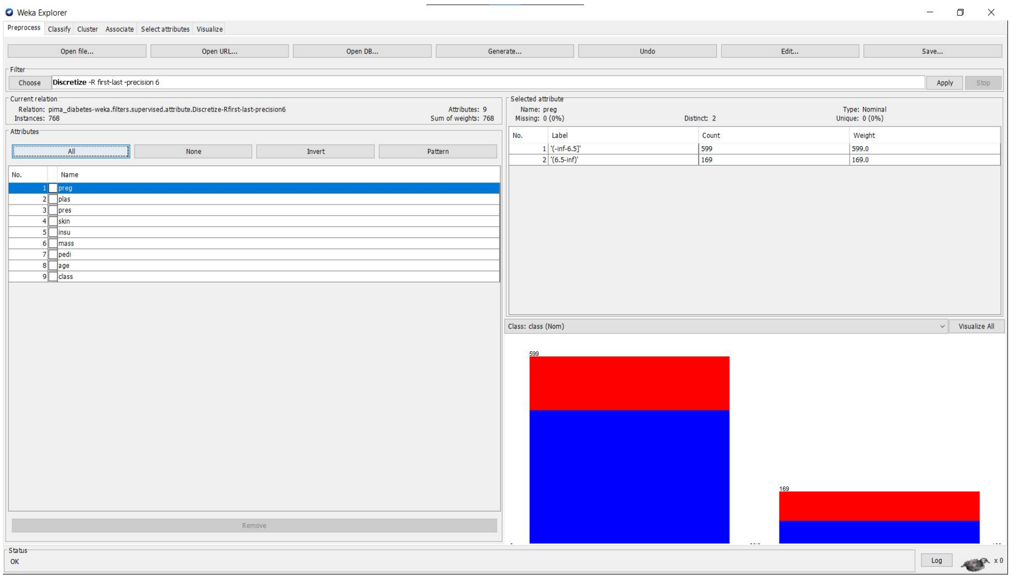
Attributes

No.		Name
1	□	preg
2	□	plas
3	□	pres
4	□	skin
5	□	insu
6	□	mass
7	□	pedi
8	□	age
9	□	class

Status
OK

The attribute value before discretization shows a continuous histogram with varying values and no of instances.

After:



Filter: Discretize -R first-last -precision 6

Current relation
Relation: pima_diabetes-weka.filters.supervised.attribute.Discretize-Rfirst-last-precision6
Instances: 768
Attributes: 9
Sum of weights: 768

Selected attribute
Name: preg
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	'(-inf-6.5]'	599	599.0
2	'(6.5-inf)'	169	169.0

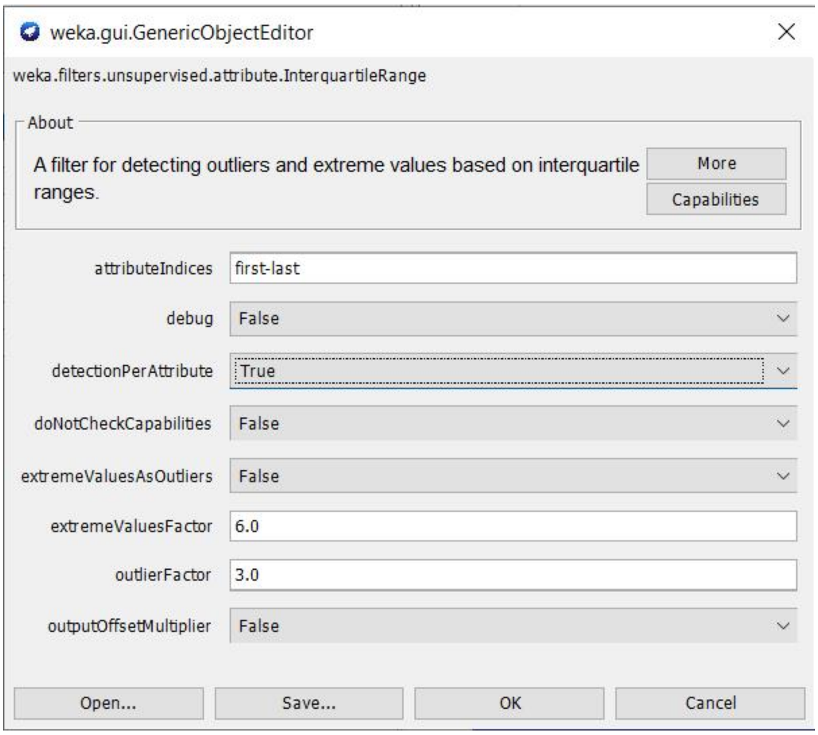
Attributes

No.		Name
1	□	preg
2	□	plas
3	□	pres
4	□	skin
5	□	insu
6	□	mass
7	□	pedi
8	□	age
9	□	class

Status
OK

Here we have applied unsupervised discrete filtering to the dataset. For each attribute discrete buckets are created, and the instances are separated into their appropriate bucket. The attribute here is nominal and hence the bucket is split based on their values and extends from -infinity to 6.5 and from 6.5 to infinity respectively.

c. **IQR**: Observe the IQR values for a selected attribute. Observe the outlier and extreme values



weka.gui.GenericObjectEditor

weka.filters.unsupervised.attribute.InterquartileRange

About

A filter for detecting outliers and extreme values based on interquartile ranges.

attributeIndices: first-last

debug: False

detectionPerAttribute: True

doNotCheckCapabilities: False

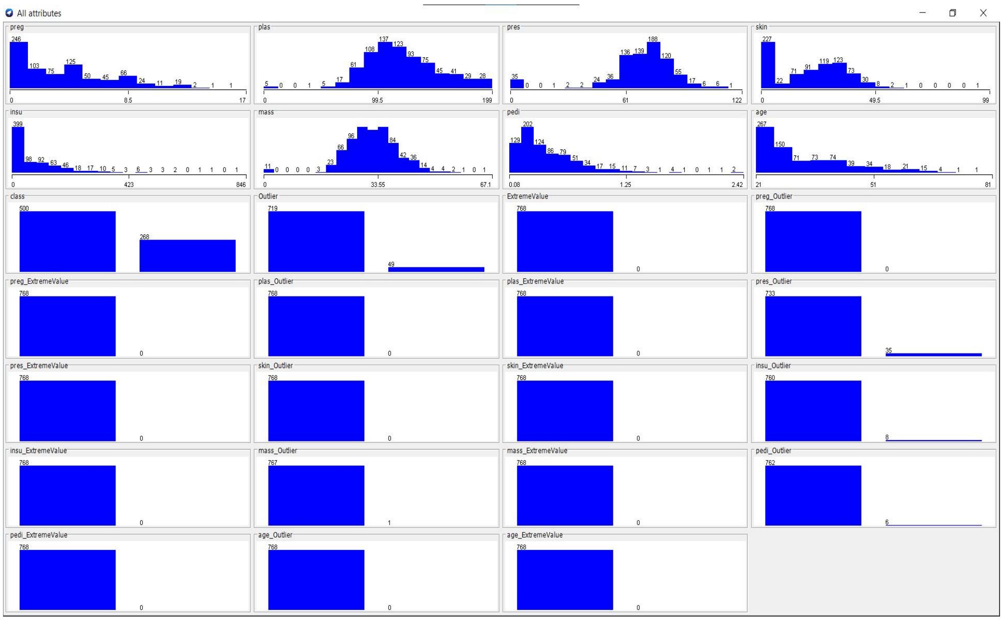
extremeValuesAsOutliers: False

extremeValuesFactor: 6.0

outlierFactor: 3.0

outputOffsetMultiplier: False

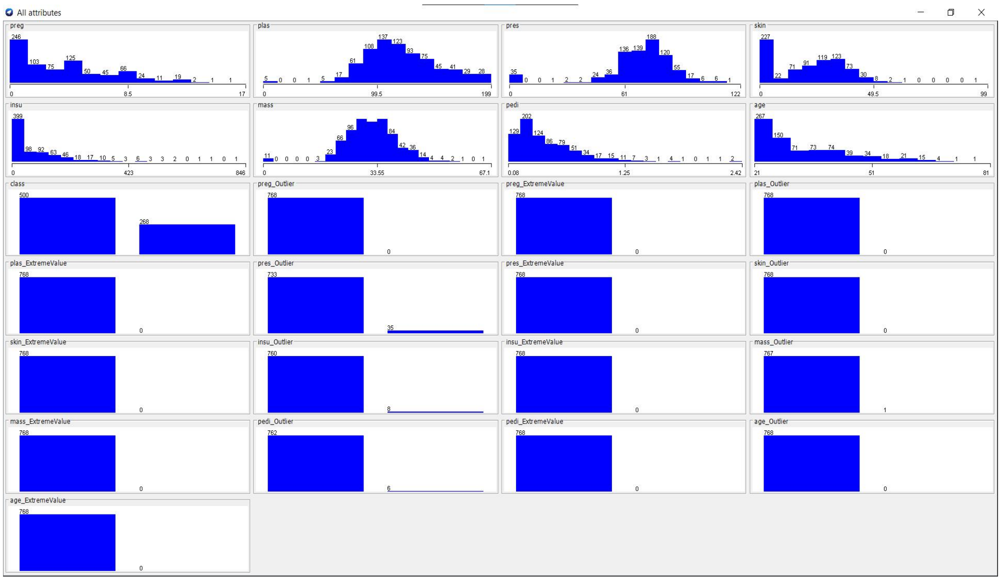
These are the settings that have been applied for IQR.



When we apply the Inter Quartile Range (IQR) per attribute, we are able to see the outliers present in each of the attributes separately along with the extreme values that are present. Outliers are reported at less than Q1-1.5*IQR and greater than Q3+1.5*IQR.

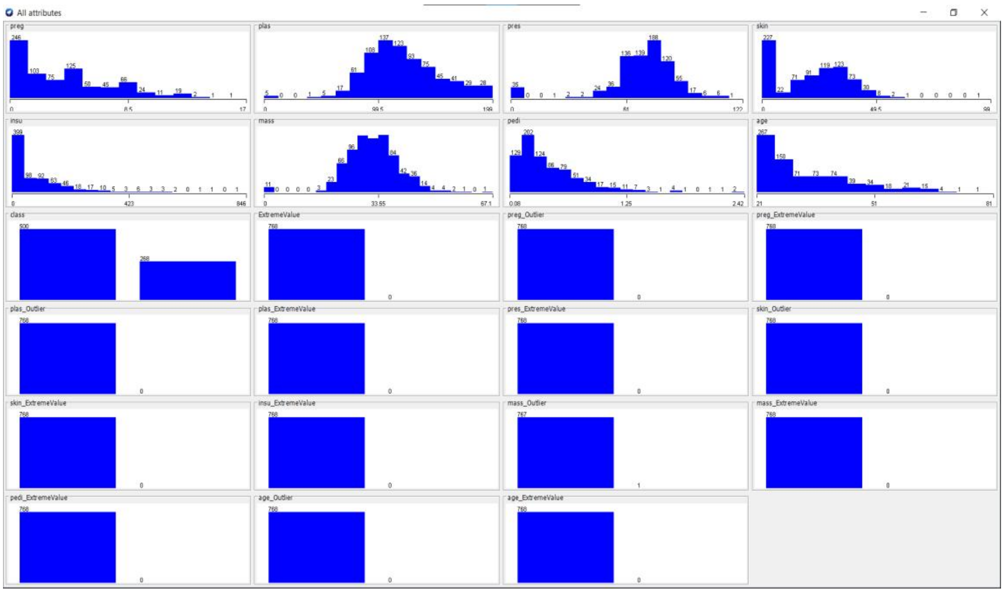
d. **Remove the value**: Remove instances with outlier values and show the screenshots of dataset before and after the removal.

Before:



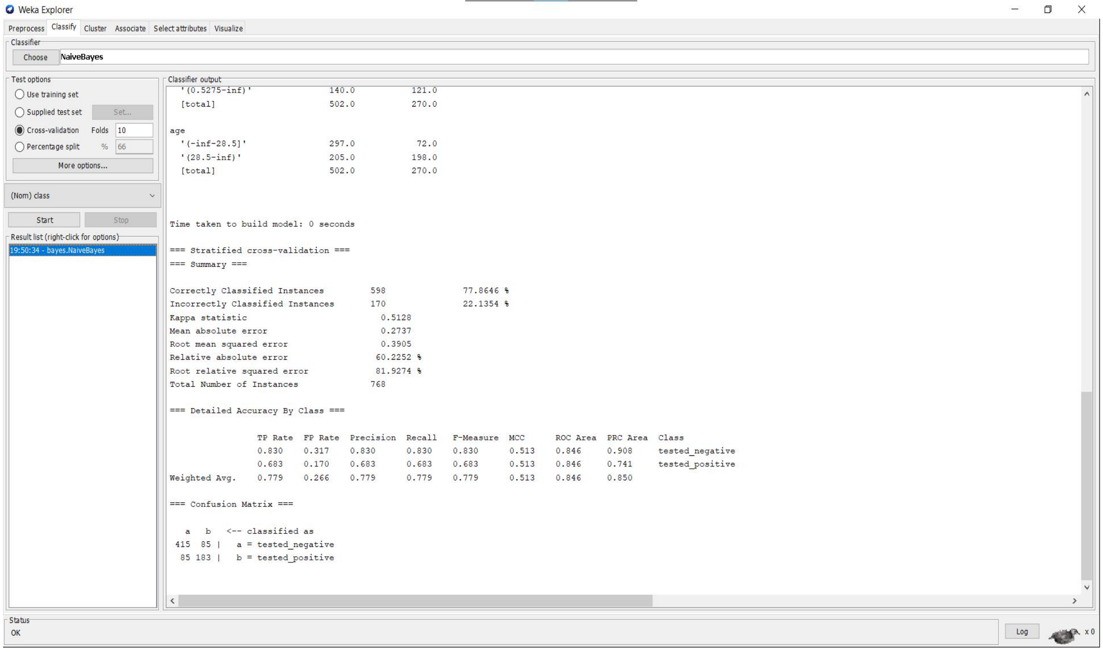
From the above IQR plot, we can determine the attributes that have the outliers. From this dataset, the attributes 'predi', 'insu' and 'pres' have outliers with a combined total of 49 instances.

After:



We can then use the attributes menu to remove those instances from the respective attributes and again visualising the attributes graphs shows that the outliers in the dataset have been removed.

4. **Classification**: Perform NB, kNN and DT/rule-based classification



Classifier: NaiveBayes

Folds: 10

%: 66

(Nom) class

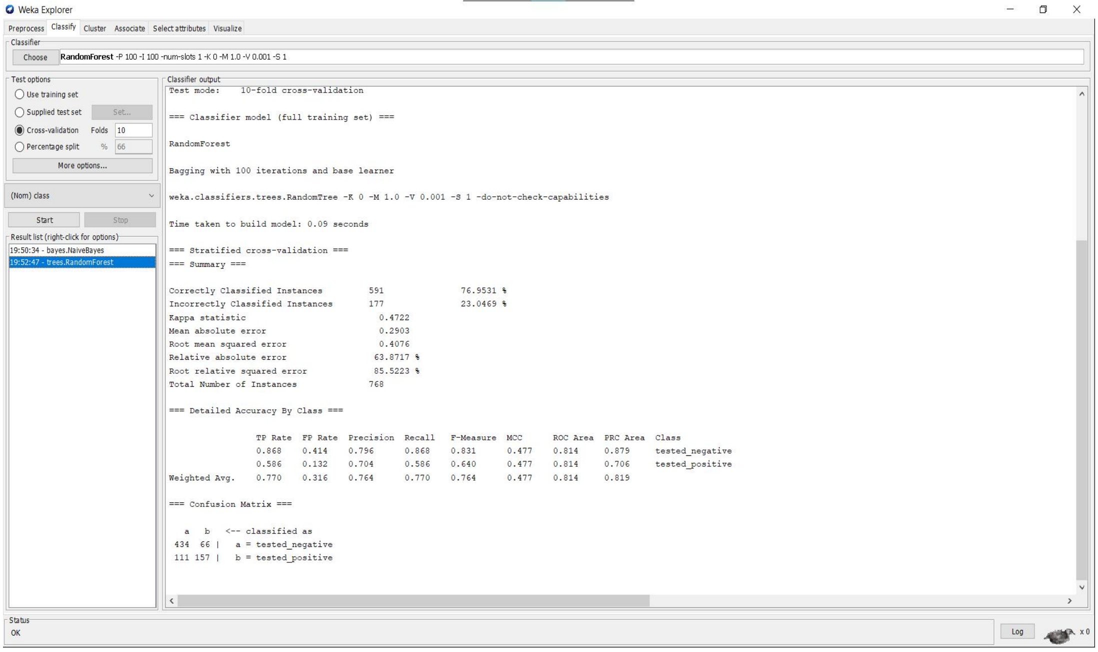
19:50:34 - bayes.NaiveBayes

Status: OK

Classifier output

```
  '(0.5275-inf)'              140.0          121.0
  [total]                     502.0          270.0

age
  '(-inf-28.5]'               297.0           72.0
  '(28.5-inf)'                205.0          198.0
  [total]                     502.0          270.0



Time taken to build model: 0 seconds

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

Correctly Classified Instances         598               77.8646 %
Incorrectly Classified Instances       170               22.1354 %
Kappa statistic                          0.5128
Mean absolute error                      0.2737
Root mean squared error                  0.3905
Relative absolute error                 60.2252 %
Root relative squared error             81.9274 %
Total Number of Instances              768

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.830    0.317    0.830      0.830    0.830      0.513    0.846     0.908     tested_negative
                 0.683    0.170    0.683      0.683    0.683      0.513    0.846     0.741     tested_positive
Weighted Avg.    0.779    0.266    0.779      0.779    0.779      0.513    0.846     0.850

=== Confusion Matrix ===

   a   b   <-- classified as
 415  85 |   a = tested_negative
  85 183 |   b = tested_positive
```

We have performed Naïve Bayes classification on the dataset. The Naïve bayes classifier correctly classified 77.86% of the instances while 22.13% were incorrectly classified.



Classifier: RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Folds: 10

%: 66

(Nom) class

19:50:34 - bayes.NaiveBayes
19:52:47 - trees.RandomForest

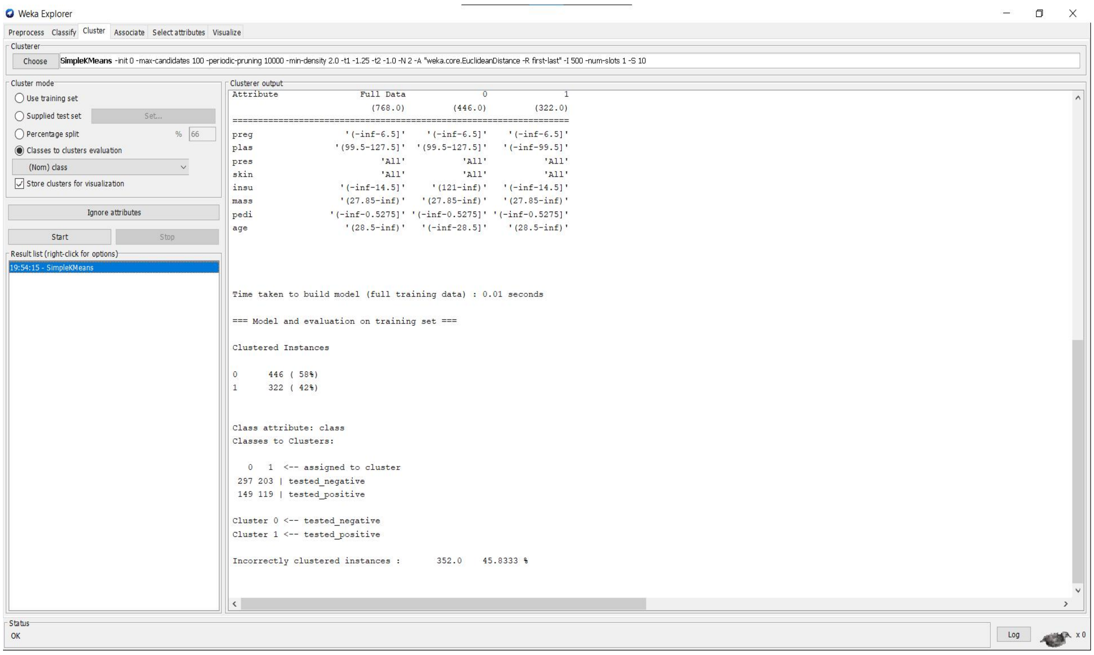
Status: OK

Classifier output

```
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.09 seconds

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

Correctly Classified Instances         591               76.9531 %
Incorrectly Classified Instances       177               23.0469 %
Kappa statistic                          0.4722
Mean absolute error                      0.2903
Root mean squared error                  0.4076
Relative absolute error                 63.8717 %
Root relative squared error             85.5223 %
Total Number of Instances              768

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.868    0.414    0.796      0.868    0.831      0.477    0.814     0.879     tested_negative
                 0.586    0.132    0.704      0.586    0.640      0.477    0.814     0.706     tested_positive
Weighted Avg.    0.770    0.316    0.764      0.770    0.764      0.477    0.814     0.819

=== Confusion Matrix ===

   a   b   <-- classified as
 434  66 |   a = tested_negative
 111 157 |   b = tested_positive
```

We have performed Random Forest (Decision Tree/Rule Based) classification on the dataset. The Random Forest classifier correctly classified 76.95% of the instances while 23.04% were incorrectly classified.

5. **Clustering**: Perform kmeans, hierarchical clustering and explain the output



SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10

Cluster mode
○ Use training set
○ Supplied test set
○ Percentage split % 66
● Classes to clusters evaluation
(Nom) class
☑ Store clusters for visualization

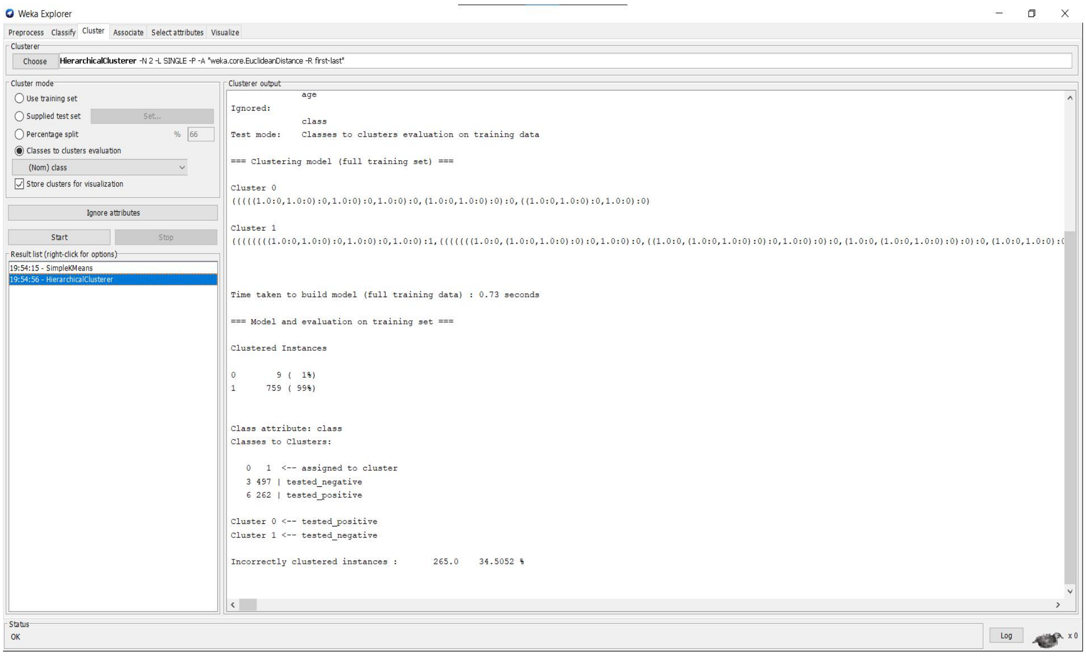
19:54:15 - SimpleKMeans

Clusterer output

```
Attribute               Full Data              0                1
                          (768.0)          (446.0)          (322.0)
==================================================================
preg                  '(-inf-6.5]'     '(-inf-6.5]'     '(-inf-6.5]'
plas               '(99.5-127.5]'   '(99.5-127.5]'     '(-inf-99.5]'
pres                        'All'            'All'            'All'
skin                        'All'            'All'            'All'
insu                 '(-inf-14.5]'      '(121-inf)'    '(-inf-14.5]'
mass                 '(27.85-inf)'    '(27.85-inf)'    '(27.85-inf)'
pedi              '(-inf-0.5275]'  '(-inf-0.5275]'  '(-inf-0.5275]'
age                  '(28.5-inf)'    '(-inf-28.5]'     '(28.5-inf)'




Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      446 ( 58%)
1      322 ( 42%)


Class attribute: class
Classes to Clusters:

   0   1  <-- assigned to cluster
 297 203 | tested_negative
 149 119 | tested_positive

Cluster 0 <-- tested_negative
Cluster 1 <-- tested_positive

Incorrectly clustered instances :       352.0    45.8333 %
```

Status
OK

We have performed K-Means Clustering on the dataset to form groups of similar instances from the data. It incorrectly clustered 45.83% of the instances.



HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"

Cluster mode
○ Use training set
○ Supplied test set
○ Percentage split % 66
● Classes to clusters evaluation
(Nom) class
☑ Store clusters for visualization

19:54:15 - SimpleKMeans
19:54:56 - HierarchicalClusterer

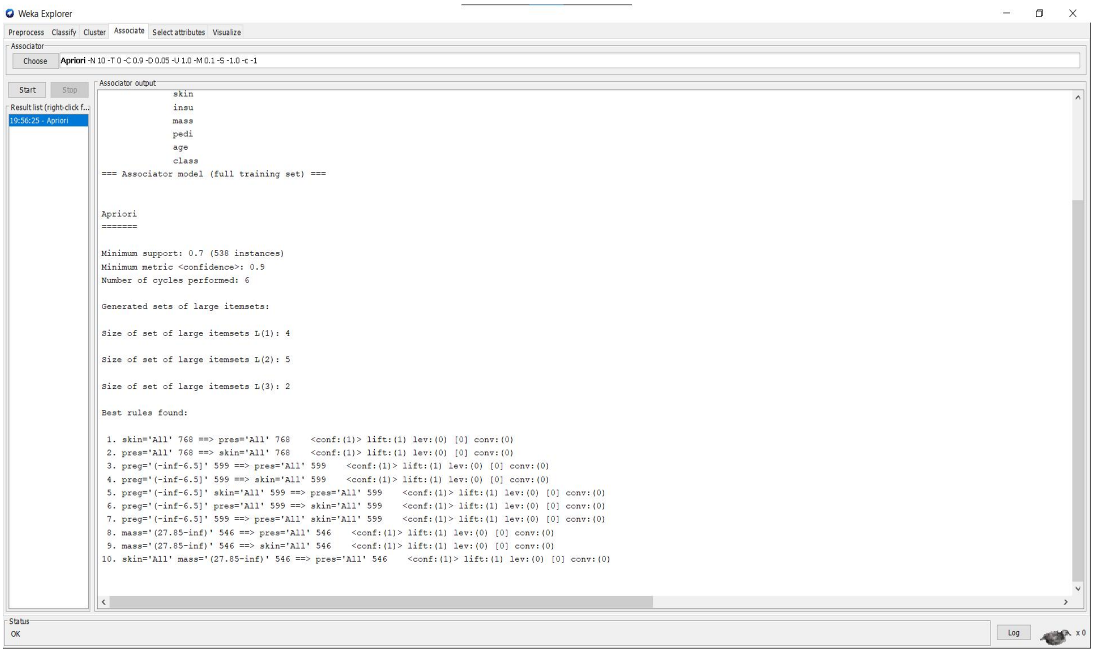
Clusterer output

```
              age
Ignored:
              class
Test mode:    Classes to clusters evaluation on training data

=== Clustering model (full training set) ===

Cluster 0
((((((1.0:0,1.0:0):0,1.0:0):0,1.0:0):0,(1.0:0,1.0:0):0):0,((1.0:0,1.0:0):0,1.0:0):0)

Cluster 1
(((((((1.0:0,1.0:0):0,1.0:0):0,1.0:0):1,((((((1.0:0,(1.0:0,1.0:0):0):0,1.0:0):0,((1.0:0,(1.0:0,1.0:0):0):0,1.0:0):0):0,(1.0:0,(1.0:0,1.0:0):0):0):0,(1.0:0,1.0:0):(



Time taken to build model (full training data) : 0.73 seconds

=== Model and evaluation on training set ===

Clustered Instances

0        9 (  1%)
1      759 ( 99%)


Class attribute: class
Classes to Clusters:

   0   1  <-- assigned to cluster
   3 497 | tested_negative
   6 262 | tested_positive

Cluster 0 <-- tested_positive
Cluster 1 <-- tested_negative

Incorrectly clustered instances :       265.0    34.5052 %
```

Status
OK

We have performed Hierarchical Clustering on the dataset to form groups of similar instances from the data. It incorrectly clustered 34.51% of the instances, thus being comparatively better than the K-Means clustering algorithm for this dataset.

6. **Association rule mining**: Perform apriori algo and show the rules created



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
              skin
              insu
              mass
              pedi
              age
              class
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.7 (538 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 6

Generated sets of large itemsets:

Size of set of large itemsets L(1): 4

Size of set of large itemsets L(2): 5

Size of set of large itemsets L(3): 2

Best rules found:

 1. skin='All' 768 ==> pres='All' 768    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 2. pres='All' 768 ==> skin='All' 768    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 3. preg='(-inf-6.5]' 599 ==> pres='All' 599    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 4. preg='(-inf-6.5]' 599 ==> skin='All' 599    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 5. preg='(-inf-6.5]' skin='All' 599 ==> pres='All' 599    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 6. preg='(-inf-6.5]' pres='All' 599 ==> skin='All' 599    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 7. preg='(-inf-6.5]' 599 ==> pres='All' skin='All' 599    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 8. mass='(27.85-inf)' 546 ==> pres='All' 546    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 9. mass='(27.85-inf)' 546 ==> skin='All' 546    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
10. skin='All' mass='(27.85-inf)' 546 ==> pres='All' 546    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
```

Associative rule mining is used to find associations in the data. Weka has an associate tab that allows us to perform associative rule mining using algorithms like apriori. Here we have applied apriori to the dataset. From the results, we can infer that:

a. The attributes skin, mass and pres are associative
b. The attributes mass and pres are associative
c. The attributes preg and skin are associative

Conclusion

Weka is a very intuitive tool for exploring datasets and performing data mining operations. It provides extensive support for data visualisations through histograms to visualise variation in the instances within the data and through scatter plots for correlating various attributes. It also provides IQR which allows us to detect and remove outliers. We can also filter the data through discretisation and other processes and visualise the state of the data while doing so. It also allows us to select attributes from the dataset and then perform classification and clustering on the data to identify similar groups. Weka also allows us to perform associative mining on the data to find associated attributes for basket analysis.