

CS 634 - DATA MINING

FINAL TERM PROJECT

Option 1 (Supervised Data Mining)

Decision Tree (Category 3)

And

Naïve Bayes (Category 5)

TO: DR. JASON T.L. WANG, PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE

NEW JERSEY INSTITUTE OF TECHNOLOGY

DONE BY: CHARANPREET KAUR DHIR

UCID: 31478357

CKD22 @NJIT.EDU

INDEX

- OBJECTIVE
- DATA SET FOR THE PROJECT
 - DATA SET INFORMATION
 - ATTRIBUTE INFORMATION
 - DATA IN CSV FILE
- DECISION TREE (C4.5)
 - INTRODUCTION
 - ALGORITHM
 - SOFTWARE USED: WEKA
 - ALGORITHM WORKING
 - CLASSIFIER OUTPUT
 - SOURCE CODE
- NAÏVE BAYES
 - INTRODUCTION
 - ALGORITHM
 - SOFTWARE USED: WEKA
 - ALGORITHM WORKING
 - CLASSIFIER OUTPUT
 - SOURCE CODE
- COMPARISON BETWEEN DECISION TREE AND NAÏVE BAYES

OBJECTIVE

Implement 2 classification algorithm i.e. Decision tree and Random Forest tree on a dataset. And for our project I chose Breast Cancer Wisconsin (Prognostic) Data Set.

DATA SET FOR THE PROJCT

URL : <http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Prognostic%29>

Title : Breast Cancer Wisconsin (Prognostic) Data Set

DATA SET INFORMATION :

Each record represents follow-up data for one breast cancer case. These are consecutive patients seen by Dr. Wolberg since 1984, and include only those cases exhibiting invasive breast cancer and no evidence of distant metastases at the time of diagnosis.

There are total of 699 breast cancer cases. Each case could be diagnosed as benign or malignant. For the diagnosis, each case has ten attributes listed in the table below. The last attribute in the table represents the class, or label with which the case was diagnosed.

Class distribution:

Benign: 458 (65.5%)

Malignant: 241 (34.5%)

ATTRIBUTE INFORMATION:

ATTRIBUTE	DOMAIN
Sample code number	id number
Clump Thickness	1 – 10
Uniformity of Cell Size	1 – 10
Uniformity of Cell Shape	1 – 10
Marginal Adhesion	1 – 10
Single Epithelial Cell Size	1 – 10
Bare Nuclei	1 – 10
Bland Chromatin	1 – 10
Normal Nucleoli	1 – 10
Mitoses	1 – 10
Class:	(2 for benign, 4 for malignant)

DATA IN CSV FILE:

The data is stored in CSV file displayed below and is used further in WEKA.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Sample code number	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class				
2	1000025	5	1	1	1	2	1	3	1	1	2				
3	1002945	5	4	4	5	7	10	3	2	1	2				
4	1015425	3	1	1	1	2	2	3	1	1	2				
5	1016277	6	8	8	1	3	4	3	7	1	2				
6	1017023	4	1	1	3	2	1	3	1	1	2				
7	1017122	8	10	10	8	7	10	9	7	1	4				
8	1018099	1	1	1	1	2	10	3	1	1	2				
9	1018561	2	1	2	1	2	1	3	1	1	2				
10	1033078	2	1	1	1	2	1	1	1	5	2				
11	1033078	4	2	1	1	2	1	2	1	1	2				
12	1035283	1	1	1	1	1	1	3	1	1	2				
13	1036172	2	1	1	1	2	1	2	1	1	2				
14	1041801	5	3	3	3	2	3	4	4	1	4				
15	1043999	1	1	1	1	2	3	3	1	1	2				
16	1044572	8	7	5	10	7	9	5	5	4	4				
17	1047630	7	4	6	4	6	1	4	3	1	4				
18	1048672	4	1	1	1	2	1	2	1	1	2				
19	1049815	4	1	1	1	2	1	3	1	1	2				
20	1050670	10	7	7	6	4	10	4	1	2	4				
21	1050718	6	1	1	1	2	1	3	1	1	2				
22	1054590	7	3	2	10	5	10	5	4	4	4				
23	1054593	10	5	5	3	6	7	7	10	1	4				
24	1056784	3	1	1	1	2	1	2	1	1	2				
25	1057013	8	4	5	1	2	?	7	3	1	4				
26	1059552	1	1	1	1	2	1	3	1	1	2				

DECISION TREE (C4.5)

INTRODUCTION

A decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node. Good for categorical attributes with non-continuous values.

ALGORITHM

- The attribute with the highest information gain is selected at each level.
- Suppose there are two classes P (positive) and N (negative) where P contains p training examples and N contains n training examples

$$I(p, n) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

- Assume that using attribute A a set S will be partitioned into subsets $\{S_1, S_2, \dots, S_v\}$
- If S_i contains p_i examples of P and n_i examples of N, the entropy, or the expected information needed to classify objects in all subtree's S_i is

$$E(A) = \sum_{i=1}^v \frac{p_i + n_i}{p + n} I(p_i, n_i)$$
$$Gain(A) = I(p, n) - E(A)$$

SOFTWARE USED: WEKA

The ID3 algorithm classifies data with discrete values; in order for WEKA to recognize the dataset values as discrete a pre-processing phase must be done in the data before running the algorithm. In this phase the Discretization of the dataset is done, along with the filtering of the “Sample code number” attribute that is not needed for the classification.

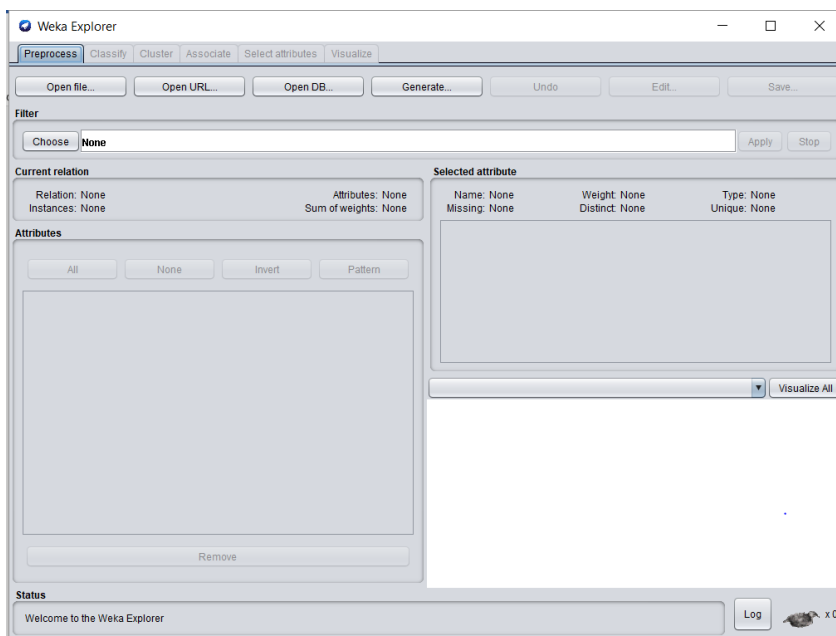
Software for Weka is downloaded from : <https://sourceforge.net/projects/weka/>

STEPS :

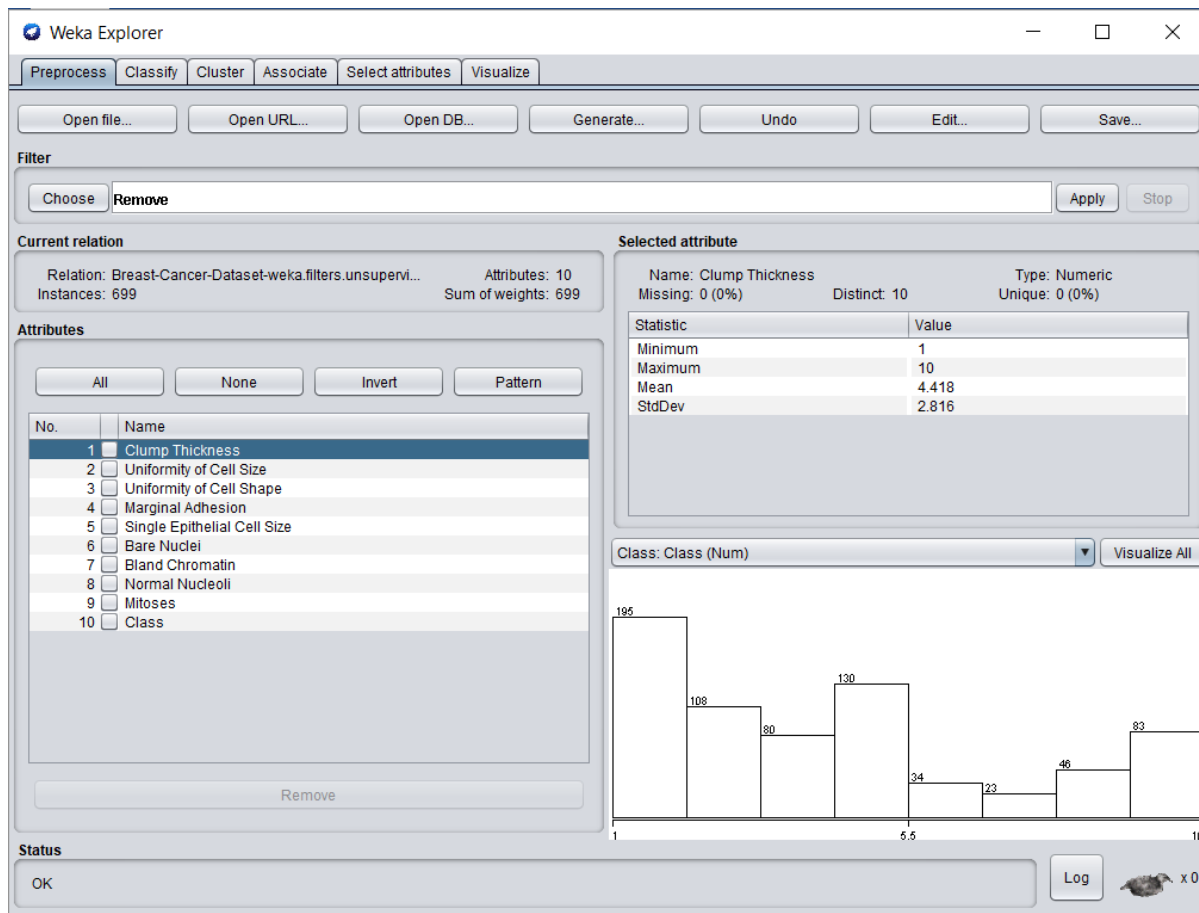
- ✓ We open the WEKA tool and click in the “Explorer” button



- ✓ Click on “OPEN FILE” and select the Breast Cancer Data Set csv file stored on the disk.



- ✓ We remove the first attribute “Sample Code Number” attribute by clicking in the Choose button located under the word Filter, and then we choose the option filters -> unsupervised -> attribute -> remove

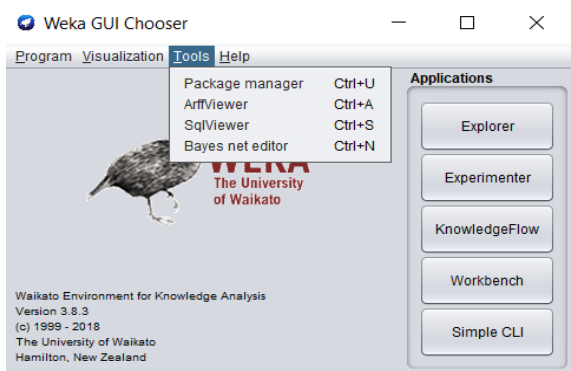


How to create .arff file using weka?

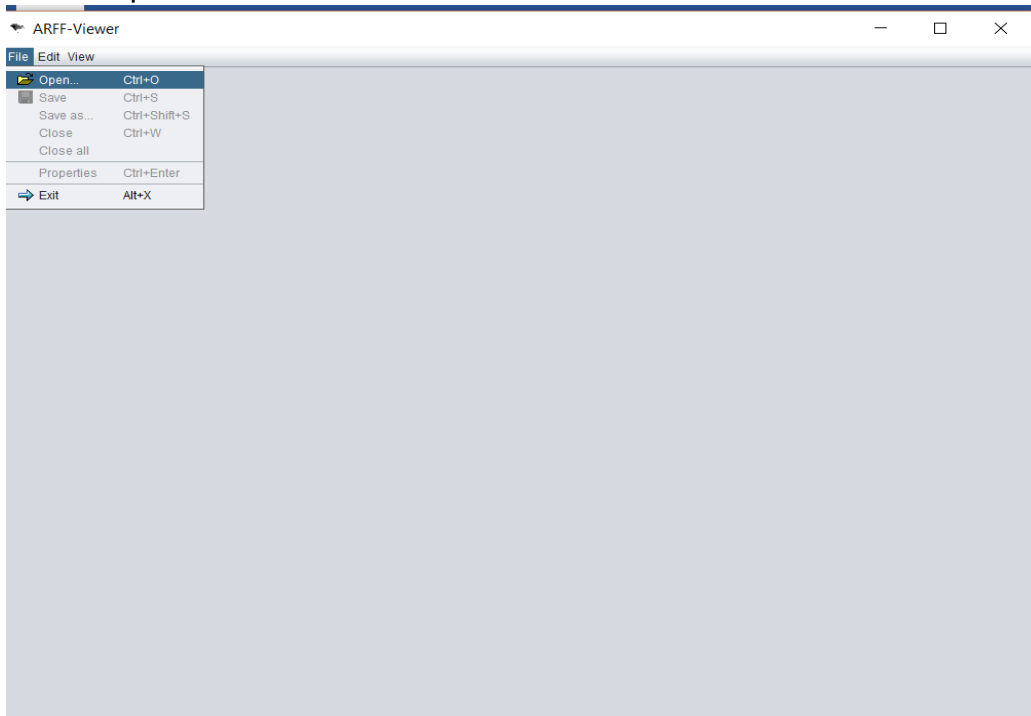


✓ Click on tools

✓ Click on ArffViewer



✓ File -> Open -> Select the csv data set file



✓ Now save it in arff format by file -> save as

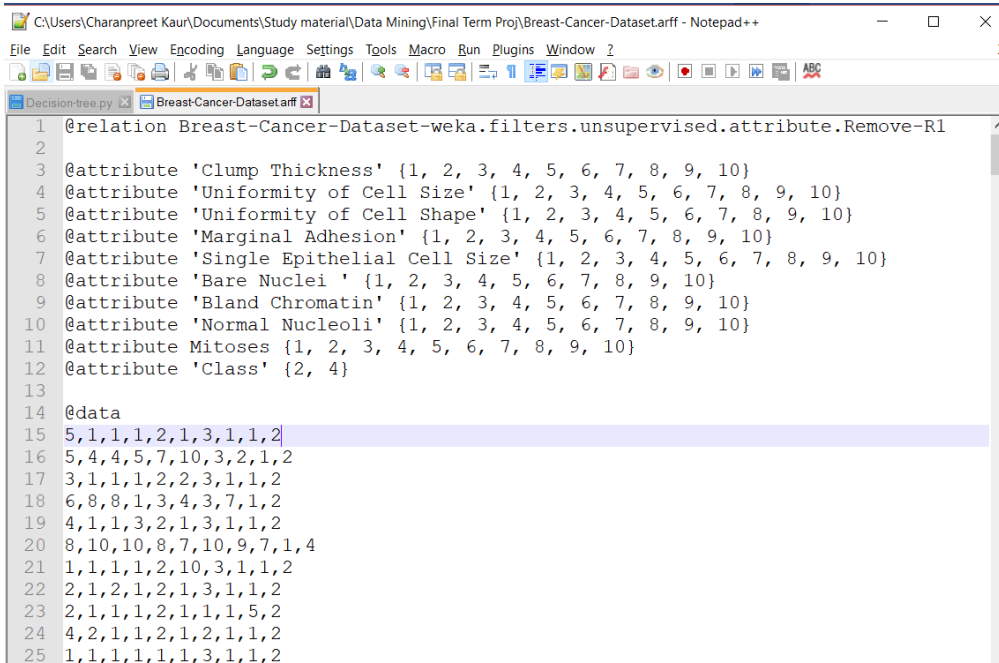
✓ Open the saved file in notepad++

```
@relation Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1

@attribute 'Clump Thickness' numeric
@attribute 'Uniformity of Cell Size' numeric
@attribute 'Uniformity of Cell Shape' numeric
@attribute 'Marginal Adhesion' numeric
@attribute 'Single Epithelial Cell Size' numeric
@attribute 'Bare Nuclei ' numeric
@attribute 'Bland Chromatin' numeric
@attribute 'Normal Nucleoli' numeric
@attribute Mitoses numeric
@attribute Class numeric

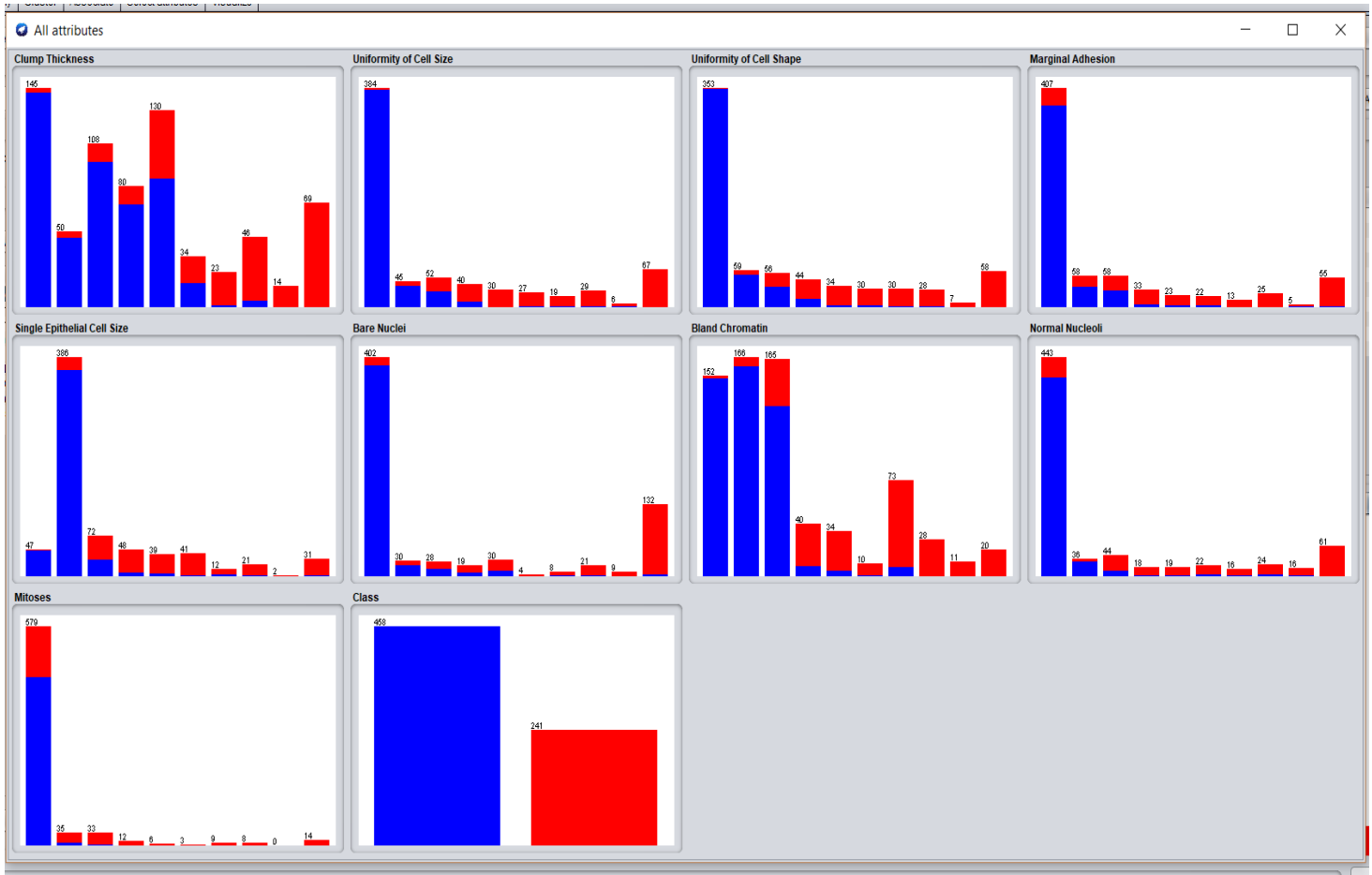
@data
5,1,1,1,2,1,3,1,1,2
5,4,4,5,7,10,3,2,1,2
3,1,1,1,2,2,3,1,1,2
6,8,8,1,3,4,3,7,1,2
4,1,1,3,2,1,3,1,1,2
8,10,10,8,7,10,9,7,1,4
1,1,1,1,2,10,3,1,1,2
2,1,2,1,2,1,3,1,1,2
-,-,-,-,-,-,-,-,-,-
```

- ✓ Now we need to Discretize the dataset: For this we need to go to the arff file and open it as text. The numeric type must be replaced by the group of discrete numbers that the attribute can have. In our dataset case, all the attributes can have a value between 1-10, so the numeric word will be replaced by {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}. The Class attribute can only have 2 and 4 as values so its discrete numbers will be {2, 4}.



```
1 @relation Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1
2
3 @attribute 'Clump Thickness' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
4 @attribute 'Uniformity of Cell Size' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
5 @attribute 'Uniformity of Cell Shape' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
6 @attribute 'Marginal Adhesion' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
7 @attribute 'Single Epithelial Cell Size' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
8 @attribute 'Bare Nuclei ' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
9 @attribute 'Bland Chromatin' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
10 @attribute 'Normal Nucleoli' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
11 @attribute Mitoses {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
12 @attribute 'Class' {2, 4}
13
14 @data
15 5,1,1,1,2,1,3,1,1,2
16 5,4,4,5,7,10,3,2,1,2
17 3,1,1,1,2,2,3,1,1,2
18 6,8,8,1,3,4,3,7,1,2
19 4,1,1,3,2,1,3,1,1,2
20 8,10,10,8,7,10,9,7,1,4
21 1,1,1,1,2,10,3,1,1,2
22 2,1,2,1,2,1,3,1,1,2
23 2,1,1,1,2,1,1,5,2
24 4,2,1,1,2,1,2,1,1,2
25 1,1,1,1,1,1,3,1,1,2
```

- ✓ After this step we reload our arff file, and now the data will be displayed with different colors depending on the class which the case belongs. Blue for benign cases and Red for malign cases.
- ✓ The data for all attributes is shown below

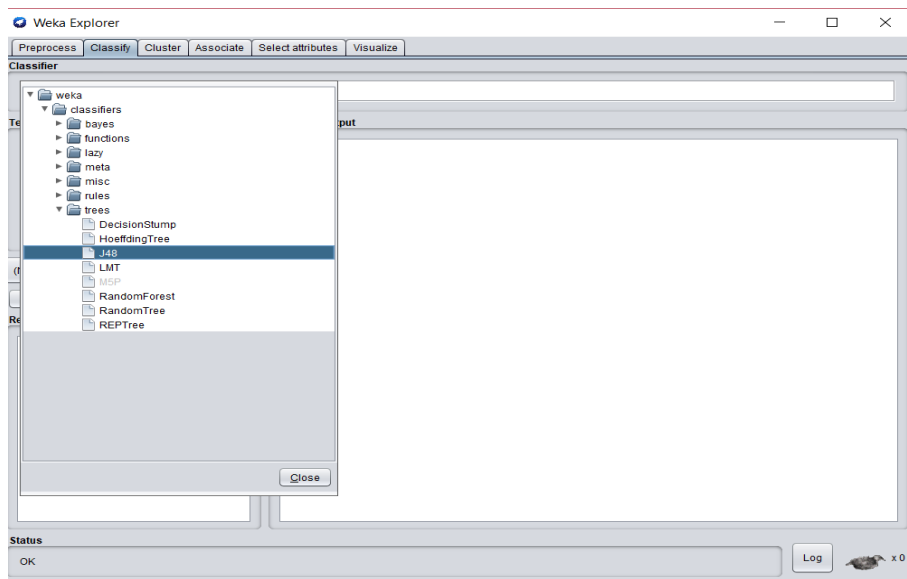
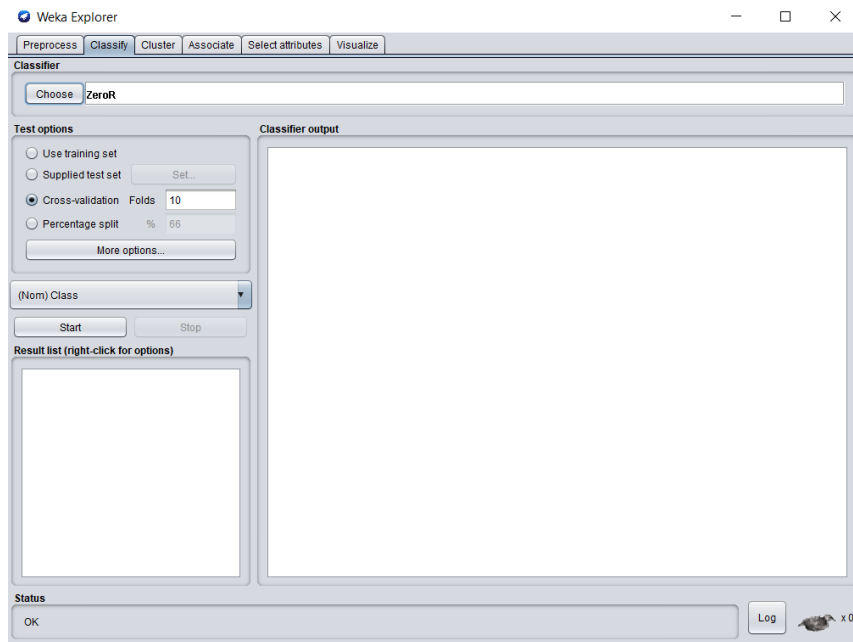


ALGORITHM WORKING:

- ✓ Choose the Algorithm: Select the Classify tab. Then click on the Choose button below the text "Classifier" and next to the text field. Then, in the list of algorithm displayed we proceed to choose the option classifiers-> trees -> J48.

WHY J48?

In WEKA the ID3 algorithm only operates on nominal attributes; and because the attributes of our dataset are numeric the J48 algorithm will be used instead. It must be clear that the ID3 algorithm is the precursor of the J48 algorithm so won't be any alteration in the results because of this change.



- ✓ According to the requirements we need to do the evaluation using the 10-Fold Cross Validation, we need to set this method in the Test options area by choosing the Cross-validation option. Also, we need to set the number of folds as ten. To run the algorithm, we click on the **Start** button and WEKA will proceed to make the evaluations in the dataset and present the results in the Classifier output area.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -C 0.25 -M 2

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)

01:30:45 - trees.J48

Classifier output

Time taken to build model: 0 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	660	94.4206 %
Incorrectly Classified Instances	39	5.5794 %
Kappa statistic	0.8769	
Mean absolute error	0.0796	
Root mean squared error	0.218	
Relative absolute error	17.6026 %	
Root relative squared error	45.8562 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.954	0.075	0.960	0.954	0.957	0.877	0.955	0.959	2
	0.925	0.046	0.914	0.925	0.920	0.877	0.955	0.905	4
Weighted Avg.	0.944	0.065	0.944	0.944	0.944	0.877	0.955	0.940	

=== Confusion Matrix ===

	a	b	<-- classified as
437	21	1	a = 2
18	223	1	b = 4

Status

OK Log x0

- ✓ To view the decision tree of the dataset make a right click on the result description displayed in the Result list area. Then select the option Visualize tree, a new window will appear with the decision tree display on it

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -C 0.25 -M 2

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)

01:30:45 - trees.J48

Classifier output

Time taken to build model: 0 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	660	94.4206 %
Incorrectly Classified Instances	39	5.5794 %
Kappa statistic	0.8769	
Mean absolute error	0.0796	
Root mean squared error	0.218	
Relative absolute error	17.6026 %	
Root relative squared error	45.8562 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.954	0.075	0.960	0.954	0.957	0.877	0.955	0.959	2
	0.925	0.046	0.914	0.925	0.920	0.877	0.955	0.905	4
Weighted Avg.	0.944	0.065	0.944	0.944	0.944	0.877	0.955	0.940	

=== Confusion Matrix ===

	a	b	<-- classified as
437	21	1	a = 2
18	223	1	b = 4

Status

OK Log x0

View in main window

View in separate window

Save result buffer

Delete result buffer(s)

Load model

Save model

Re-evaluate model on current test set

Re-apply this model's configuration

Visualize classifier errors

Visualize tree

Visualize margin curve

Visualize threshold curve

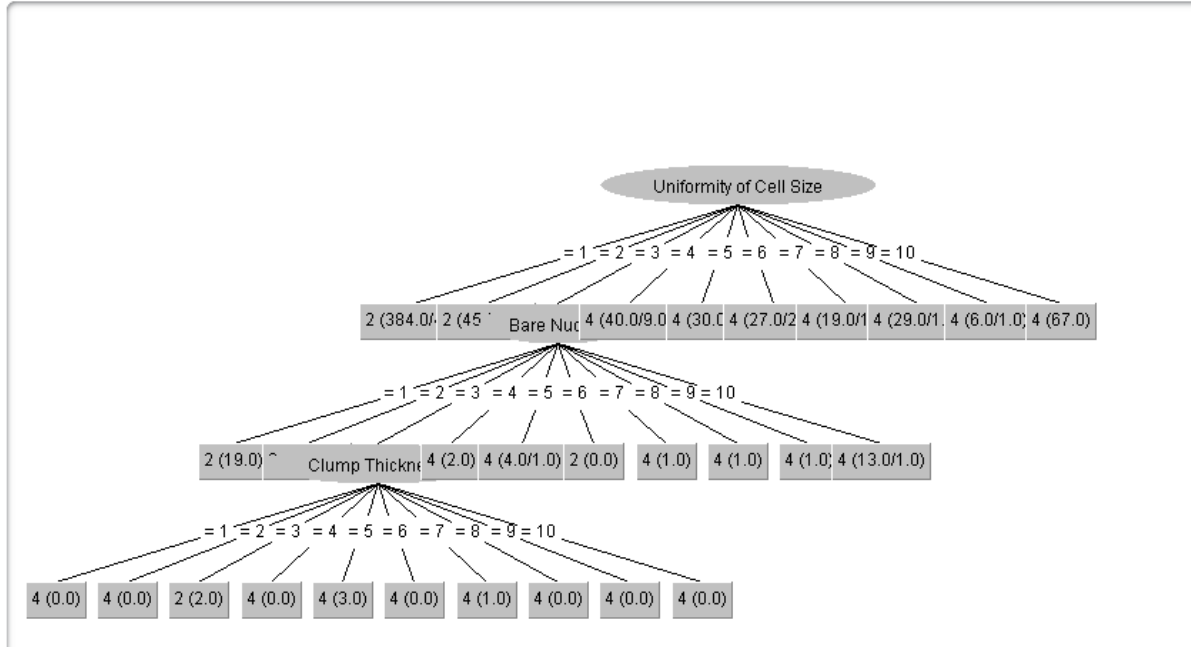
Cost/Benefit analysis

Visualize cost curve

DECISION TREE FOR THE DATASET:

Weka Classifier Tree Visualizer: 18:14:28 - trees.J48 (Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1)

Tree View



Number of Leaves: 28

Size of the Tree: 31

CLASSIFIER OUTPUT:

=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1

Instances: 699

Attributes: 10

Clump Thickness

Uniformity of Cell Size

Uniformity of Cell Shape

Marginal Adhesion

Single Epithelial Cell Size

Bare Nuclei

Bland Chromatin

Normal Nucleoli

Mitoses

Class

Test mode: 10-fold cross-validation

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

J48 pruned tree

Uniformity of Cell Size = 1: 2 (384.0/4.0)

Uniformity of Cell Size = 2: 2 (45.0/8.0)

Uniformity of Cell Size = 3

| Bare Nuclei = 1: 2 (19.0)

| Bare Nuclei = 2: 2 (5.0/1.0)

| Bare Nuclei = 3

| | Clump Thickness = 1: 4 (0.0)

| | Clump Thickness = 2: 4 (0.0)

| | Clump Thickness = 3: 2 (2.0)

| | Clump Thickness = 4: 4 (0.0)

| | Clump Thickness = 5: 4 (3.0)

| | Clump Thickness = 6: 4 (0.0)

| | Clump Thickness = 7: 4 (1.0)

| | Clump Thickness = 8: 4 (0.0)

| | Clump Thickness = 9: 4 (0.0)

| | Clump Thickness = 10: 4 (0.0)

| Bare Nuclei = 4: 4 (2.0)

| Bare Nuclei = 5: 4 (4.0/1.0)

| Bare Nuclei = 6: 2 (0.0)

| Bare Nuclei = 7: 4 (1.0)

| Bare Nuclei = 8: 4 (1.0)

| Bare Nuclei = 9: 4 (1.0)

| Bare Nuclei = 10: 4 (13.0/1.0)

Uniformity of Cell Size = 4: 4 (40.0/9.0)

Uniformity of Cell Size = 5: 4 (30.0)

Uniformity of Cell Size = 6: 4 (27.0/2.0)

Uniformity of Cell Size = 7: 4 (19.0/1.0)

Uniformity of Cell Size = 8: 4 (29.0/1.0)

Uniformity of Cell Size = 9: 4 (6.0/1.0)

Uniformity of Cell Size = 10: 4 (67.0)

Number of Leaves : 28

Size of the tree : 31

Time taken to build model: 0 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	660	94.4206 %
Incorrectly Classified Instances	39	5.5794 %
Kappa statistic	0.8769	
Mean absolute error	0.0796	
Root mean squared error	0.218	
Relative absolute error	17.6026 %	
Root relative squared error	45.8562 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.954	0.075	0.960	0.954	0.957	0.877	0.955	0.959	2
	0.925	0.046	0.914	0.925	0.920	0.877	0.955	0.905	4
Weighted Avg.	0.944	0.065	0.944	0.944	0.944	0.944	0.877	0.955	0.940

=== Confusion Matrix ===

```
a    b  <-- classified as
437  21 |  a = 2
 18 223 |  b = 4
```

RESULTS:

True Benign: 437

False Benign: 21

True Malign:223

False Malign:18

Class	TP Rate	FP Rate	Precision	Recall	F- Measure	MCC	ROC Area	PRC Area
2	0.954	0.075	0.960	0.954	0.957	0.877	0.955	0.959
4	0.925	0.046	0.914	0.925	0.920	0.877	0.955	0.905
Weighted Avg.	0.944	0.065	0.944	0.944	0.944	0.877	0.955	0.940

SOURCE CODE FOR DECISION TREE

```
1
2 # Decision Tree Classifier
3
4 # Importing the libraries
5
6 import numpy as np
7 import matplotlib.pyplot as plt
8 import pandas as pd
9
10 # Importing the datasets
11
12 datasets = pd.read_csv('Social_Network_Ads.csv')
13 X = datasets.iloc[:, [2,3]].values
14 Y = datasets.iloc[:, 4].values
15
16 # Splitting the dataset into the Training set and Test set
17
18 from sklearn.model_selection import train_test_split
19 X_Train, X_Test, Y_Train, Y_Test = train_test_split(X, Y, test_size = 0.25, random_state = 0)
20
21 # Feature Scaling
22
23 from sklearn.preprocessing import StandardScaler
24 sc_X = StandardScaler()
25 X_Train = sc_X.fit_transform(X_Train)
26 X_Test = sc_X.transform(X_Test)
27
28 # Fitting the classifier into the Training set
29
30 from sklearn.tree import DecisionTreeClassifier
31 classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
32
33 # Predicting the test set results
34
35 Y_Pred = classifier.predict(X_Test)
36
37 # Making the Confusion Matrix
38
39 from sklearn.metrics import confusion_matrix
40 cm = confusion_matrix(Y_Test, Y_Pred)
41
42 # Visualising the Training set results
43
44 from matplotlib.colors import ListedColormap
45 X_Set, Y_Set = X_Train, Y_Train
46 X1, X2 = np.meshgrid(np.arange(start = X_Set[:, 0].min() - 1, stop = X_Set[:, 0].max() + 1, step = 0.01),
47                      np.arange(start = X_Set[:, 1].min() - 1, stop = X_Set[:, 1].max() + 1, step = 0.01))
48 plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
49             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
50 plt.xlim(X1.min(), X1.max())
51 plt.ylim(X2.min(), X2.max())
52 for i, j in enumerate(np.unique(Y_Set)):
53     plt.scatter(X_Set[Y_Set == j, 0], X_Set[Y_Set == j, 1],
54               c = ListedColormap(('red', 'green'))(i), label = j)
55 plt.title('Decision Tree Classifier (Training set)')
56 plt.xlabel('Age')
57 plt.ylabel('Estimated Salary')
58 plt.legend()
59 plt.show()
60
61 # Visualising the Test set results
62
63 from matplotlib.colors import ListedColormap
64 X_Set, Y_Set = X_Test, Y_Test
65 X1, X2 = np.meshgrid(np.arange(start = X_Set[:, 0].min() - 1, stop = X_Set[:, 0].max() + 1, step = 0.01),
66                      np.arange(start = X_Set[:, 1].min() - 1, stop = X_Set[:, 1].max() + 1, step = 0.01))
67 plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
68             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
```



```
69 plt.xlim(X1.min(), X1.max())
70 plt.ylim(X2.min(), X2.max())
71 for i, j in enumerate(np.unique(Y_Set)):
72     plt.scatter(X_Set[Y_Set == j, 0], X_Set[Y_Set == j, 1],
73               c = ListedColormap(('red', 'green'))(i), label = j)
74 plt.title('Decision Tree Classifier (Test set)')
75 plt.xlabel('Age')
76 plt.ylabel('Estimated Salary')
77 plt.legend()
78 plt.show()
```

NAÏVE BAYES (NAÏVE BAYES)

INTRODUCTION

The Naive Bayes classification algorithm is a probabilistic classifier. It is based on probability models that incorporate strong independence assumptions.

The independence assumptions often do not have an impact on reality. Therefore they are considered as naive.

BAYES THEOREM

Conditional probability

$$P(A,B) = P(A|B)P(B) = P(B|A)P(A)$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- Applying Bayes rule

$$\begin{aligned} P(Y_i|X_1 \dots X_n) &= \frac{P(Y_i) P(X_1 \dots X_n|Y_i)}{P(X_1 \dots X_n)} \\ &= \frac{P(Y_i) P(X_1|Y_i)P(X_2|Y_i) \dots P(X_n|Y_i)}{P(X_1 \dots X_n)} \end{aligned}$$

$$Y \leftarrow \arg \max P(Y_i) P(X_1|Y_i)P(X_2|Y_i) \dots P(X_n|Y_i)$$

Assumption: X_i 's are conditionally independent given Y

ALGORITHM

Algorithmic Details (Training Phase)

- Naïve Bayes Algorithm

- Learning/training phase: given a training data set T ,
For each label Y_i
 $P(Y_i) \leftarrow$ calculate $P(Y=Y_i)$ using training examples in T ;
For each attribute value X_j
calculate $P(X_j|Y_i)$ using training examples in T .

Algorithmic Details (Testing Phase)

- Naïve Bayes Algorithm

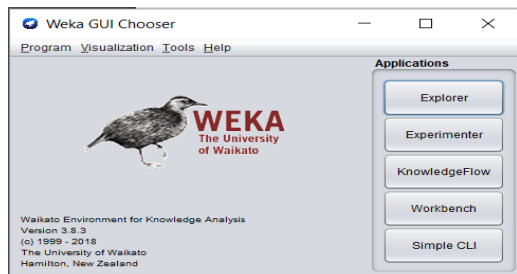
- Testing phase: given an unlabeled test record
 $X^* = \langle X_1 \dots X_n \rangle$
- Assign label Y to X^* based on
 $\max\{P(Y_i)P(X_1|Y_i)P(X_2|Y_i)\dots P(X_n|Y_i)\}$

SOFTWARE USED: WEKA

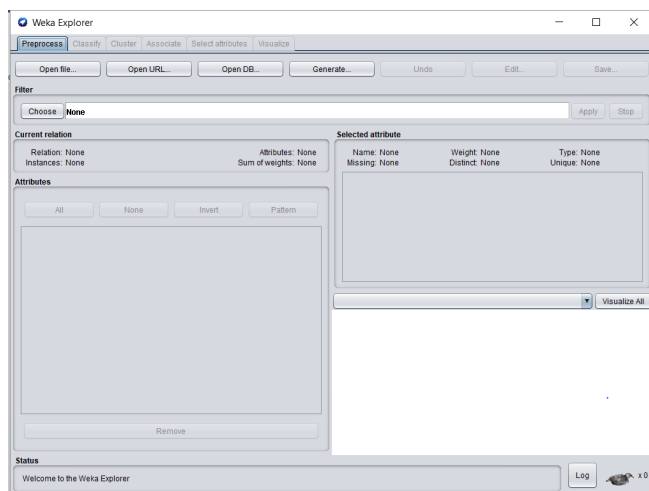
Class for a Naive Bayes classifier using estimator classes. Numeric estimator precision values are chosen based on analysis of the training data. For this reason, the classifier is not an Updateable Classifier (which in typical usage are initialized with zero training instances) -- if you need the Updateable Classifier functionality, use the Naïve Bayes Updateable classifier. The Naïve Bayes Updateable classifier will use a default precision of 0.1 for numeric attributes when build Classifier is called with zero training instances.

STEPS :

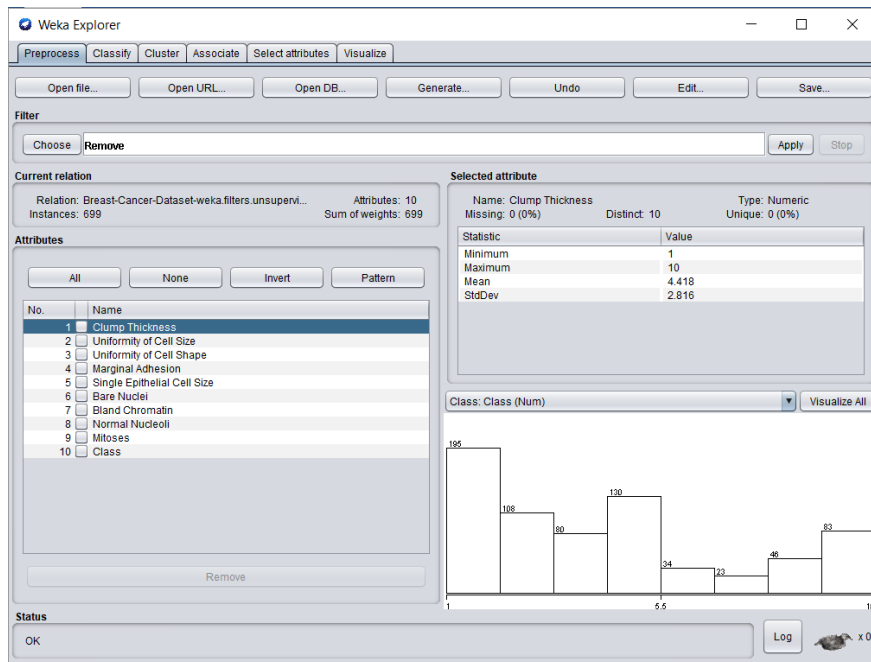
- ✓ We open the WEKA tool and click in the “Explorer” button



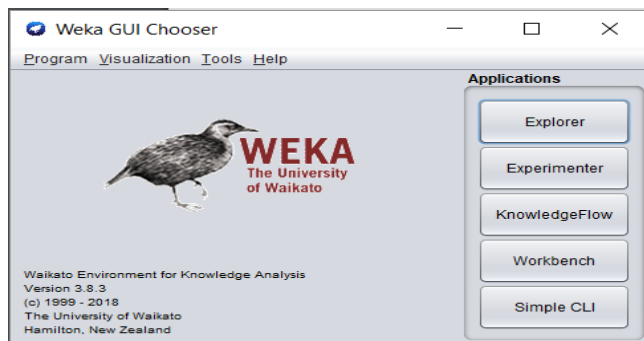
- ✓ Click on “OPEN FILE” and select the Breast Cancer Data Set csv file stored on the disk.



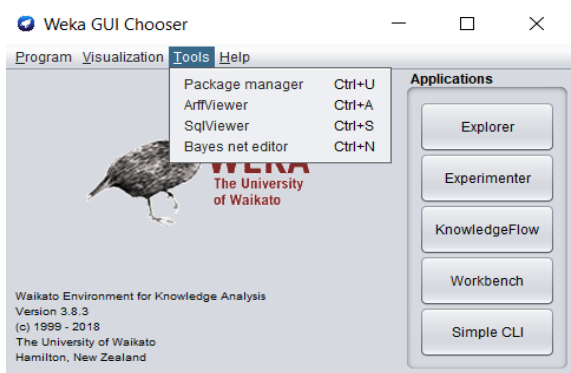
- ✓ We remove the first attribute “Sample Code Number” attribute by clicking in the Choose button located under the word Filter, and then we choose the option filters -> unsupervised -> attribute -> remove



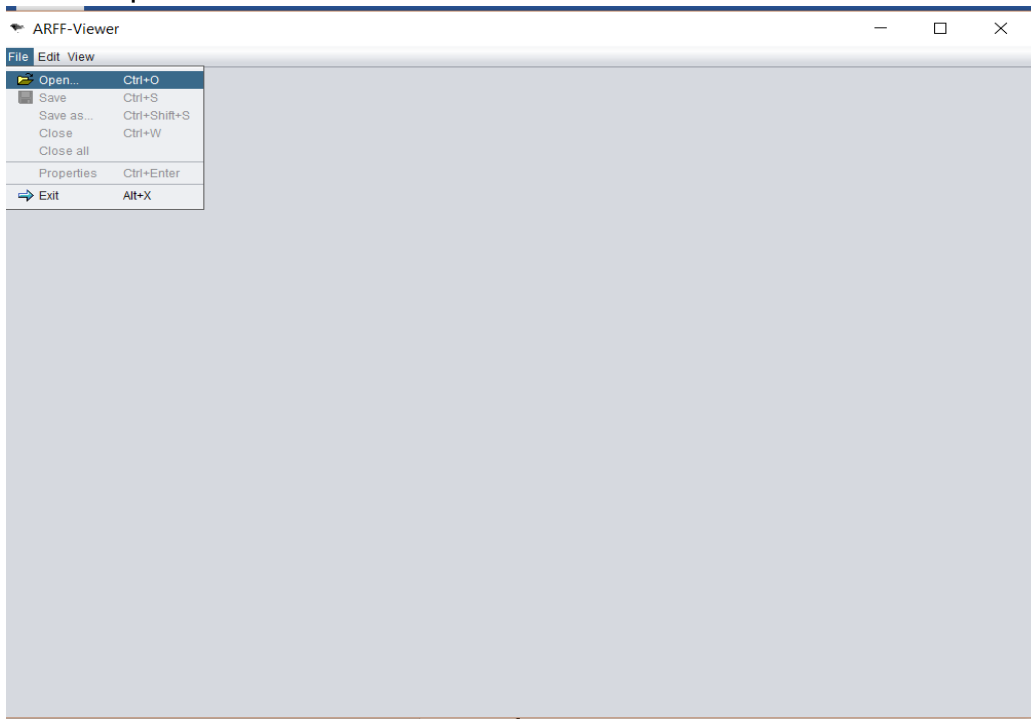
How to create .arff file using weka?



- ✓ Click on tools
- ✓ Click on ArffViewer



✓ File -> Open -> Select the csv data set file



✓ Now save it in arff format by file -> save as

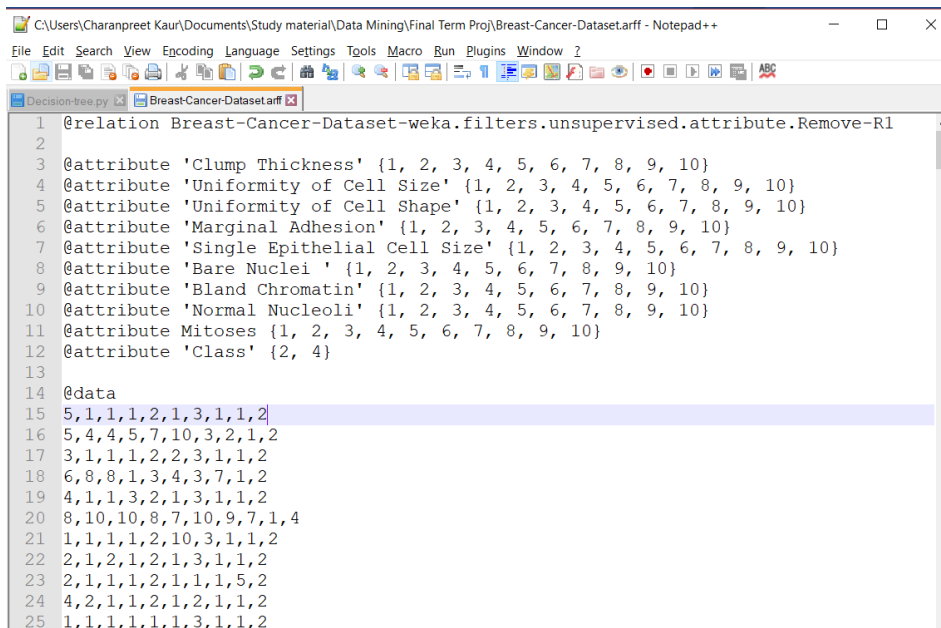
- ✓ Open the saved file in notepad++

```
@relation Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1

@attribute 'Clump Thickness' numeric
@attribute 'Uniformity of Cell Size' numeric
@attribute 'Uniformity of Cell Shape' numeric
@attribute 'Marginal Adhesion' numeric
@attribute 'Single Epithelial Cell Size' numeric
@attribute 'Bare Nuclei ' numeric
@attribute 'Bland Chromatin' numeric
@attribute 'Normal Nucleoli' numeric
@attribute Mitoses numeric
@attribute Class numeric

@data
5,1,1,1,2,1,3,1,1,2
5,4,4,5,7,10,3,2,1,2
3,1,1,1,2,2,3,1,1,2
6,8,8,1,3,4,3,7,1,2
4,1,1,3,2,1,3,1,1,2
8,10,10,8,7,10,9,7,1,4
1,1,1,1,2,10,3,1,1,2
2,1,2,1,2,1,3,1,1,2
```

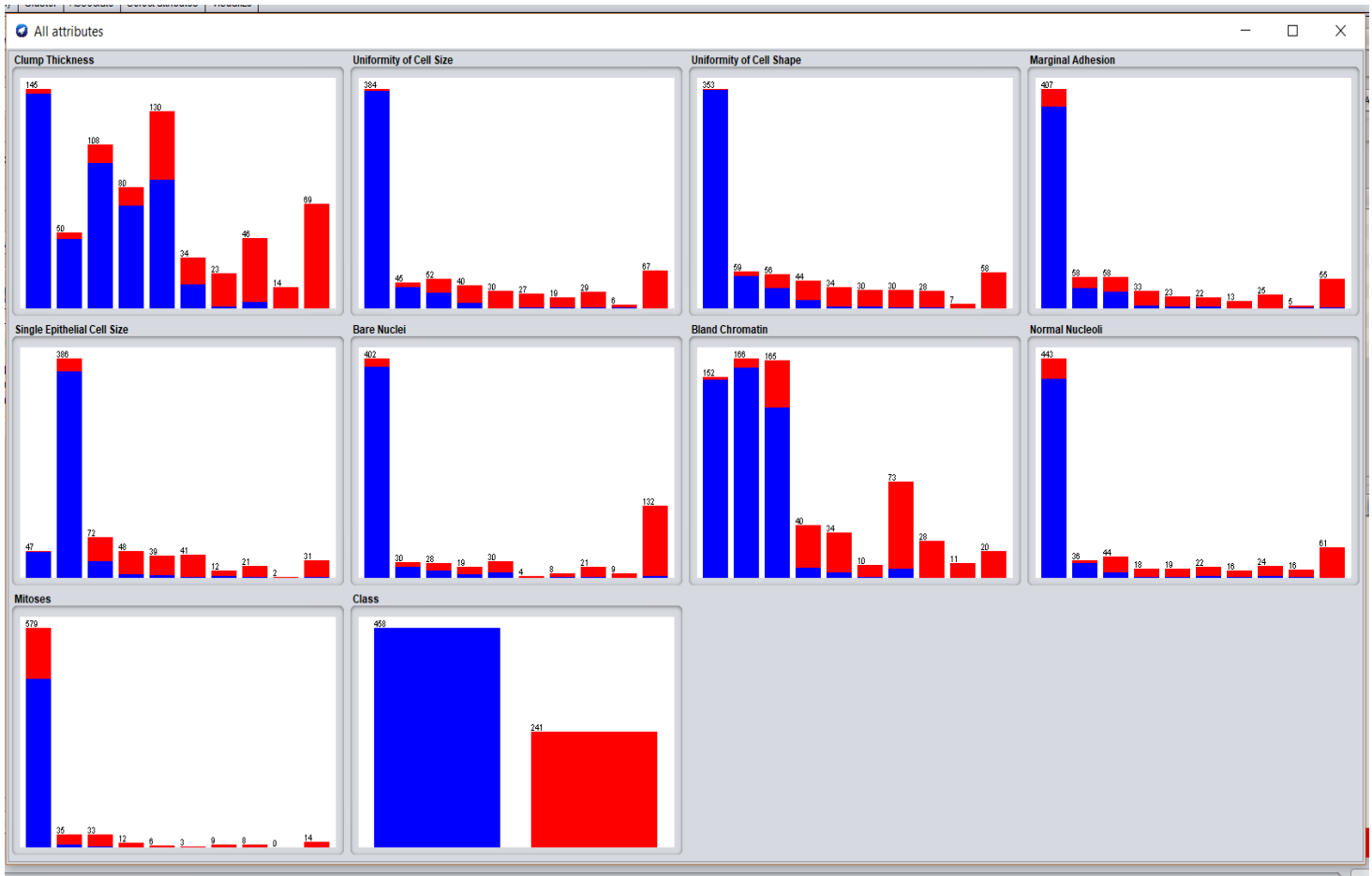
- ✓ Now we need to Discretize the dataset: For this we need to go to the arff file and open it as text. The numeric type must be replaced by the group of discrete numbers that the attribute can have. In our dataset case, all the attributes can have a value between 1-10, so the numeric word will be replaced by {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}. The Class attribute can only have 2 and 4 as values so its discrete numbers will be {2, 4}.



```
1 @relation Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1
2
3 @attribute 'Clump Thickness' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
4 @attribute 'Uniformity of Cell Size' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
5 @attribute 'Uniformity of Cell Shape' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
6 @attribute 'Marginal Adhesion' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
7 @attribute 'Single Epithelial Cell Size' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
8 @attribute 'Bare Nuclei ' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
9 @attribute 'Bland Chromatin' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
10 @attribute 'Normal Nucleoli' {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
11 @attribute Mitoses {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
12 @attribute 'Class' {2, 4}
13
14 @data
15 5,1,1,1,2,1,3,1,1,2
16 5,4,4,5,7,10,3,2,1,2
17 3,1,1,1,2,2,3,1,1,2
18 6,8,8,1,3,4,3,7,1,2
19 4,1,1,3,2,1,3,1,1,2
20 8,10,10,8,7,10,9,7,1,4
21 1,1,1,1,2,10,3,1,1,2
22 2,1,2,1,2,1,3,1,1,2
23 2,1,1,1,2,1,1,1,5,2
24 4,2,1,1,2,1,2,1,1,2
25 1,1,1,1,1,1,3,1,1,2
```

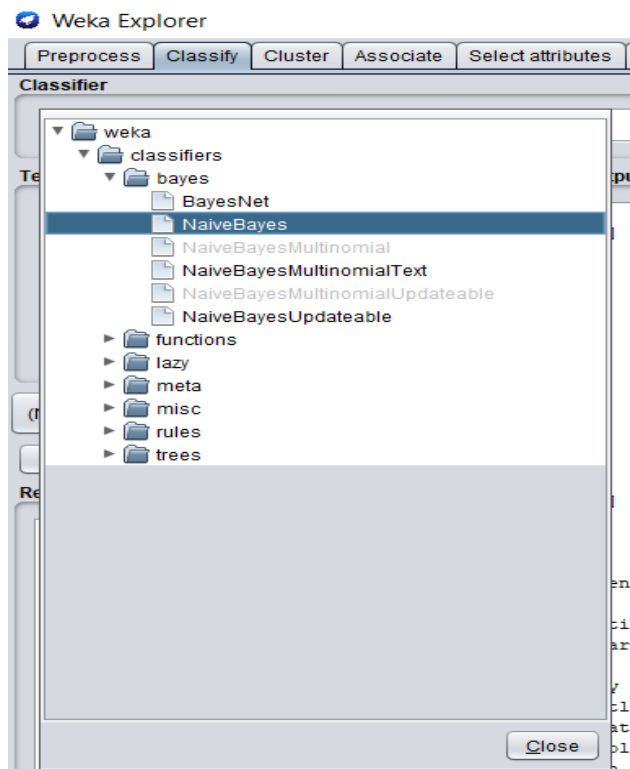
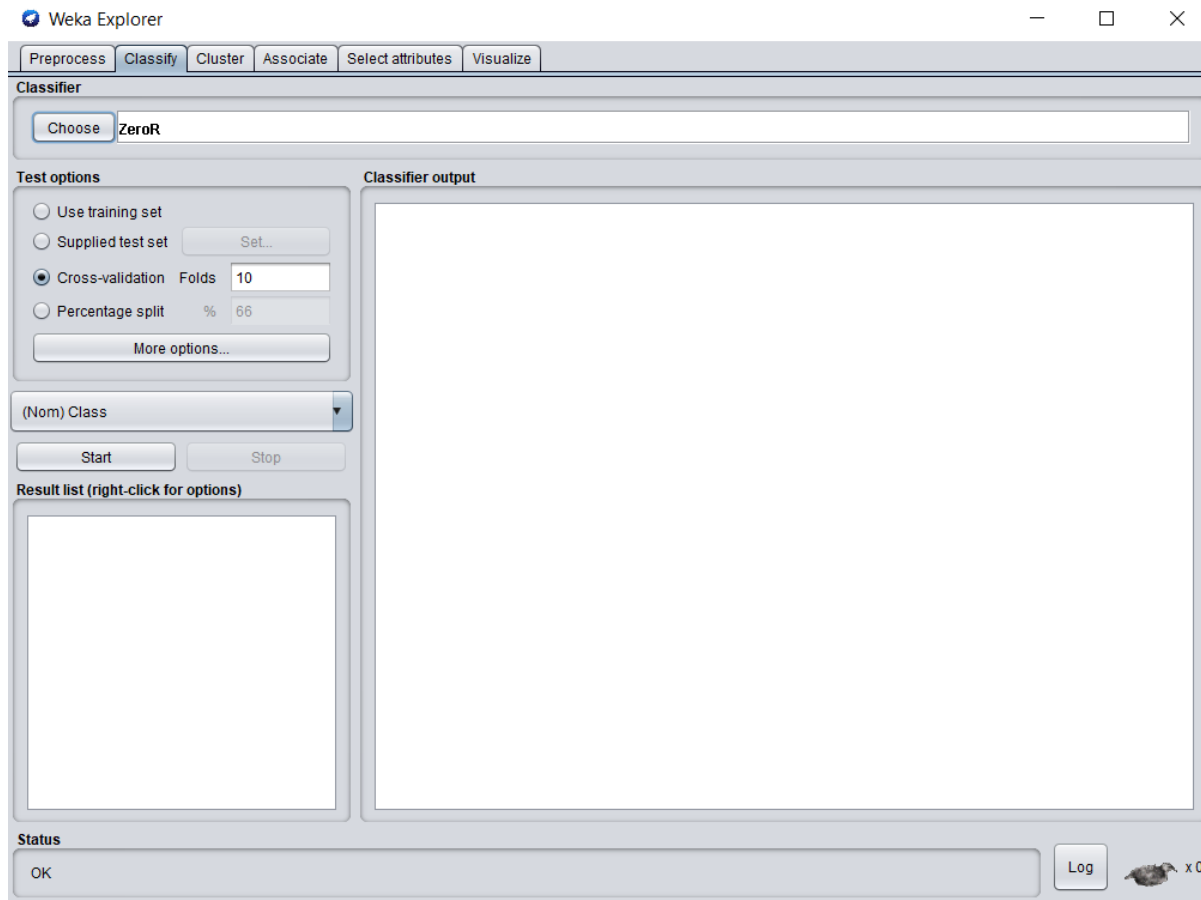
- ✓ After this step we reload our arff file, and now the data will be displayed with different colors depending on the class which the case belongs. Blue for benign cases and Red for malign cases.

- ✓ The data for all attributes is shown below

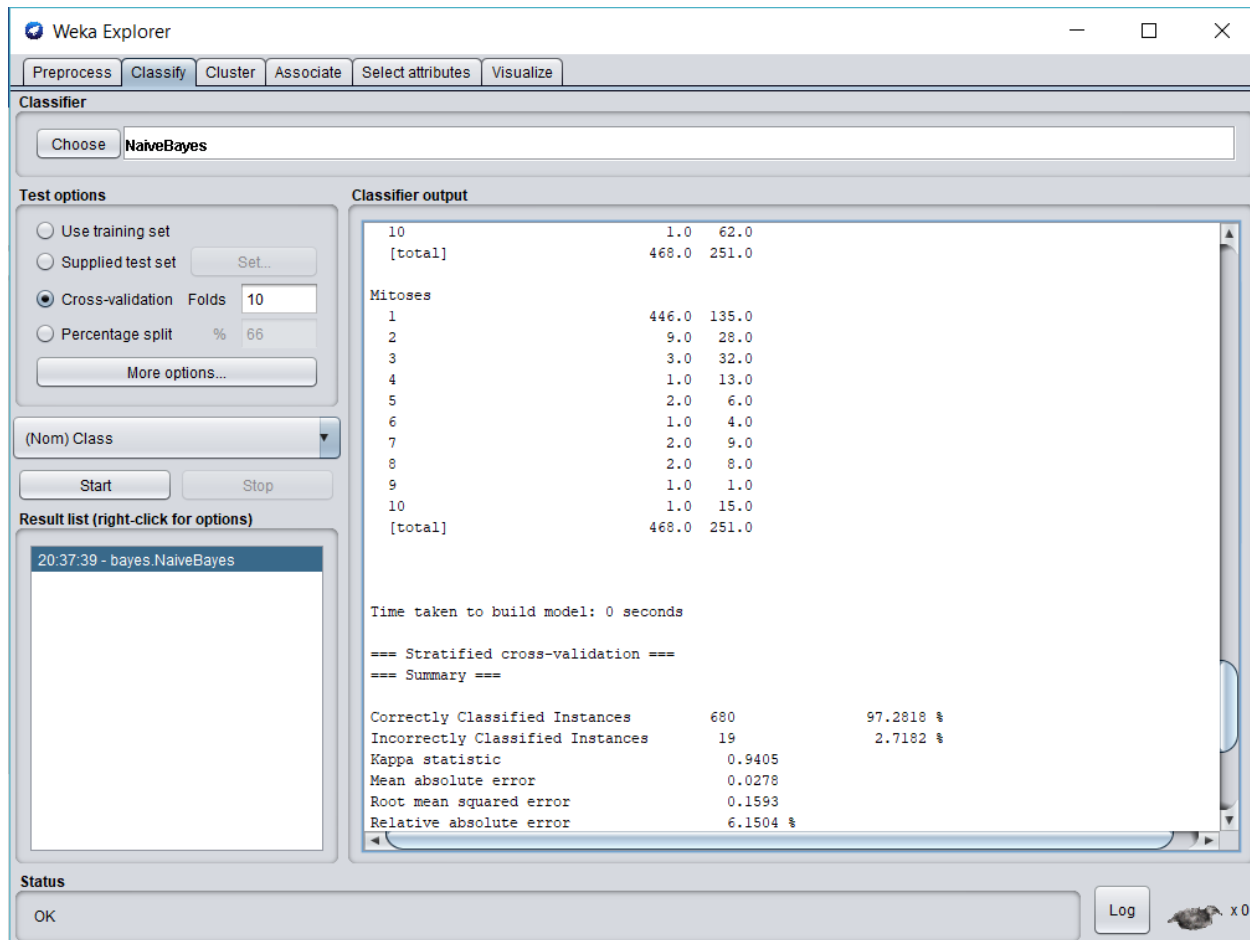


ALGORITHM WORKING:

- ✓ Choose the Algorithm: Select the Classify tab. Then click on the Choose button below the text “Classifier” and next to the text field. Then, in the list of algorithm displayed we proceed to choose the option classifiers-> Bayes -> Naïve Bayes.



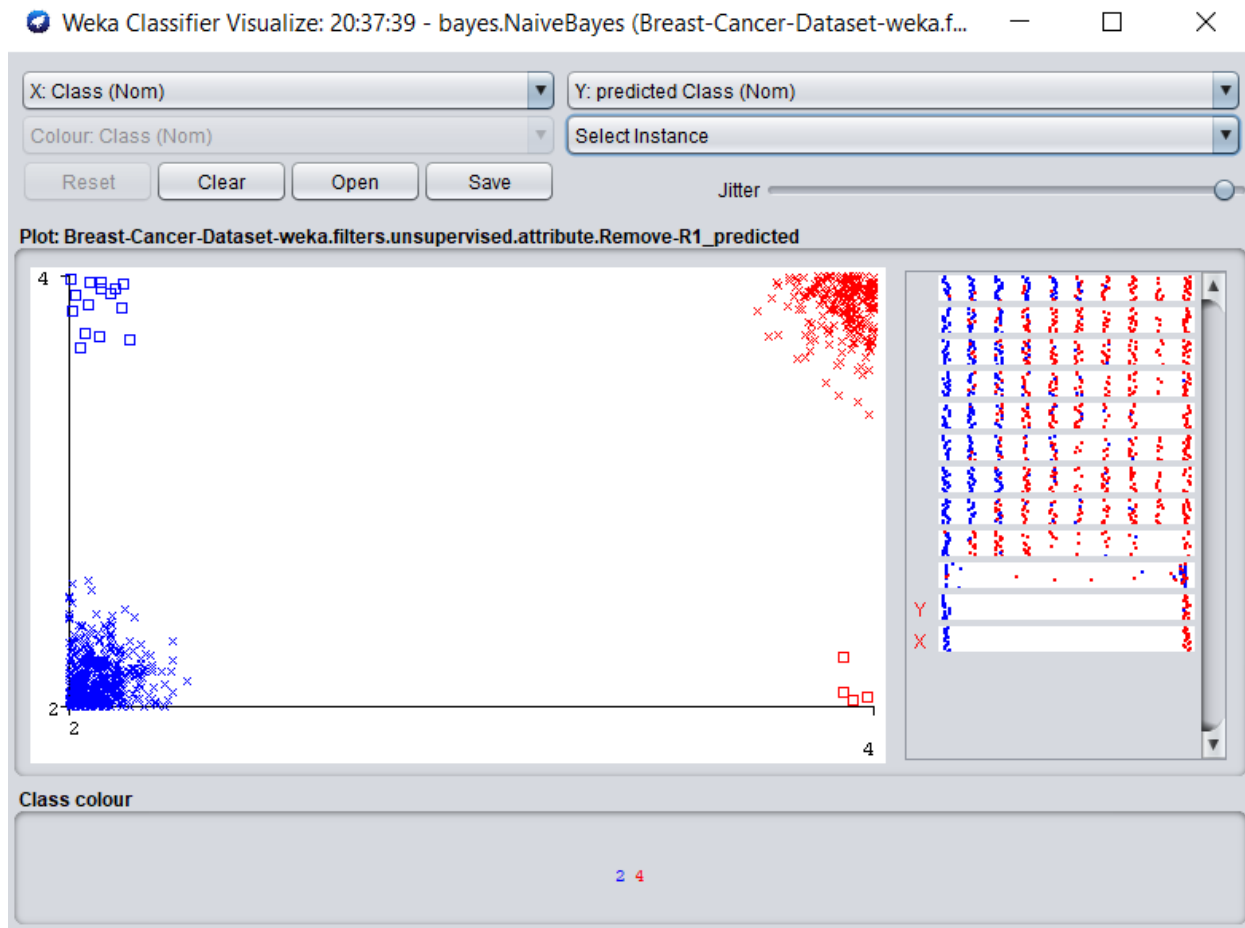
- ✓ According to the requirements we need to do the evaluation using the 10-Fold Cross Validation, we need to set this method in the Test options area by choosing the Cross-validation option. Also, we need to set the number of folds as ten. To run the algorithm, we click on the **Start** button and WEKA will proceed to make the evaluations in the dataset and present the results in the Classifier output area.



Correctly Classified Instances 680 97.2818 %

Incorrectly Classified Instances 19 2.7182 %

VISUALIZE CLASSIFIER ERROR



CLASSIFIER OUTPUT:

=== Run information ===

Scheme: weka.classifiers.bayes.NaiveBayes

Relation: Breast-Cancer-Dataset-weka.filters.unsupervised.attribute.Remove-R1

Instances: 699

Attributes: 10

- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape
- Marginal Adhesion
- Single Epithelial Cell Size
- Bare Nuclei
- Bland Chromatin
- Normal Nucleoli
- Mitoses
- Class

Test mode: 10-fold cross-validation

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

Naive Bayes Classifier

Attribute	Class	
	2	4
	(0.65)	(0.35)

=====

Clump Thickness

1	143.0	4.0
2	47.0	5.0
3	97.0	13.0
4	69.0	13.0
5	86.0	46.0
6	17.0	19.0
7	2.0	23.0
8	5.0	43.0
9	1.0	15.0
10	1.0	70.0
[total]	468.0	251.0

Uniformity of Cell Size

1	381.0	5.0
2	38.0	9.0
3	28.0	26.0
4	10.0	32.0
5	1.0	31.0
6	3.0	26.0
7	2.0	19.0
8	2.0	29.0
9	2.0	6.0
10	1.0	68.0
[total]	468.0	251.0

Uniformity of Cell Shape

1	352.0	3.0
2	53.0	8.0
3	34.0	24.0
4	14.0	32.0
5	4.0	32.0
6	4.0	28.0
7	3.0	29.0

8	2.0	28.0
9	1.0	8.0
10	1.0	59.0
[total]	468.0	251.0

Marginal Adhesion

1	376.0	33.0
2	38.0	22.0
3	32.0	28.0
4	6.0	29.0
5	5.0	20.0
6	5.0	19.0
7	1.0	14.0
8	1.0	26.0
9	2.0	5.0
10	2.0	55.0
[total]	468.0	251.0

Single Epithelial Cell Size

1	47.0	2.0
2	364.0	24.0
3	30.0	44.0
4	8.0	42.0
5	6.0	35.0
6	3.0	40.0
7	4.0	10.0
8	3.0	20.0
9	1.0	3.0
10	2.0	31.0
[total]	468.0	251.0

Bare Nuclei

1	388.0	16.0
2	22.0	10.0
3	15.0	15.0
4	7.0	14.0
5	11.0	21.0
6	1.0	5.0
7	2.0	8.0
8	3.0	20.0
9	1.0	10.0
10	4.0	130.0
[total]	454.0	249.0

Bland Chromatin

1	151.0	3.0
2	160.0	8.0
3	130.0	37.0
4	9.0	33.0
5	5.0	31.0
6	2.0	10.0
7	8.0	67.0
8	1.0	29.0
9	1.0	12.0
10	1.0	21.0
[total]	468.0	251.0

Normal Nucleoli

1	403.0	42.0
2	31.0	7.0
3	13.0	33.0
4	2.0	18.0
5	3.0	18.0
6	5.0	19.0
7	3.0	15.0
8	5.0	21.0
9	2.0	16.0
10	1.0	62.0
[total]	468.0	251.0

Mitoses

1	446.0	135.0
2	9.0	28.0
3	3.0	32.0
4	1.0	13.0
5	2.0	6.0
6	1.0	4.0
7	2.0	9.0
8	2.0	8.0
9	1.0	1.0
10	1.0	15.0
[total]	468.0	251.0

Time taken to build model: 0 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	680	97.2818 %
Incorrectly Classified Instances	19	2.7182 %
Kappa statistic	0.9405	
Mean absolute error	0.0278	
Root mean squared error	0.1593	
Relative absolute error	6.1504 %	
Root relative squared error	33.5215 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.967	0.017	0.991	0.967	0.979	0.941	0.993	0.996	2
0.983	0.033	0.940	0.983	0.961	0.941	0.993	0.985	4
Weighted Avg.	0.973	0.022	0.974	0.973	0.973	0.941	0.993	0.992

=== Confusion Matrix ===

```
a  b  <-- classified as
443 15 | a = 2
 4 237 | b = 4
```

RESULTS:

True Benign: 443

False Benign: 15

True Malign:237

False Malign:4

Class	TP Rate	FP Rate	Precision	Recall	F- Measure	MCC	ROC Area	PRC Area
2	0.967	0.017	0.991	0.967	0.979	0.941	0.993	0.996
4	0.983	0.033	0.940	0.983	0.961	0.941	0.993	0.985
Weighted Avg.	0.973	0.022	0.974	0.973	0.973	0.941	0.993	0.992

SOURCE CODE FOR NAÏVE BAYES

```
1  import csv
2  import math
3  import random
4
5  #Handle data
6  def loadCsv(filename):
7      lines = csv.reader(open(filename, "r"))
8      dataset = list(lines)
9      for i in range(len(dataset)):
10         dataset[i] = [float(x) for x in dataset[i]]
11     return dataset
12
13     #Split dataset with ratio
14     def splitDataset(dataset, splitRatio):
15         trainSize = int(len(dataset) * splitRatio)
16         trainSet = []
17         copy = list(dataset)
18         while len(trainSet) < trainSize:
19             index = random.randrange(len(copy))
20             trainSet.append(copy.pop(index))
21         return [trainSet, copy]
22
23     #Separate by Class
24     def separateByClass(dataset):
25         separated = {}
26         for i in range(len(dataset)):
27             vector = dataset[i]
28             if (vector[-1] not in separated):
29                 separated[vector[-1]] = []
30             separated[vector[-1]].append(vector)
31         return separated
32
33     #Calculate Mean
34     def mean(numbers):
```



```

35         return sum(numbers)/float(len(numbers))
36
37     def stdev(numbers):
38         avg = mean(numbers)
39         variance = sum([pow(x-avg,2) for x in numbers])/float(len(numbers)-1)
40         return math.sqrt(variance)
41
42     #Summarize Dataset
43     def summarize(dataset):
44         summaries = [(mean(attribute), stdev(attribute)) for attribute in zip(*dataset)]
45         del summaries[-1]
46         return summaries
47
48     #Summarize attributes by class
49     def summarizeByClass(dataset):
50         separated = separateByClass(dataset)
51         summaries = {}
52         for classValue, instances in separated.items():
53             summaries[classValue] = summarize(instances)
54         return summaries
55
56     #Calculate Gaussian Probability Density Function
57     def calculateProbability(x, mean, stdev):
58         exponent = math.exp(-(math.pow(x-mean,2)/(2*math.pow(stdev,2))))
59         return (1/(math.sqrt(2*math.pi)*stdev))*exponent
60
61     #Calculate Class Probabilities
62     def calculateClassProbabilities(summaries, inputVector):
63         probabilities = {}
64         for classValue, classSummaries in summaries.items():
65             probabilities[classValue] = 1
66             for i in range(len(classSummaries)):
67                 mean, stdev = classSummaries[i]
68                 x = inputVector[i]

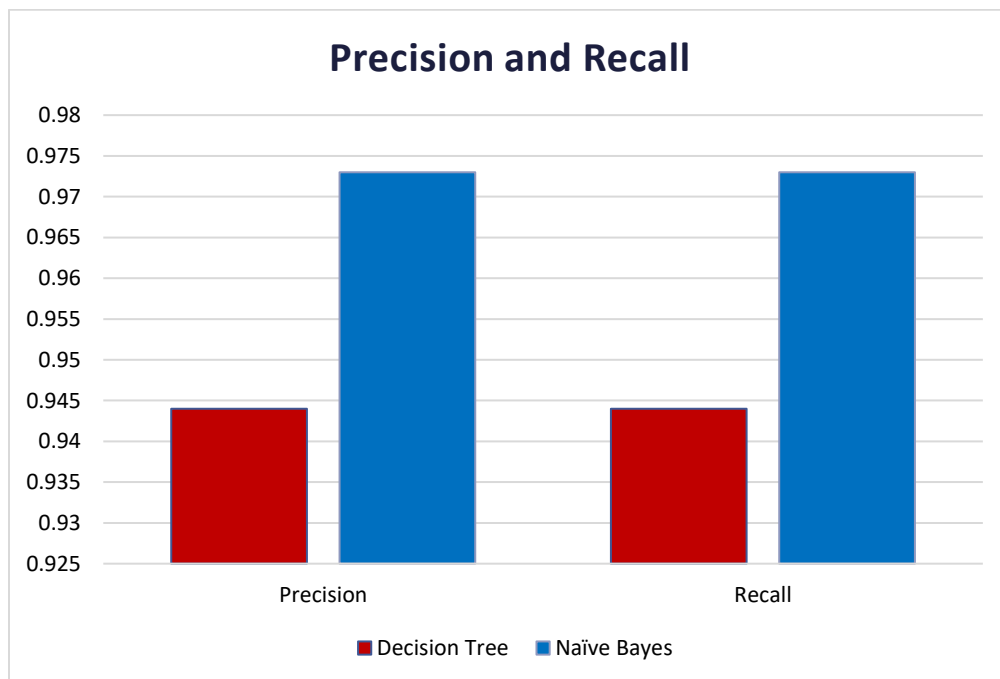
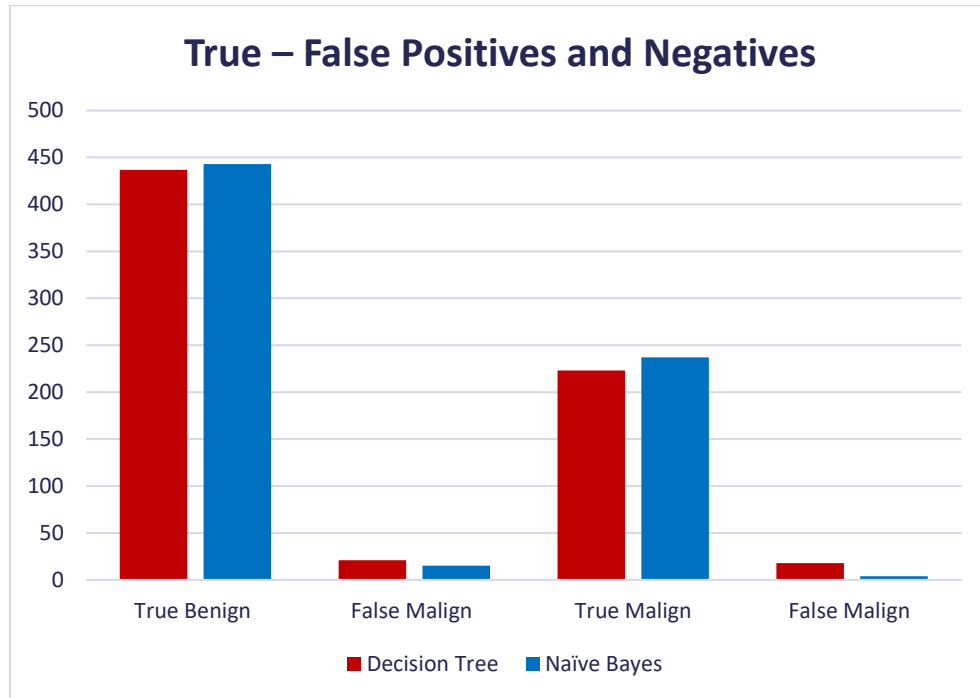
```

```

69         probabilities[classValue] *= calculateProbability(x, mean, stdev)
70     return probabilities
71
72     #Make a prediction
73     def predict(summaries, inputVector):
74         probabilities = calculateClassProbabilities(summaries, inputVector)
75         bestLabel, bestProb = None, -1
76         for classValue, probability in probabilities.items():
77             if bestLabel is None or probability > bestProb:
78                 bestProb = probability
79                 bestLabel = classValue
80         return bestLabel
81
82     #Get predictions
83     def getPredictions(summaries, testSet):
84         predictions = []
85         for i in range(len(testSet)):
86             result = predict(summaries, testSet[i])
87             predictions.append(result)
88         return predictions
89
90     #Get Accuracy
91     def getAccuracy(testSet, predictions):
92         correct = 0
93         for x in range(len(testSet)):
94             if testSet[x][-1] == predictions[x]:
95                 correct += 1
96         return (correct/float(len(testSet)))*100.0
97
98     def main():
99         filename = 'pima-indians-diabetes.data.csv'
100         splitRatio = 0.67
101         dataset = loadCsv(filename)
102         trainingSet, testSet = splitDataset(dataset, splitRatio)
103
104         print('Split {0} rows into train = {1} and test = {2} rows'.format(len(dataset),len(trainingSet),len(testSet)))
105         #prepare model
106         summaries = summarizeByClass(trainingSet)
107         #test model
108         predictions = getPredictions(summaries, testSet)
109         accuracy = getAccuracy(testSet, predictions)
110         print('Accuracy: {0}%'.format(accuracy))
111     main()

```

COMPARISON OF DECISION TREE AND NAÏVE BAYES



With respect to the chart above, we can infer from the results that the Naïve Bayes had better performance as compared to the decision tree algorithm. As we can observe from chart of True -False positive negatives that both of the algorithms gave us the close results. However, the Naïve Bayes Algorithm predicted more cases correctly than decision tree algorithm as well as number of cases predicted wrong were less compared to decision tree. Precision and Recall chart shows that Naïve Bayes classifies the cases more accurately and the results are more relevant. So, we can conclude that Naïve Bayes is more optimal algorithm that can be used for this dataset.