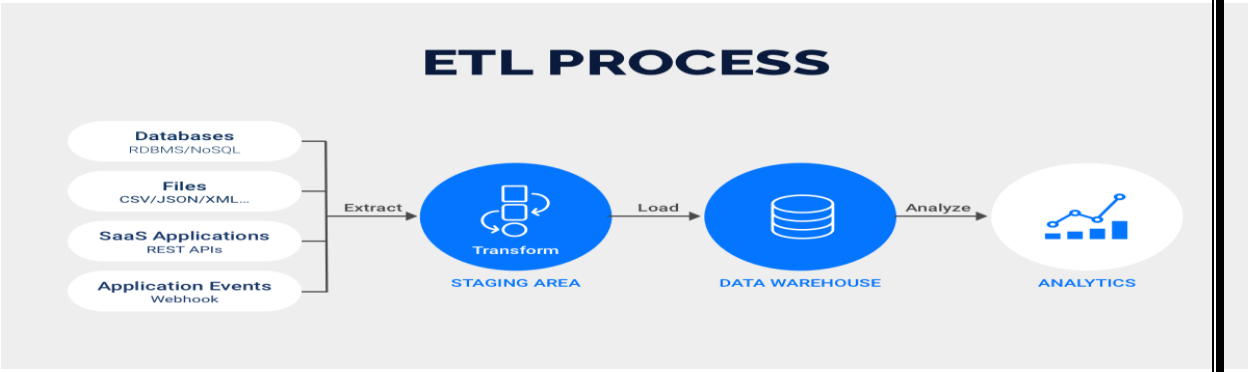
Week 1

Creation of a Data Warehouse.Build Data Warehouse/Data Mart (using open source tools like Pentaho Data Integration Tool, Pentaho Business Analytics; or other data warehouse tools like Microsoft-SSIS, Informatica, Business Objects,etc.,) Design multi-dimensional data models namely Star, Snowflake and Fact Constellation schemas for any one enterprise (ex. Banking, Insurance, Finance, Healthcare, manufacturing, Automobiles, sales etc).Write ETL scripts and implement using data warehouse tools.Perform Various OLAP operations such as slice, dice, roll up, drill up and pivot

ETL(Extract, Transform, and Load) is a process that extracts data from multiple source systems, changes it and then puts it into the Data Warehouse system. It's easy to building a Data warehouse as simple as pulling data from numerous sources and feeding it into a Data warehouse database.

The ETL process, which is technically complex, involves active participation from a variety of stakeholders, including developers, analysts, testers, and senior executives.

To preserve its value as a decision-making tool, the data warehouse system must develop in sync with business developments. ETL is a regular (daily, weekly, monthly) process of a data warehouse system that must be agile, automated, and properly documented.



Steps for ETL process :

Step 1) Extraction

Data is extracted from the source system and placed in the staging area during extraction. If any transformations are required, they are performed in the staging area so that the performance of the source system is not harmed. Rollback will be difficult if damaged data is transferred directly from the source into the Data warehouse database.

Data warehouses can combine systems with different hardware, database management systems, operating systems, and communication protocols. Data warehouses must combine systems with disparate DBMS, hardware, operating systems, and communication protocols. Sources may include legacy programs such as mainframes, customized applications, point-of-contact devices such as ATMs and call switches, text files, spreadsheets, ERP, data from vendors and partners, and so on. Before extracting data and loading it physically, a logical data map is required.

Step 2 : Transformation

The data retrieved from the source server is raw and unusable in its original state, it must be cleaned, mapped, and transformed. It is a key ETL concept in which you apply a collection of functions to extracted data. **Direct move** or **pass through dat**a is the type of data that does not require any transformation.

Step 3 : Loading The final stage in the ETL process is to load data into the target data warehouse database. A large volume of data is loaded in a relatively short period of time in a typical data warehouse. As a result, the load process should be optimized for performance.

In the occurrence of a load failure, recovery procedures should be put in place so that operations can restart from the point of failure without compromising data integrity. Data Warehouse administrators must monitor, continue, and stop loads based on server performance.

Types of Loading:

Initial Load — filling all of the Data Warehouse tables

Incremental Load — implementing ongoing modifications as needed on a regular basis

Full Refresh — clearing the contents of one or more tables and reloading them with fresh data

Load verification

Check that the key field data is not missing or null.

Modelling views based on target tables should be tested.

Examine the combined values3 and computed measures.

Data checks in the dimension and history tables.

Examine the BI reports on the loaded fact and dimension table.

Step 1 Create database Data Warehouse :

*Step 2 Create **Customer dimension** table in Data Warehouse which will hold customer personal details. Fill the **Customer dimension** with sample Values*

*Step 3 Create basic level of **Product Dimension** table without considering any Category or Subcategory Fill the **Product dimension** with sample Values*

*Step 4 Create **Store Dimension** table which will hold details related stores available across various places. Fill the **Store Dimension** with sample Values*

*Step 5 Create **Dimension Sales Person** table which will hold details related stores available across various places. Fill the Dimension **Sales Person** with sample values:*

*Step 6 Create **Date Dimension** table which will create and populate date data divided on various levels.*

*Step 7 Create **Time Dimension** table which will create and populate Time data for the entire day with various time buckets.*

*Step 8 Create **Fact table** to hold all your transactional entries of previous day sales with appropriate foreign key columns which refer to primary key column of your dimensions;*

```
Createdatabase Sales_DW
Create table DimCustomer (
CustomerID int primary key identity,
CustomerAltID varchar(10) not null,
CustomerName varchar(50),
Gender varchar(20)
)
Insert into DimCustomer(CustomerAltID,CustomerName,Gender) values
('IMI-001','Henry Ford','M'),
('IMI-002','Bill Gates','M'),
('IMI-003','Muskan Shaikh','F'),
('IMI-004','Richard Thrubin','M'),
('IMI-005','Emma Wattson','F');)


Create table DimProduct
(
ProductKey int primary key identity,
```

```
ProductAltKey varchar(10)not null,
ProductName varchar(100),
ProductActualCost money,
ProductSalesCost money

)



Insert into DimProduct (ProductAltKey,ProductName, ProductActualCost, ProductSalesCost)values
('ITM-001','Wheat Floor 1kg',5.50,6.50),
('ITM-002','Rice Grains 1kg',22.50,24),
('ITM-003','SunFlower Oil 1 ltr',42,43.5),
('ITM-004','Nirma Soap',18,20),
('ITM-005','Arial Washing Powder 1kg',135,139);)

Create table DimStores
(
StoreID int primary key identity,
StoreAltID varchar(10)not null,
StoreName varchar(100),
StoreLocation varchar(100),
City varchar(100),
State varchar(100),
Country varchar(100) )

Insert into DimStores(StoreAltID,StoreName,StoreLocation,City,State,Country )values
('LOC-A1','X-Mart','S.P. RingRoad','Ahmedabad','Guj','India'),
('LOC-A2','X-Mart','Maninagar','Ahmedabad','Guj','India'),
('LOC-A3','X-Mart','Sivranjani','Ahmedabad','Guj','India');)
```

```
Create table DimSalesPerson
(
SalesPersonID int primary key identity,
SalesPersonAltID varchar(10)not null,
SalesPersonName varchar(100),
StoreID int,
City varchar(100),
State varchar(100),
Country varchar(100)
)


Insert into DimSalesPerson(SalesPersonAltID,SalesPersonName,StoreID,City,State,Country
)values
('SP-DMSPR1','Ashish',1,'Ahmedabad','Guj','India'),
('SP-DMSPR2','Ketan',1,'Ahmedabad','Guj','India'),
('SP-DMNGR1','Srinivas',2,'Ahmedabad','Guj','India'),
('SP-DMNGR2','Saad',2,'Ahmedabad','Guj','India'),
('SP-DMSVR1','Jasmin',3,'Ahmedabad','Guj','India'),
('SP-DMSVR2','Jacob',3,'Ahmedabad','Guj','India');)
```

Week – 9

Write a program of cluster analysis using simple k-means algorithm Python programming language

```
from copy import deepcopy
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt


df = pd.read_csv('/content/iris.csv')
#df.drop('Id',axis=1,inplace=True)
Double-click (or enter) to edit
1 print(df)
```

```
     sepal.length  sepal.width  petal.length  petal.width    variety
0             5.1          3.5           1.4          0.2     Setosa
1             4.9          3.0           1.4          0.2     Setosa
2             4.7          3.2           1.3          0.2     Setosa
3             4.6          3.1           1.5          0.2     Setosa
4             5.0          3.6           1.4          0.2     Setosa
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  Virginica
146           6.3          2.5           5.0          1.9  Virginica
147           6.5          3.0           5.2          2.0  Virginica
148           6.2          3.4           5.4          2.3  Virginica
149           5.9          3.0           5.1          1.8  Virginica

[150 rows x 5 columns]
```

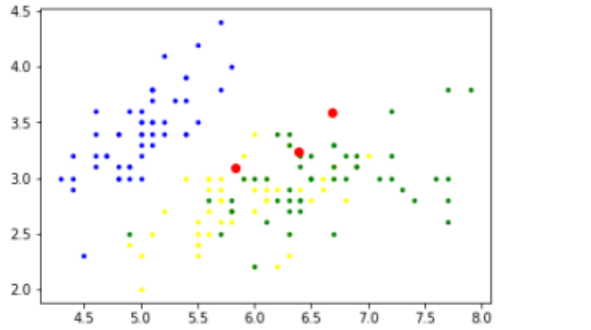
```
df["variety"] = pd.Categorical(df["variety"])
df["variety"] = df["variety"].cat.codes
# Changing dataframe to numpy matrix
data = df.values[:, 0:4]
-= df.values[:, 4]
```

>Time to create the K Means cluster. To make things easier, we'll be creating a plot using the matplotlib module

```
k = 3
# Training data
n = data.shape[0]
# Number of features in the data
c = data.shape[1]
# Generating random centers
mean = np.mean(data, axis = 0)
std = np.std(data, axis = 0)
centers = np.random.randn(k,c)*std + mean
<matplotlib.collections.PathCollection at 0x7fed6de0e690>


# Plotting data
colors=['blue', 'yellow', 'green']
for i in range(n):
```

```
plt.scatter(data[i, 0], data[i,1], s=7, color = colors[int(category[i])])
plt.scatter(centers[:,0], centers[:,1], marker='.', c='r', s=150)
```

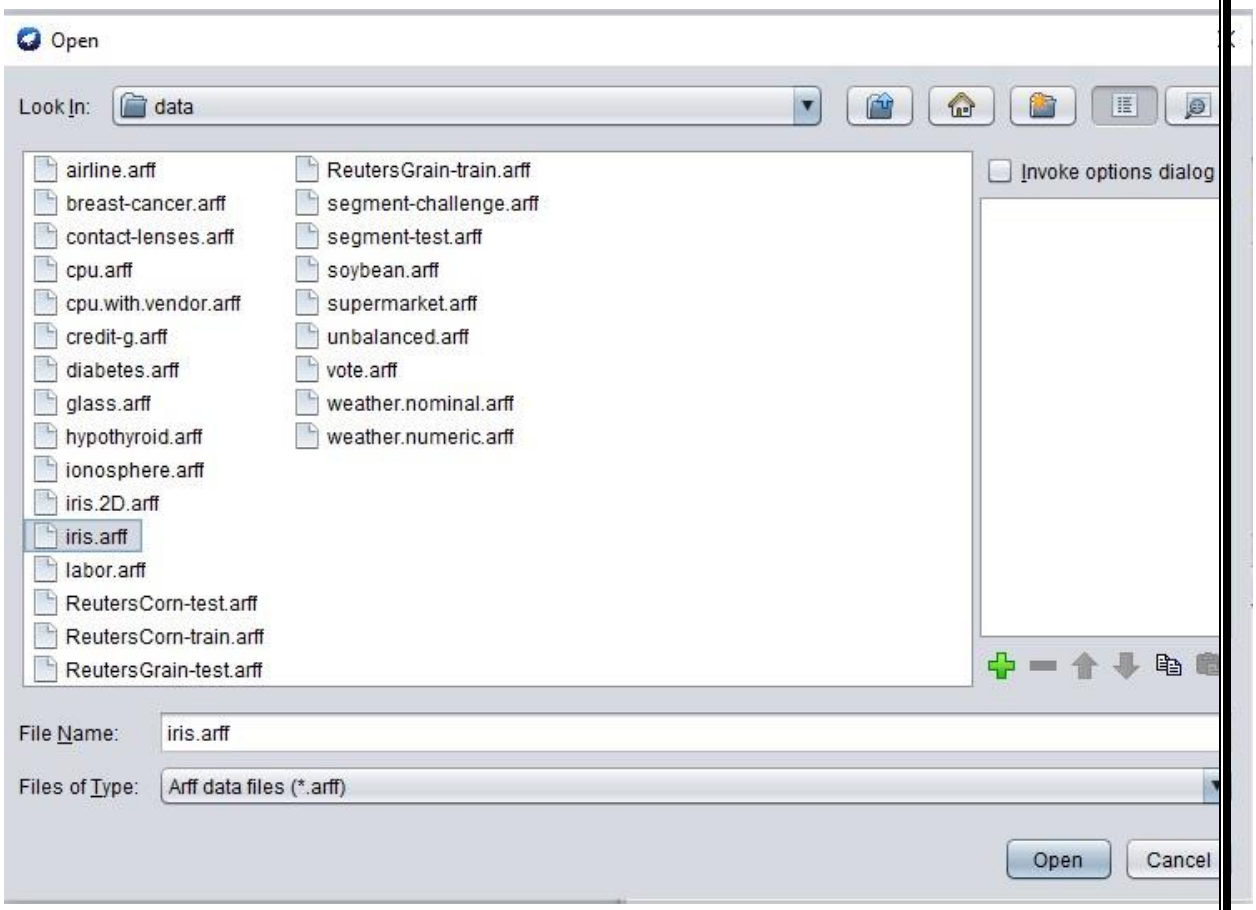
```
<matplotlib.collections.PathCollection at 0x7fed6de0e690>
```



Week 8:

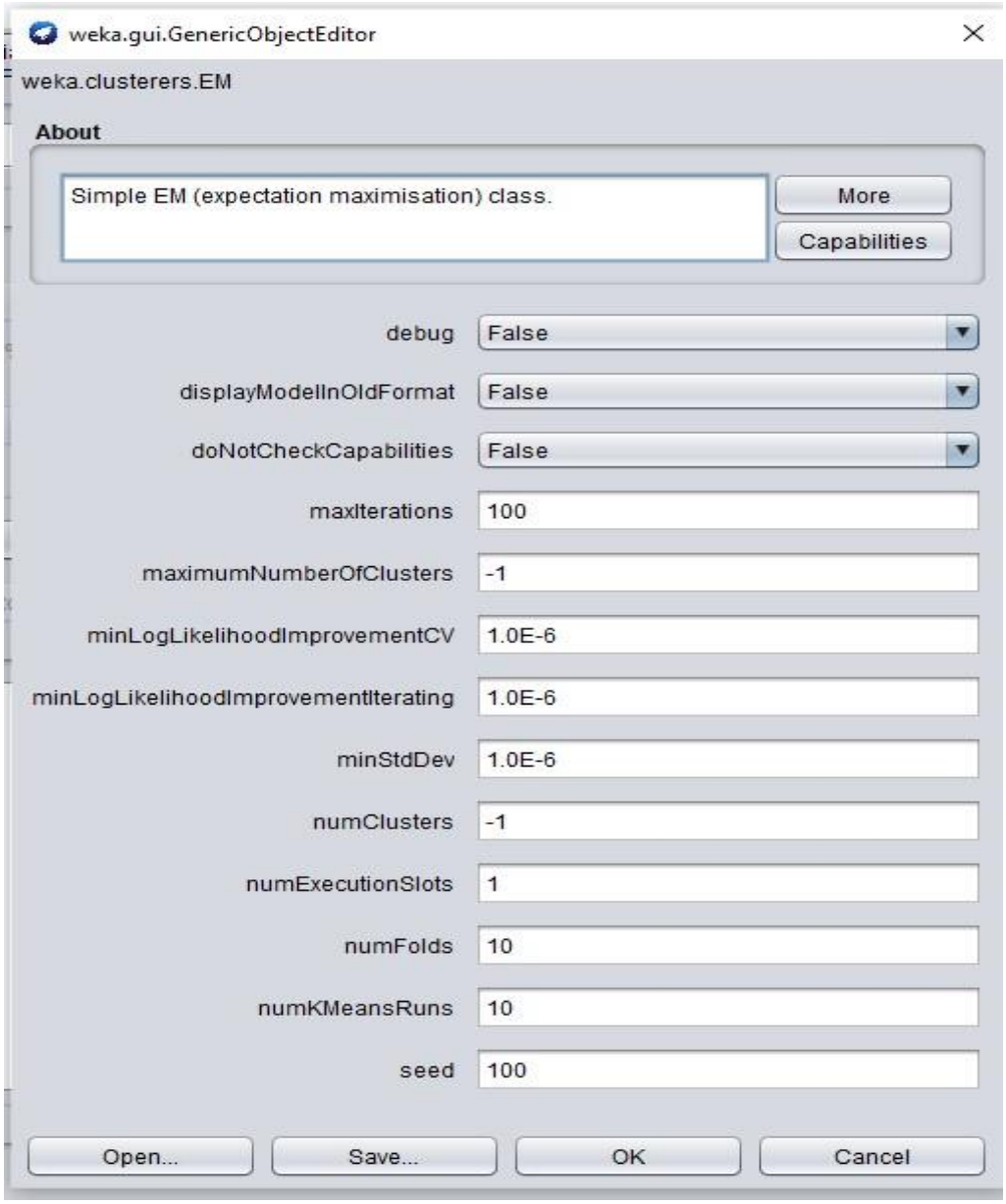
Demonstrate performing clustering of data sets Load each dataset into Weka and run simple k-means clustering algorithm with different values of k (number of desired clusters). Study the clusters formed. Observe the sum of squared errors and centroids, and derive insights. Explore other clustering techniques available in Weka. Explore visualization features of Weka to visualize the clusters. Derive interesting insights and explain **Steps to be followed:**

Step 1: In the preprocessing interface, open the Weka Explorer and load the required dataset, and we are taking the iris.arff dataset.



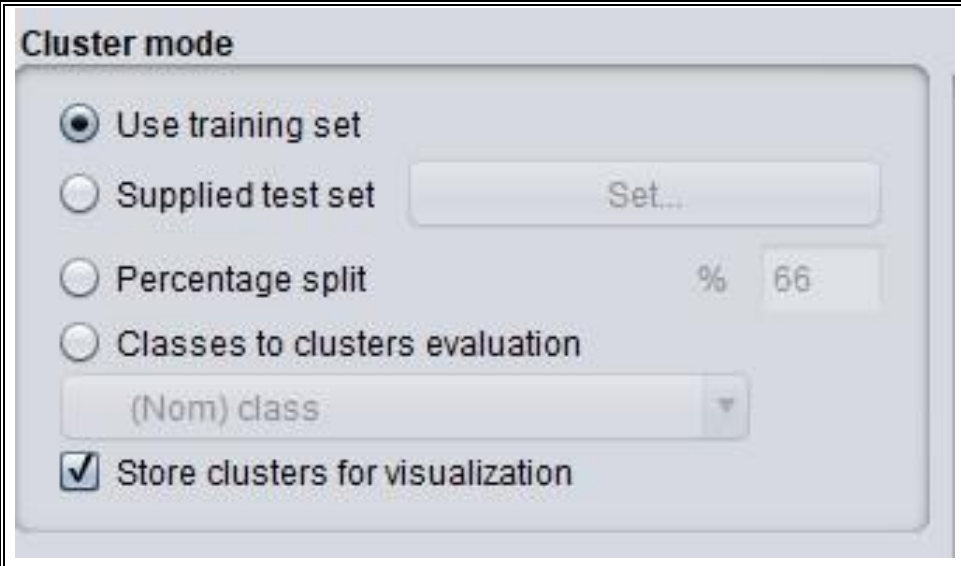
Step 2: Find the 'cluster' tab in the explorer and press the choose button to execute clustering. A dropdown list of available clustering algorithms appears as a result of this step and selects the simple-k means algorithm.

Step 3: Then, to the right of the choose icon, press the text button to bring up the popup window shown in the screenshots. We enter three for the number of clusters in this window and leave the seed value alone. The seed value is used to generate a random number that is used to make internal assignments of instances of clusters



Step 4: One of the choices has been chosen. We must ensure that they are in the 'cluster mode' panel before running the clustering algorithm.

The choice to use a training set is selected, and then the 'start' button is pressed. The screenshots below display the process and the resulting window.



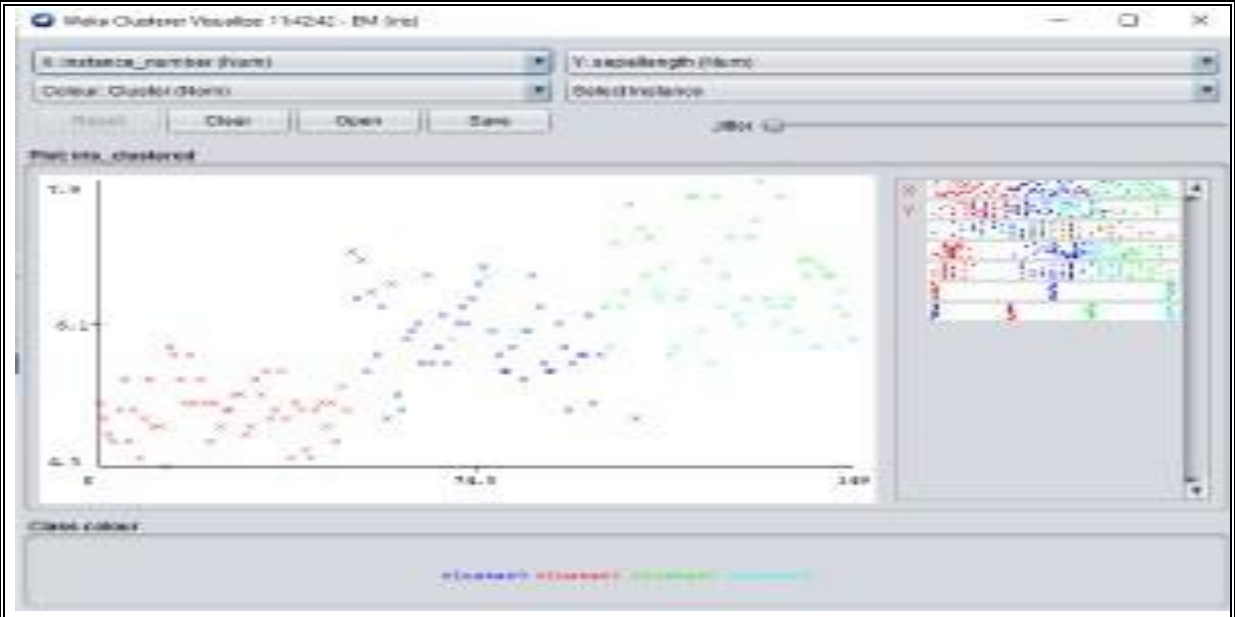
Step 5: The centroid of each cluster is shown in the result window, along with statistics on the number and percent of instances allocated to each cluster. Each cluster centroid is represented by a mean vector.

This cluster can be used to describe a cluster.

```
Number of clusters selected by cross validation: 4
Number of iterations performed: 16


                    Cluster
Attribute                0       1       2       3
                    (0.32)  (0.33)   (0.2)  (0.14)
=========================================================
sepallength
  mean                5.897   5.006  6.9426  6.1304
  std. dev.          0.5279  0.3489   0.498  0.2943

sepalwidth
  mean                2.7519   3.418  3.1103  2.8088
  std. dev.          0.3103  0.3772  0.2952  0.2361

petallength
  mean                4.2267   1.464  5.8559  5.0993
  std. dev.           0.445  0.1718  0.4626  0.2462

petalwidth
  mean                1.3134   0.244  2.1495  1.8254
  std. dev.          0.1864  0.1061   0.232  0.2152

class
  Iris-setosa              1      51       1       1
  Iris-versicolor    48.1125       1  1.0182  3.8693
  Iris-virginica      2.0983       1 31.0375 19.8641
  [total]            51.2108      53 33.0557 24.7335
```

Step 6: Another way to grasp the characteristics of each cluster is to visualize them. To do so, right-click the result set on the result.

Selecting to visualize cluster assignments from the list column.

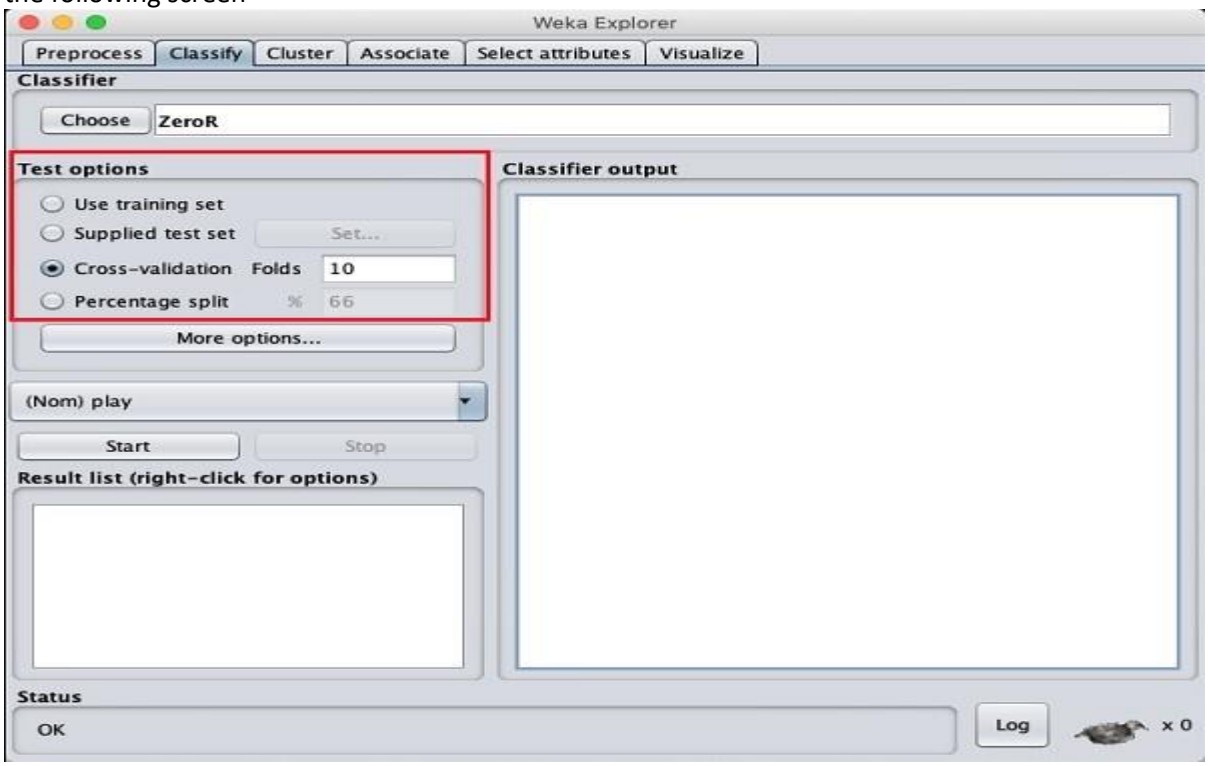

Clear
Open
Save

Week – 6

Demonstrate ZeroR technique on Iris dataset (by using necessary preprocessing technique(s)) and share your observations (using WEKA

Setting Test Data

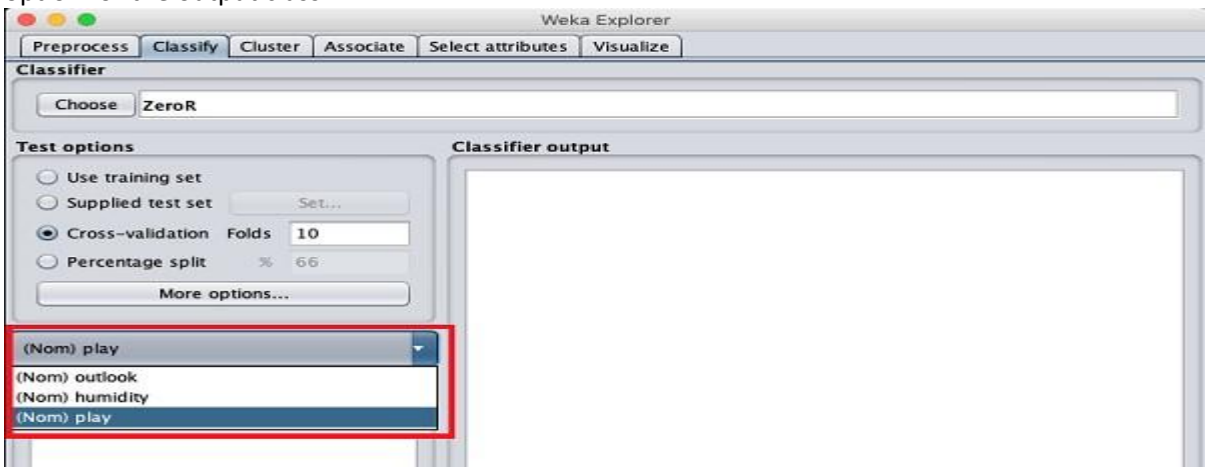
We will use the preprocessed weather data file from the previous lesson. Open the saved file by using the Open file ... option under the Preprocess tab, click on the Classify tab, and you would see the following screen –



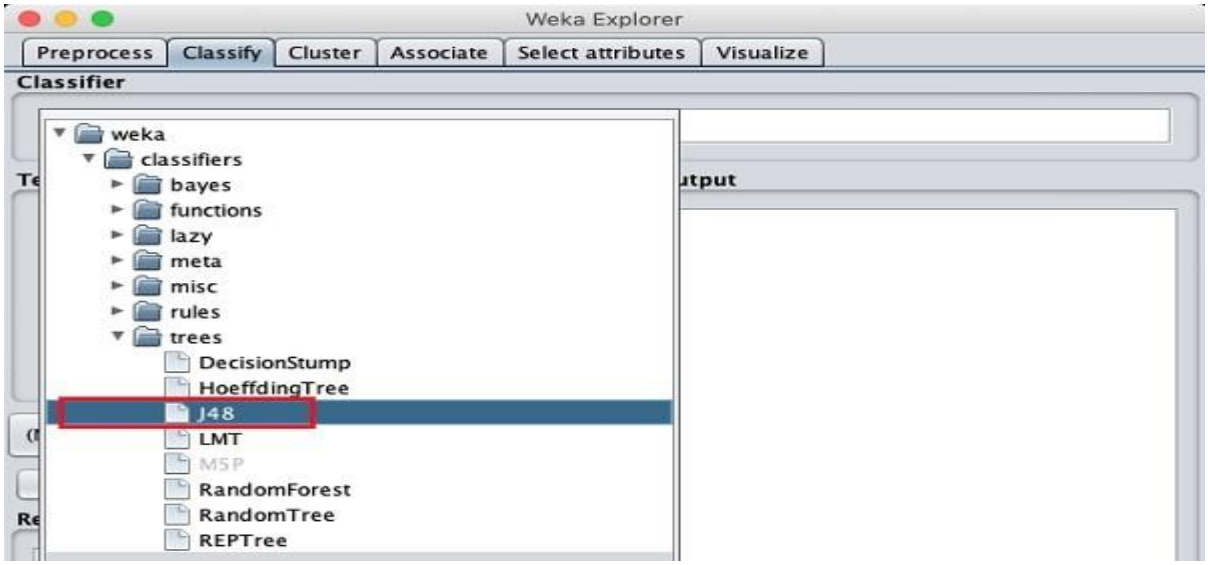
Before you learn about the available classifiers, let us examine the Test options. You will notice four testing options as listed below –

- Training set
- Supplied test set
- Cross-validation
- Percentage split

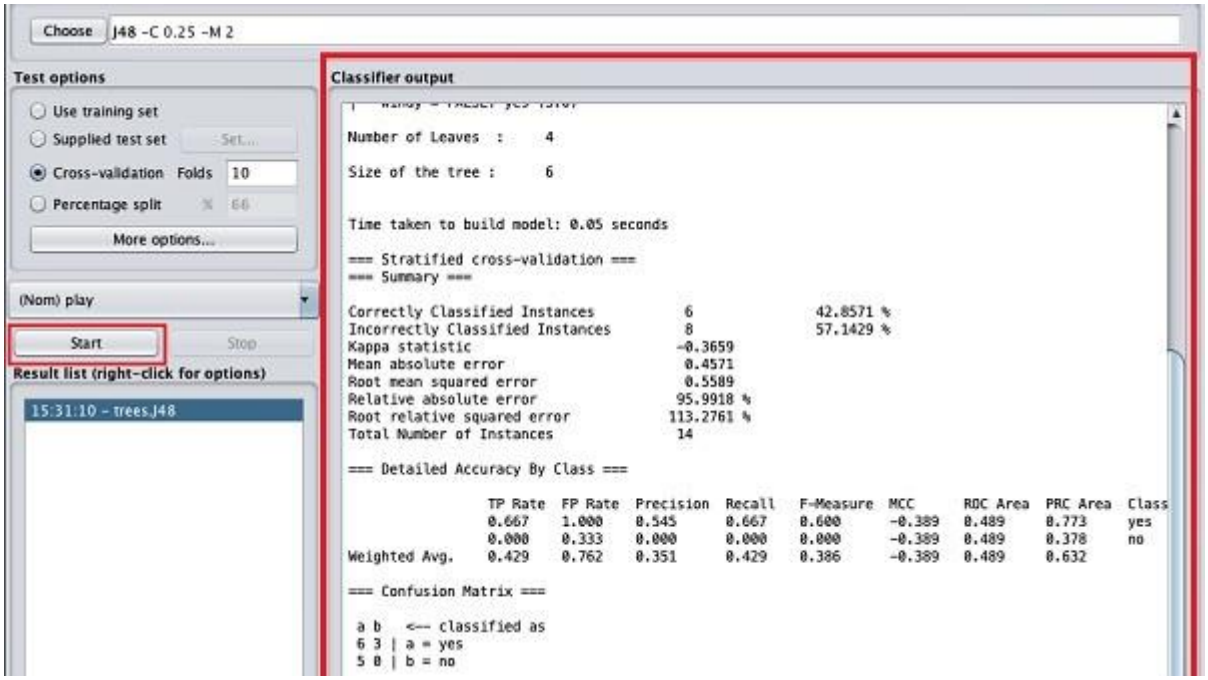
Unless you have your own training set or a client supplied test set, you would use crossvalidation or percentage split options. Under cross-validation, you can set the number of folds in which entire data would be split and used during each iteration of training. In the percentage split, you will split the data between training and testing using the set split percentage.Now, keep the default play option for the output class –



Next, you will select the classifier.
Selecting Classifier
Click on the Choose button and select the following classifier –
weka→classifiers>trees>J48
This is shown in the screenshot below –



Click on the Start button to start the classification process. After a while, the classification results would be presented on your screen as shown here –



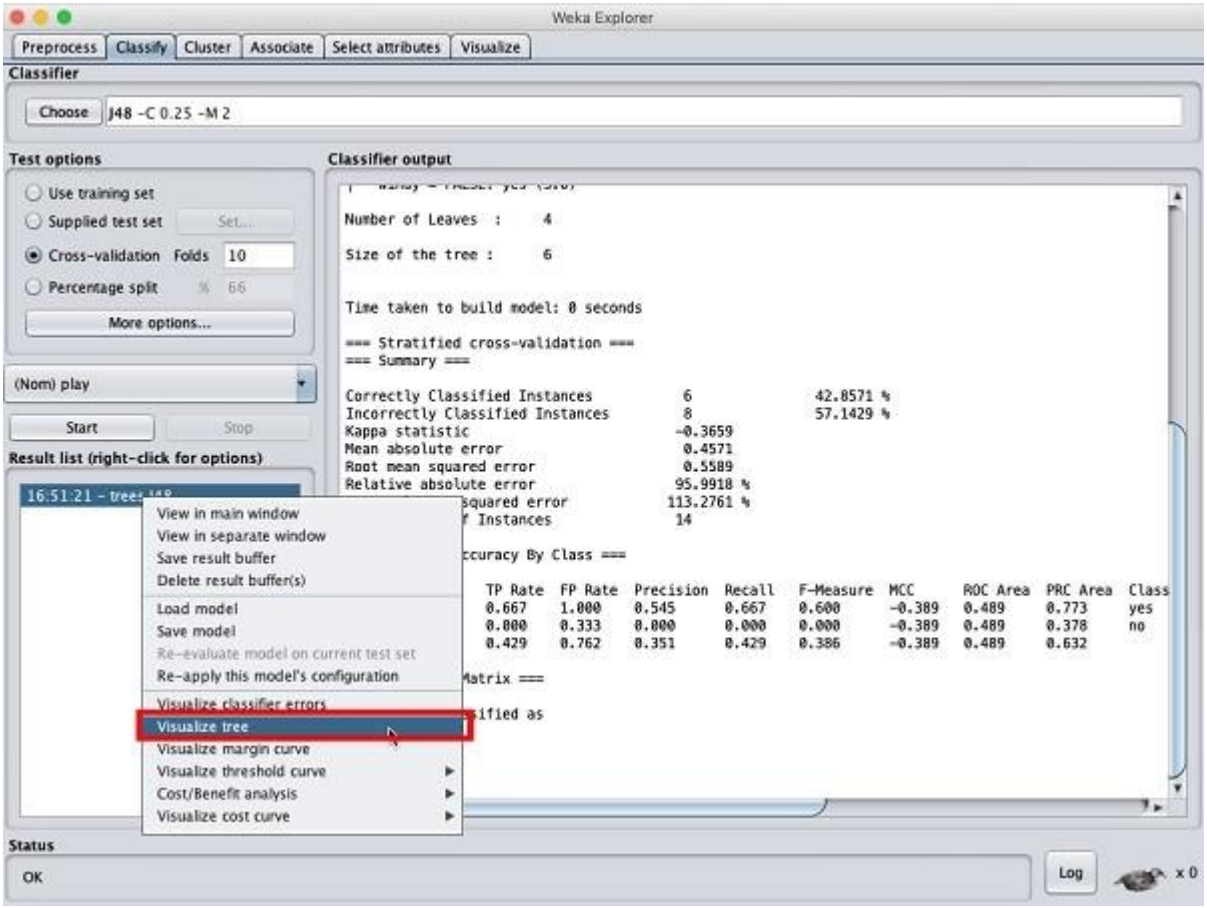
Let us examine the output shown on the right hand side of the screen.

It says the size of the tree is 6. You will very shortly see the visual representation of the tree. In the Summary, it says that the correctly classified instances as 2 and the incorrectly classified instances as 3, It also says that the Relative absolute error is 110%. It also shows the Confusion Matrix. Going into the analysis of these results is beyond the scope of this tutorial. However, you can easily make out from these results that the classification is not acceptable and you will need more data for analysis, to refine your features selection, rebuild the model and so on until you are satisfied with the model's accuracy. Anyway, that's what WEKA is all about. It allows you to test your ideas quickly.

Visualize Results

To see the visual representation of the results, right click on the result in the Result list box.

Several options would pop up on the screen as shown here −



Select Visualize tree to get a visual representation of the traversal tree as seen in the screenshot below −