

**MACHINE LEARNING FOR COMMUNICATION**

**(MEIC501P)**

**LAB TASK 2**

**2a-PERFORMANCE ANALYSIS OF LINEAR REGRESSION USING EEG SIGNAL**

**NAME: Praveen M**

**REGISTRATION NO: 24MEC0016**

**PHONE NUMBER: +91 99524622525**

**MAIL ID: praveen.m2024b@vitstudent.ac.in**

## Task 2a - Performance Analysis of Linear Regression using EEG Signal

### AIM:

This experiment aims to investigate how anxiety influences EEG signal patterns, focusing on beta activity, alpha asymmetry, ERPs, and connectivity patterns, and to understand the variability of these effects based on anxiety intensity and context.

### PROGRAM:

```
1  from glob import glob
2  import numpy as np
3  import pandas as pd
4  import mne # library used for extracting EEG
5  from matplotlib import pyplot as plt
6
7  # Get the list of .edf files in the specified directory
8  data_set = glob('D:/Machine Learning/Lab Task 2/dataverse_files/*.edf')
9
10 # Define a function to extract data from our EEG signal
11 def read_data(file_path):
12     # Reading our raw EEG data
13     data = mne.io.read_raw_edf(file_path, preload=True)
14     # Creating fixed length epochs of 5 seconds with 1 second overlap
15     epochs = mne.make_fixed_length_epochs(data, duration=5, overlap=1)
16     # Extracting data in the form of an array
17     array = epochs.get_data()
18     # Taking out the individual max value and multiplying by 1e6 for efficient plotting
19     return max(array[0][0] * 1000000)
20
21 # Apply the read_data function to each .edf file and store the results in a list
22 data_array = [read_data(i) for i in data_set]
23
24 # Creating a DataFrame using pandas
25 df = pd.DataFrame()
26 # Appending our data into the DataFrame
27 df['EEG_epochs'] = data_array
28
29 # Reading our source file using pandas
30 df2 = pd.read_csv('Sor.csv')
31 # Appending our ANX range into the DataFrame
32 df['ANX'] = df2['ANX']
33 # Converting the DataFrame into a CSV file
34 df.to_csv('epo2.csv')
```

```

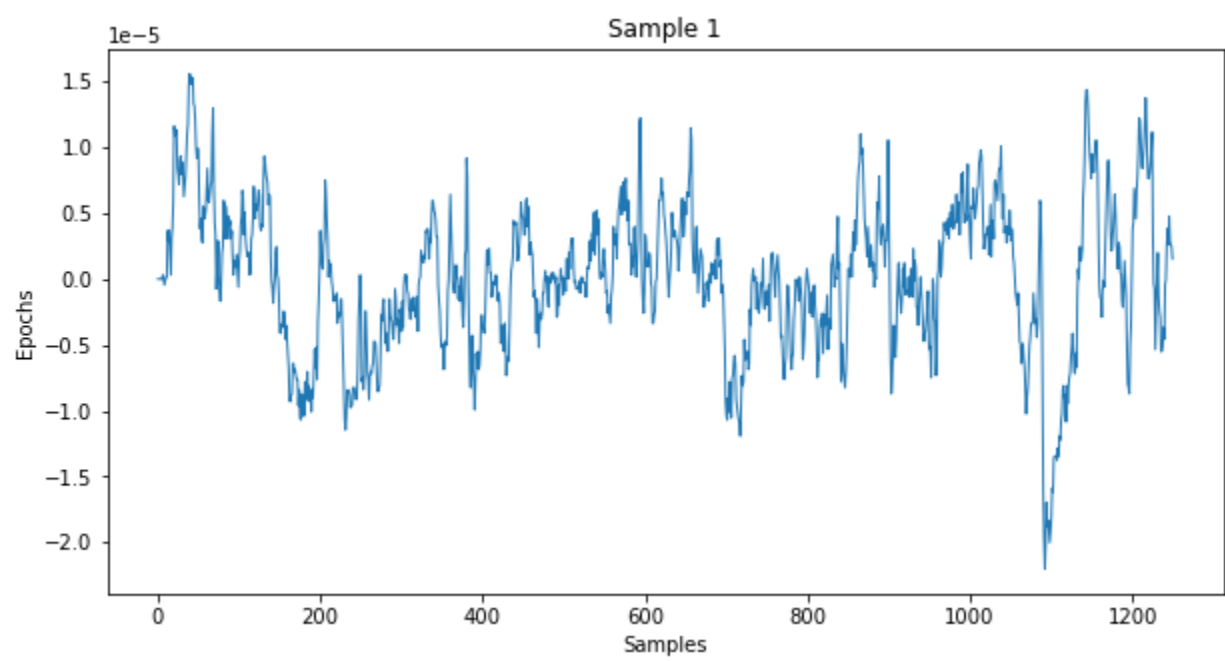
36 # Signal Representation
37 # Reading raw EEG data from the first .edf file
38 raw = mne.io.read_raw_edf(data_set[0])
39 # Creating fixed length epochs of 5 seconds with 1 second overlap
40 epochs1 = mne.make_fixed_length_epochs(raw, duration=5, overlap=1)
41 # Extracting data in the form of an array
42 arr = epochs1.get_data()
43
44 # Plotting the data
45 pd.Series(arr[0][0]).plot(figsize=(10, 5), lw=1, title='Sample 1')
46 plt.xlabel('Samples')
47 plt.ylabel('Epochs')
48 # Printing the data of the first epoch
49 print(arr[0][0])
50 # Displaying the plot
51 plt.show()
52
53 # Printing the DataFrame
54 print(df)
55

```

## OUTPUT:

	EEG_epochs	ANX
0	15.598824	0.3571
1	19.574357	0.7142
2	19.880167	1.0713
3	20.491788	1.4284
4	22.632460	1.7855
5	23.123123	2.1426
6	24.008606	2.4997
7	25.843468	2.8568
8	27.525424	3.2139
9	28.442855	3.5710

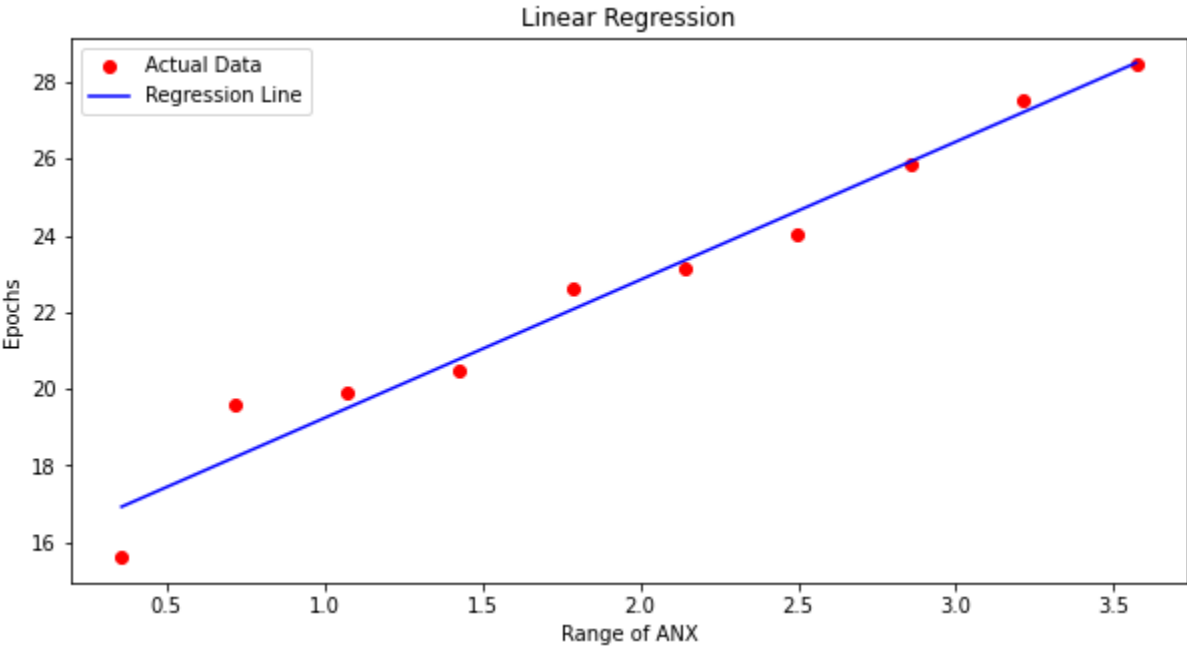
**OUTPUT WAVEFORM:**



## PROGRAM:

```
1  import pandas as pd
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from sklearn import linear_model
5  # library used for Machine learning, we are using this for Linear Regression
6  import warnings
7
8  warnings.filterwarnings("ignore") # Ignore warnings for a cleaner output
9
10 # Read the CSV file
11 df = pd.read_csv('epo2.csv') # Load data from a CSV file into a pandas DataFrame
12
13 # Define a linear regression model
14 reg = linear_model.LinearRegression() # Initialize a Linear Regression model
15
16 # Fit the model using ANX as the predictor and EEG_epochs as the response variable
17 reg.fit(df[['ANX']], df[['EEG_epochs']])
18 # Train the model with 'ANX' as the input and 'EEG_epochs' as the output
19
20 # Predicting an EEG_epochs value by providing a random ANX value
21 predicted_value = reg.predict([[4.2]])
22 # Use the trained model to predict 'EEG_epochs' for ANX value of 4.2
23 print(predicted_value) # Print the predicted value
24
25 # Plotting our Linear regression graph
26 plt.figure(figsize=(10, 5)) # Set the figure size for the plot
27 plt.title('Linear Regression') # Set the title of the plot
28 plt.xlabel('Range of ANX') # Set the x-axis label
29 plt.ylabel('Epochs') # Set the y-axis label
30
31 # Scatter plot of the actual data
32 plt.scatter(df['ANX'], df['EEG_epochs'], color='red', label='Actual Data')
33 # Plot the actual data points in red
34
35 # Line plot of the regression line
36 plt.plot(df['ANX'], reg.predict(df[['ANX']]), color='blue', label='Regression Line')
37 # Plot the regression line in blue
38
39 plt.legend() # Add a legend to the plot
40 plt.show() # Display the plot
41
```

WAVEFORM:



## TASK 2b - ANALYSIS OF MACHINE LEARNING TOOL

### TOOLS USED

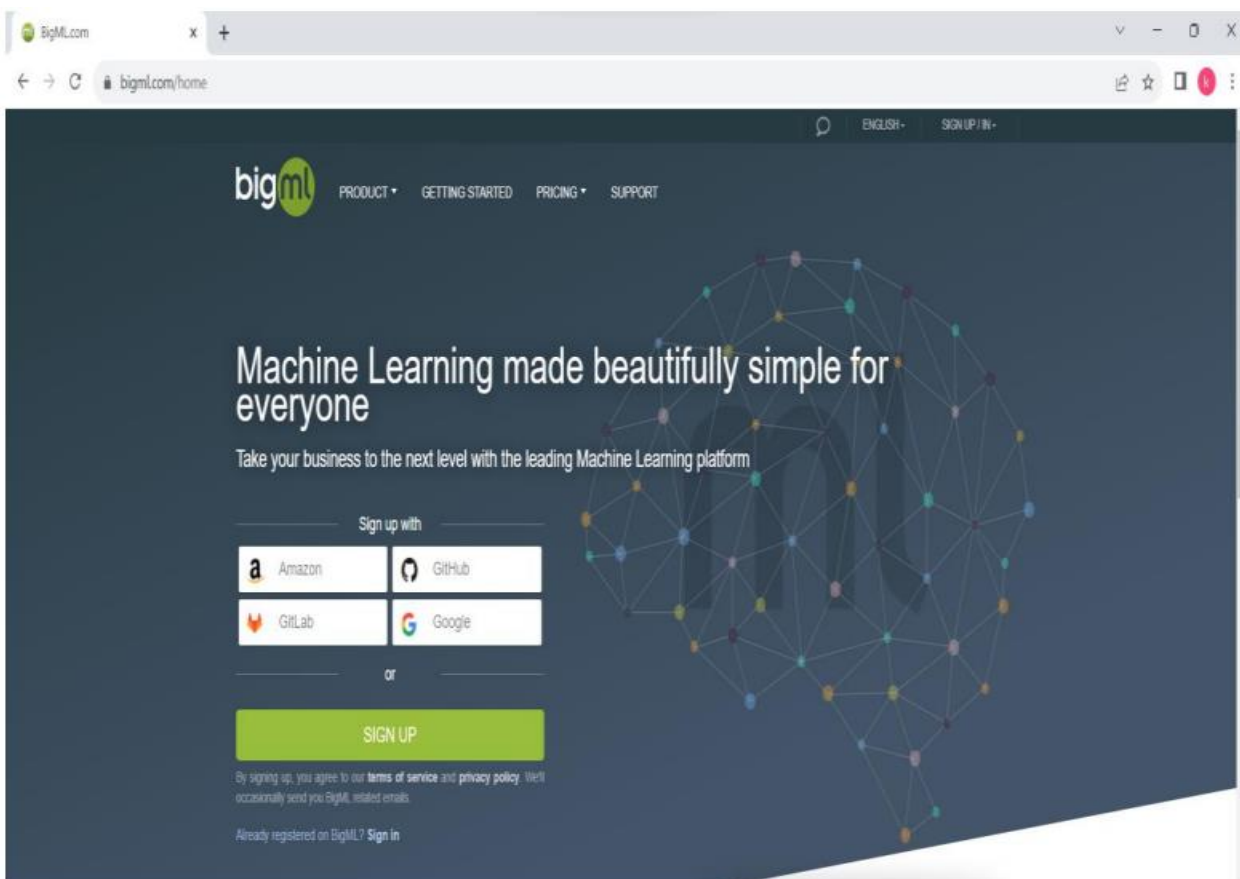
#### 1.BIGML 2. WEKA

### INTRODUCTION

BigML is a consumable, programmable, and scalable Machine Learning platform that makes it easy to solve and automate Classification, Regression, Time Series Forecasting, Cluster Analysis, Anomaly Detection, Association Discovery, and Topic Modelling tasks.

In this tool I have used various machine learning algorithms to study the dataset. The dataset was downloaded from [www.kaggle.com](http://www.kaggle.com)

STEP 1: OPEN BIGML WEBSITE FROM ANY BROWSER. THIS IS ONLINE SOFTWARE INTERFACE



## STEP 2: SIGN UP WITH YOUR GOOGLE ACCOUNT OR AMAZON, GITHUB ETC

## STEP 3: DOWNLOAD ANY DATA SET FROM KAGGLE



## STEP 4: UPLOAD YOUR DATASET IN BIGML

Sources

Datasets

Supervised ▾

Unsupervised ▾

Predictions ▾

Tasks

WhizzML ▾

Sources

🗑️

🔍

🧩+

📊+

📁+

📄+

Type ↕	Name			
📄 CSV	<b>archive (1).zip</b> closed, table, 6 fields (4 categorical, 2 numeric)			
📄 CSV	<b>heart_attack_dataset.csv</b> open, table, 8 fields (5 categorical, 3 numeric)			
🧩	<b>hot-dog-or-not.zip</b> open, image, 160 sources, 237 fields (1 categorical, 234 numeric, 1 path, 1 image)	11h 50min	60.8 KB	0
🧩	<b>firetruck.zip</b> open, image, 118 sources, 236 fields (234 numeric, 1 path, 1 image)	1d 1h	2.2 MB	0
🧩	<b>grape-strawberry.zip</b> open, image, 70 sources, 237 fields (1 categorical, 234 numeric, 1 path, 1 image)	1d 1h	1.5 MB	0
📄 CSV	<b>Country Stats Mashup</b> open, table, 8 fields (8 numeric)	1d 1h	1.9 MB	0
📄 TSV	<b>Fictional Wine Sales</b> open, table, 6 fields (3 categorical, 3 numeric)	1d 1h	12.0 KB	0
📄 CSV	<b>Titanic Survival</b> open, table, 5 fields (3 categorical, 2 numeric)	1d 1h	51.9 KB	0
📄 BZ2	<b>US Car Accidents in 2011</b> open, table, 16 fields (8 categorical, 7 numeric, 1 datetime)	1d 1h	78.0 KB	0
📄 CSV	<b>Premier League 2011-2012 Season</b> open, table, 26 fields (1 categorical, 24 numeric, 1 datetime)	1d 1h	885.5 KB	0

📄

UPLOAD A LOCAL FILE

CSV, TSV, TXT, JSON, ARFF, XLS, GZ, BZ2, PNG, JPG, GIF, TIF, BMP, WEBP AND MORE...

🔗

CREATE A SOURCE FROM A URL

📄

CREATE AN INLINE SOURCE

Show

10 ▾

sources

1 to 10 of 15 sources

⏪

⏴


1

2

⏵

⏩

## STEP 4: NOW YOUR DATASET WILL BE VISIBLE LIKE THIS



[PRODUCT](#)
[GETTING STARTED](#)
[PRICING](#)
[SUPPORT](#)

PRAVEENMAHE...

Dashboard

**PRAVEENMAHEN2001** - My Dashboard

All

Sources

**Datasets**

Supervised

Unsupervised

Predictions

Tasks

WhizzML

Datasets

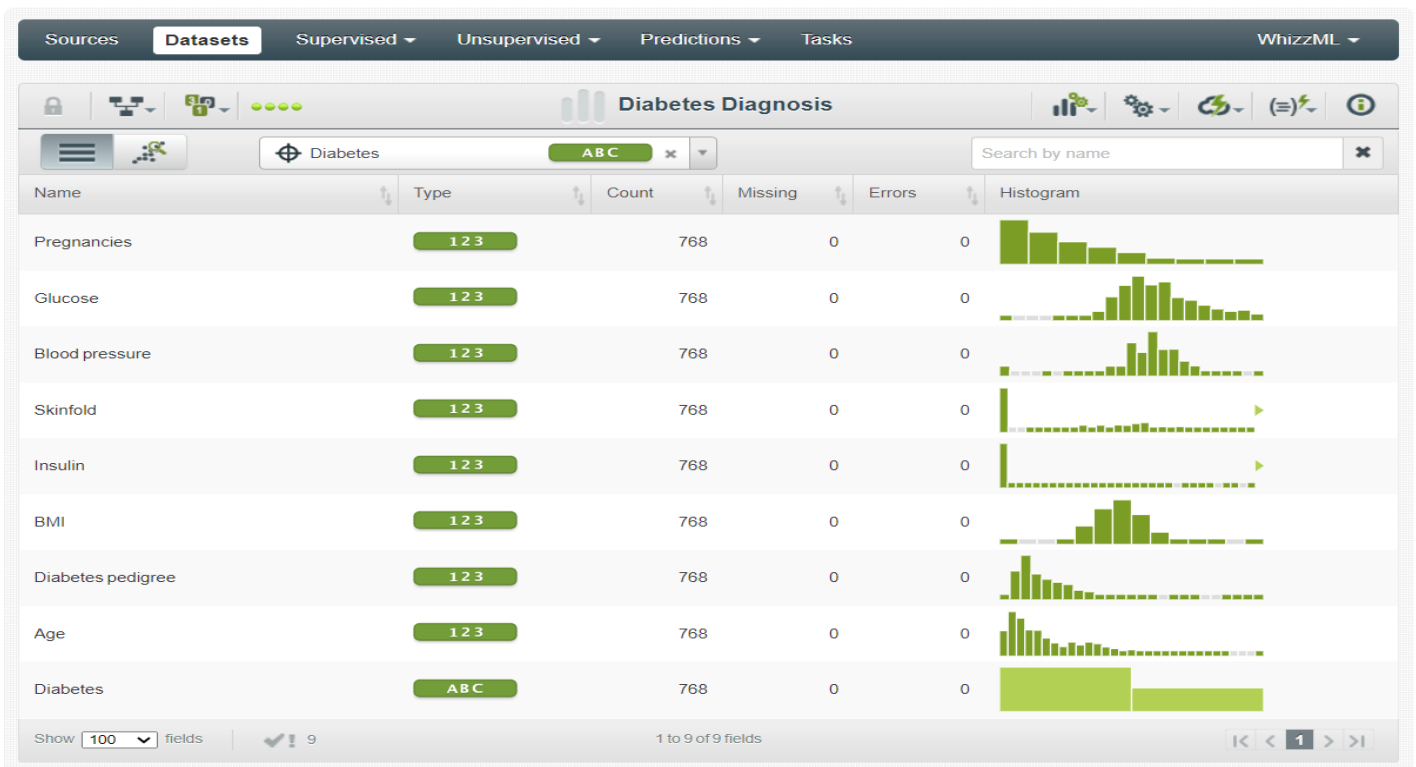
	Name																
	<b>archive (1)</b> 1000 instances, 6 fields (4 categorical, 2 numeric)	11h	42 K	0	0	1	0	0	0	0	0	0	0	0	0	0	0
	<b>Diabetes Diagnosis</b> 768 instances, 9 fields (1 categorical, 8 numeric)	11h	26 K	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Show  datasets

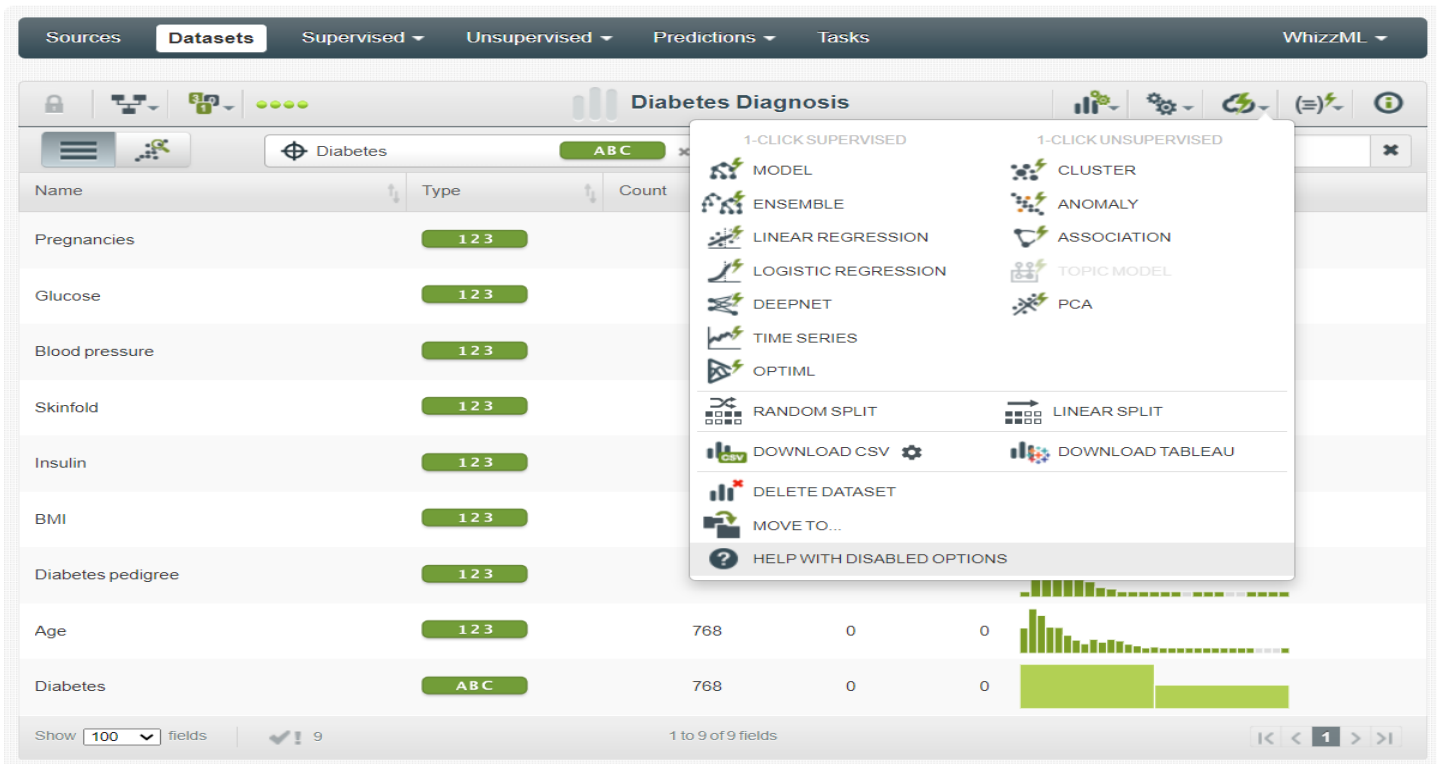
1 to 2 of 2 datasets

1

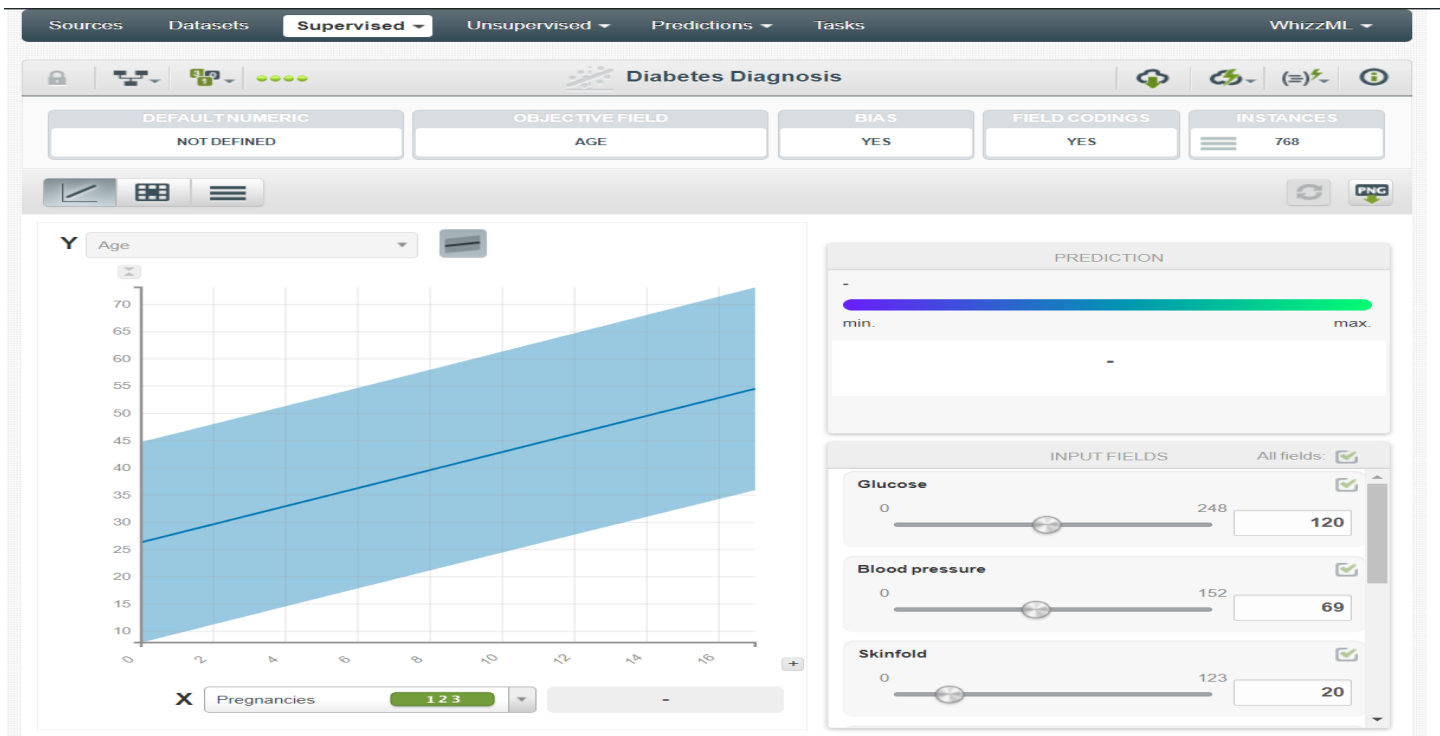
## STEP 5: FULL DETAILED REPRESENTATION OF DATASET CAN BE REPRESENTED



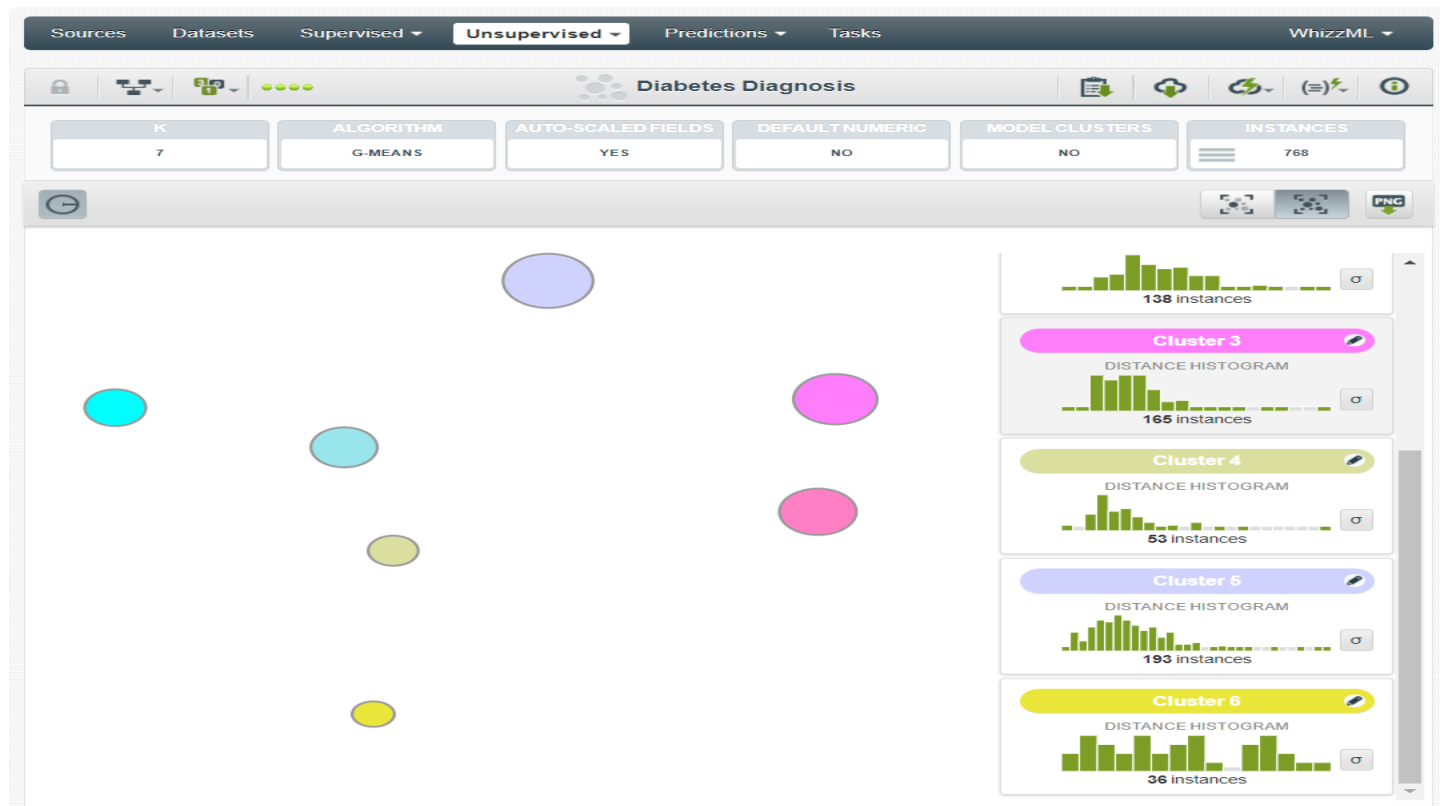
## STEP 6: HERE YOU CAN SEE SUPERVISED AND UNSUPERVISED CLASSIFIER



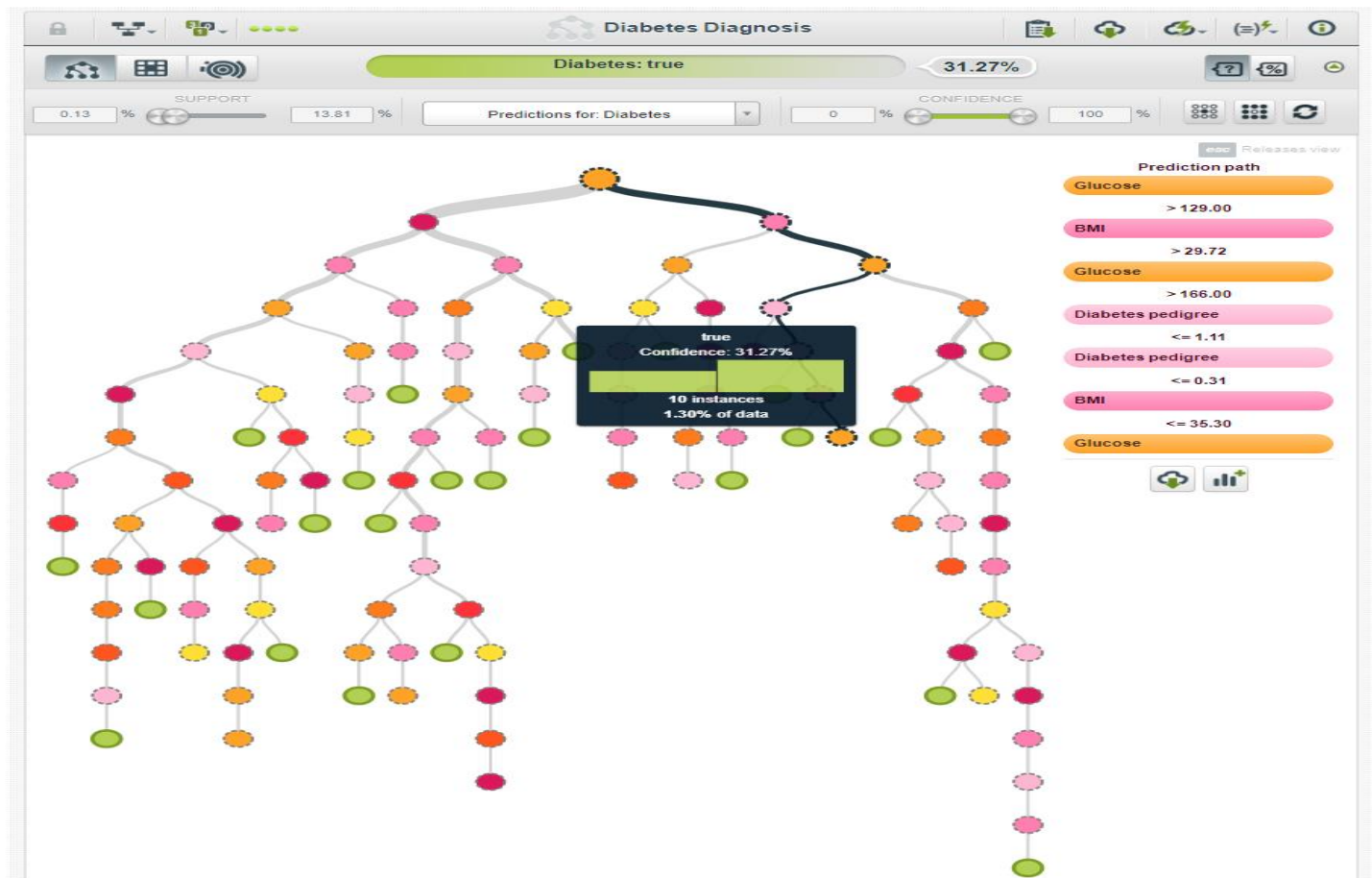
STEP 7: YOU CAN SEE LINEAR REGRESSION CURVE FOR GIVEN DATASET



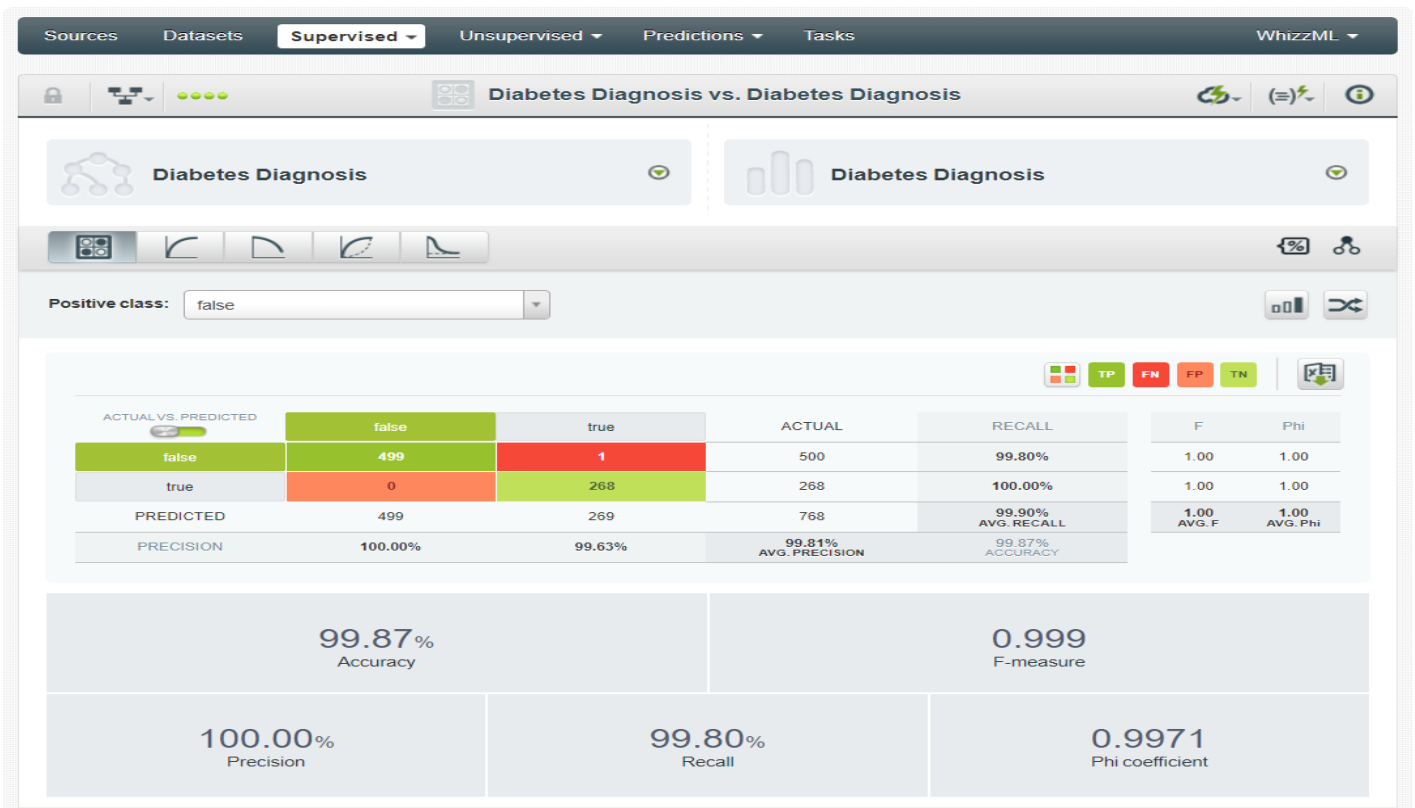
STEP 8: THIS BELOW IMAGE SHOWS UNSUPERVISED KMEANS CLUSTERING ALGORITHM FOR THE FOLLOWING DATASET



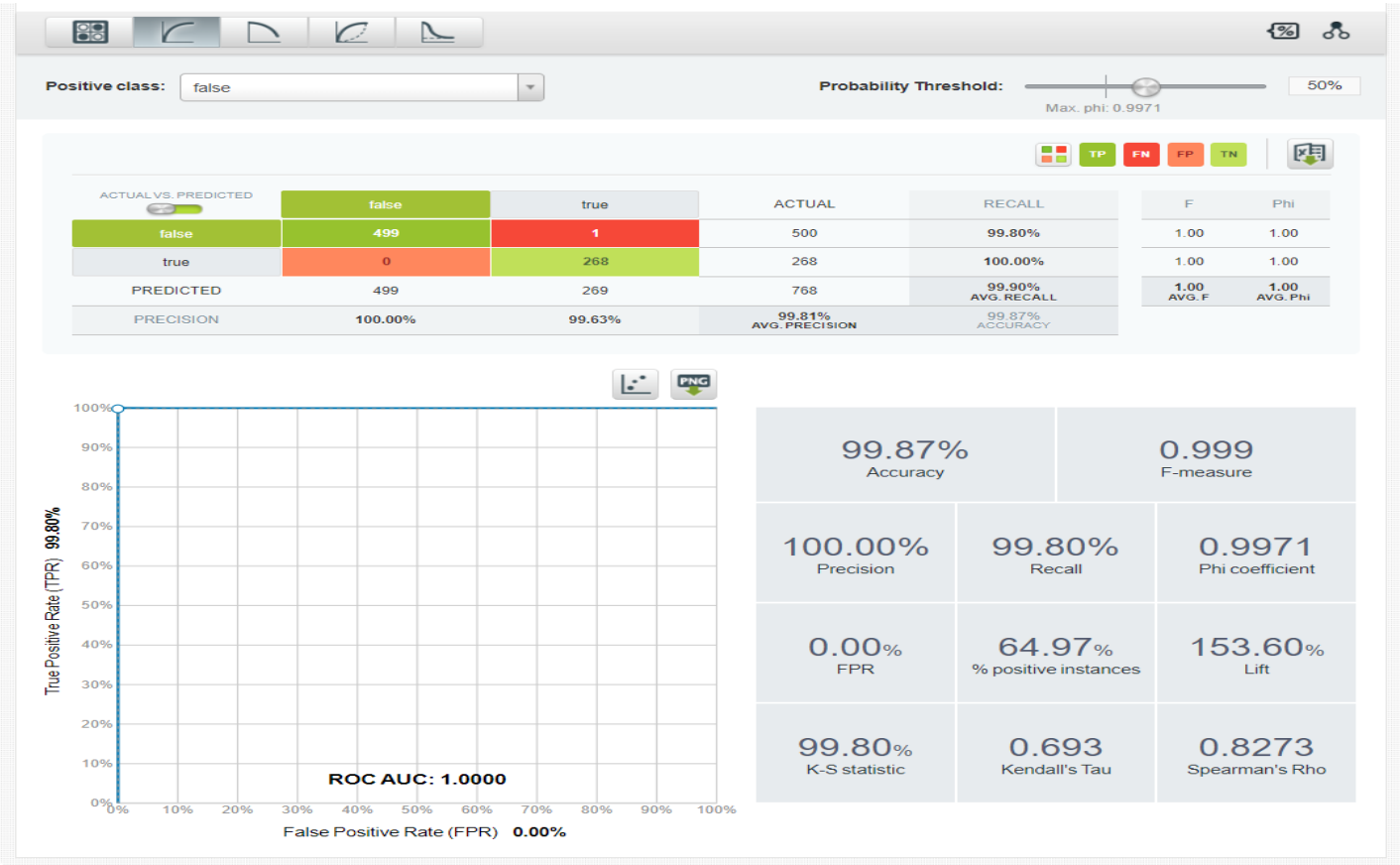
## MODEL:



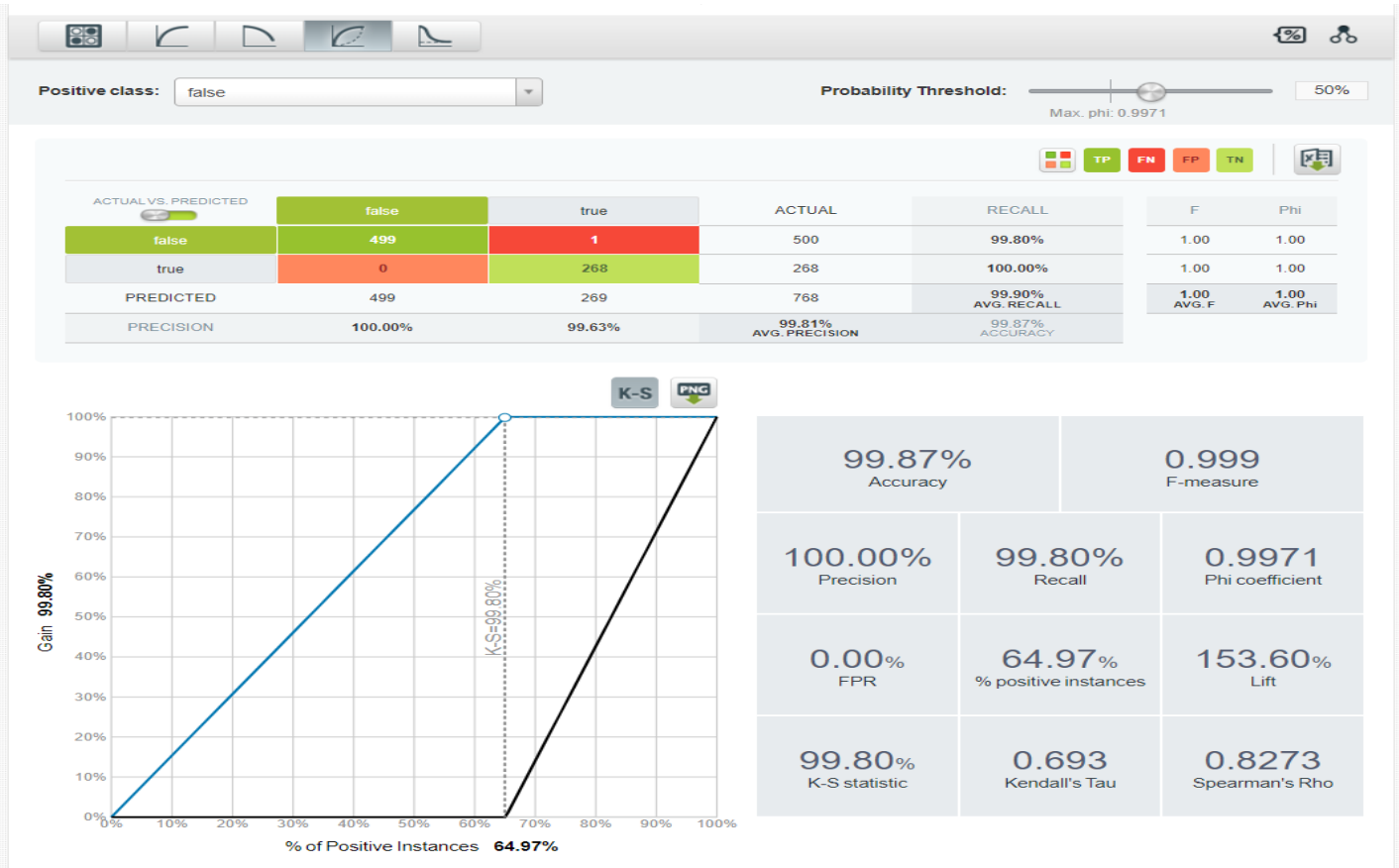
## EVALUATION:



ROC:



K MEANS:

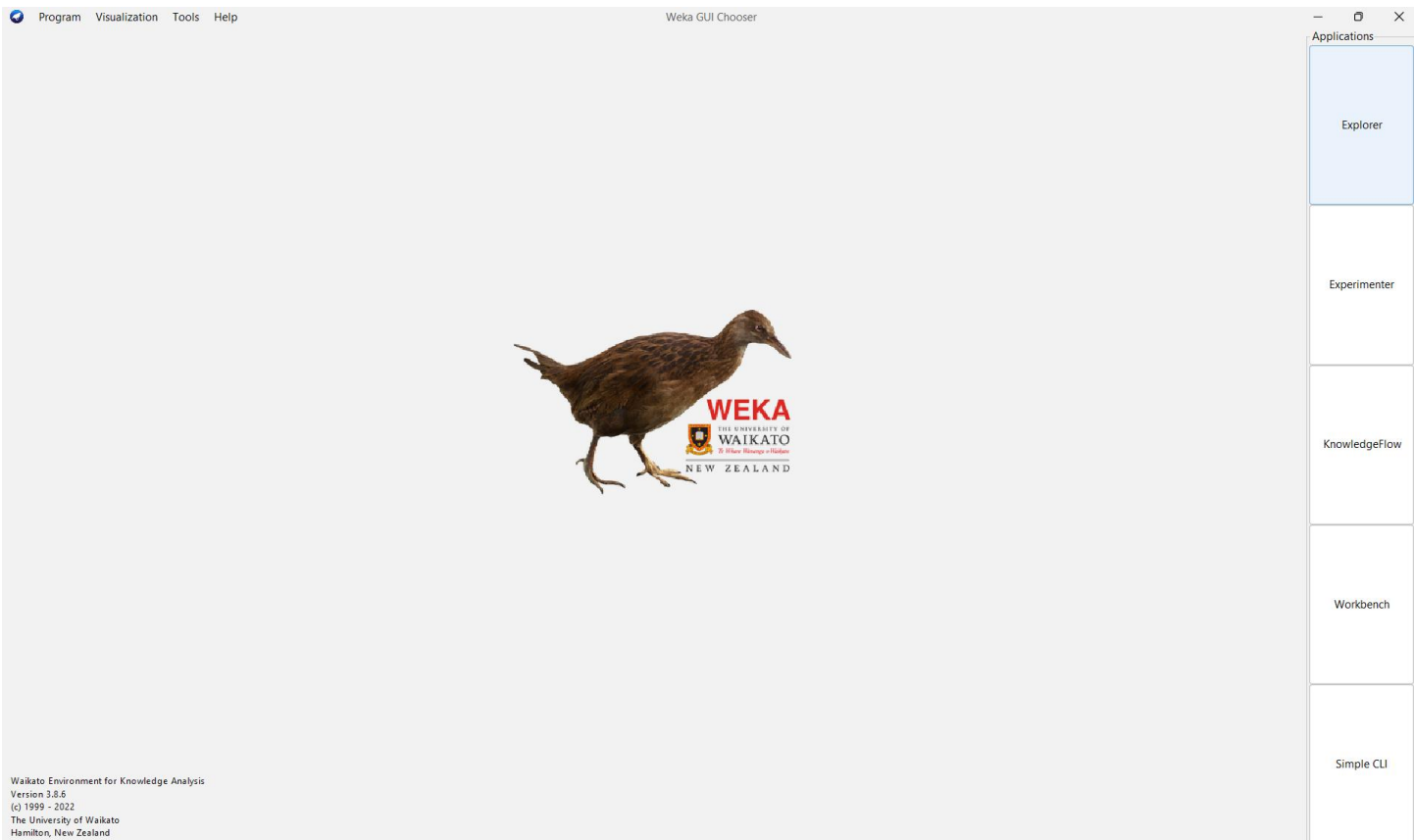


## TOOL 2: WEKA

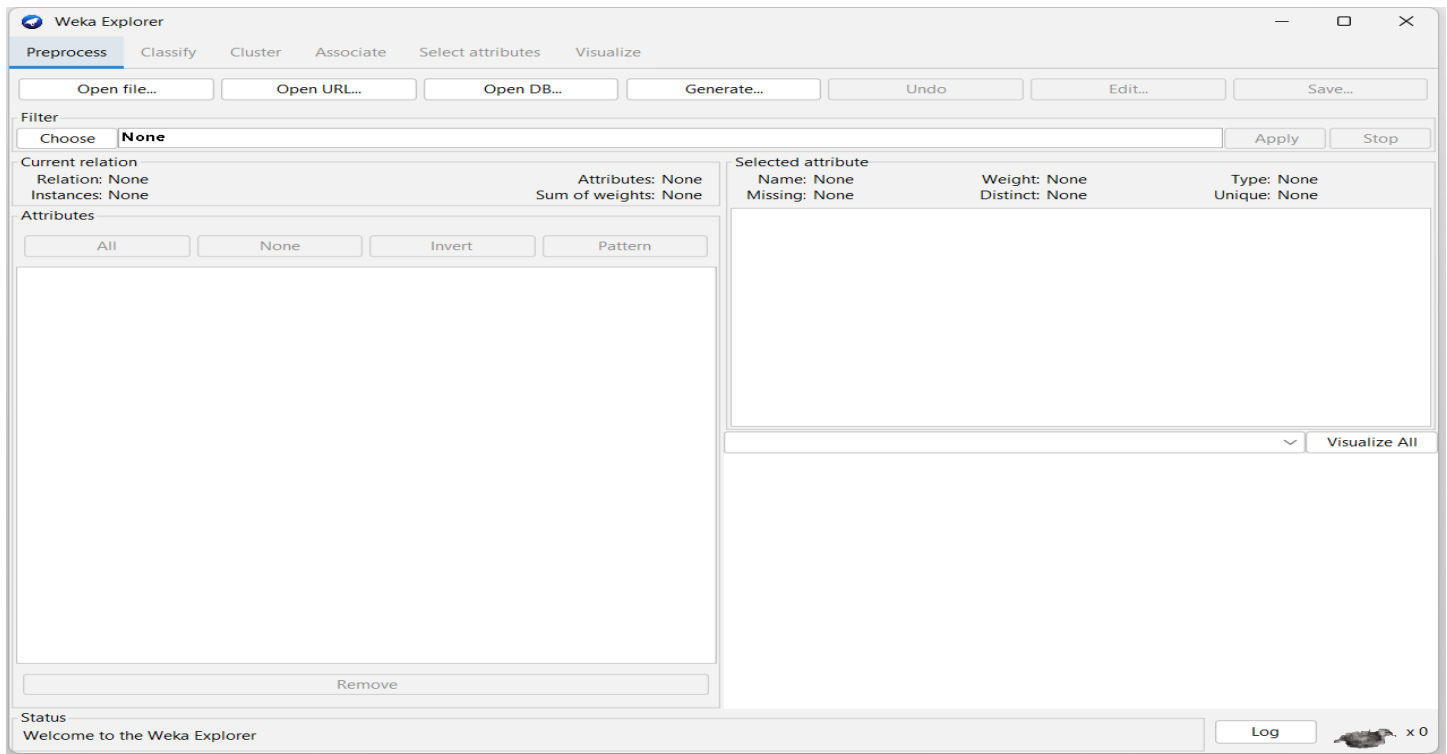
### INTRODUCTION

Waikato Environment for Knowledge Analysis (Weka) is a collection of machine learning and data analysis free software. Weka contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to these functions.

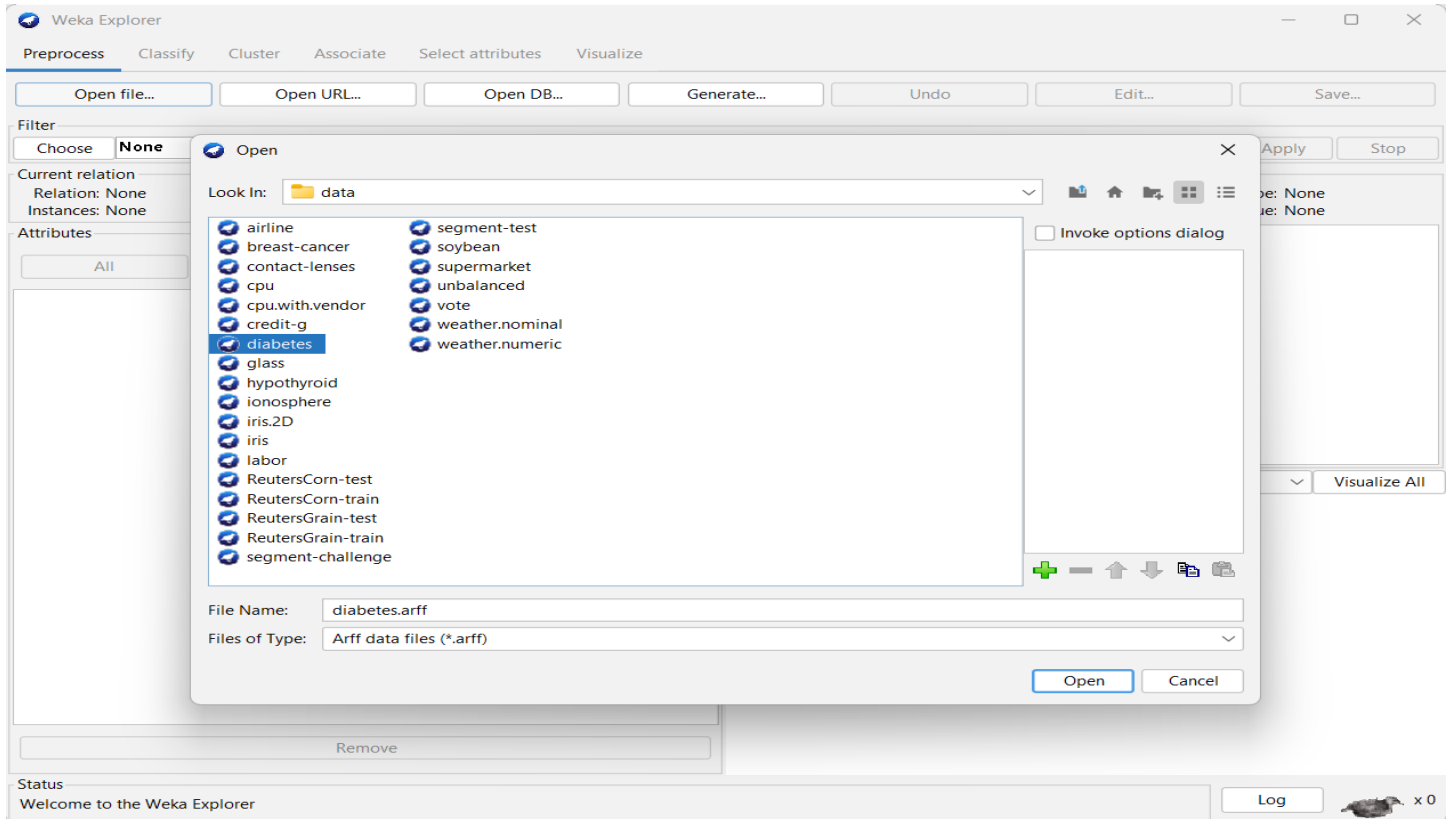
### STEP 1: DOWNLOAD AND INSTALL THE WEKA SOFTWARE FROM THE GOOGLE



## STEP 2: CLICK ON THE EXPLORER YOU CAN SEE THIS WINDOW



## STEP 4: INSIDE THE DATA FOLDER YOU CAN SEE DIFFERENT TYPES OF INBUILT DATASETS YOU CAN CHOOSE ANY ONE OF IT



## STEP 5: DATASET AND PREPROCESSED USING NAÏVE BAYES CLASSIFIER

Weka Explorer

Preprocess   **Classify**   Cluster   Associate   Select attributes   Visualize

Open file...   Open URL...   Open DB...   Generate...   Undo   Edit...   Save...

Filter  
Choose: **None**   Apply   Stop

Current relation  
Relation: pima\_diabetes  
Instances: 768  
Attributes: 9  
Sum of weights: 768

Attributes  
All   None   Invert   Pattern

No.	Name
1	<input type="checkbox"/> preg
2	<input type="checkbox"/> plas
3	<input type="checkbox"/> pres
4	<input type="checkbox"/> skin
5	<input type="checkbox"/> insu
6	<input type="checkbox"/> mass
7	<input type="checkbox"/> pedi
8	<input type="checkbox"/> age
9	<input type="checkbox"/> class

Remove

Status  
OK

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

Class: class (Nom)   Visualize All

Log   x 0

## STEP 6: NOW GO TO CLASSIFY IN THE MENU

Weka Explorer

Preprocess   **Classify**   Cluster   Associate   Select attributes   Visualize

Classifier  
Choose: **ZeroR**

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)

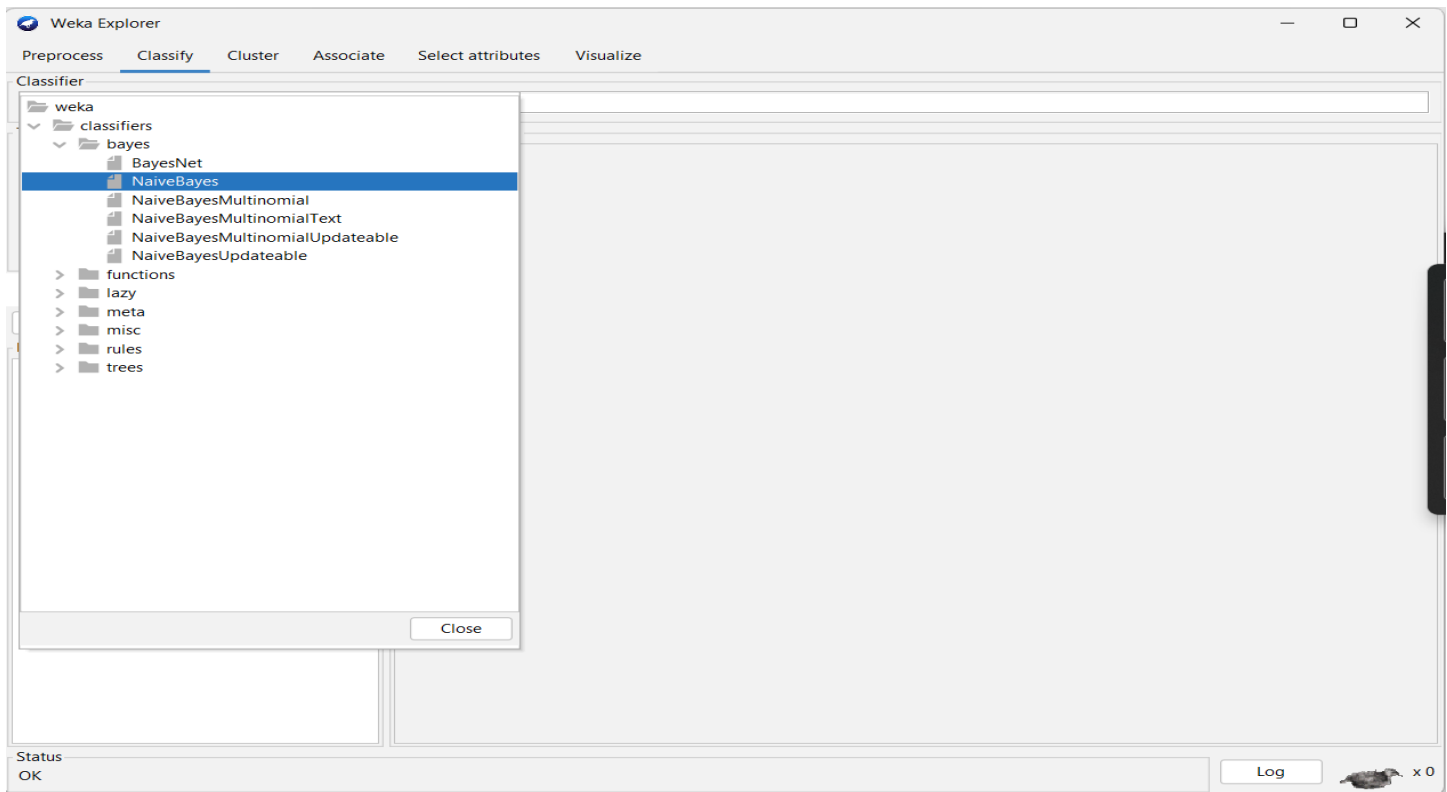
Classifier output

Status  
OK

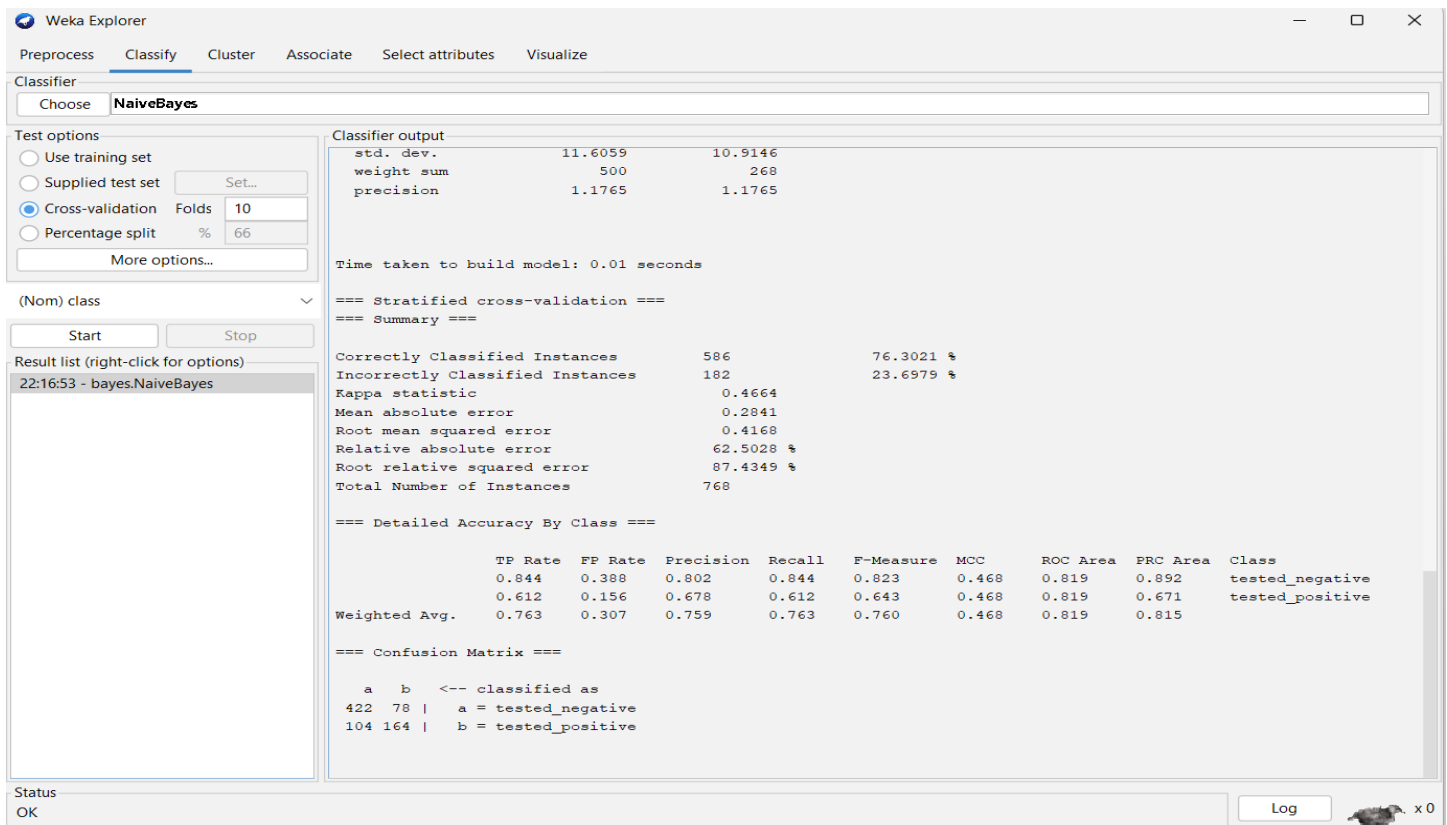
Log   x 0



## STEP 8: SELECT NAÏVE BAYES CLASSIFIER



## STEP 9: NOW CLICK ON CROSS VALIDATION AND CLICK ON START BUTTON



## RANDOM FOREST:

**Weka Explorer**

Preprocess **Classify** Cluster Associate Select attributes Visualize

Classifier

Choose **RandomForest** -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options

☐ Use training set

☐ Supplied test set

☒ Cross-validation Folds

☐ Percentage split %

(Nom) class

Result list (right-click for options)

19:47:19 - trees.RandomForest

Classifier output

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: 0.18 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	582	75.7813 %
Incorrectly Classified Instances	186	24.2188 %
Kappa statistic	0.4566	
Mean absolute error	0.3106	
Root mean squared error	0.4031	
Relative absolute error	68.3405 %	
Root relative squared error	84.5604 %	
Total Number of Instances	768	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.836	0.388	0.801	0.836	0.818	0.458	0.820	0.886	tested_negative
	0.612	0.164	0.667	0.612	0.638	0.458	0.820	0.679	tested_positive
Weighted Avg.	0.758	0.310	0.754	0.758	0.755	0.458	0.820	0.814	

=== Confusion Matrix ===

a	b	-- classified as
418	82	a = tested_negative
104	164	b = tested_positive

Status OK  x0

## K MEAN:

**Weka Explorer**

Preprocess **Cluster** Associate Select attributes Visualize

Clusterer

Choose **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 %

☐ Classes to clusters evaluation

(Nom) class

☒ Store clusters for visualization

Result list (right-click for options)

19:48:12 - SimpleKMeans

Clusterer output

kMeans

=====

Number of iterations: 4

Within cluster sum of squared errors: 149.5177664581119

Initial starting points (random):

Cluster 0: 1,126,56,29,152,28.7,0.801,21,tested\_negative

Cluster 1: 8,95,72,0,0,36.8,0.485,57,tested\_negative

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data (768.0)	Cluster# 0 (500.0)	1 (268.0)
preg	3.8451	3.298	4.8657
plas	120.8945	109.98	141.2575
pres	69.1055	68.184	70.8246
skin	20.5365	19.664	22.1642
insu	79.7995	68.792	100.3358
mass	31.5926	30.3042	35.1425
pedi	0.4719	0.4297	0.5505
age	33.2409	31.19	37.0672
class		tested_negative	tested_negative tested_positive

Time taken to build model (full training data) : 0.03 seconds

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

Clustered Instances

0	500 ( 65%)
1	268 ( 35%)

Status OK  x0

## STEP 10: NOW YOU CAN SEE THIS NOMINAL TEMPERATURE IN THE VISUALIZE WINDOW



## STEP 11: ON CLICKING ON ANY OF THE PLOT MATRIX

