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:

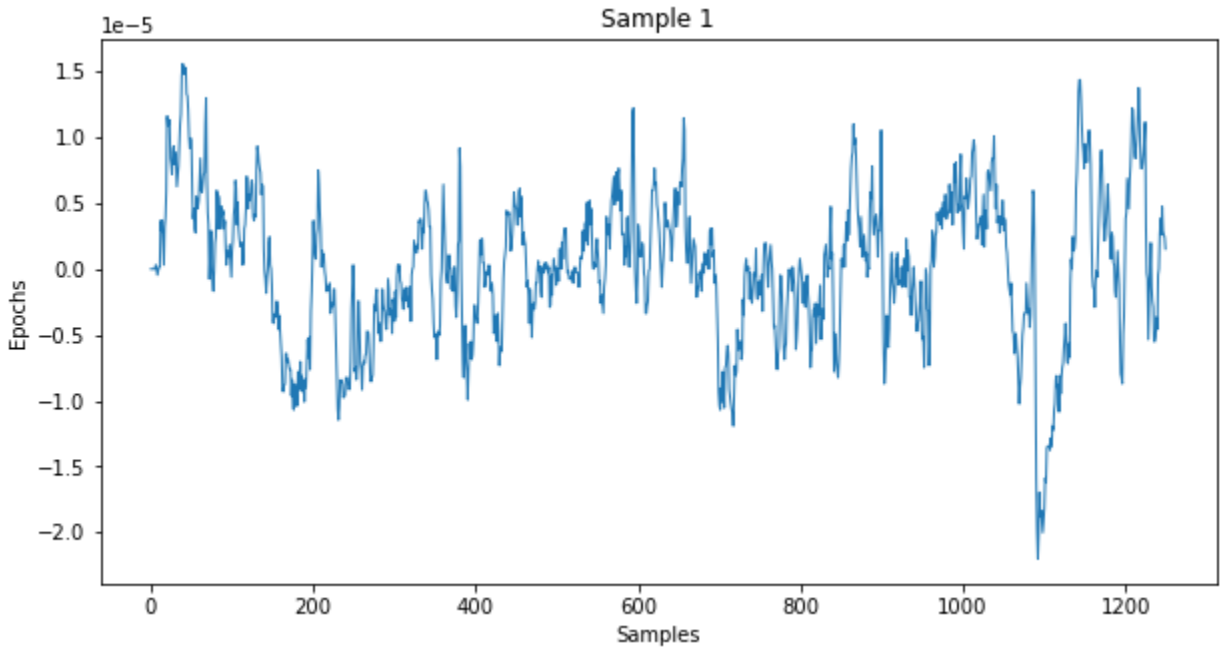
```python
from glob import glob
import numpy as np
import pandas as pd
import mne  # library used for extracting EEG
from matplotlib import pyplot as plt

# Get the list of .edf files in the specified directory
data_set = glob('D:/Machine Learning/Lab Task 2/dataverse_files/*.edf')

# Define a function to extract data from our EEG signal
def read_data(file_path):
    # Reading our raw EEG data
    data = mne.io.read_raw_edf(file_path, preload=True)
    # Creating fixed length epochs of 5 seconds with 1 second overlap
    epochs = mne.make_fixed_length_epochs(data, duration=5, overlap=1)
    # Extracting data in the form of an array
    array = epochs.get_data()
    # Taking out the individual max value and multiplying by 1e6 for efficient plotting
    return max(array[0][0] * 1000000)

# Apply the read_data function to each .edf file and store the results in a list
data_array = [read_data(i) for i in data_set]

# Creating a DataFrame using pandas
df = pd.DataFrame()
# Appending our data into the DataFrame
df['EEG_epochs'] = data_array

# Reading our source file using pandas
df2 = pd.read_csv('Sor.csv')
# Appending our ANX range into the DataFrame
df['ANX'] = df2['ANX']
# Converting the DataFrame into a CSV file
df.to_csv('epo2.csv')

```

```
# Signal Representation
# Reading raw EEG data from the first .edf file
raw = mne.io.read_raw_edf(data_set[0])
# Creating fixed length epochs of 5 seconds with 1 second overlap
epochs1 = mne.make_fixed_length_epochs(raw, duration=5, overlap=1)
# Extracting data in the form of an array
arr = epochs1.get_data()

# Plotting the data
pd.Series(arr[0][0]).plot(figsize=(10, 5), lw=1, title='Sample 1')
plt.xlabel('Samples')
plt.ylabel('Epochs')
# Printing the data of the first epoch
print(arr[0][0])
# Displaying the plot
plt.show()

# Printing the DataFrame
print(df)

```

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

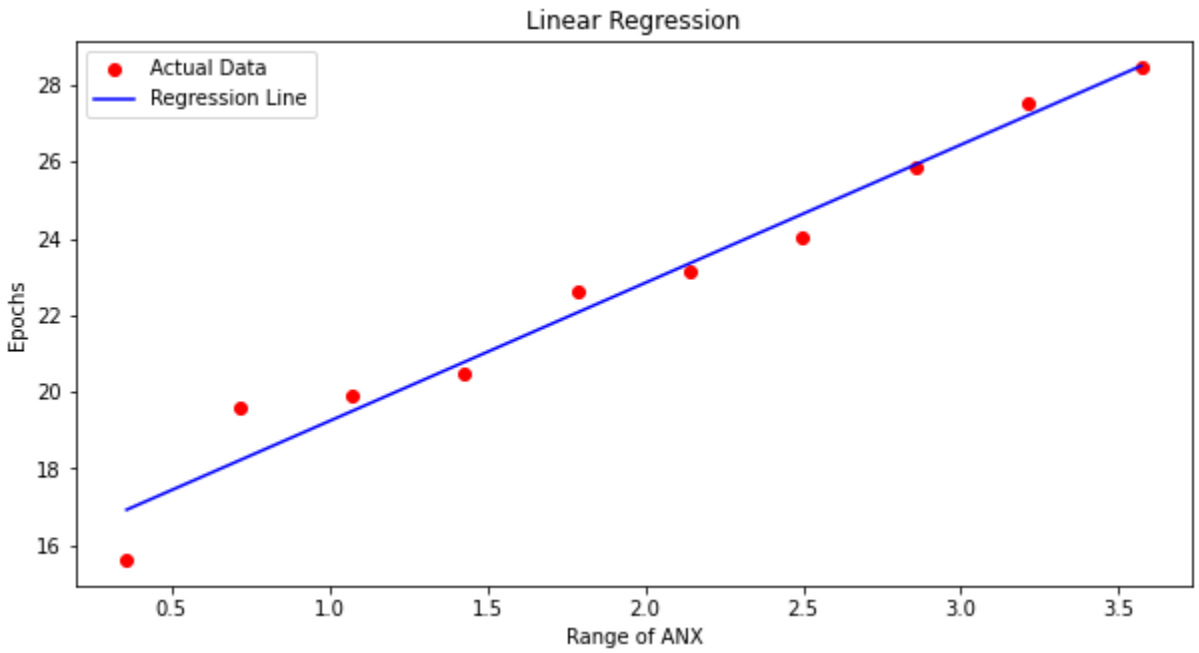
OUTPUT WAVEFORM:



PROGRAM:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
# library used for Machine learning, we are using this for Linear Regression
import warnings

warnings.filterwarnings("ignore")  # Ignore warnings for a cleaner output

# Read the CSV file
df = pd.read_csv('epo2.csv')  # Load data from a CSV file into a pandas DataFrame

# Define a linear regression model
reg = linear_model.LinearRegression()  # Initialize a Linear Regression model

# Fit the model using ANX as the predictor and EEG_epochs as the response variable
reg.fit(df[['ANX']], df[['EEG_epochs']])
 # Train the model with 'ANX' as the input and 'EEG_epochs' as the output

# Predicting an EEG_epochs value by providing a random ANX value
predicted_value = reg.predict([[4.2]])
# Use the trained model to predict 'EEG_epochs' for ANX value of 4.2
print(predicted_value)  # Print the predicted value

# Plotting our Linear regression graph
plt.figure(figsize=(10, 5))  # Set the figure size for the plot
plt.title('Linear Regression')  # Set the title of the plot
plt.xlabel('Range of ANX')  # Set the x-axis label
plt.ylabel('Epochs')  # Set the y-axis label

# Scatter plot of the actual data
plt.scatter(df['ANX'], df['EEG_epochs'], color='red', label='Actual Data')
 # Plot the actual data points in red

# Line plot of the regression line
plt.plot(df['ANX'], reg.predict(df[['ANX']]), color='blue', label='Regression Line')
# Plot the regression line in blue

plt.legend()  # Add a legend to the plot
plt.show()  # Display the plot

```

WAVEFORM:


Linear Regression
Actual Data
Regression Line
Epochs
Range of ANX
16
18
20
22
24
26
28
0.5
1.0
1.5
2.0
2.5
3.0
3.5

TASK 2b - ANALYSIS OF MACHINE LEARNING TOOL

TOOLS USED

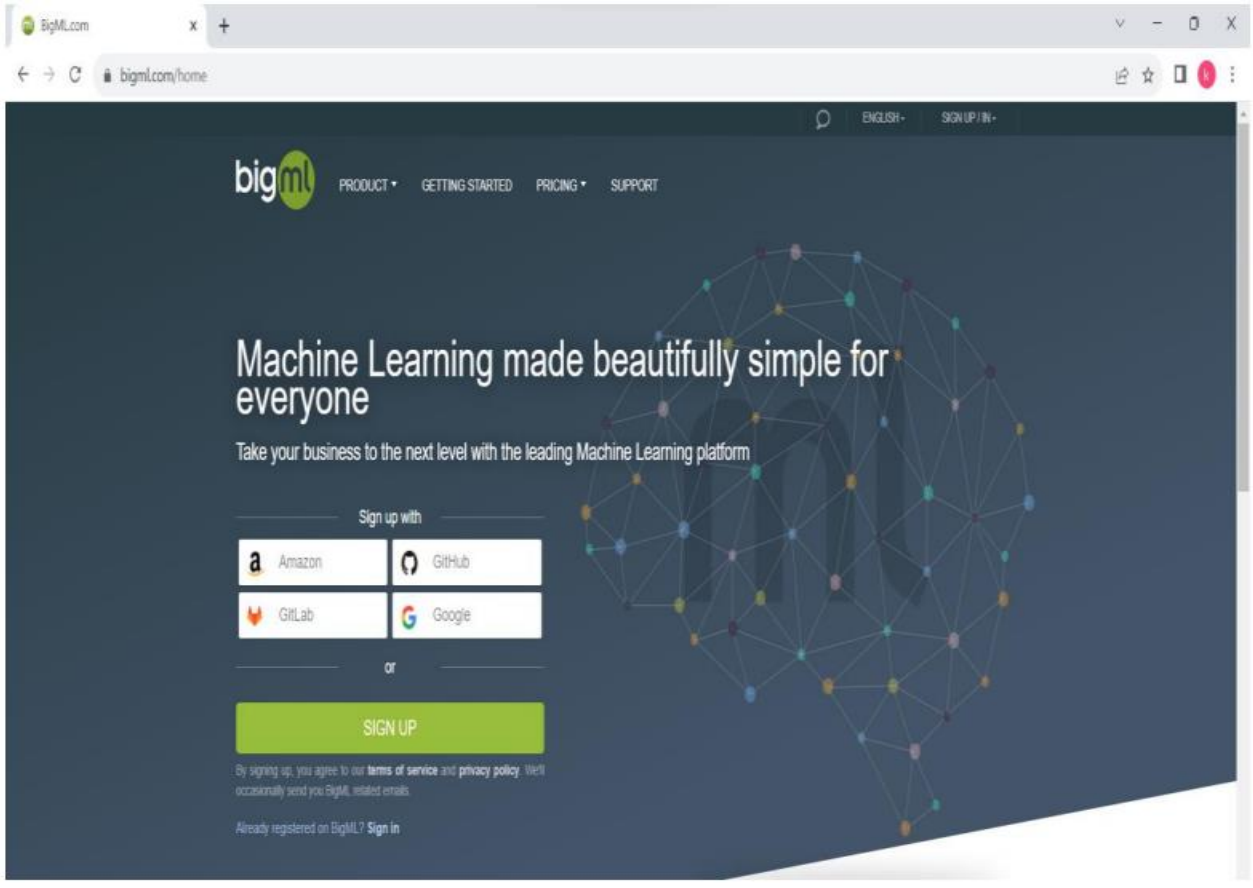
1.BIGML 2. WEKA

INTRODUCTION

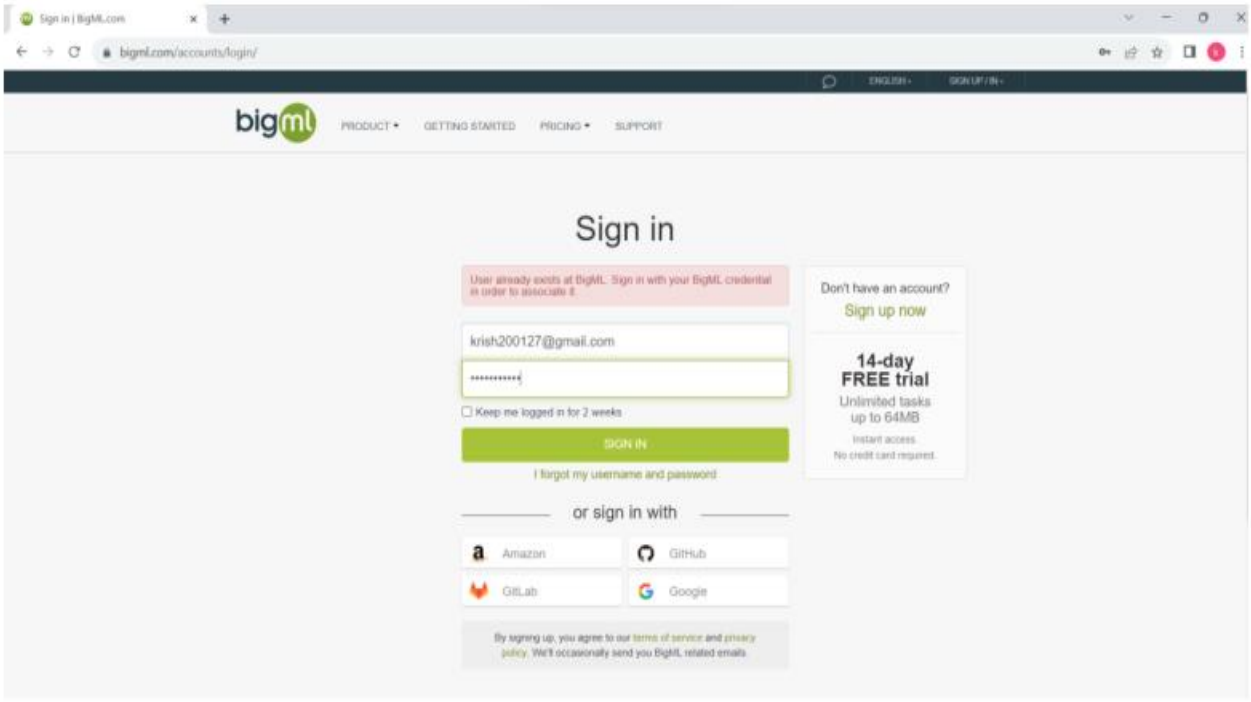
BigML is a consumable, programmable, and scalable Machine Learning platform that makes it easy to solve and automate Classification, Regression, Time Series Forecasting, Cluster Analysis, Anomaly Detection, Association Discovery, and Topic Modelling tasks.

In this tool I have used various machine learning algorithms to study the dataset. The dataset was downloaded from www.kaggle.com

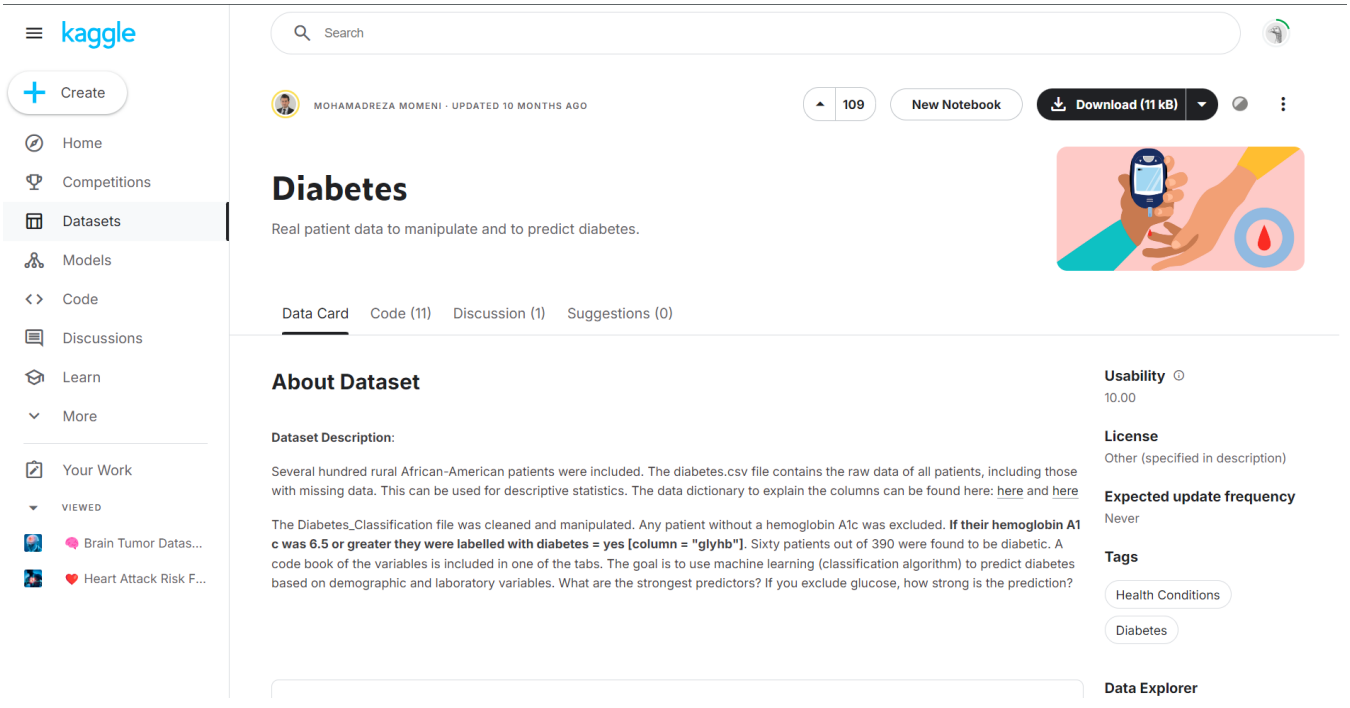
STEP 1: OPEN BIGML WEBSITE FROM ANY BROWSER. THIS IS ONLINE SOFTWARE INTERFACE



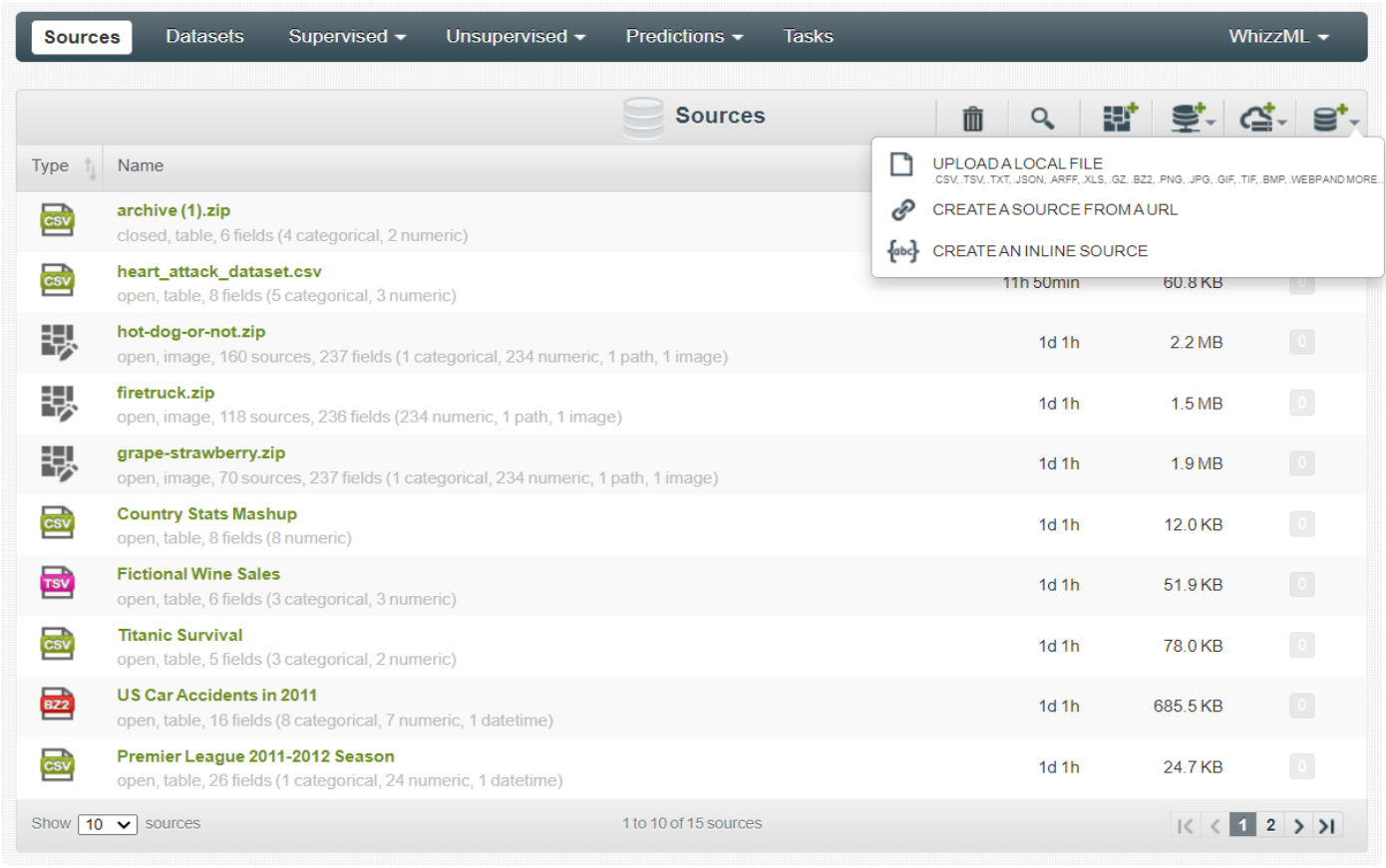
STEP 2: SIGN UP WITH YOUR GOOGLE ACCOUNT OR AMAZON, GITHUB ETC



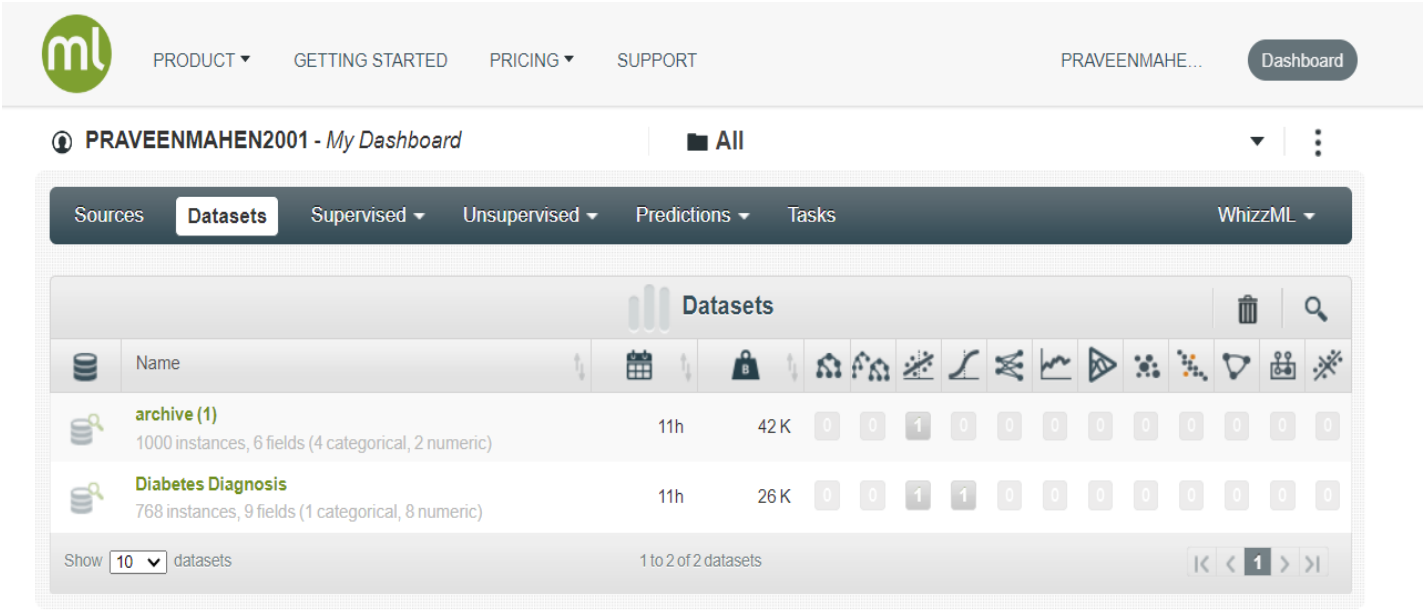
STEP 3: DOWNLOAD ANY DATA SET FROM KAGGLE



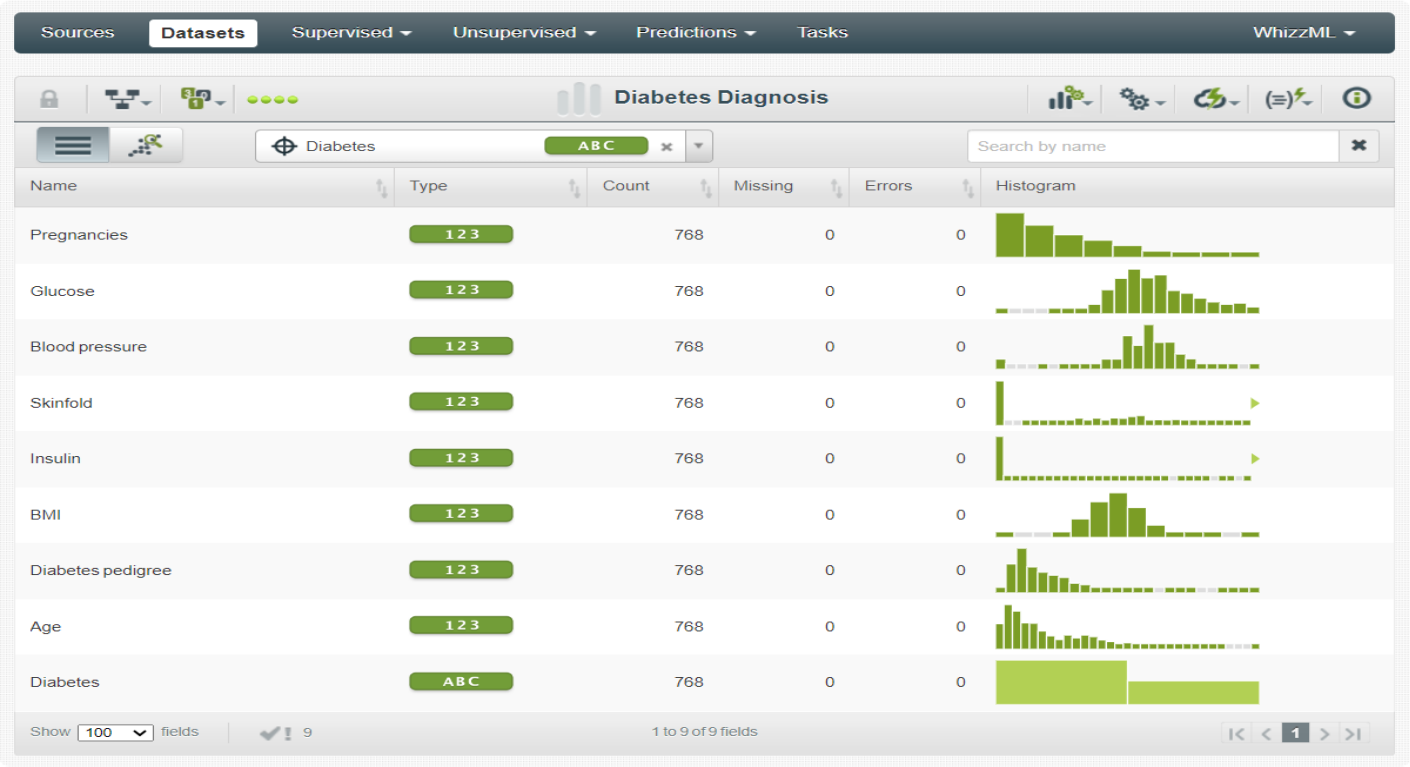
STEP 4: UPLOAD YOUR DATASET IN BIGML



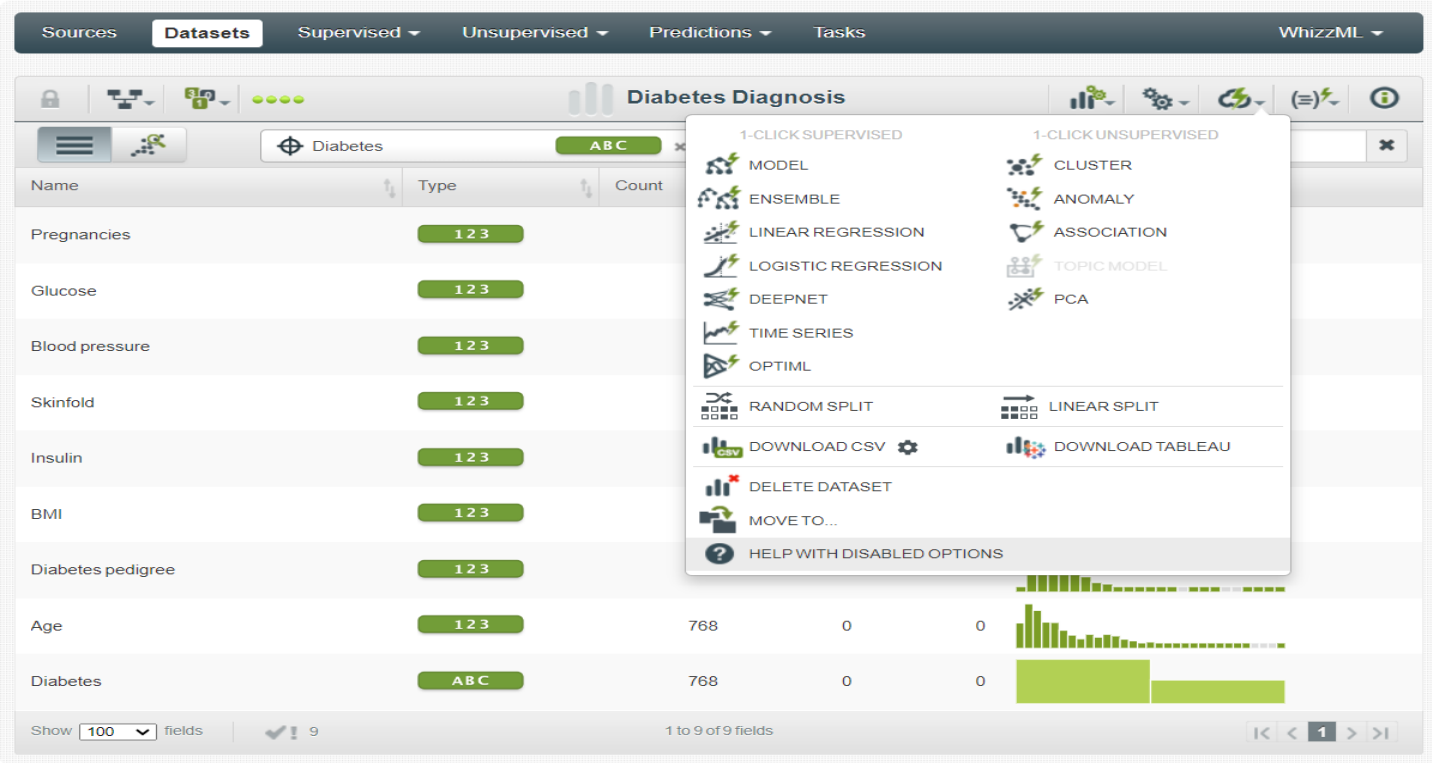
STEP 4: NOW YOUR DATASET WILL BE VISIBLE LIKE THIS



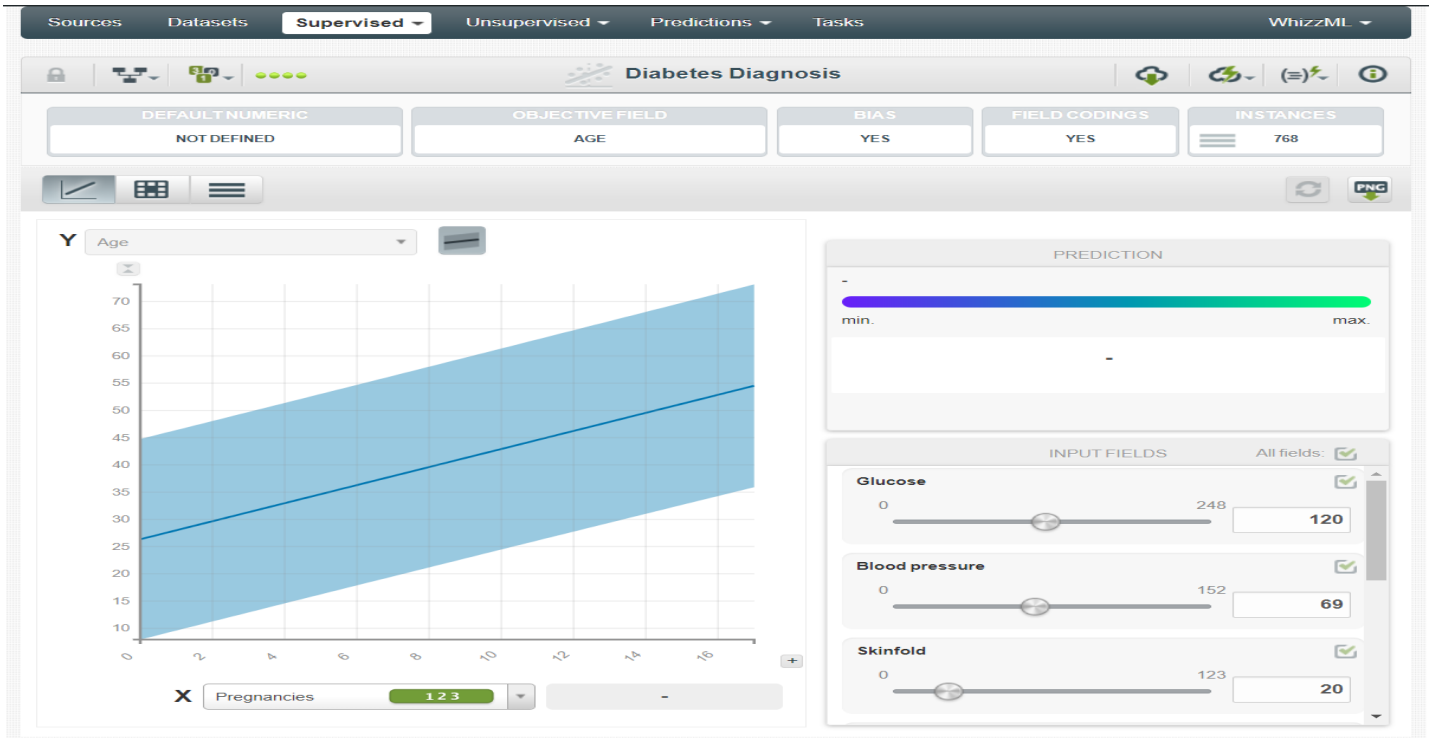
STEP 5: FULL DETAILED REPRESENTATION OF DATASET CAN BE REPRESENTED



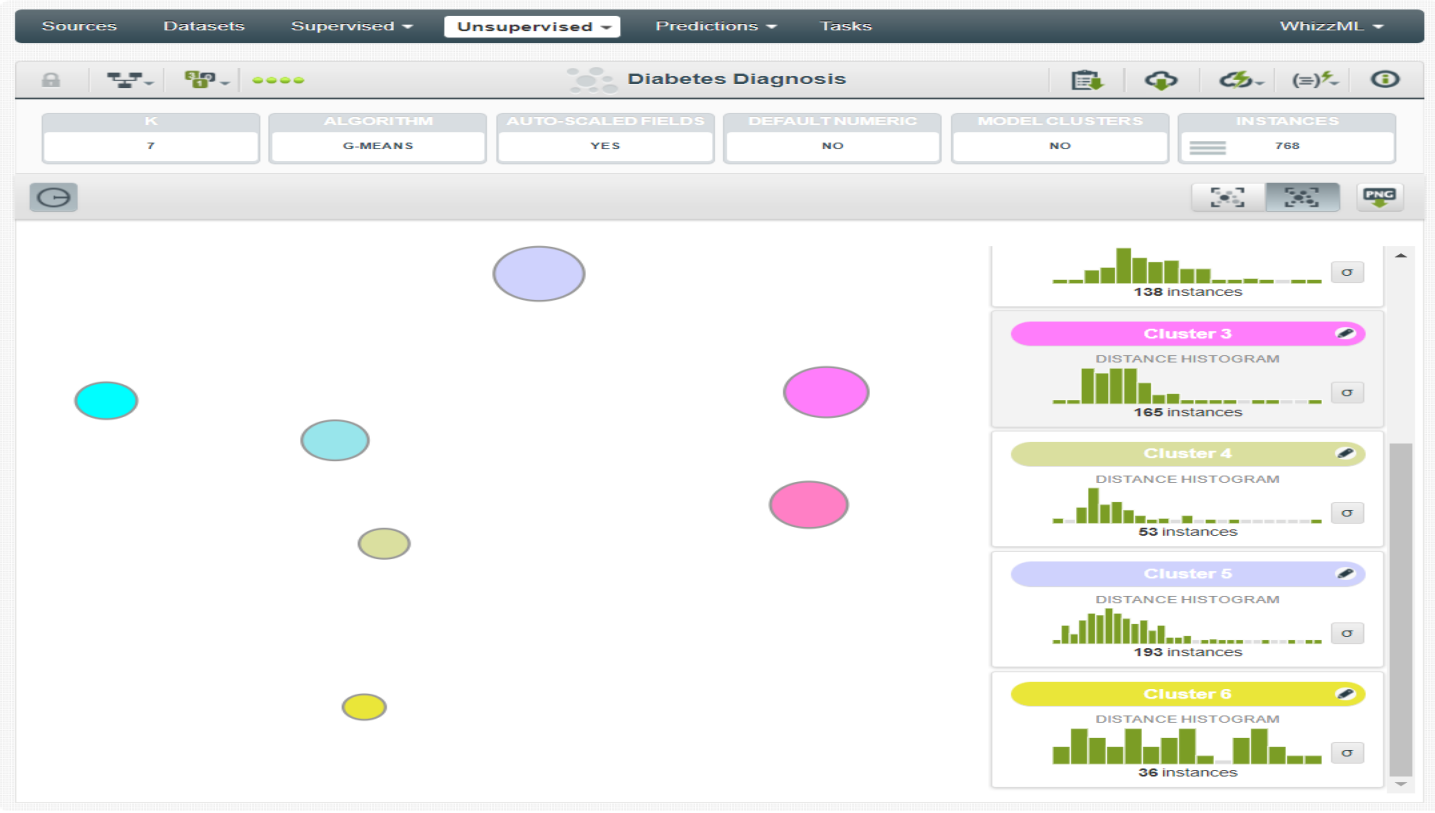
STEP 6: HERE YOU CAN SEE SUPERVISED AND UNSUPERVISED CLASSIFIER



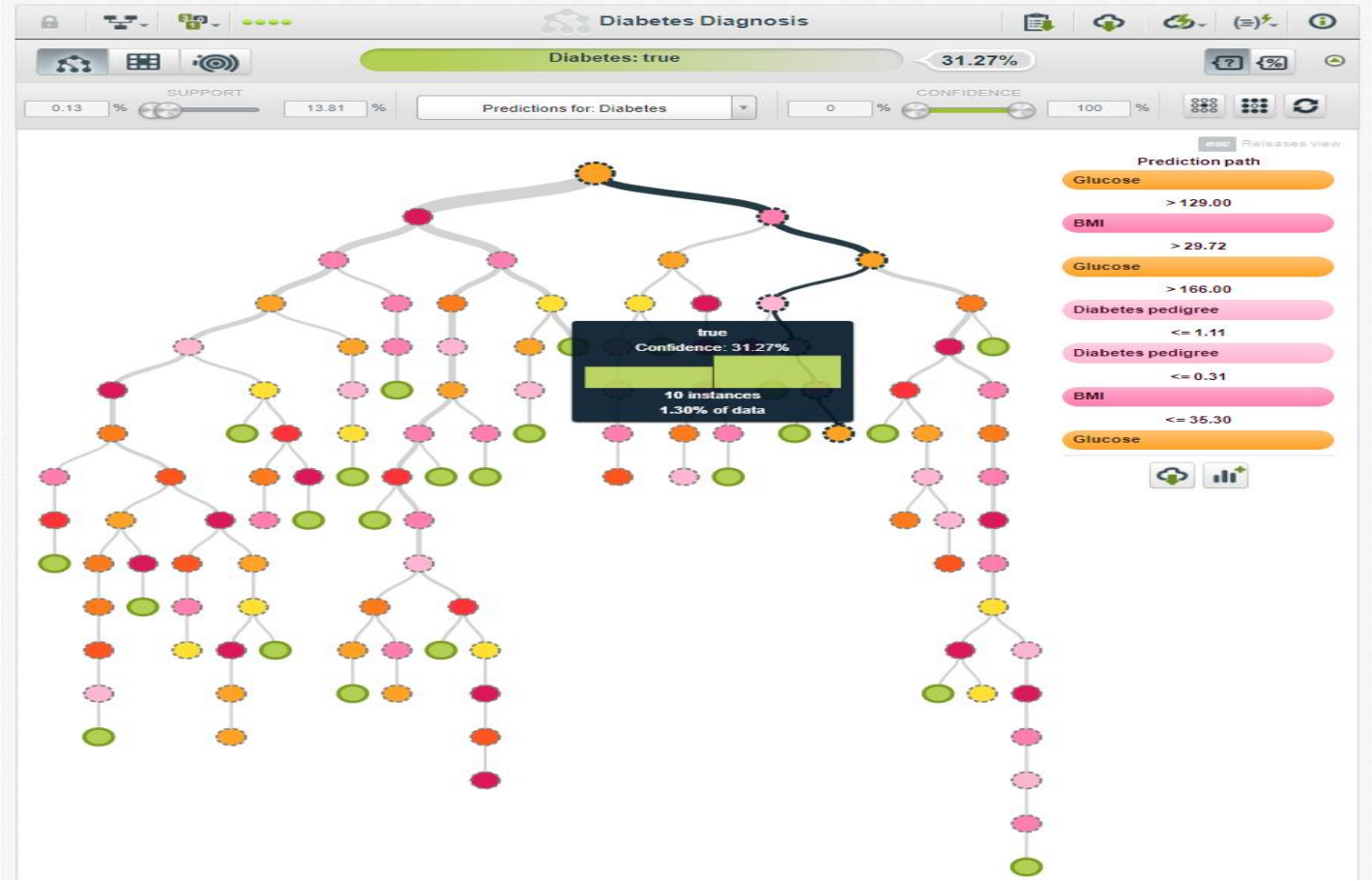
STEP 7: YOU CAN SEE LINEAR REGRESSION CURVE FOR GIVEN DATASET



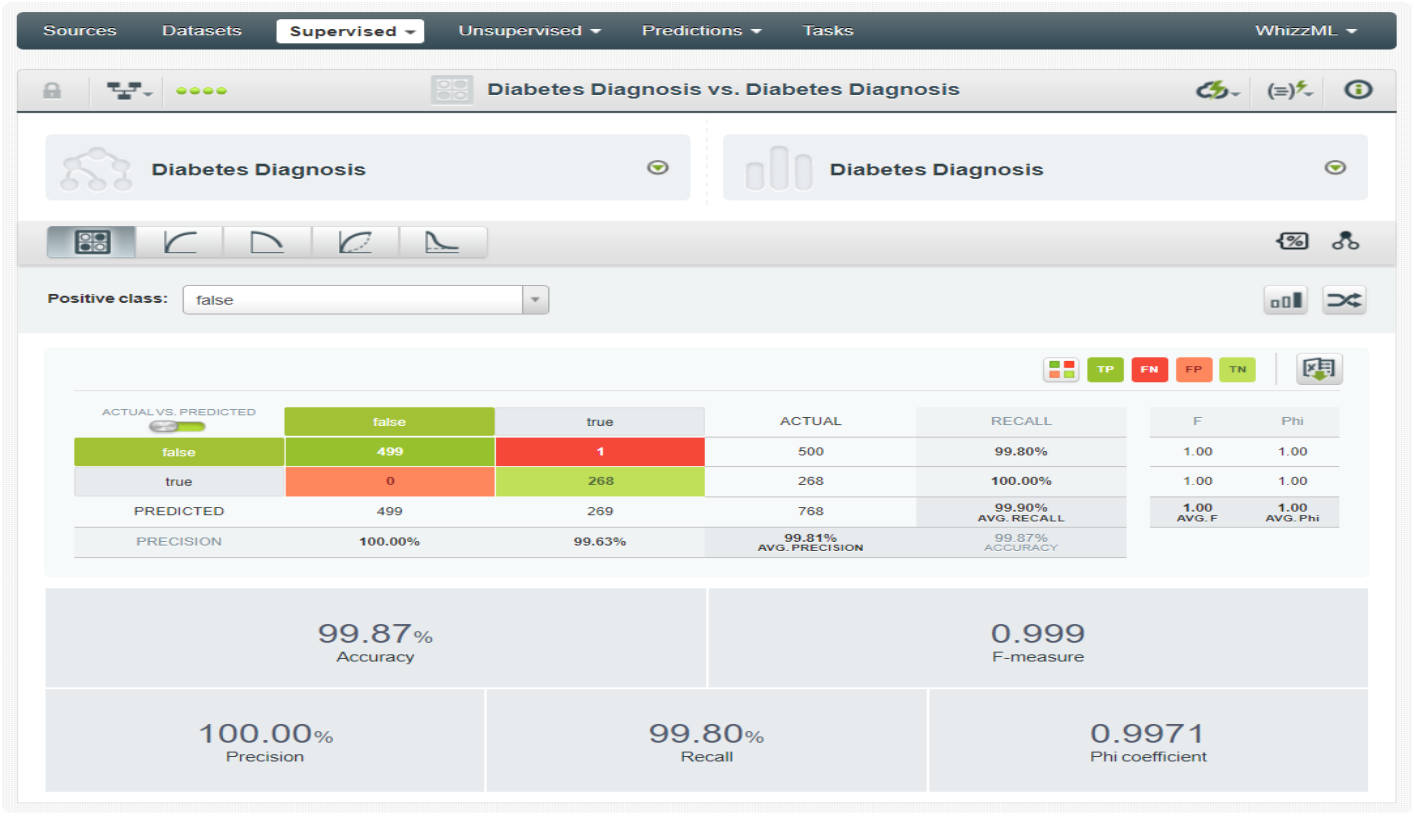
STEP 8: THIS BELOW IMAGE SHOWS UNSUPERVISED KMEANS CLUSTERING ALGORITHM FOR THE FOLLOWING DATASET



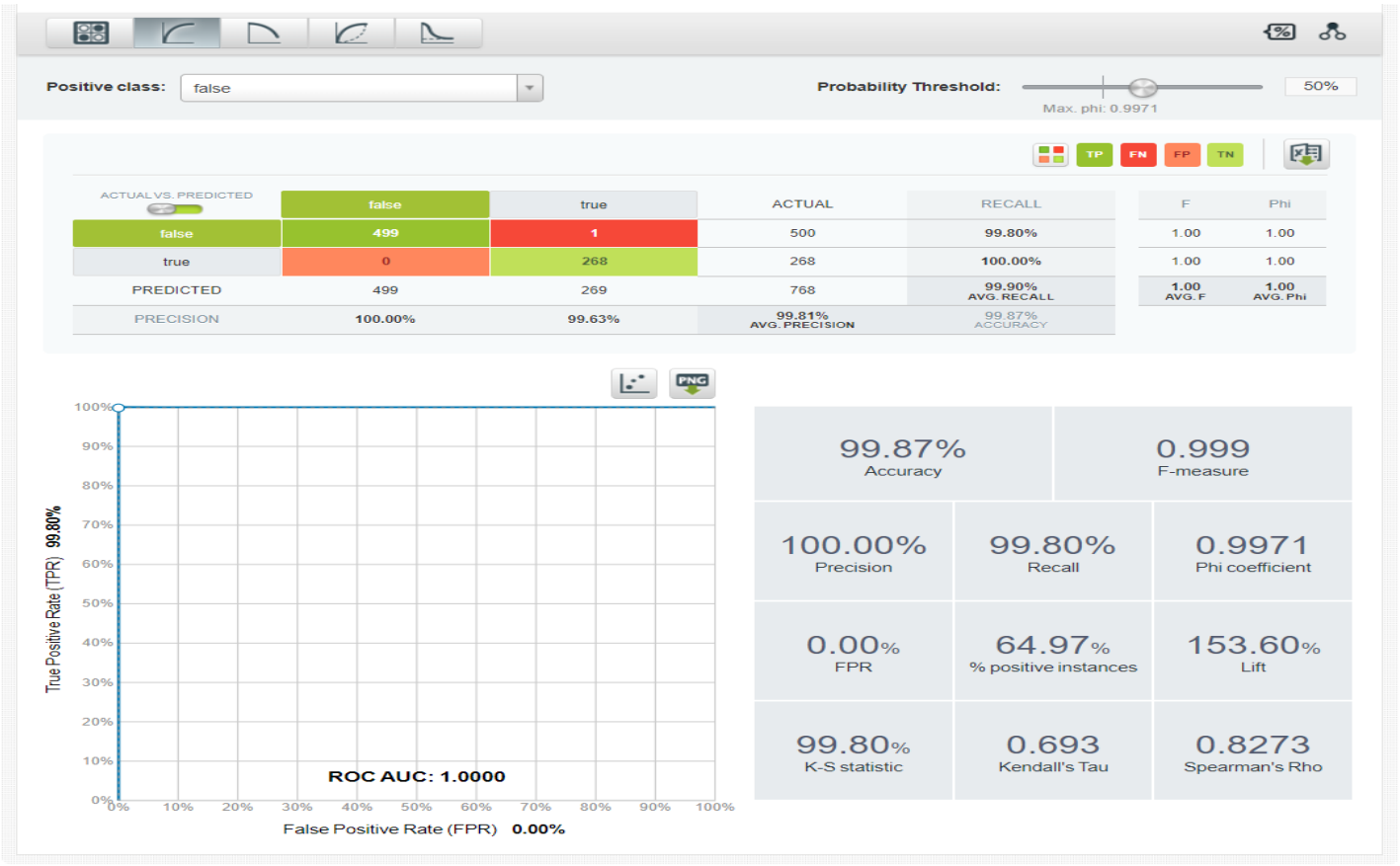
MODEL:



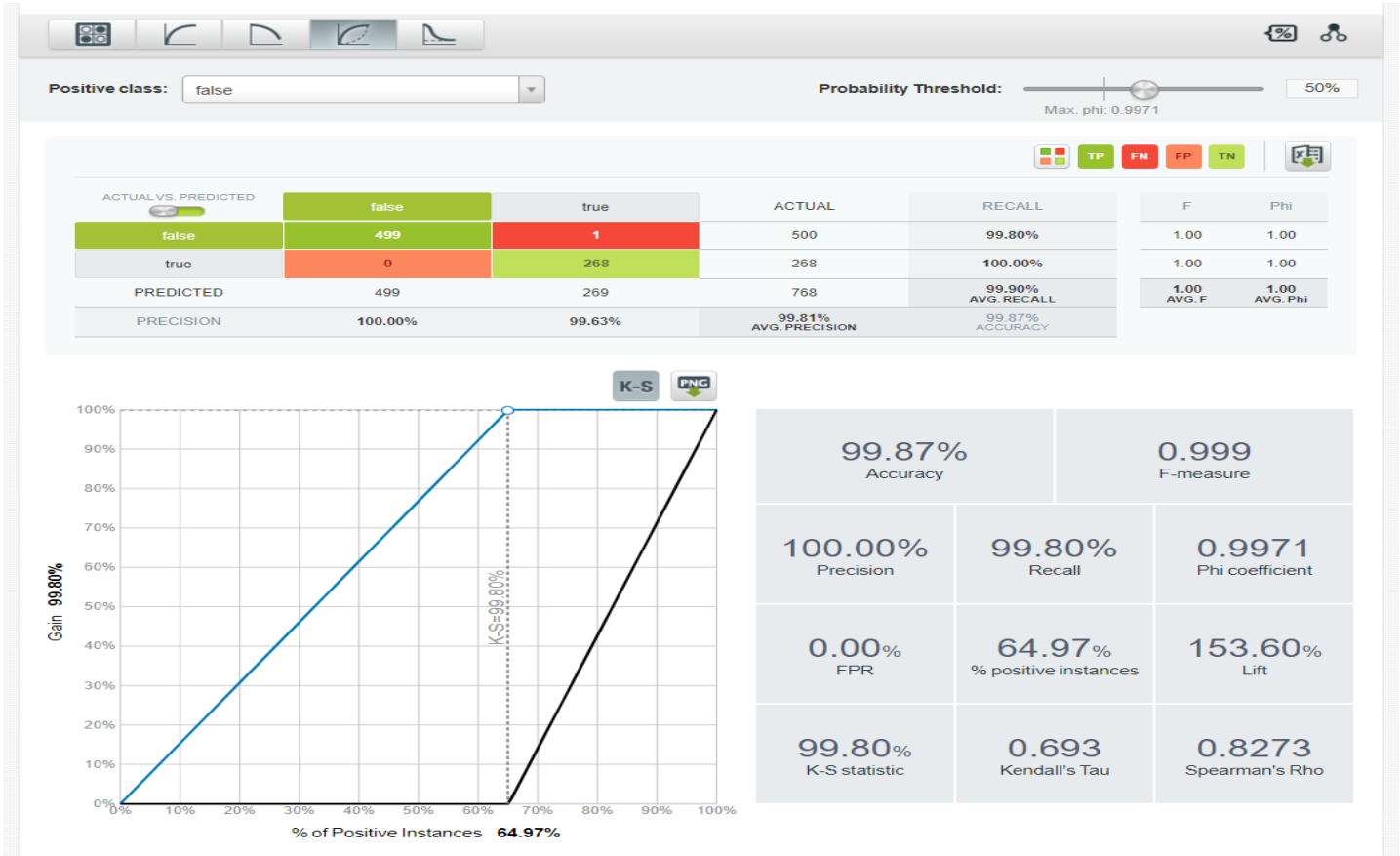
EVALUATION:



ROC:


Positive class: false
Probability Threshold: 50%
Max. phi: 0.9971
TP FN FP TN
ACTUAL VS. PREDICTED
false true ACTUAL RECALL F Phi
false 499 1 500 99.80% 1.00 1.00
true 0 268 268 100.00% 1.00 1.00
PREDICTED 499 269 768 99.90% AVG. RECALL 1.00 AVG. F 1.00 AVG. Phi
PRECISION 100.00% 99.63% 99.81% AVG. PRECISION 99.87% ACCURACY
PNG
True Positive Rate (TPR) 99.80%
ROC AUC: 1.0000
False Positive Rate (FPR) 0.00%
99.87% Accuracy
0.999 F-measure
100.00% Precision
99.80% Recall
0.9971 Phi coefficient
0.00% FPR
64.97% % positive instances
153.60% Lift
99.80% K-S statistic
0.693 Kendall's Tau
0.8273 Spearman's Rho

K MEANS:

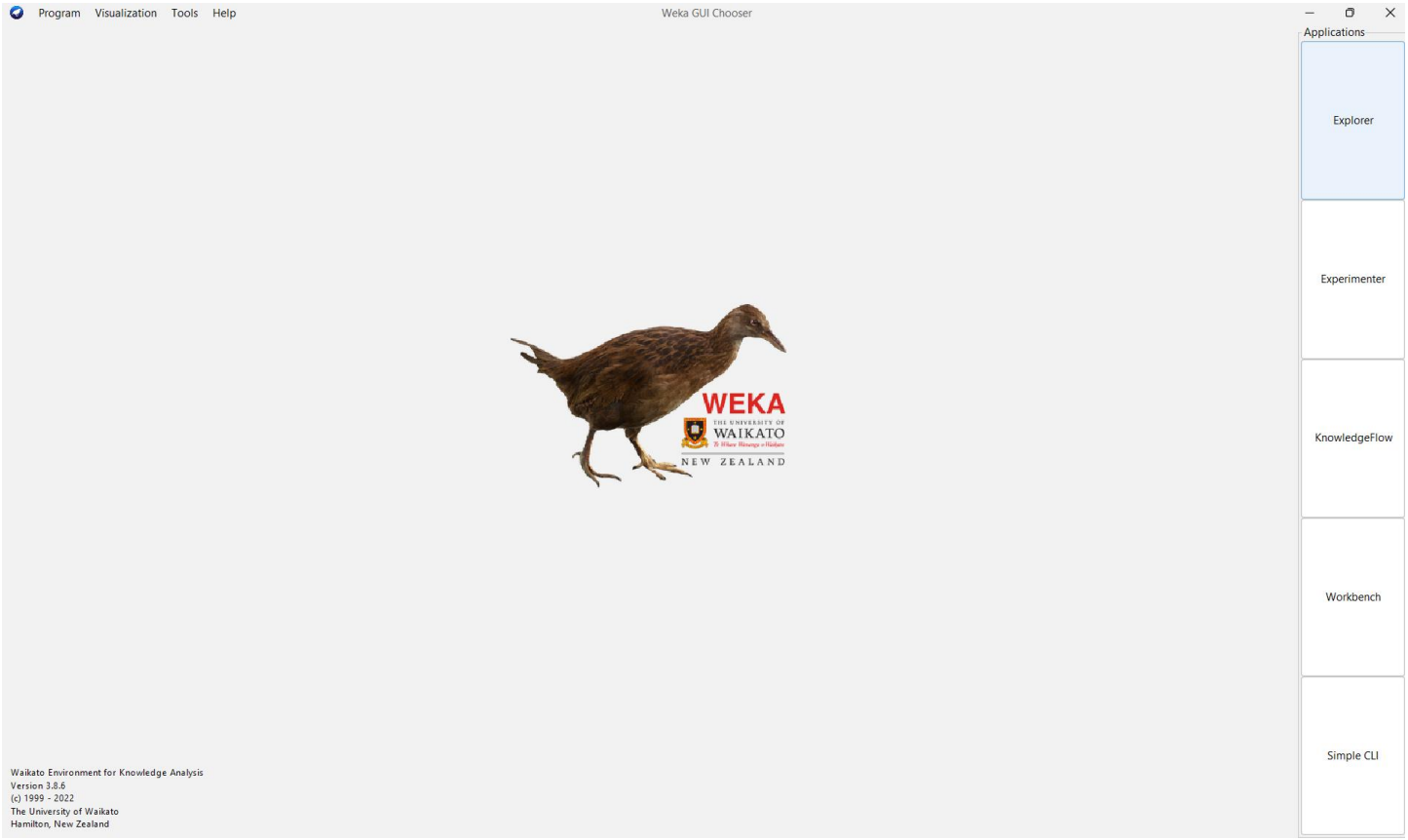

Positive class: false
Probability Threshold: 50%
Max. phi: 0.9971
TP FN FP TN
ACTUAL VS. PREDICTED
false true ACTUAL RECALL F Phi
false 499 1 500 99.80% 1.00 1.00
true 0 268 268 100.00% 1.00 1.00
PREDICTED 499 269 768 99.90% AVG. RECALL 1.00 AVG. F 1.00 AVG. Phi
PRECISION 100.00% 99.63% 99.81% AVG. PRECISION 99.87% ACCURACY
K-S PNG
Gain 99.80%
K-S=99.80%
% of Positive Instances 64.97%
99.87% Accuracy
0.999 F-measure
100.00% Precision
99.80% Recall
0.9971 Phi coefficient
0.00% FPR
64.97% % positive instances
153.60% Lift
99.80% K-S statistic
0.693 Kendall's Tau
0.8273 Spearman's Rho

TOOL 2: WEKA

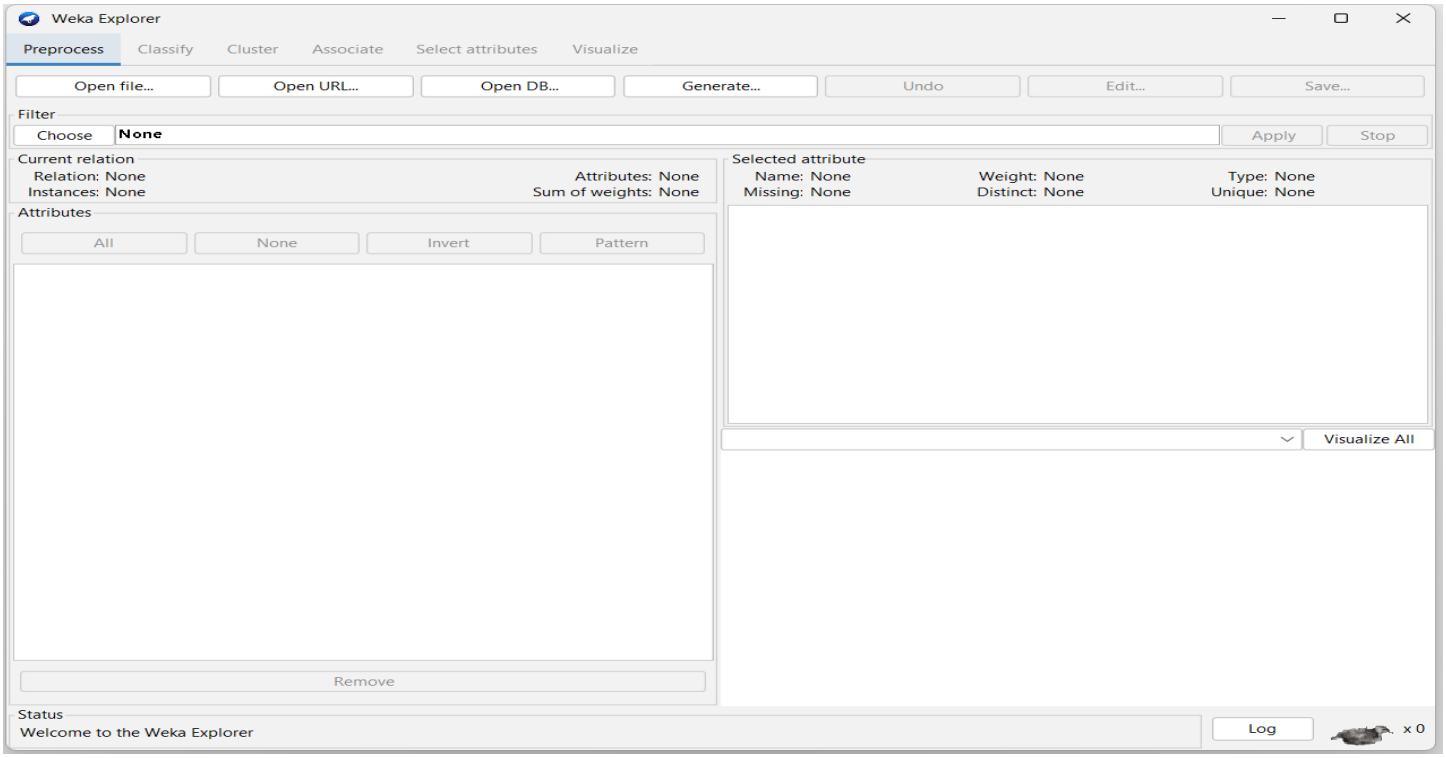
INTRODUCTION

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.

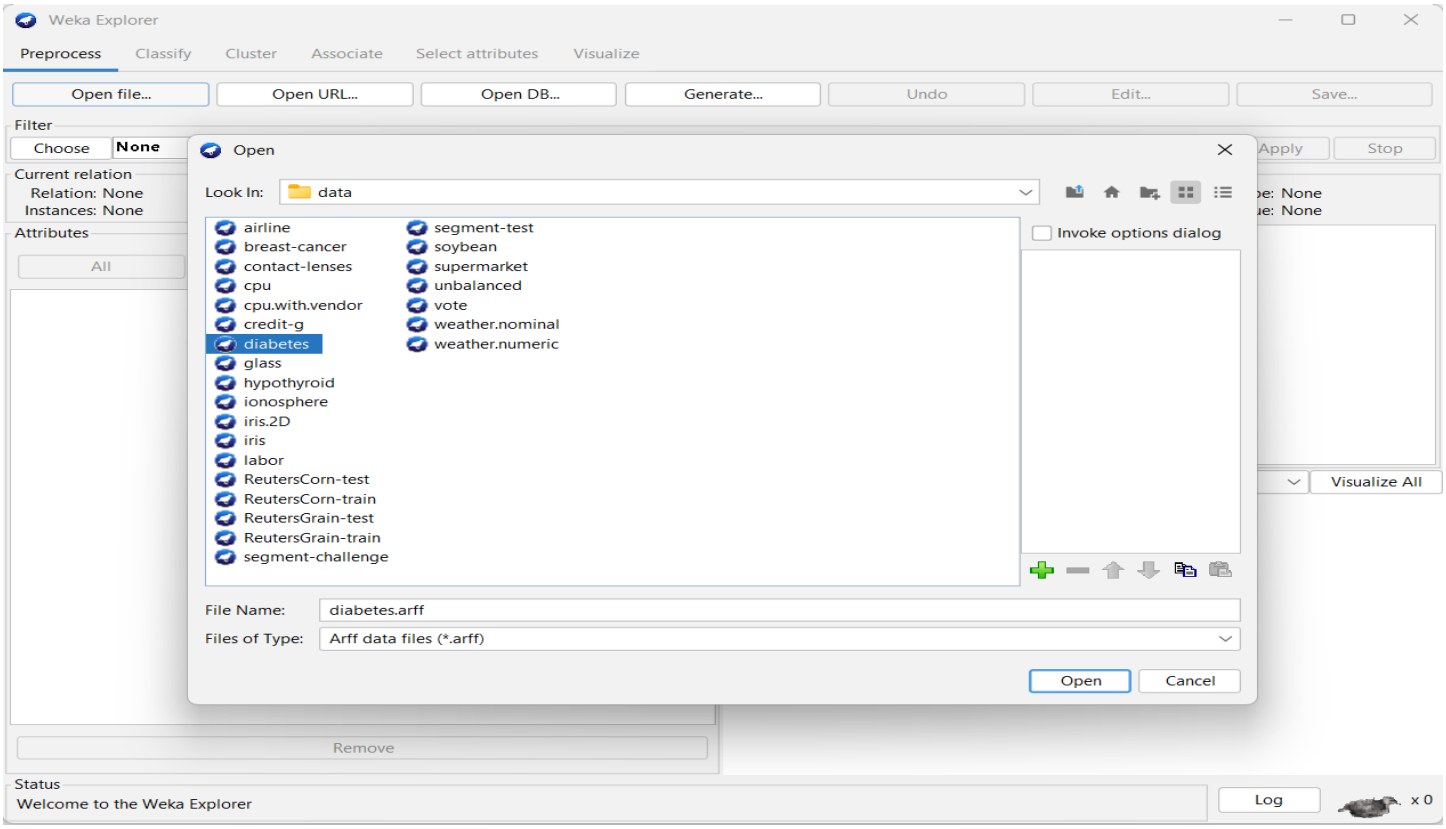
STEP 1: DOWNLOAD AND INSTALL THE WEKA SOFTWARE FROM THE GOOGLE



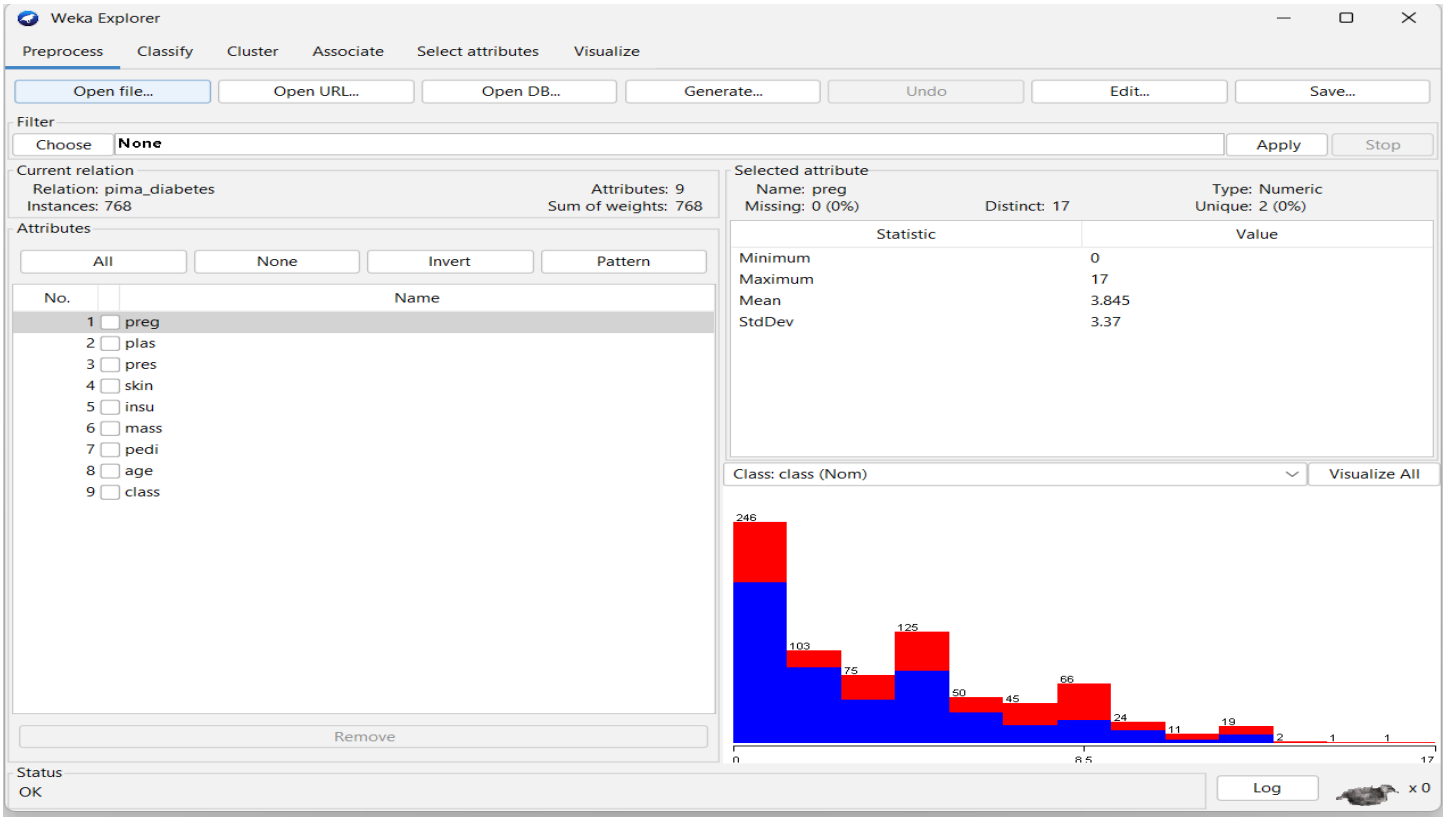
STEP 2: CLICK ON THE EXPLORER YOU CAN SEE THIS WINDOW



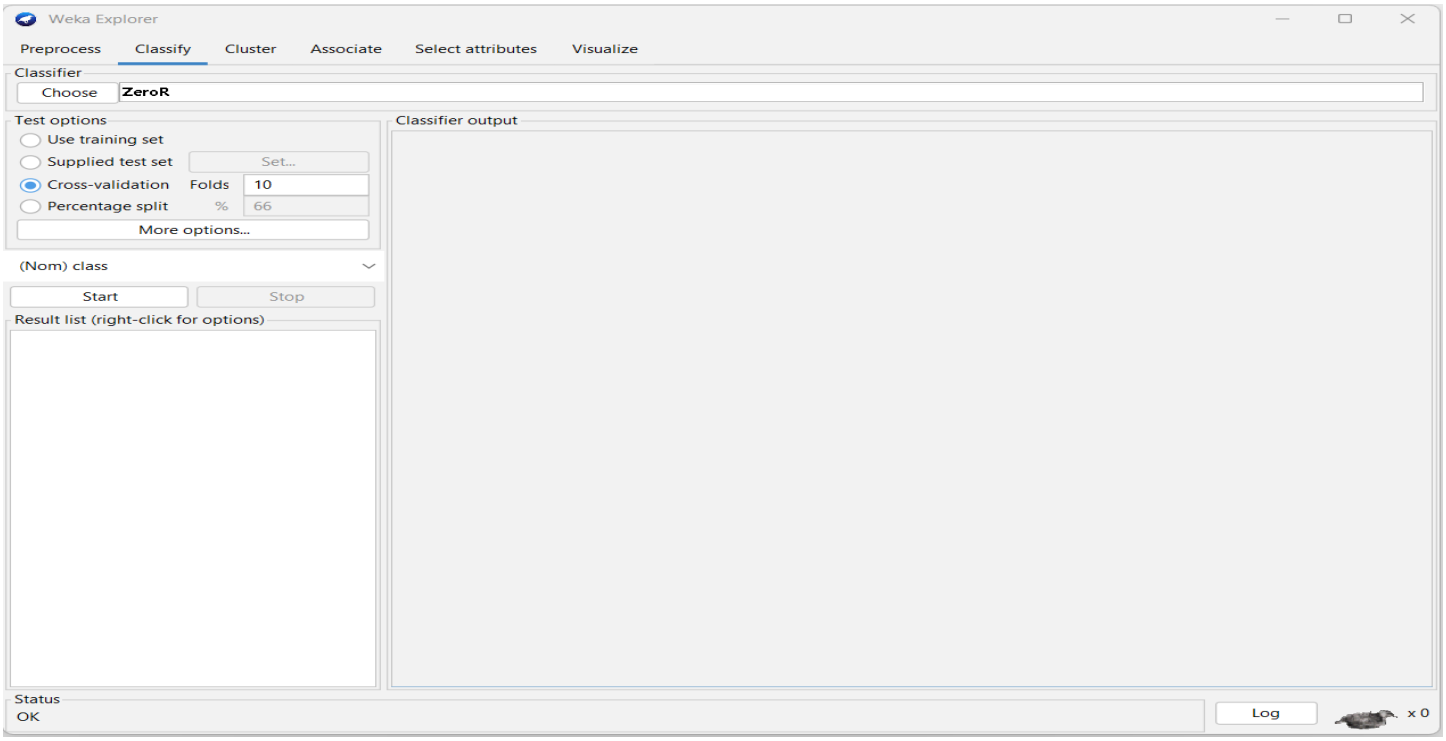
STEP 4: INSIDE THE DATA FOLDER YOU CAN SEE DIFFERENT TYPES OF INBUILT DATASETS YOU CAN CHOOSE ANY ONE OF IT



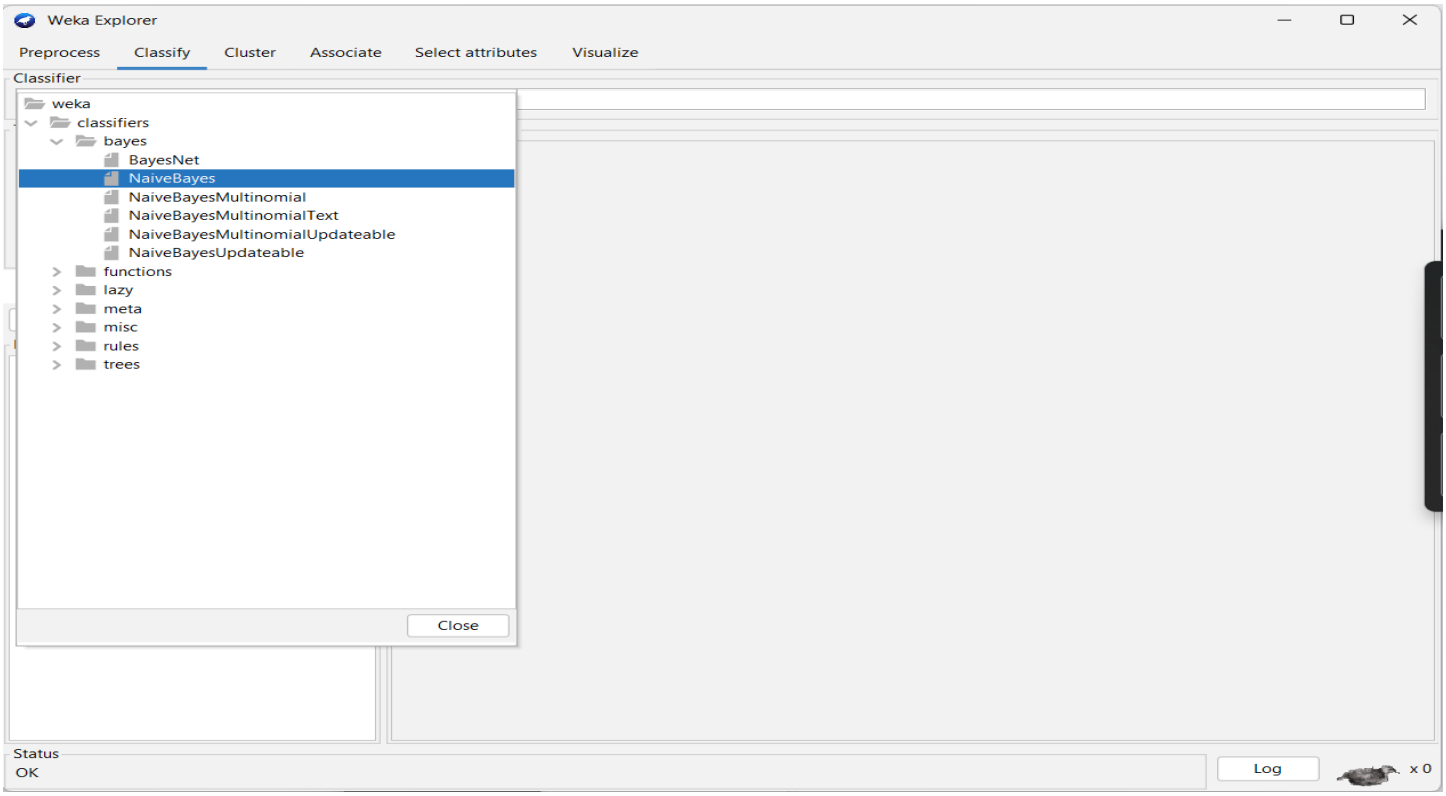
STEP 5: DATASET AND PREPROCESSED USING NAÏVE BAYES CLASSIFIER



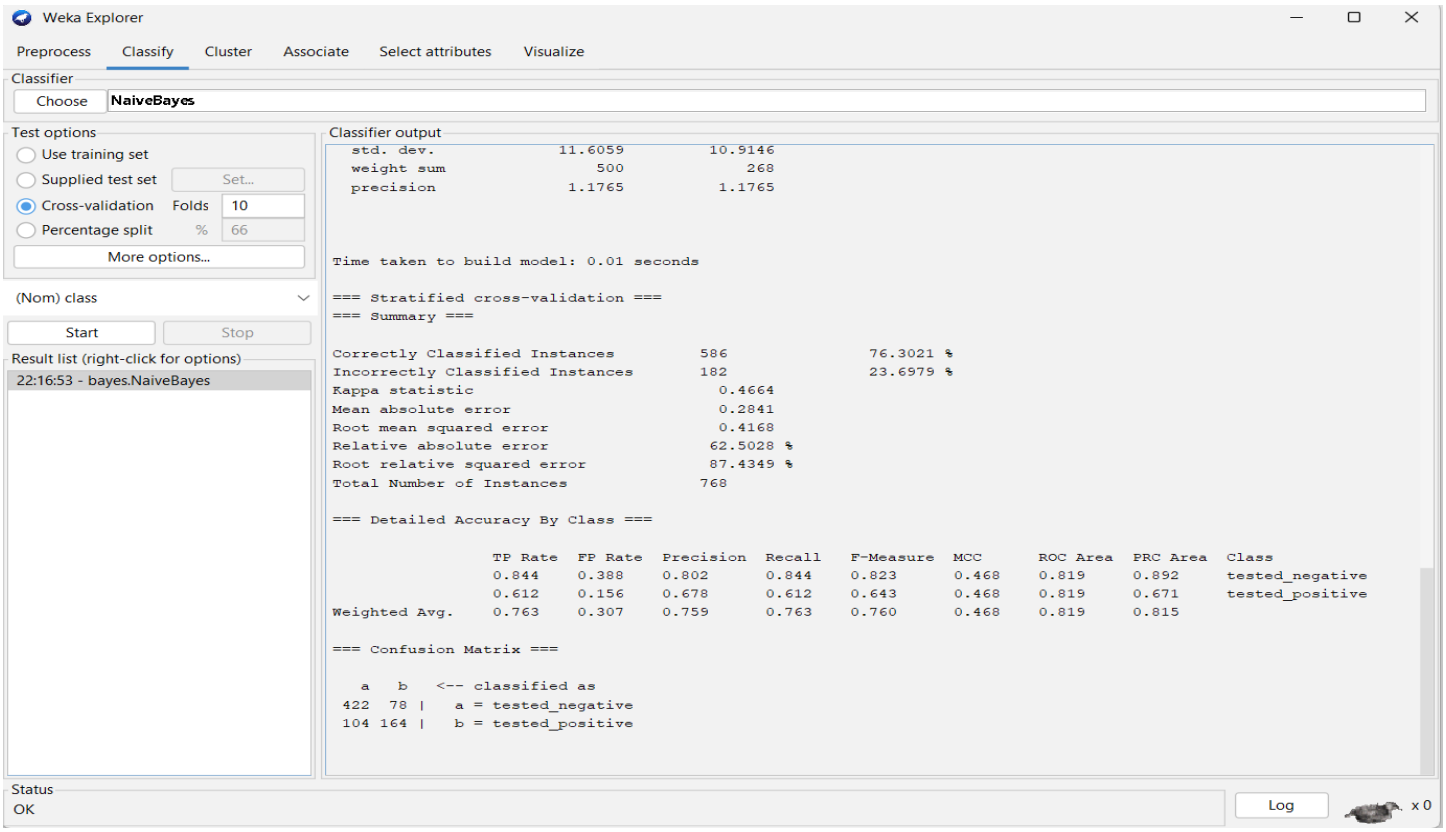
STEP 6: NOW GO TO CLASSIFY IN THE MENU



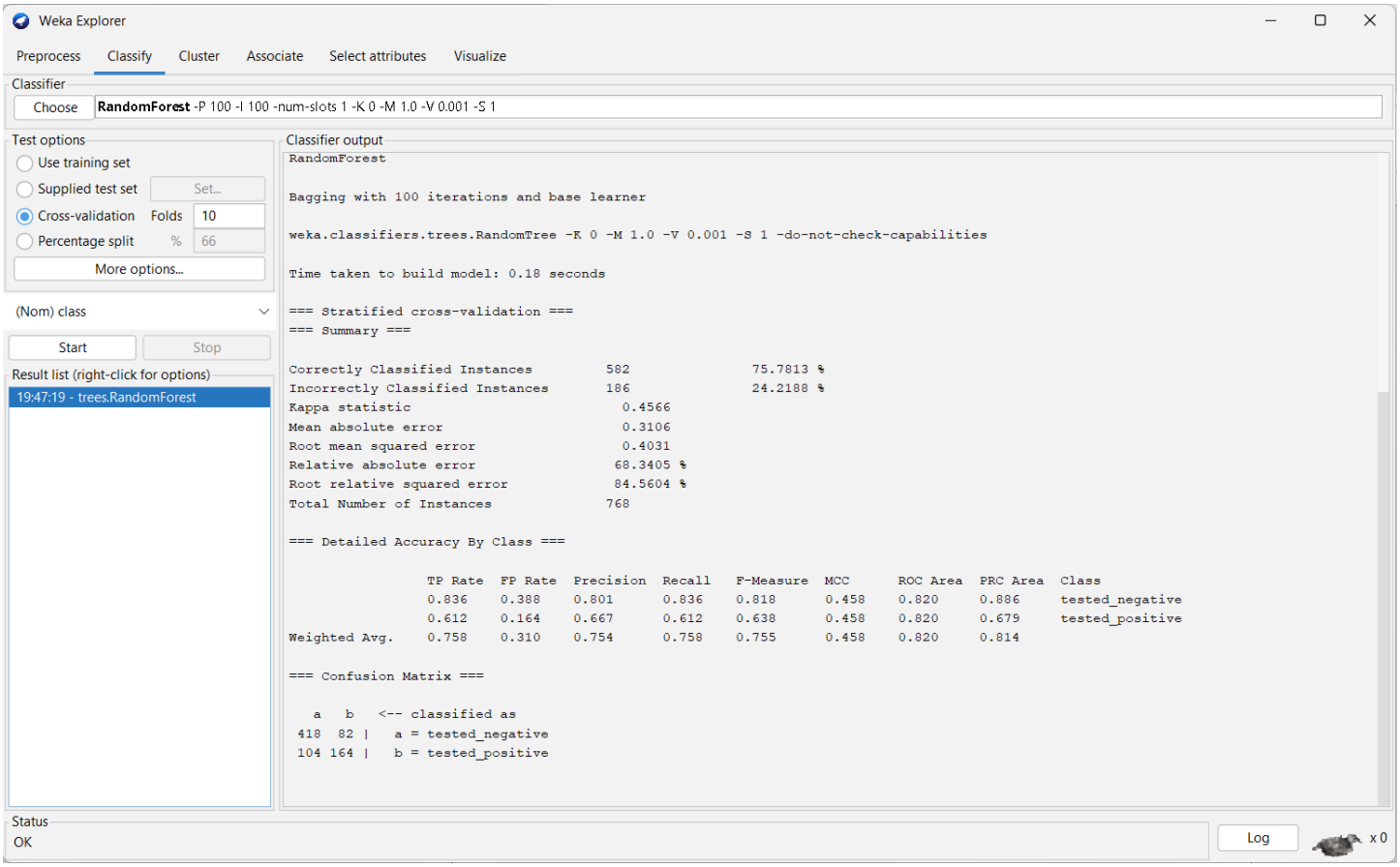
STEP 8: SELECT NAÏVE BAYES CLASSIFIER



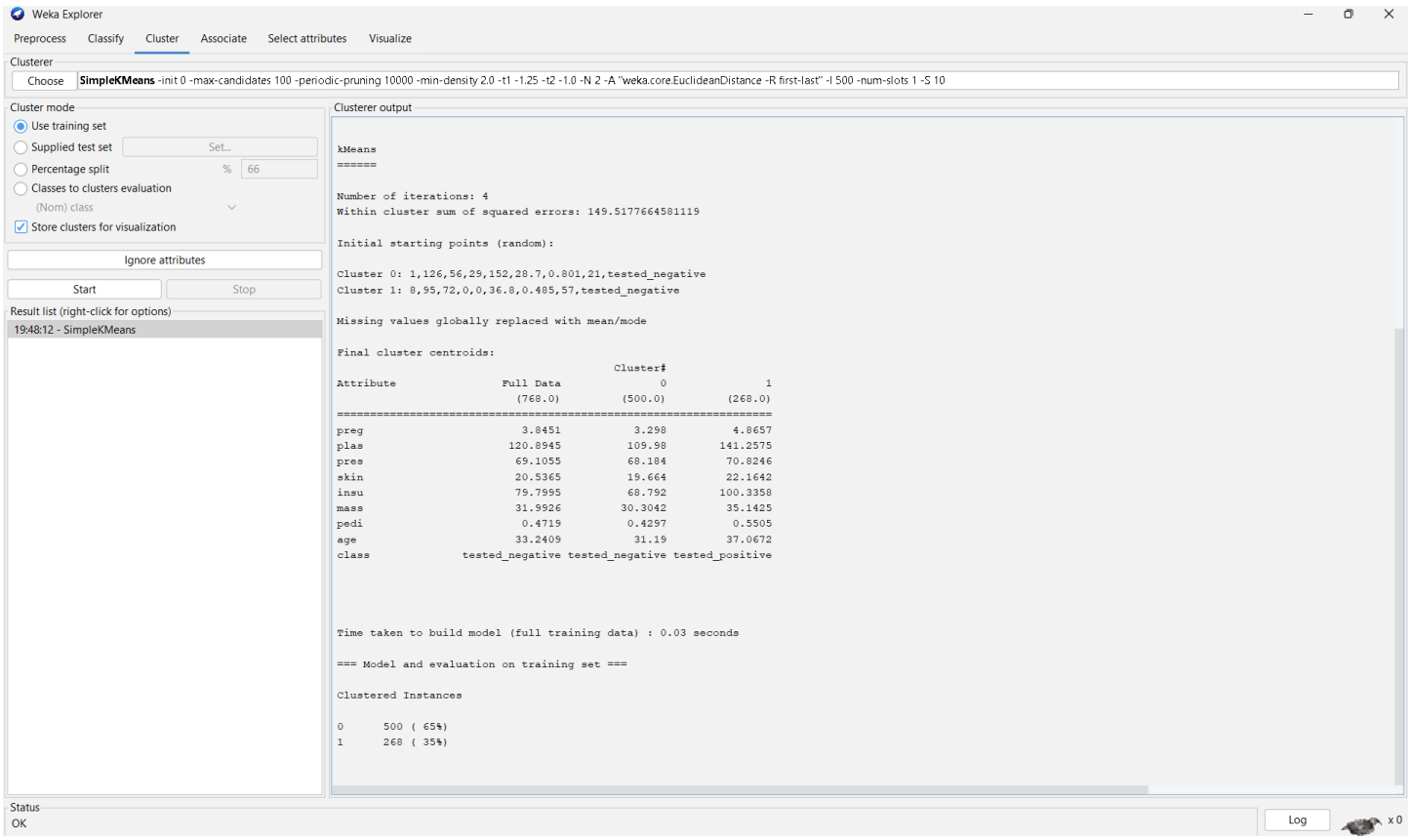
STEP 9: NOW CLICK ON CROSS VALIDATION AND CLICK ON START BUTTON



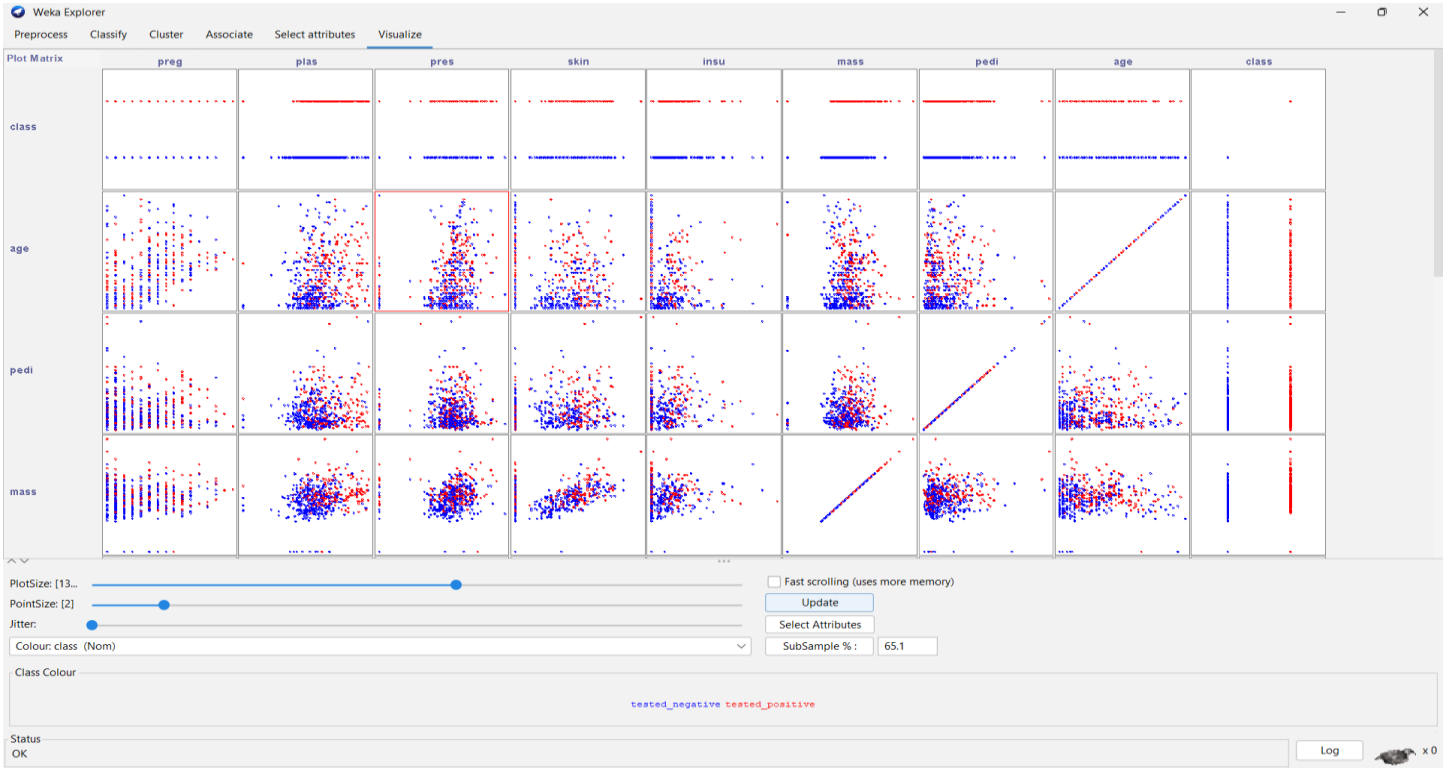
RANDOM FOREST:



K MEAN:



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



STEP 11: ON CLICKING ON ANY OF THE PLOT MATRIX