NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

CLASSIFICATION OF IRIS DATASET USING GAUSSIAN NAIVE BAYES CLASSIFIER

Python Mini Project Report Submitted by

Rushab Shah

(4NM15CS141)

Safa Suleman Shaikh

(4NM15CS144)

UNDER THE GUIDANCE OF

Mr. Vijaya Murari

Assistant Professor

Department of Computer Science and Engineering

November 2018

NITTE
EDUCATION TRUST

N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled "**Classfication of Iris Dataset using Gaussian Naive Bayes Classifier**" is a bonafide work carried out by **Rushab Shah(4NM15CS141)** and **Safa Suleman Shaikh(4NM15CS144)** in partial fulfillment for the award of Degree of Bachelor of Engineering in Computer Sceience and Engineering of the Visvesvaraya Technology Unversity, Belgaum during the year 2016. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the bachelor of Engineering Degree.

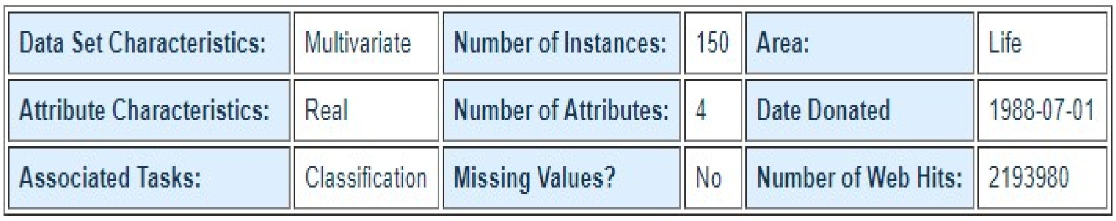
____

Signature of Guide

INTRODUCTION

The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in his 1936 paper "*The use of multiple measurements in taxonomic problems"* as an example of linear discriminant analysis. It is sometimes called Anderson's Iris data set because Edgar Anderson collected the data to quantify the morphologic variation of Iris flowers of three related species. Two of the three species were collected in the Gaspé Peninsula - "all from the same pasture, and picked on the same day and measured at the same time by the same person with the same apparatus".

The data set consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters. Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other.Based on Fisher's linear discriminant model, this data set became a typical test case for many statistical classification techniques in machine learning.



Data Set Characteristics:	Multivariate	Number of Instances:	150	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	4	Date Donated	1988-07-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	2193980

Table 1: The characrteristics of the Iris Dataset

PROBLEM STATEMENT

The aim of the project is to develop a classifier which can classify the flowers according to their species based on the Iris dataset.

We aim to implement the classifier using the Gaussian Naive Bayes algorithm and Python programming language. We will make use of the the Scikit-learn library for designing the machine learning algorithm.

Also the Iris dataset available from the UCI Machine Learning Repository will be stored in a MySQL Database and accessed from there to train and test the classifier.

The front end will be an application designed using the Tkinter python library which will take user inputs and predict the class or the species of the flower.

IMPLEMENTATION:

Back End:

The back end is designed using the MySQL Database. We create a database named Iris_dataset under which we create a table called iris with fields – Serial no, sepal length, sepal width, petal length, petal width and species.

The SQL commands are as follows:

```
CREATE DATABASE Iris_dataset;

USE Iris_dataset;

CREATE TABLE Iris(    No INT PRIMARY KEY,

                      Sepal_length DOUBLE NOT NULL,

                      Sepal_width DOUBLE NOT NULL,

                      Petal_length DOUBLE NOT NULL,

                      Petal_width DOUBLE NOT NULL,

                      Species VARCHAR(100) NOT NULL);
```

We then insert the 150 tuples to the database using the INSERT query. A sample query is shown below:

```
INSERT INTO Iris VALUES (5.1,3.4,1.4,0.2,"Iris-setosa");
```

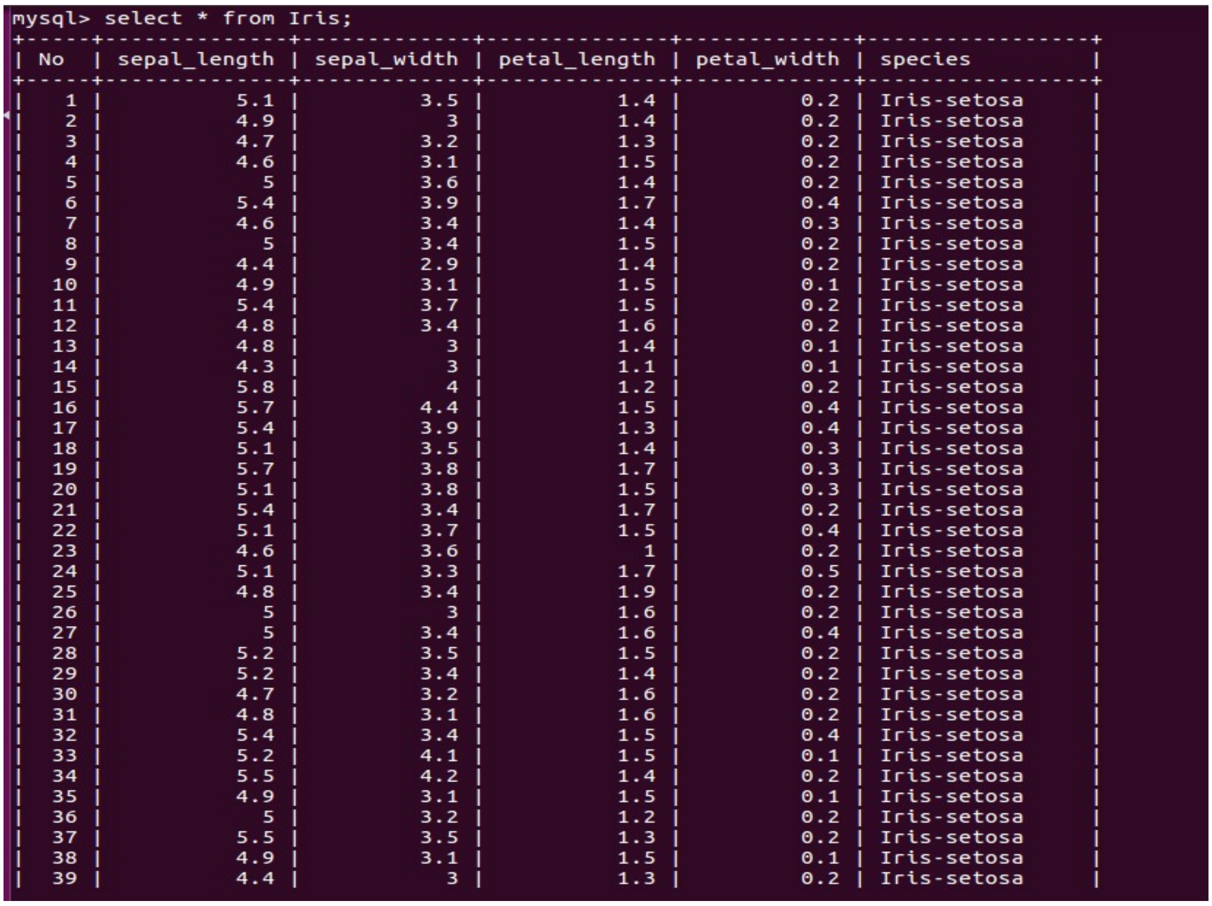
FrontEnd:

```
#!/usr/bin/env
python3
            import mysql.connector
            import numpy as np
            import pandas as pd
            from tkinter import *
            from PIL import ImageTk, Image
            from sklearn.metrics import confusion_matrix
            from sklearn.model_selection import train_test_split
            from sklearn.preprocessing import StandardScaler
            global predict_window
            # GUI Initialization
            home_screen = Tk()
            home_screen.title("Python Project")
            iris_color = '#43348c'
            # GUI - User-entered values
            sepal_length = DoubleVar()
            sepal_width = DoubleVar()
            petal_length = DoubleVar()
            petal_width = DoubleVar()
            # Naive Bayes algorithm
            class GaussianNB(object):
                def __init__(self):
                    pass
                def fit(self, X, y):
                    """Fit Gaussian Naive Bayes according to X, y
                    Parameters
                    ----------
                    X : array-like, shape (n_samples, n_features)
                        Training vectors, where n_samples is the number of samples
                        and n_features is the number of features.
                    y : array-like, shape (n_samples,)
                        Target values."""
                    separated = [
                        [
                            x for x,
                            t in zip(X, y)
                            if t == c
                        ]
                        for c in np.unique(y)
                    ]
                    self.model = np.array(
                        [
                            np.c_[
                                np.mean(i, axis=0),
                            np.std(i, axis=0)
                        ]
                        for i in separated
                    ]
                )
                return self
            def _prob(self, x, mean, std):
                exponent = np.exp(- ((x - mean)**2 / (2 * std**2)))
                return np.log(exponent / (np.sqrt(2 * np.pi) * std))
            def predict_log_proba(self, X):
                """
                Return log-probability estimates for the test vector X.
                Parameters
                ----------
                X : array-like, shape = [n_samples, n_features]"""
                return [
                    [
                        sum(
                            self._prob(i, *s) for s,
                            i in zip(summaries, x)
                        )
                        for summaries in self.model
                    ]
                    for x in X
                ]
            def predict(self, X):
                """
                Perform classification on an array of test vectors X.
                Parameters
                ----------
                X : array-like, shape = [n_samples, n_features]"""
                return np.argmax(self.predict_log_proba(X), axis=1)
            # Importing the dataset
            # dataset = pd.read_csv('iris_uci.csv')
            mydb = mysql.connector.connect(
                host="localhost", user="root", passwd="root", database="iris_dataset")
            mycursor = mydb.cursor()
```

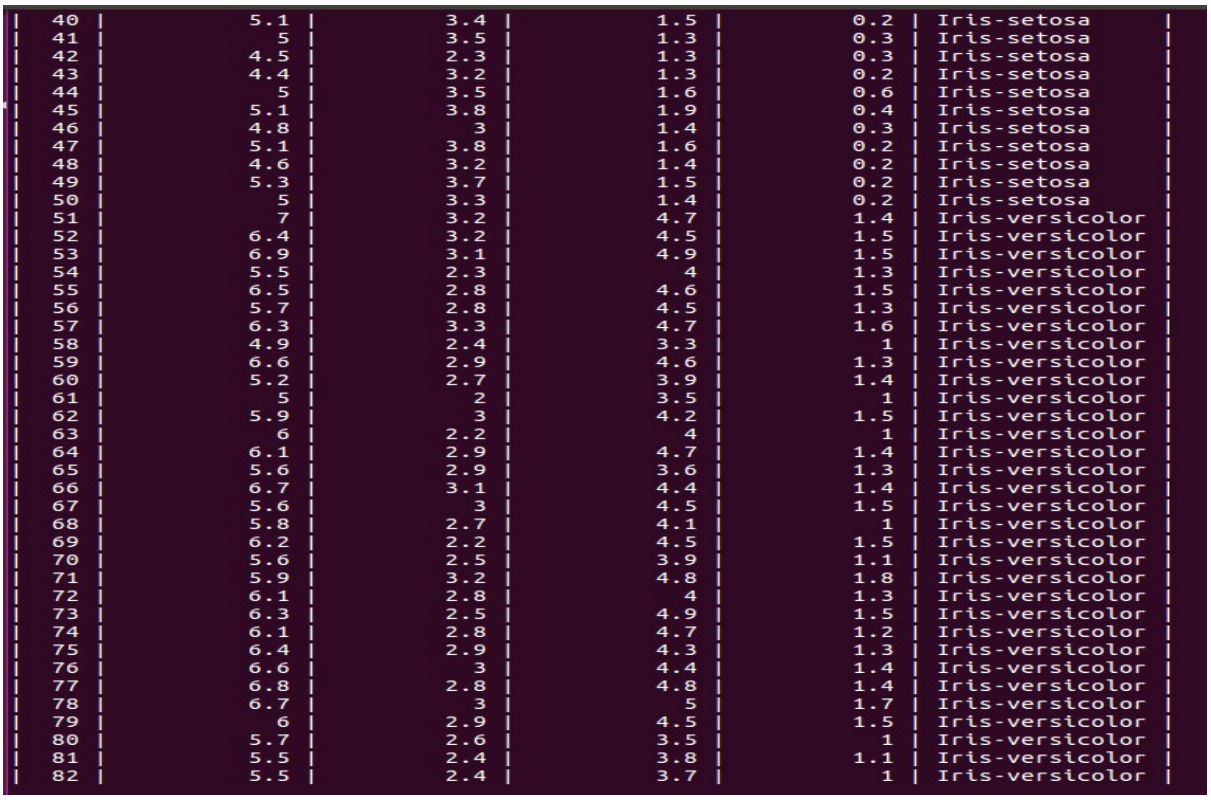
```
sql = "SELECT * FROM Iris"
dataset = pd.read_sql(sql, mydb)
# Spliting the dataset into independent and dependent variables
X = dataset.iloc[:, 1:5].values
y = dataset['species'].values
# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.20, random_state=82)
'''
This function transforms the data such that its distribution will have a mean
value 0 and standard deviation of 1.
Here each value in the dataset will have the mean value subtracted,
and then divided by the standard deviation of the whole dataset.
x′=(x−μ)/σ
fit() calculates μ and σ
fit_transform() calls fit() and transform() internally.
'''
sc = StandardScaler()
# Now sc is fitted, so just use sc.transform(test_data) from now on.
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
# Fitting Naive Bayes Classification to the Training set with linear kernel
nvclassifier = GaussianNB().fit(X_train, y_train)
# Predicting the Test set results. (0 = setosa, 1 = versicolor, 2 = virginica)
predictions = nvclassifier.predict(X_test)
print(X_test)
# Assigning flower class based on the prediction results.
y_pred = []
for x in range(len(predictions)):
    if predictions[x] == 0:
        y_pred.append('Iris-setosa')
    elif predictions[x] == 1:
        y_pred.append('Iris-versicolor')
    else:
        y_pred.append('Iris-virginica')
y_pred = np.asarray(y_pred)
print("Actual values: \n", y_test)
print("Predicted values: \n", y_pred)
# Side-by-side view of actual and predicted values
y_compare = np.vstack((y_test, y_pred)).T
# Actual value on the left side and predicted value on the right hand side
print("\n\nSide-by-side comparison of Actual vs Predicted values")
print("------------------------------------------------------")
print(y_compare[:, :])
# Making the Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
# Finding accuracy from the confusion matrix.
a = cm.shape
corrPred = 0
falsePred = 0
for row in range(a[0]):
    for c in range(a[1]):
        if row == c:
            corrPred += cm[row, c]
        else:
            falsePred += cm[row, c]
print('\nCorrect predictions: ', corrPred)
print('False predictions', falsePred)
print('\n\nAccuracy of the Bayesian Classification is: ',
      corrPred / (cm.sum()) * 100, '%')
# GUI - Show flower window
def show_flower(flower):
    flower_window = Toplevel(home_screen)
    flower_window.title("Predicted Flower")
    flower_canvas = Canvas(flower_window, width=300, height=300)
    flower_canvas.pack()
    if flower == 0:
        img = ImageTk.PhotoImage(Image.open("iris_setosa.jpg"))
        l_flower = Label(flower_window, text="The flower is Iris - Setosa")
        l_flower.config(font=("Ubuntu Mono", 13))
    elif flower == 1:
        img = ImageTk.PhotoImage(Image.open("iris_versicolor.jpg"))
        l_flower = Label(flower_window, text="The flower is Iris -
Versicolor")
```

```
        l_flower.config(font=("Ubuntu Mono", 13))
    else:
        img = ImageTk.PhotoImage(Image.open("iris_virginica.jpg"))
        l_flower = Label(flower_window, text="The flower is Iris -
Virginica")
        l_flower.config(font=("Ubuntu Mono", 13))
    l_flower.pack()
    flower_canvas.create_image(0, 0, anchor=NW, image=img)
    flower_window.mainloop()
# GUI - Prediction
def predict():
    print("The entries are ", sepal_length.get(), " ", sepal_width.get(),
          " ", petal_length.get(), " ", petal_width.get(), "\n\n")
    test = np.array([[sepal_length.get(), sepal_width.get(),
                      petal_length.get(), petal_width.get()]])
    test = sc.transform(test)
    prediction = nvclassifier.predict(test)
    if prediction[0] == 0:
        show_flower(0)
    elif prediction[0] == 1:
        show_flower(1)
    else:
        show_flower(2)
def prediction_window():
    predict_window = Toplevel(home_screen)
    predict_window.title("Predict Flower")
    l_input_desc = Label(
        predict_window, text="Enter the following parameters of iris flower:",
font=("Ubuntu Mono", 13))
    l_input_sepal_length = Label(predict_window, text="Sepal Length:")
    l_input_sepal_length.config(font=("Ubuntu Mono", 13))
    entry_sepal_length = Entry(
        predict_window, textvariable=sepal_length)
    l_input_sepal_width = Label(predict_window, text="Sepal Width:")
    l_input_sepal_width.config(font=("Ubuntu Mono", 13))
    entry_sepal_width = Entry(predict_window, textvariable=sepal_width)
    l_input_petal_length = Label(predict_window, text="Petal Length:")
    l_input_petal_length.config(font=("Ubuntu Mono", 13))
    entry_petal_length = Entry(
        predict_window, textvariable=petal_length)
    l_input_petal_width = Label(predict_window, text="Petal Width:")
    l_input_petal_width.config(font=("Ubuntu Mono", 13))
    entry_petal_width = Entry(predict_window, textvariable=petal_width)
    predict_button = Button(predict_window, text="Predict!",
                            bd=0, font=20, justify="center", height=1,
activebackground="#cc575d", command=predict)
    l_input_desc.grid(row=0, column=0)
    l_input_sepal_length.grid(row=1, column=0)
    entry_sepal_length.grid(row=1, column=1)
    l_input_sepal_width.grid(row=2, column=0)
    entry_sepal_width.grid(row=2, column=1)
    l_input_petal_length.grid(row=3, column=0)
    entry_petal_length.grid(row=3, column=1)
    l_input_petal_width.grid(row=4, column=0)
    entry_petal_width.grid(row=4, column=1)
    predict_button.grid(row=5, column=0, columnspan=2)
    predict_window.mainloop()
def home_page():
    title = Label(
        home_screen, text="Hello! Welcome to Iris flower predictor!",
height=5)
    title.config(font=("Ubuntu Mono", 16))
    about_iris_button = Button(home_screen, text="About Iris flowers",
bd=0, font=20, justify="center", height=1,
                               command=about_iris_flower,
activebackground="#cc575d")
    about_iris_button.config(font=("Ubuntu Mono", 13))
    predict_window_button = Button(home_screen, text="Predict the
flowers!", bd=0, font=20, justify="center", height=1,
                                   command=prediction_window,
activebackground="#cc575d")
    predict_window_button.config(font=("Ubuntu Mono", 13))
    about_team_button = Button(home_screen, text="About Team", bd=0,
font=20, justify="center", height=1,
                    command=about_team, activebackground="#cc575d")
    about_team_button.config(font=("Ubuntu Mono", 13))
    canvas = Canvas(home_screen, width=300, height=300)
    canvas.grid(row=1, column=0)
    img = ImageTk.PhotoImage(Image.open("about_iris.jpg"))
    canvas.create_image(0, 0, anchor=NW, image=img)
    canvas2 = Canvas(home_screen, width=300, height=300)
    canvas2.grid(row=1, column=1)
    img2 = ImageTk.PhotoImage(Image.open("predict.jpg"))
    canvas2.create_image(0, 0, anchor=NW, image=img2)
    canvas3 = Canvas(home_screen, width=300, height=300)
    canvas3.grid(row=1, column=2)
    img3 = ImageTk.PhotoImage(Image.open("about_team.jpg"))
    canvas3.create_image(0, 0, anchor=NW, image=img3)
    title.grid(row=0, columnspan=3)
    about_iris_button.grid(row=2)
    predict_window_button.grid(row=2, column=1)
    about_team_button.grid(row=2, column=2)
    home_screen.mainloop()
def about_iris_flower():
    about_iris_window = Toplevel(home_screen)
    about_iris_window.title("About Iris Flowers")
    iris_desc = "Iris is a genus of 260–300 species of flowering plants with
showy flowers.\n\nIt takes its name from the Greek word for a rainbow,
which is also the name\nfor the Greek goddess of the rainbow,
Iris.\n\nSome authors state that the name refers to the wide variety of
flower colors\nfound among the many species.\n\nAs well as being the
scientific name, iris is also widely used as a common name for all Iris
species,\nas well as some belonging to other closely related genera.\n"
    iris_desc += "\nWe will be looking at the following three classes of Iris
flower:\n"
    iris_desc += "-----------------------------------------------------------------\n"
    title = Label(about_iris_window, text=iris_desc,
              justify="left", font=("Ubuntu Mono", 13))
    title.grid(row=0, columnspan=3)
    setosa_head = "1) Iris Setosa:\n"
    setosa_head += "--------------\n"
    l_setosa = Label(about_iris_window, text=setosa_head,
                     anchor=W, justify="left", font=("Ubuntu Mono", 13))
    l_setosa.grid(row=2)
    canvas = Canvas(about_iris_window, width=300, height=300)
    canvas.grid(row=3, column=0)
    img = ImageTk.PhotoImage(Image.open("iris_setosa.jpg"))
    canvas.create_image(0, 0, anchor=NW, image=img)
    versicolor_head = "2) Iris Versicolor:\n"
    versicolor_head += "--------------\n"
    l_setosa = Label(about_iris_window, text=versicolor_head,
                     anchor=W, justify="left", font=("Ubuntu Mono", 13))
    l_setosa.grid(row=2, column=1)
    canvas2 = Canvas(about_iris_window, width=300, height=300)
    canvas2.grid(row=3, column=1)
    img2 = ImageTk.PhotoImage(Image.open("iris_versicolor.jpg"))
    canvas2.create_image(0, 0, anchor=NW, image=img2)
    virginica_head = "2) Iris Virginica:\n"
    virginica_head += "--------------\n"
    l_setosa = Label(about_iris_window, text=virginica_head,
                     anchor=W, justify="left", font=("Ubuntu Mono", 13))
    l_setosa.grid(row=2, column=2)
    canvas3 = Canvas(about_iris_window, width=300, height=300)
    canvas3.grid(row=3, column=2)
    img3 = ImageTk.PhotoImage(Image.open("iris_virginica.jpg"))
    canvas3.create_image(0, 0, anchor=NW, image=img3)
    about_iris_window.mainloop()
def about_team():
    about_team_window = Toplevel(home_screen)
    about_team_window.title("About the Team")
    team_name_1 = "Safa Suleman Shaikh"
    team_USN_1 = "USN:          4NM15CS144"
    team_name_2 = "Rushab Shah"
    team_USN_2 = "USN:          4NM15CS141"
    team_semester = "Semester:     7th"
    team_section = "Section:      C"
    team_dept = "Department:    Computer Science and Engineering"
    l_name_1 = Label(about_team_window, text=team_name_1,
                     anchor='w', width=50, font=("Ubuntu Mono", 15))
    l_usn_1 = Label(about_team_window, text=team_USN_1,
                    anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_semester_1 = Label(about_team_window, text=team_semester,
                         anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_section_1 = Label(about_team_window, text=team_section,
                        anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_dept_1 = Label(about_team_window, text=team_dept,
                     anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_name_2 = Label(about_team_window, text="\n"+team_name_2,
                     anchor='w', width=50, font=("Ubuntu Mono", 15))
    l_usn_2 = Label(about_team_window, text=team_USN_2,
                    anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_semester_2 = Label(about_team_window, text=team_semester,
                         anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_section_2 = Label(about_team_window, text=team_section,
                        anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_dept_2 = Label(about_team_window, text=team_dept,
                     anchor='w', width=50, font=("Ubuntu Mono", 13))
    l_name_1.grid(row=0, columnspan=2)
    l_usn_1.grid(row=1, columnspan=2)
    l_semester_1.grid(row=2, columnspan=2)
    l_section_1.grid(row=3, columnspan=2)
    l_dept_1.grid(row=4, columnspan=2)
    l_name_2.grid(row=6, columnspan=2)
    l_usn_2.grid(row=7, columnspan=2)
    l_semester_2.grid(row=8, columnspan=2)
    l_section_2.grid(row=9, columnspan=2)
    l_dept_2.grid(row=10, columnspan=2)
    about_team_window.mainloop()
home_page()
```

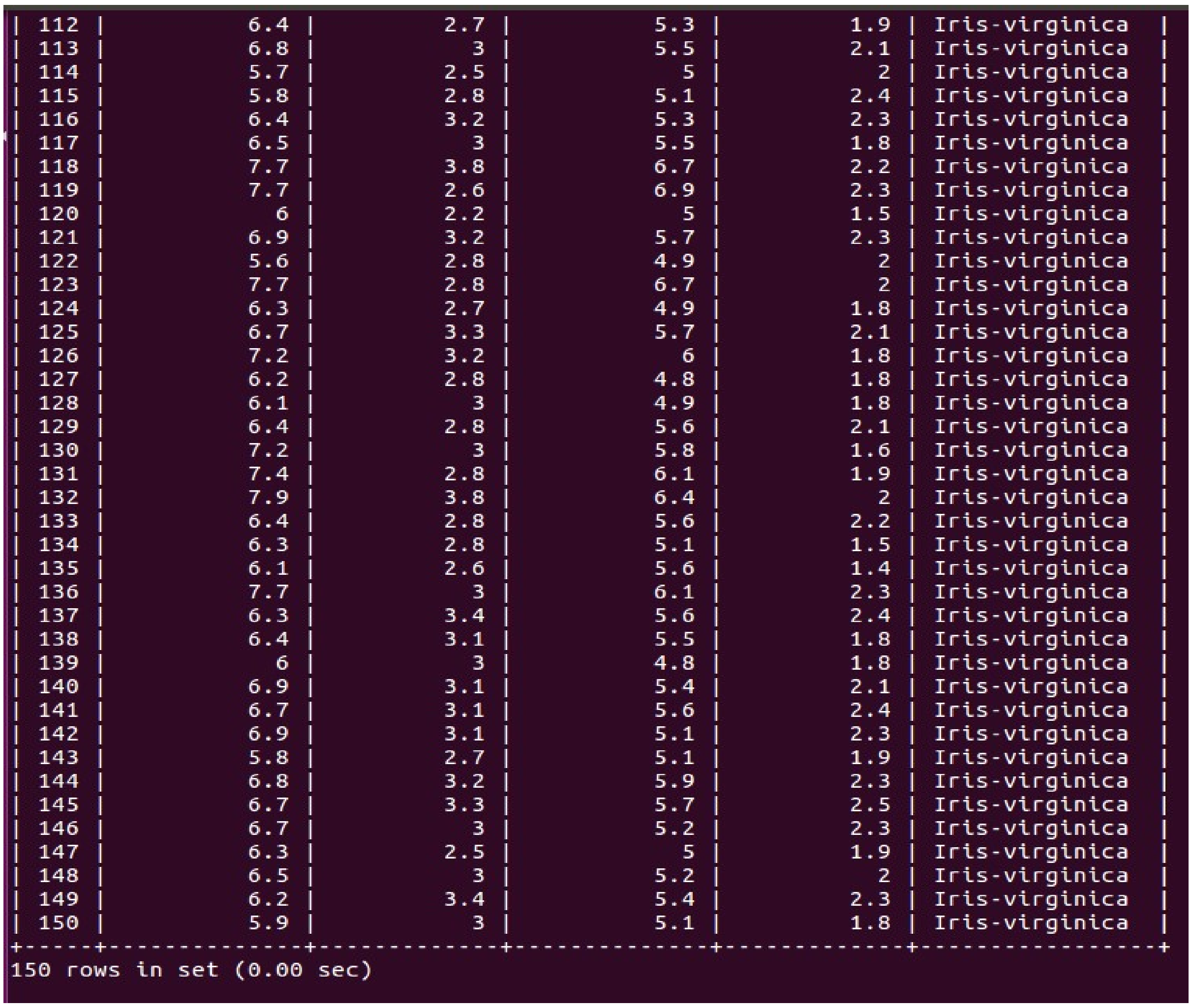
OUTPUT:



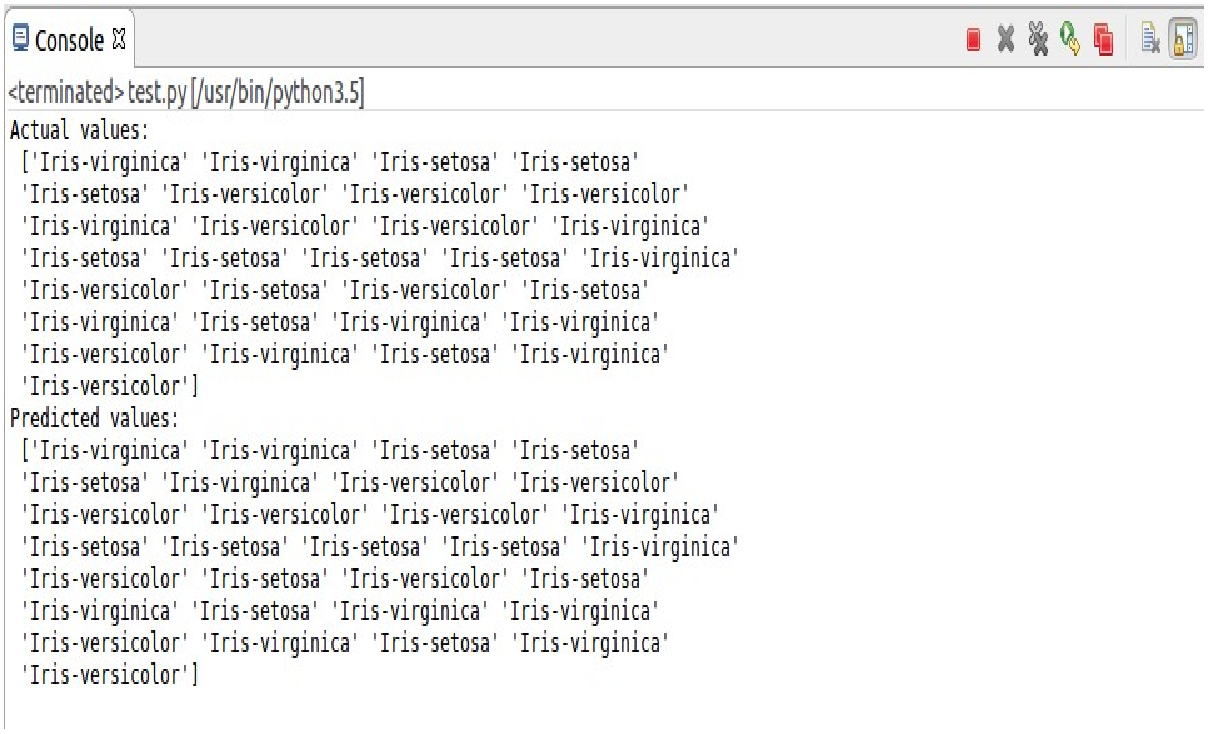
```
mysql> select * from Iris;
+-----+--------------+-------------+--------------+-------------+------------------+
| No  | sepal_length | sepal_width | petal_length | petal_width | species          |
+-----+--------------+-------------+--------------+-------------+------------------+
|   1 |          5.1 |         3.5 |          1.4 |         0.2 | Iris-setosa      |
|   2 |          4.9 |           3 |          1.4 |         0.2 | Iris-setosa      |
|   3 |          4.7 |         3.2 |          1.3 |         0.2 | Iris-setosa      |
|   4 |          4.6 |         3.1 |          1.5 |         0.2 | Iris-setosa      |
|   5 |            5 |         3.6 |          1.4 |         0.2 | Iris-setosa      |
|   6 |          5.4 |         3.9 |          1.7 |         0.4 | Iris-setosa      |
|   7 |          4.6 |         3.4 |          1.4 |         0.3 | Iris-setosa      |
|   8 |            5 |         3.4 |          1.5 |         0.2 | Iris-setosa      |
|   9 |          4.4 |         2.9 |          1.4 |         0.2 | Iris-setosa      |
|  10 |          4.9 |         3.1 |          1.5 |         0.1 | Iris-setosa      |
|  11 |          5.4 |         3.7 |          1.5 |         0.2 | Iris-setosa      |
|  12 |          4.8 |         3.4 |          1.6 |         0.2 | Iris-setosa      |
|  13 |          4.8 |           3 |          1.4 |         0.1 | Iris-setosa      |
|  14 |          4.3 |           3 |          1.1 |         0.1 | Iris-setosa      |
|  15 |          5.8 |           4 |          1.2 |         0.2 | Iris-setosa      |
|  16 |          5.7 |         4.4 |          1.5 |         0.4 | Iris-setosa      |
|  17 |          5.4 |         3.9 |          1.3 |         0.4 | Iris-setosa      |
|  18 |          5.1 |         3.5 |          1.4 |         0.3 | Iris-setosa      |
|  19 |          5.7 |         3.8 |          1.7 |         0.3 | Iris-setosa      |
|  20 |          5.1 |         3.8 |          1.5 |         0.3 | Iris-setosa      |
|  21 |          5.4 |         3.4 |          1.7 |         0.2 | Iris-setosa      |
|  22 |          5.1 |         3.7 |          1.5 |         0.4 | Iris-setosa      |
|  23 |          4.6 |         3.6 |            1 |         0.2 | Iris-setosa      |
|  24 |          5.1 |         3.3 |          1.7 |         0.5 | Iris-setosa      |
|  25 |          4.8 |         3.4 |          1.9 |         0.2 | Iris-setosa      |
|  26 |            5 |           3 |          1.6 |         0.2 | Iris-setosa      |
|  27 |            5 |         3.4 |          1.6 |         0.4 | Iris-setosa      |
|  28 |          5.2 |         3.5 |          1.5 |         0.2 | Iris-setosa      |
|  29 |          5.2 |         3.4 |          1.4 |         0.2 | Iris-setosa      |
|  30 |          4.7 |         3.2 |          1.6 |         0.2 | Iris-setosa      |
|  31 |          4.8 |         3.1 |          1.6 |         0.2 | Iris-setosa      |
|  32 |          5.4 |         3.4 |          1.5 |         0.4 | Iris-setosa      |
|  33 |          5.2 |         4.1 |          1.5 |         0.1 | Iris-setosa      |
|  34 |          5.5 |         4.2 |          1.4 |         0.2 | Iris-setosa      |
|  35 |          4.9 |         3.1 |          1.5 |         0.1 | Iris-setosa      |
|  36 |            5 |         3.2 |          1.2 |         0.2 | Iris-setosa      |
|  37 |          5.5 |         3.5 |          1.3 |         0.2 | Iris-setosa      |
|  38 |          4.9 |         3.1 |          1.5 |         0.1 | Iris-setosa      |
|  39 |          4.4 |           3 |          1.3 |         0.2 | Iris-setosa      |
```



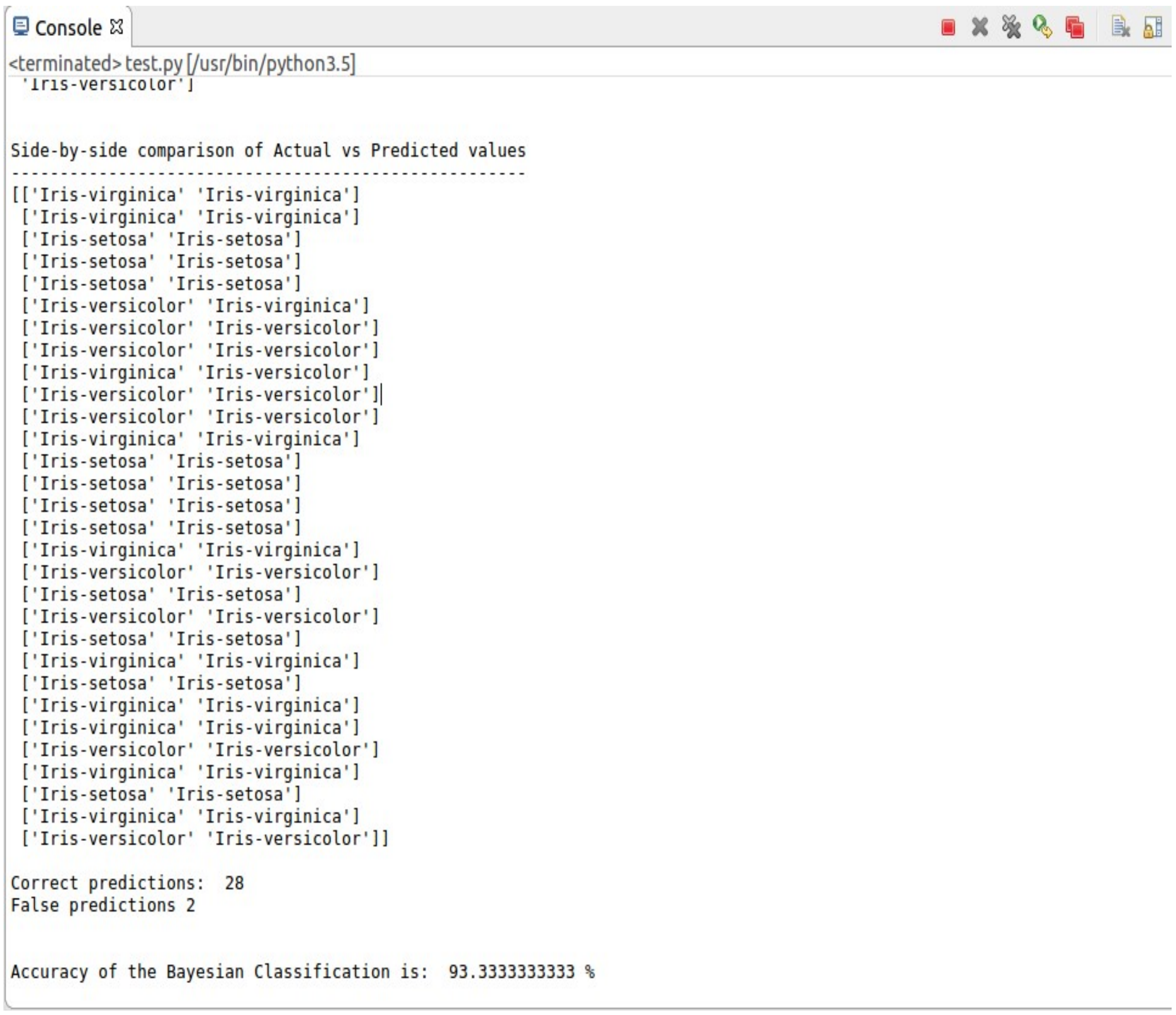
```
|   40 |            5.1 |           3.4 |            1.5 |           0.2 | Iris-setosa     |
|   41 |              5 |           3.5 |            1.3 |           0.3 | Iris-setosa     |
|   42 |            4.5 |           2.3 |            1.3 |           0.3 | Iris-setosa     |
|   43 |            4.4 |           3.2 |            1.3 |           0.2 | Iris-setosa     |
|   44 |              5 |           3.5 |            1.6 |           0.6 | Iris-setosa     |
|   45 |            5.1 |           3.8 |            1.9 |           0.4 | Iris-setosa     |
|   46 |            4.8 |             3 |            1.4 |           0.3 | Iris-setosa     |
|   47 |            5.1 |           3.8 |            1.6 |           0.2 | Iris-setosa     |
|   48 |            4.6 |           3.2 |            1.4 |           0.2 | Iris-setosa     |
|   49 |            5.3 |           3.7 |            1.5 |           0.2 | Iris-setosa     |
|   50 |              5 |           3.3 |            1.4 |           0.2 | Iris-setosa     |
|   51 |              7 |           3.2 |            4.7 |           1.4 | Iris-versicolor |
|   52 |            6.4 |           3.2 |            4.5 |           1.5 | Iris-versicolor |
|   53 |            6.9 |           3.1 |            4.9 |           1.5 | Iris-versicolor |
|   54 |            5.5 |           2.3 |              4 |           1.3 | Iris-versicolor |
|   55 |            6.5 |           2.8 |            4.6 |           1.5 | Iris-versicolor |
|   56 |            5.7 |           2.8 |            4.5 |           1.3 | Iris-versicolor |
|   57 |            6.3 |           3.3 |            4.7 |           1.6 | Iris-versicolor |
|   58 |            4.9 |           2.4 |            3.3 |             1 | Iris-versicolor |
|   59 |            6.6 |           2.9 |            4.6 |           1.3 | Iris-versicolor |
|   60 |            5.2 |           2.7 |            3.9 |           1.4 | Iris-versicolor |
|   61 |              5 |             2 |            3.5 |             1 | Iris-versicolor |
|   62 |            5.9 |             3 |            4.2 |           1.5 | Iris-versicolor |
|   63 |              6 |           2.2 |              4 |             1 | Iris-versicolor |
|   64 |            6.1 |           2.9 |            4.7 |           1.4 | Iris-versicolor |
|   65 |            5.6 |           2.9 |            3.6 |           1.3 | Iris-versicolor |
|   66 |            6.7 |           3.1 |            4.4 |           1.4 | Iris-versicolor |
|   67 |            5.6 |             3 |            4.5 |           1.5 | Iris-versicolor |
|   68 |            5.8 |           2.7 |            4.1 |             1 | Iris-versicolor |
|   69 |            6.2 |           2.2 |            4.5 |           1.5 | Iris-versicolor |
|   70 |            5.6 |           2.5 |            3.9 |           1.1 | Iris-versicolor |
|   71 |            5.9 |           3.2 |            4.8 |           1.8 | Iris-versicolor |
|   72 |            6.1 |           2.8 |              4 |           1.3 | Iris-versicolor |
|   73 |            6.3 |           2.5 |            4.9 |           1.5 | Iris-versicolor |
|   74 |            6.1 |           2.8 |            4.7 |           1.2 | Iris-versicolor |
|   75 |            6.4 |           2.9 |            4.3 |           1.3 | Iris-versicolor |
|   76 |            6.6 |             3 |            4.4 |           1.4 | Iris-versicolor |
|   77 |            6.8 |           2.8 |            4.8 |           1.4 | Iris-versicolor |
|   78 |            6.7 |             3 |              5 |           1.7 | Iris-versicolor |
|   79 |              6 |           2.9 |            4.5 |           1.5 | Iris-versicolor |
|   80 |            5.7 |           2.6 |            3.5 |             1 | Iris-versicolor |
|   81 |            5.5 |           2.4 |            3.8 |           1.1 | Iris-versicolor |
|   82 |            5.5 |           2.4 |            3.7 |             1 | Iris-versicolor |
```



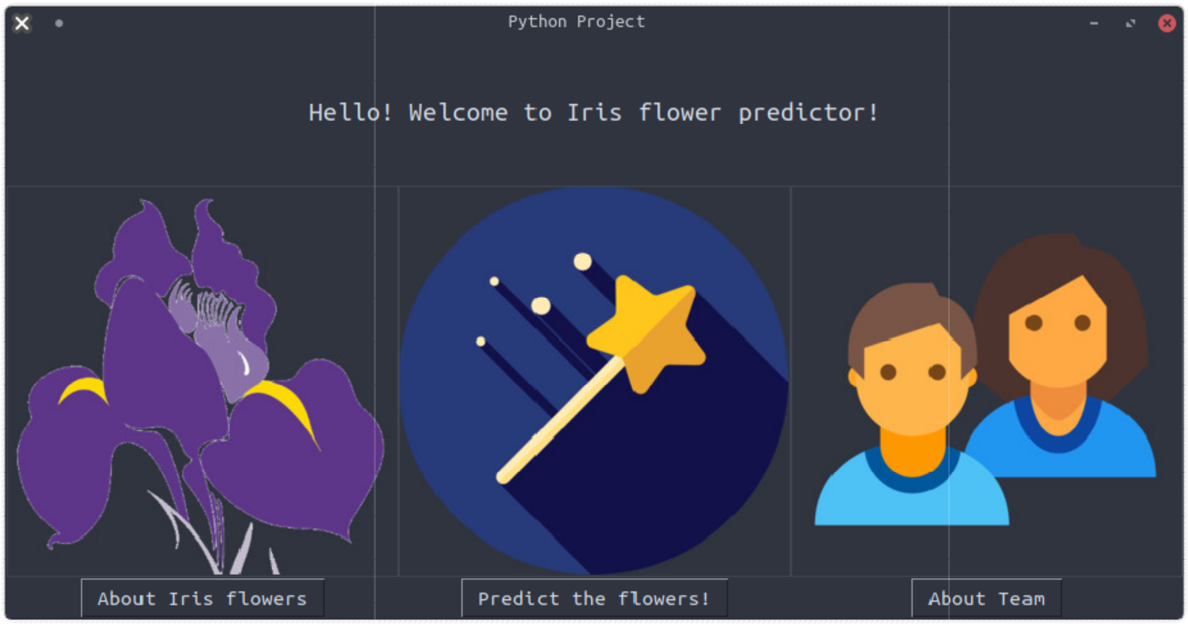
```
| 112 |          6.4 |          2.7 |           5.3 |          1.9 | Iris-virginica  |
| 113 |          6.8 |            3 |           5.5 |          2.1 | Iris-virginica  |
| 114 |          5.7 |          2.5 |             5 |            2 | Iris-virginica  |
| 115 |          5.8 |          2.8 |           5.1 |          2.4 | Iris-virginica  |
| 116 |          6.4 |          3.2 |           5.3 |          2.3 | Iris-virginica  |
| 117 |          6.5 |            3 |           5.5 |          1.8 | Iris-virginica  |
| 118 |          7.7 |          3.8 |           6.7 |          2.2 | Iris-virginica  |
| 119 |          7.7 |          2.6 |           6.9 |          2.3 | Iris-virginica  |
| 120 |            6 |          2.2 |             5 |          1.5 | Iris-virginica  |
| 121 |          6.9 |          3.2 |           5.7 |          2.3 | Iris-virginica  |
| 122 |          5.6 |          2.8 |           4.9 |            2 | Iris-virginica  |
| 123 |          7.7 |          2.8 |           6.7 |            2 | Iris-virginica  |
| 124 |          6.3 |          2.7 |           4.9 |          1.8 | Iris-virginica  |
| 125 |          6.7 |          3.3 |           5.7 |          2.1 | Iris-virginica  |
| 126 |          7.2 |          3.2 |             6 |          1.8 | Iris-virginica  |
| 127 |          6.2 |          2.8 |           4.8 |          1.8 | Iris-virginica  |
| 128 |          6.1 |            3 |           4.9 |          1.8 | Iris-virginica  |
| 129 |          6.4 |          2.8 |           5.6 |          2.1 | Iris-virginica  |
| 130 |          7.2 |            3 |           5.8 |          1.6 | Iris-virginica  |
| 131 |          7.4 |          2.8 |           6.1 |          1.9 | Iris-virginica  |
| 132 |          7.9 |          3.8 |           6.4 |            2 | Iris-virginica  |
| 133 |          6.4 |          2.8 |           5.6 |          2.2 | Iris-virginica  |
| 134 |          6.3 |          2.8 |           5.1 |          1.5 | Iris-virginica  |
| 135 |          6.1 |          2.6 |           5.6 |          1.4 | Iris-virginica  |
| 136 |          7.7 |            3 |           6.1 |          2.3 | Iris-virginica  |
| 137 |          6.3 |          3.4 |           5.6 |          2.4 | Iris-virginica  |
| 138 |          6.4 |          3.1 |           5.5 |          1.8 | Iris-virginica  |
| 139 |            6 |            3 |           4.8 |          1.8 | Iris-virginica  |
| 140 |          6.9 |          3.1 |           5.4 |          2.1 | Iris-virginica  |
| 141 |          6.7 |          3.1 |           5.6 |          2.4 | Iris-virginica  |
| 142 |          6.9 |          3.1 |           5.1 |          2.3 | Iris-virginica  |
| 143 |          5.8 |          2.7 |           5.1 |          1.9 | Iris-virginica  |
| 144 |          6.8 |          3.2 |           5.9 |          2.3 | Iris-virginica  |
| 145 |          6.7 |          3.3 |           5.7 |          2.5 | Iris-virginica  |
| 146 |          6.7 |            3 |           5.2 |          2.3 | Iris-virginica  |
| 147 |          6.3 |          2.5 |             5 |          1.9 | Iris-virginica  |
| 148 |          6.5 |            3 |           5.2 |            2 | Iris-virginica  |
| 149 |          6.2 |          3.4 |           5.4 |          2.3 | Iris-virginica  |
| 150 |          5.9 |            3 |           5.1 |          1.8 | Iris-virginica  |
+-----+--------------+--------------+---------------+--------------+-----------------+
150 rows in set (0.00 sec)
```

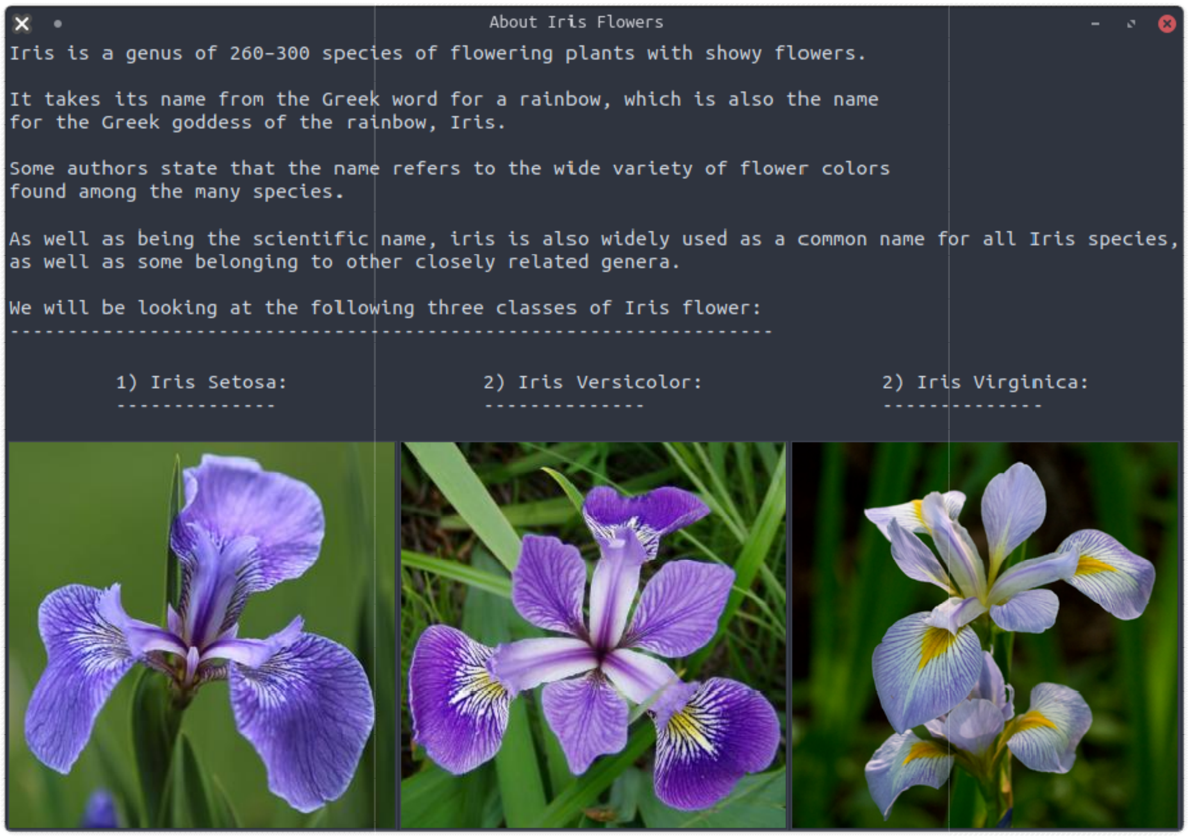


```
<terminated> test.py [/usr/bin/python3.5]
Actual values:
 ['Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-virginica' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor']
Predicted values:
 ['Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-virginica' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor']
```

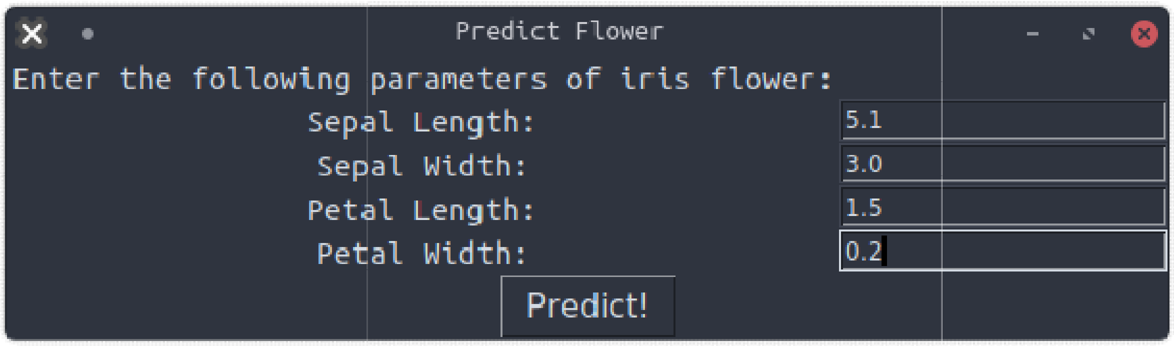


```
<terminated> test.py [/usr/bin/python3.5]
 'Iris-versicolor']


Side-by-side comparison of Actual vs Predicted values
------------------------------------------------------
[['Iris-virginica' 'Iris-virginica']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-versicolor' 'Iris-virginica']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-virginica' 'Iris-versicolor']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-versicolor' 'Iris-versicolor']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-setosa' 'Iris-setosa']
 ['Iris-virginica' 'Iris-virginica']
 ['Iris-versicolor' 'Iris-versicolor']]

Correct predictions:  28
False predictions 2


Accuracy of the Bayesian Classification is:  93.3333333333 %
```

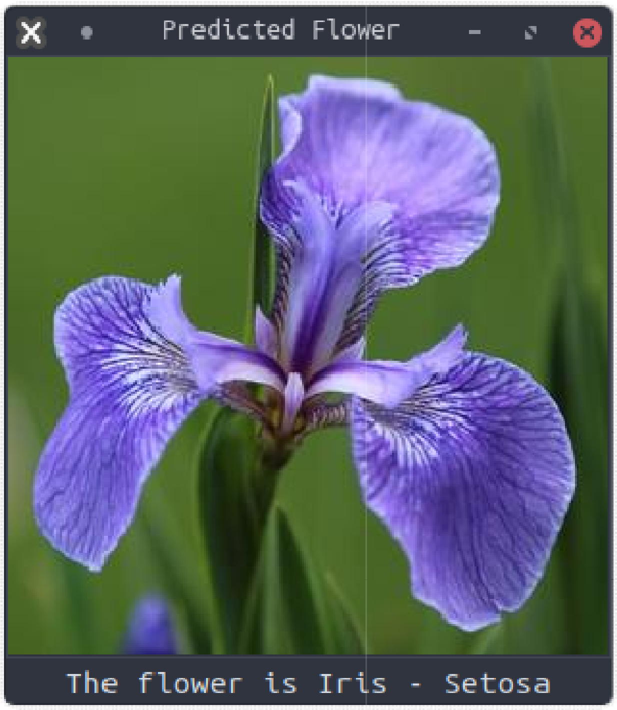




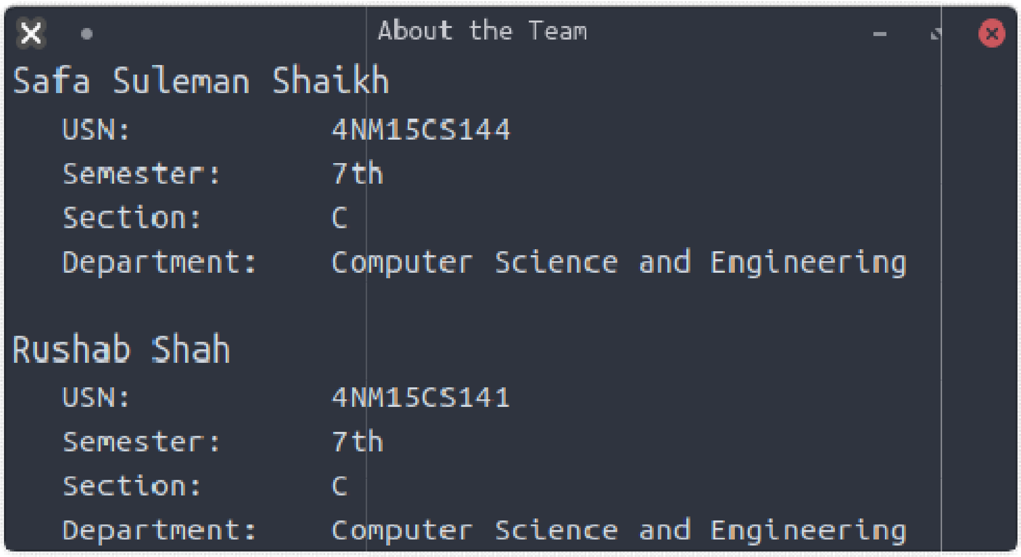
```
Iris is a genus of 260-300 species of flowering plants with showy flowers.

It takes its name from the Greek word for a rainbow, which is also the name
for the Greek goddess of the rainbow, Iris.

Some authors state that the name refers to the wide variety of flower colors
found among the many species.

As well as being the scientific name, iris is also widely used as a common name for all Iris species,
as well as some belonging to other closely related genera.

We will be looking at the following three classes of Iris flower:
-----------------------------------------------------------------

         1) Iris Setosa:                2) Iris Versicolor:                2) Iris Virginica:
         --------------                 --------------                     --------------
```



```
Enter the following parameters of iris flower:
          Sepal Length:        5.1
           Sepal Width:        3.0
          Petal Length:        1.5
           Petal Width:        0.2
```



```
The flower is Iris - Setosa
```



```
Safa Suleman Shaikh
   USN:          4NM15CS144
   Semester:     7th
   Section:      C
   Department:   Computer Science and Engineering

Rushab Shah
   USN:          4NM15CS141
   Semester:     7th
   Section:      C
   Department:   Computer Science and Engineering
```

CONCLUSION

Python is a popular and powerful interpreted language. Unlike R, Python is a complete language and platform that you can use for both research and development and developing production systems. The best way to really come to terms with a new platform or tool is to work through a mini project end-to-end and cover the key steps.

- The key steps involved in the project are loading data, summarizing data, evaluating algorithms and making some predictions.
- Attributes are numeric so we can easily load and store the data in a database.
- It is a multi-class classification problem (multi-nominal) that requires some specialized handling.
- It only has 4 attributes and 150 rows, meaning it is small and easily fits into memory (and a screen or A4 page).
- All of the numeric attributes are in the same units and the same scale, not requiring any special scaling or transforms to get started.

The Gaussian Naive Bayes Classifier developed using the python programming language predicts the species of the flowers with an accuracy of 93.33%. We have learnt the usage of the Tkinter python library and the various widgets so as to design the GUI for the proposed problem statement. Also we have learnt how python can be used for the backend programming of the database and how to access the database from the python programs.