Iris Dataset Naive Bayes Classifier

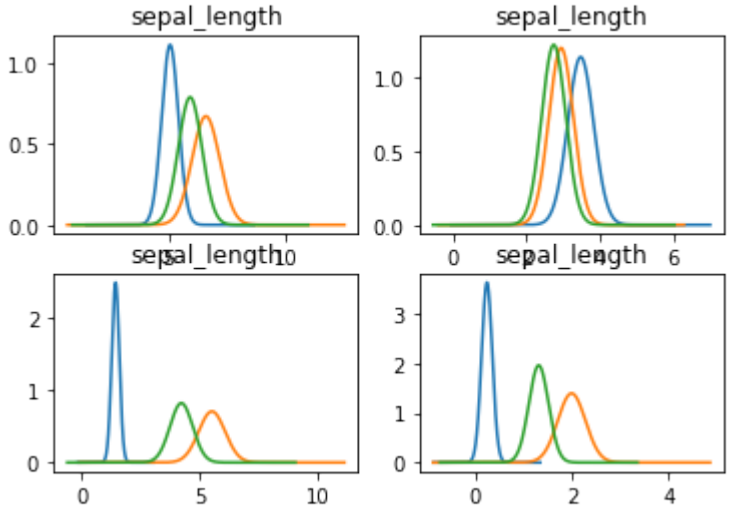
In [16]:

```
import random
from math import sqrt
from math import exp
from math import pi

import pandas
from colorama import Fore
import matplotlib.pyplot as plt
import scipy.stats as stats
import numpy as np

lookup = dict()
separated = dict()
data_frame = 'None'


def load_csv(file_name):
    dataset = list()
    global data_frame
    data_frame = pandas.read_csv(file_name)
    for row in data_frame.values.tolist():
        if not row:
            continue
        vector = [float(x) for x in row[:4]]
        vector.append(row[4])
        dataset.append(vector)
    random.shuffle(dataset)
    return dataset


def tagg_to_int(dataset, column):
    class_values = [row[column] for row in dataset]
    unique = set(class_values)
    for i, value in enumerate(unique):
        lookup[value] = i
    for row in dataset:
        row[column] = lookup[row[column]]
    return lookup


def separate_by_class(dataset):
    for i in range(len(dataset)):
        vector = dataset[i]
        tagg = vector[-1]
        if tagg not in separated:
            separated[tagg] = list()
        separated[tagg].append(vector)
    return separated


def mean(numbers):
    return sum(numbers) / float(len(numbers))


def standard_deviation(numbers):
    avg = mean(numbers)
    variance = sum([(x - avg) ** 2 for x in numbers]) / float(len(numbers) -
    return sqrt(variance)


def train_model(dataset):
```

```
    separated = separate_by_class(dataset)
    model = dict()
    for tagg, rows in separated.items():
        s = [(mean(column), standard_deviation(column), len(column)) for col
        del (s[-1])
        model[tagg] = s
    return model


def normal_distribution(x, mean, stdev):
    exponent = exp(-((x - mean) ** 2 / (2 * stdev ** 2)))
    return (1 / (sqrt(2 * pi) * stdev)) * exponent


def predict(model, row):
    total_rows = sum([model[label][0][2] for label in model])
    probabilities = dict()
    for tagg, features in model.items():
        probabilities[tagg] = model[tagg][0][2] / float(total_rows)
        for i in range(len(features)):
            mean, stdev, _ = features[i]
            probabilities[tagg] *= normal_distribution(row[i], mean, stdev)
    best_label, best_prob = None, -1
    for tagg, probability in probabilities.items():
        if best_label is None or probability > best_prob:
            best_prob = probability
            best_label = tagg
    return best_label


def plot(model):
    figure, axis = plt.subplots(2, 2)
    for j in range(3):
        for i in range(4):
            sigma = model[j][i][1]
            x = np.linspace(model[j][i][0] - 10 * sigma, model[j][i][0] + 10
            axis[int(i / 2), i % 2].plot(x, stats.norm.pdf(x, model[j][i][0]
            axis[int(i / 2), i % 2].set_title(data_frame.columns.values[0])
    plt.show()


if __name__ == '__main__':

    filename = 'iris (6).csv'

    dataset = load_csv(filename)
    train_dateset = dataset[:120]

    tagg_to_int(train_dateset, len(train_dateset[0]) - 1)
    model = train_model(train_dateset)

    test_dataset = dataset[120:]
    dictionary = {v: k for k, v in lookup.items()}

    error = 0
    for row in test_dataset:
        test = [float(x) for x in row[:4]]
        label = predict(model, test)
        if dictionary[label] == row[4]:
            print(Fore.GREEN + 'Data=%s, Predicted: %s , Tag: %s' % (test, d
        else:
```

```
            print(Fore.RED + 'Data=%s, Predicted: %s , Tag: %s' % (test, dic
            error += 1
    print(Fore.YELLOW + 'Prediction Accuracy= ', (len(test_dataset) - error)
    plot(model)
```

```
Data=[5.7, 2.8, 4.1, 1.3], Predicted: Iris-versicolor , Tag: Iris-versicolo
r
Data=[4.8, 3.4, 1.9, 0.2], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[6.1, 3.0, 4.6, 1.4], Predicted: Iris-versicolor , Tag: Iris-versicolo
r
Data=[6.4, 3.2, 5.3, 2.3], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[7.7, 3.0, 6.1, 2.3], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[6.7, 3.1, 4.4, 1.4], Predicted: Iris-versicolor , Tag: Iris-versicolo
r
Data=[5.2, 2.7, 3.9, 1.4], Predicted: Iris-versicolor , Tag: Iris-versicolo
r
Data=[6.7, 3.1, 5.6, 2.4], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[5.8, 2.8, 5.1, 2.4], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[4.5, 2.3, 1.3, 0.3], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[5.6, 2.8, 4.9, 2.0], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[4.9, 3.1, 1.5, 0.1], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[5.7, 3.0, 4.2, 1.2], Predicted: Iris-versicolor , Tag: Iris-versicolo
r
Data=[5.0, 3.0, 1.6, 0.2], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[7.2, 3.2, 6.0, 1.8], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[6.6, 2.9, 4.6, 1.3], Predicted: Iris-versicolor , Tag: Iris-versicolo
r
Data=[7.9, 3.8, 6.4, 2.0], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[4.9, 3.1, 1.5, 0.1], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[6.5, 3.2, 5.1, 2.0], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[4.6, 3.4, 1.4, 0.3], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[6.3, 2.9, 5.6, 1.8], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[5.1, 3.8, 1.9, 0.4], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[5.8, 2.7, 5.1, 1.9], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[5.1, 3.4, 1.5, 0.2], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[7.6, 3.0, 6.6, 2.1], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[4.4, 2.9, 1.4, 0.2], Predicted: Iris-setosa , Tag: Iris-setosa
Data=[6.8, 3.0, 5.5, 2.1], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[5.8, 2.7, 5.1, 1.9], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[6.3, 3.3, 6.0, 2.5], Predicted: Iris-virginica , Tag: Iris-virginica
Data=[6.7, 3.0, 5.0, 1.7], Predicted: Iris-virginica , Tag: Iris-versicolor
Prediction Accuracy=  96.66666666666667 %
```



In [ ]: