**Experiment**

**Implementation of uncertain methods for an application**

**Name:** Sayan Pramanick  **Reg. No.:** RA1811030010023

**Problem taken:** Implementing Uncertain Methods.

**Aim:**  To create correlation matrix using Naive Bayes from the data-set provided.

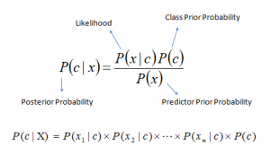
**Problem Description:**

There are typically several different perspectives one can take on the same statistical/machine learning algorithm. Many classifiers have a statistical interpretation that involves fitting a statistical distribution to the data. There can also be a more geometric or “heuristic” interpretation. For example, you can view linear regression fitting a Gaussian distribution to the errors of a linear model or as minimizing the sum of squared residuals.

**Problem Formulation:**

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



* P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
* P(c) is the prior probability of class.
* P(x|c) is the likelihood which is the probability of predictor given class.
* P(x) is the prior probability of predictor.

How Naive Bayes algorithm works?

Step 1: Convert the data set into a frequency table

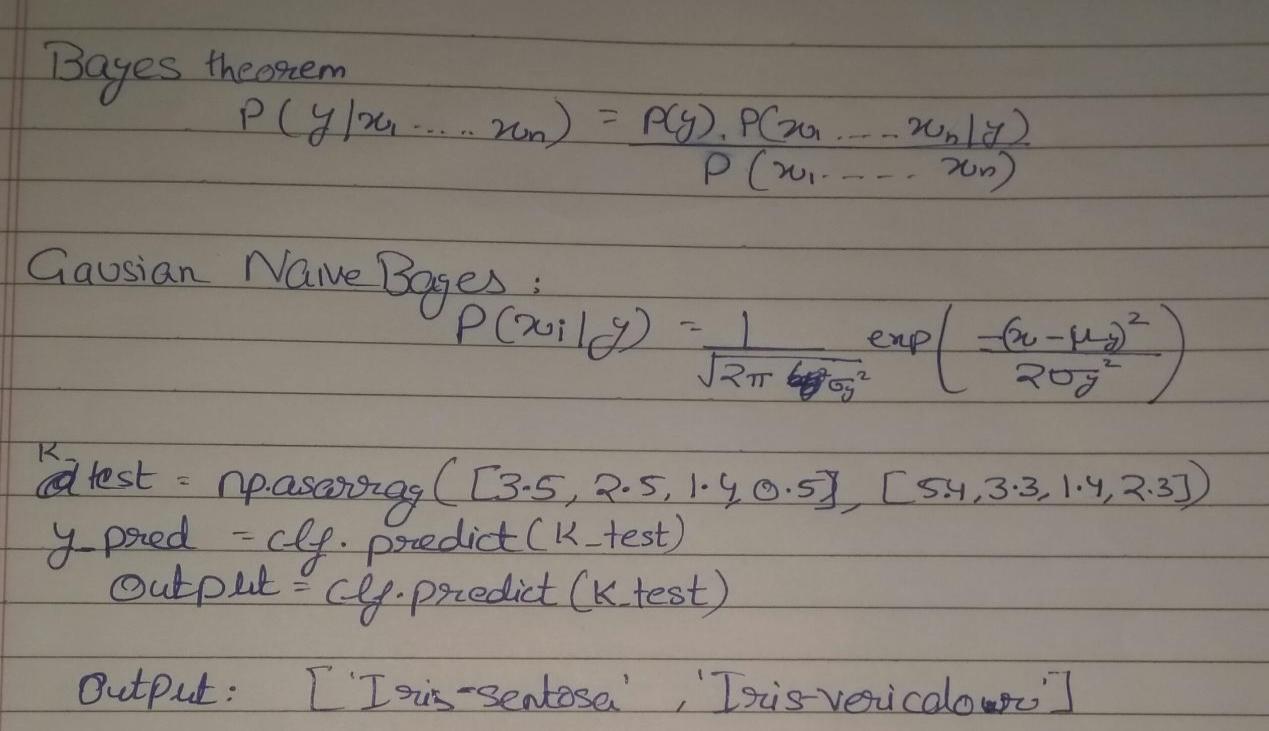
Step 2: Create Likelihood table by finding the probabilities.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

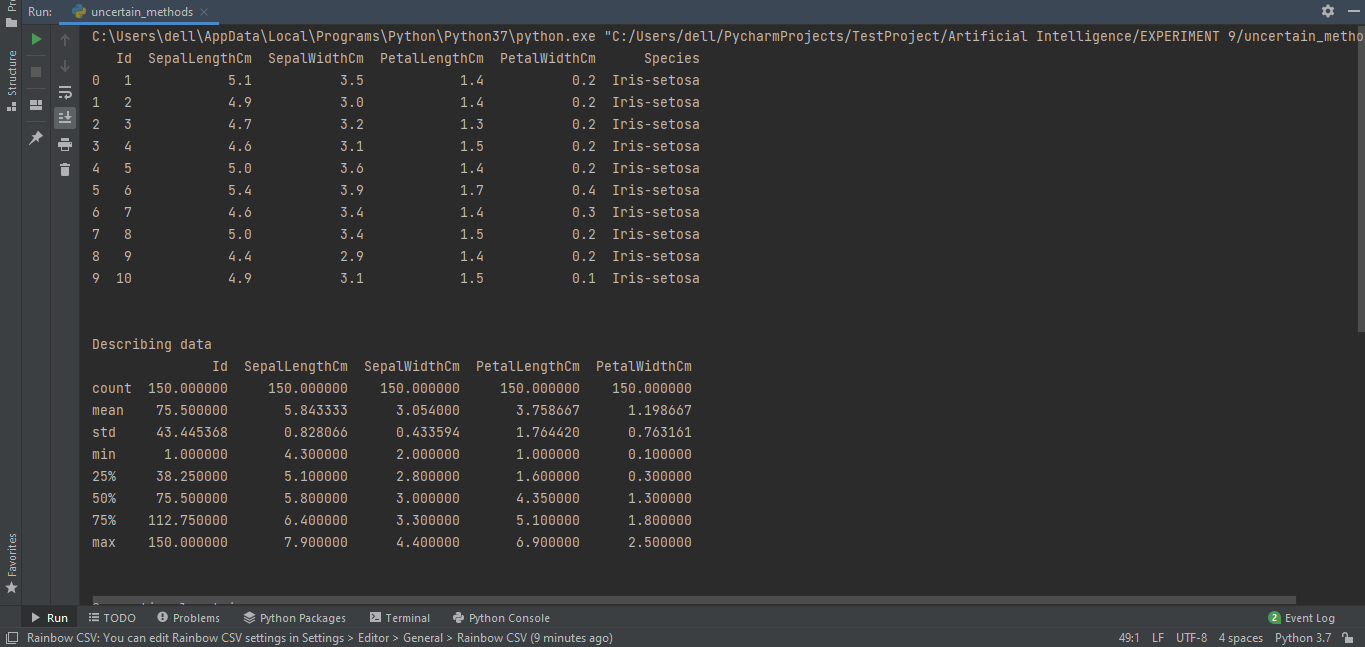
**Source code:**

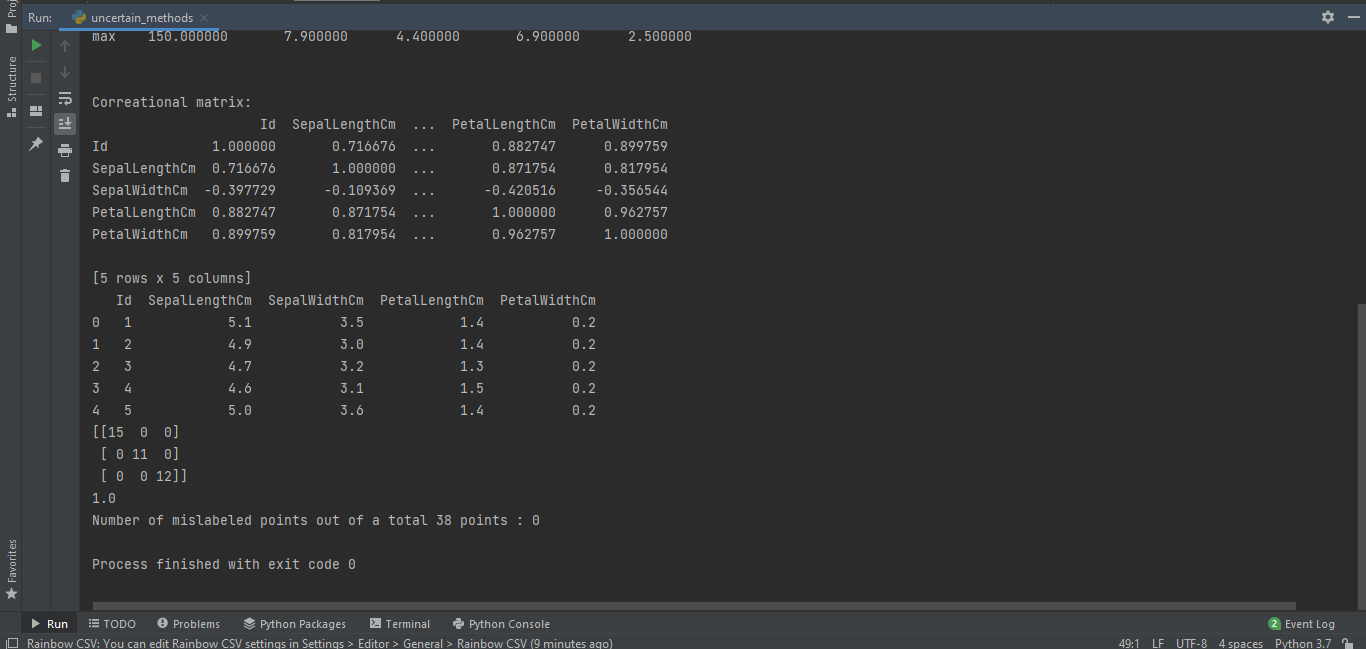
from sklearn.metrics import f1\_score  
from sklearn.metrics import confusion\_matrix  
import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.naive\_bayes import GaussianNB  
data = pd.read\_csv("iris.csv", delimiter=',')  
print(data.head(10))  
  
# describe stats summary of data  
print("\n\nDescribing data")  
print(data.describe())  
  
# correlation matrix  
print("\n\nCorreational matrix:")  
print(data.corr(method='pearson'))  
  
X = data.drop("Species", axis=1)  
print(X.head())  
y = data['Species']  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42, shuffle=True)  
  
clf = GaussianNB()  
clf.fit(X\_train, y\_train)  
y\_pred = clf.predict(X\_test)  
print(confusion\_matrix(y\_test, y\_pred))  
print(f1\_score(y\_test, y\_pred, average='macro'))  
print("Number of mislabeled points out of a total %d points : %d" % (X\_test.shape[0], (y\_test != y\_pred).sum()))

**Test Cases:**

****

**Verification:**





**Result:**

The correlation matrix using Naive Bayes from the data-set provided, was analyzed and an optimal solution was devised. This solution was the coded & tested against various test cases and documented.