# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

| Algorithm: **Naïve-Bayes Algorithm** | |
|---|---|
| USN : 1MS17CS130 | NAME: UTKARSHA VERMA |
| USN : 1MS17CS145 | NAME: ANU VIKRAM K |

**Description of the Algorithm:**

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification tasks. The crux of the classifier is based on the Bayes theorem. Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred, which is as per the following formula:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where A and B are events and P(B) is unknown.

- Basically, we are trying to find the probability of event A, given the event B is true. Event B is also termed as evidence.
- P(A) is the prior probability of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
- P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

The Naive assumption is applied to the Bayes' theorem, which is independence among the features.

$$P(A, B) = P(A)P(B)$$

RAMAIAH Institute of Technology

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

---

**Algorithm Pseudocode:**

Input:

Training dataset T,

F= ($f_1$, $f_2$, $f_3$,.., $f_n$) // value of the predictor variable in testing dataset.

Output:

A class of testing dataset.

Step:

1. Read the training dataset T;
2. Calculate the mean and standard deviation of the predictor variables in each class;
3. Repeat

   Calculate the probability of $f_i$ using the gauss density equation in each class;

   Until the probability of all predictor variables ($f_1$, $f_2$, $f_3$,.., $f_n$) has been calculated.
4. Calculate the likelihood for each class;
5. Get the greatest likelihood;

RAMAIAH Institute of Technology

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

---

**Data set Used: (Attach Screenshot of the few rows):**

| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| .. | ... | ... | ... | ... | ... |
| 145 | 6.7 | 3.0 | 5.2 | 2.3 | virginica |
| 146 | 6.3 | 2.5 | 5.0 | 1.9 | virginica |
| 147 | 6.5 | 3.0 | 5.2 | 2.0 | virginica |
| 148 | 6.2 | 3.4 | 5.4 | 2.3 | virginica |
| 149 | 5.9 | 3.0 | 5.1 | 1.8 | virginica |

[150 rows x 5 columns]

**Challenges faced during the implementation of the program:**

Understanding the working of the Naive Bayes Algorithm and applying that theoretical logic into a code that implements the same.

RAMAIAH Institute of Technology

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

---

**Code:**

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import matplotlib.colors as colors
import seaborn as sns
import itertools
from scipy.stats import norm
import scipy.stats
from sklearn.naive_bayes import GaussianNB


%matplotlib inline
sns.set()
```

**Loading the data set and plotting the data using scatter plot**

```
#Load the data set
iris = sns.load_dataset("iris")
print(iris)
iris = iris.rename(index = str, columns = {'sepal_length':'1_sepal_length','sepal_width':'2_sepal_width',
'petal_length':'3_petal_length', 'petal_width':'4_petal_width'})


#Plot the scatter of sepal length vs sepal width
```

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

---

```
sns.FacetGrid(iris, hue="species", height=7) .map(plt.scatter,"1_sepal_length", "2_sepal_width", )
.add_legend()

plt.title('Scatter plot')

df1 = iris[["1_sepal_length", "2_sepal_width",'species']]
```

## Using sklearn.naive_bayes and printing accuracy

```
from sklearn.naive_bayes import GaussianNB


#Setup X and y data

X_data = df1.iloc[:,0:2]

y_labels = df1.iloc[:,2].replace({'setosa':0,'versicolor':1,'virginica':2}).copy()


#Fit model

model_sk = GaussianNB(priors = None)

model_sk.fit(X_data,y_labels)


#Sklearn accuracy

display(model_sk.score(X_data,y_labels))
```

## Plotting 2D classified fitted model

```
# Our 2-dimensional classifier will be over variables X and Y

N = 100

X = np.linspace(4, 8, N)

Y = np.linspace(1.5, 5, N)

X, Y = np.meshgrid(X, Y)
```

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

---

```
#fig = plt.figure(figsize = (10,10))

#ax = fig.gca()

color_list = ['Blues','Greens','Reds']

my_norm = colors.Normalize(vmin=-1.,vmax=1.)

g = sns.FacetGrid(iris, hue="species", height=10, palette = 'colorblind') .map(plt.scatter,
"1_sepal_length", "2_sepal_width",)  .add_legend()

my_ax = g.ax


#Computing the predicted class function for each value on the grid

zz = np.array(  [model_sk.predict( [[xx,yy]])[0] for xx, yy in zip(np.ravel(X), np.ravel(Y)) ] )


#Reshaping the predicted class into the meshgrid shape

Z = zz.reshape(X.shape)

#Plot the filled and boundary contours

my_ax.contourf( X, Y, Z, 2, alpha = .1, colors = ('blue','green','red'))

my_ax.contour( X, Y, Z, 2, alpha = 1, colors = ('blue','green','red'))


# Add axis and title

my_ax.set_xlabel('Sepal length')

my_ax.set_ylabel('Sepal width')

my_ax.set_title('Gaussian Naive Bayes decision boundaries')

plt.show()
```

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

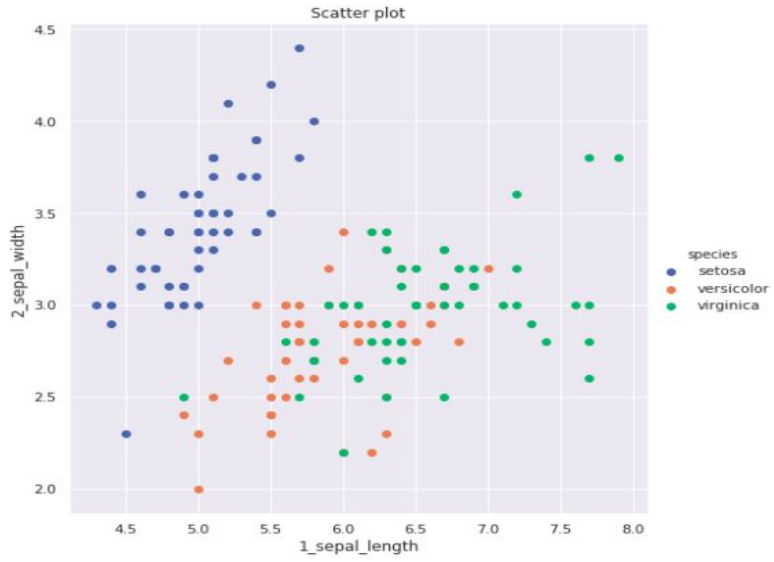
**Semester: VI** **Date: 23/05/2020**

---

**Output: (Screen shots)**

| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| .. | ... | ... | ... | ... | ... |
| 145 | 6.7 | 3.0 | 5.2 | 2.3 | virginica |
| 146 | 6.3 | 2.5 | 5.0 | 1.9 | virginica |
| 147 | 6.5 | 3.0 | 5.2 | 2.0 | virginica |
| 148 | 6.2 | 3.4 | 5.4 | 2.3 | virginica |
| 149 | 5.9 | 3.0 | 5.1 | 1.8 | virginica |

[150 rows x 5 columns]




**Using sklearn.naive_bayes and printing accuracy**

In [3]:
```
from sklearn.naive_bayes import GaussianNB

#Setup X and y data
X_data = df1.iloc[:,0:2]
y_labels = df1.iloc[:,2].replace({'setosa':0,'versicolor':1,'virginica':2}).copy()

#Fit model
model_sk = GaussianNB(priors = None)
model_sk.fit(X_data,y_labels)

#Sklearn accuracy
display(model_sk.score(X_data,y_labels))
```

```
0.78
```

# Ramaiah Institute of Technology

(Autonomous Institute, Affiliated to VTU)

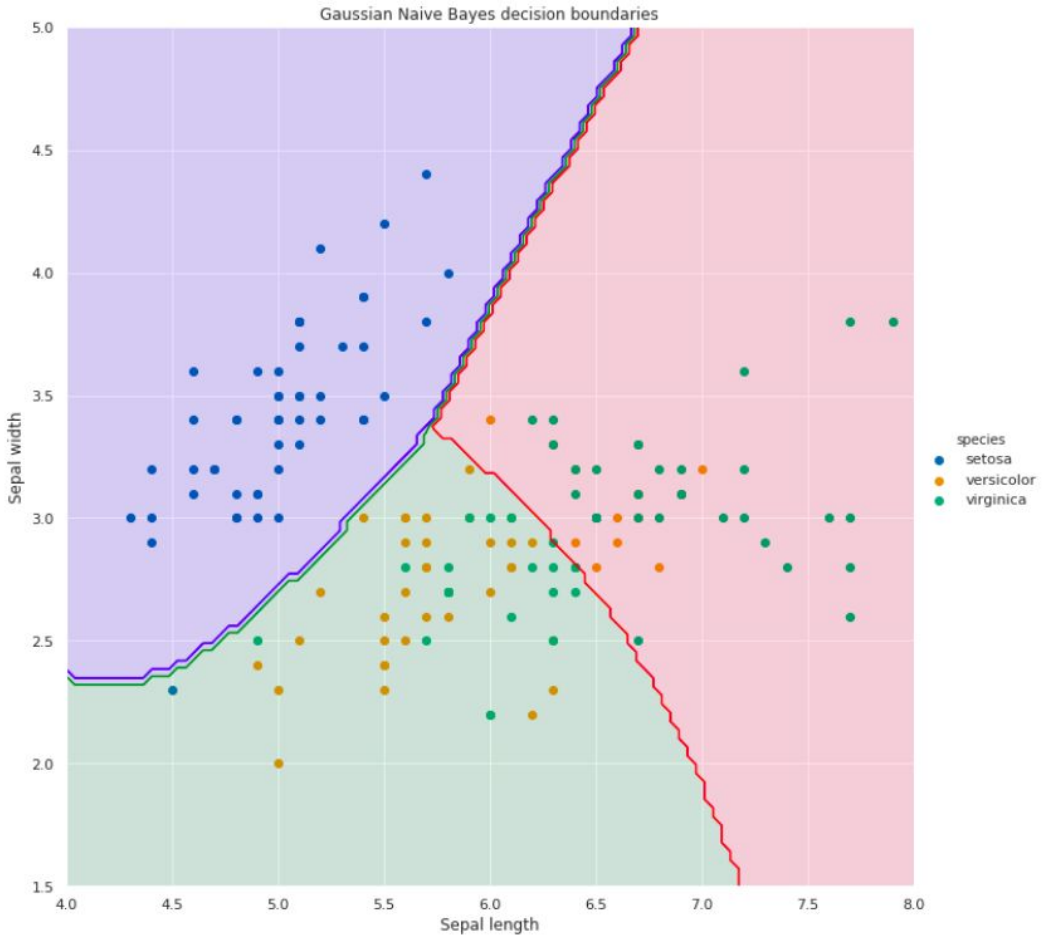
**Department of Computer Science & Engineering**

**Machine Learning (CSE11)**

**Week #: 9**

**Semester: VI** **Date: 23/05/2020**

---




**References:**


https://machinelearningmastery.com/naive-bayes-for-machine-learning/

https://www.edureka.co/blog/naive-bayes-tutorial/

https://xavierbourretsicotte.github.io/Naive_Bayes_Classifier.html