**PRACTICAL NO – 1**

**TITLE- Naive Bayes**

**THEORY-**- It comes from family of simple "probabilistic classifiers"

* based on Bayes theorem

**Advantages**: requires a small number of training data to estimate the parameters necessary for classification

Bayes Theorem 🡪

P(A|B) = [P(B|A) \*P(A)] / P(B)

P(A) 🡪 prior probability

P(B) 🡪 marginal likelihood

P(B|A) 🡪 likelihood

P(A|B) 🡪 posterior likelihood

It’s just the likelihood probability of the given data

**PROBLEM STATEMENT**- To develop an algorithm for Naïve Bayes Classifier

**ALGORITHM-**

**1.Importing the libraries.**

**2.Import the datasets.**

**3.Using the data pre-processing for cleaning and optimising data**

**4. Fitting Naïve Bayes to the dataset**

**5. Visualising the Results**

**6.End**

**CODE-**

# -\*- coding: utf-8 -\*-

"""

Created on Wed Sep 11 21:43:51 2019

Naive Bayes Based Classifier

@author: RoKu

"""

#import libraries

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

#import the dataset

dataset = pd.read\_csv('data.csv')

X=dataset.iloc[:,[2,3]].values

y=dataset.iloc[:,-1].values

#clean data

#encode data

#preprocess

#split the dataset

from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.25,random\_state=0)

#go for feature scaling

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.fit\_transform(X\_test)

#Fit the Naive bayes classifier model to training set

from sklearn.naive\_bayes import GaussianNB

classifier = GaussianNB()

classifier.fit(X\_train,y\_train)

#predicting the test set results

y\_pred = classifier.predict(X\_test)

#make confusion matrix

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test,y\_pred)

tn, fp, fn, tp = confusion\_matrix(y\_test,y\_pred).reshape(-1)

# Visualising the Test set results

from matplotlib.colors import ListedColormap

X\_set, y\_set = X\_test, y\_test

X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 1, stop = X\_set[:, 0].max() + 1, step = 0.01),

np.arange(start = X\_set[:, 1].min() - 1, stop = X\_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),

alpha = 0.75, cmap = ListedColormap(('red', 'green')))

plt.xlim(X1.min(), X1.max())

plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y\_set)):

plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1],

c = ListedColormap(('red', 'green'))(i), label = j)

plt.title('Naive Bayes Based (Testining set)')

plt.xlabel('Age')

plt.ylabel('Estimated Salary')

plt.legend()

plt.show()

**RESULTS**-

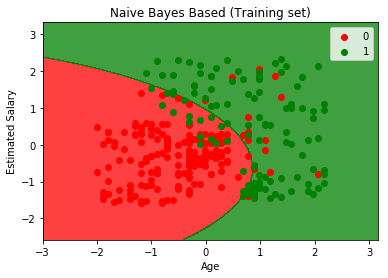
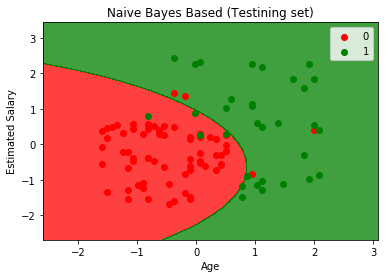


Fig.1- Visualization.



**DISCUSSIONS-**

We firstly discussed the Bayes theorem and the mathematics behind it. Then we went on for how it is beneficial to use the Bayes theorem for classification and how we can use it in machine learning terms. Then the algorithm was discussed and we went on for actual implementation.

**CONCLUSIONS-** Naïve Bayes may not be the best of classifiers but it works great with the data that is purely in numeric form.

|  |  |
| --- | --- |
| **SUBMISSION DATE**-  11 Sept 2019 | **SIGN OF COURSE INSTRUCTOR**- |
| **ROLL NO OF THE STUDENT-**  **I-62** | **NAME OF THE STUDENT-**  **Rohit Kulkarni** |