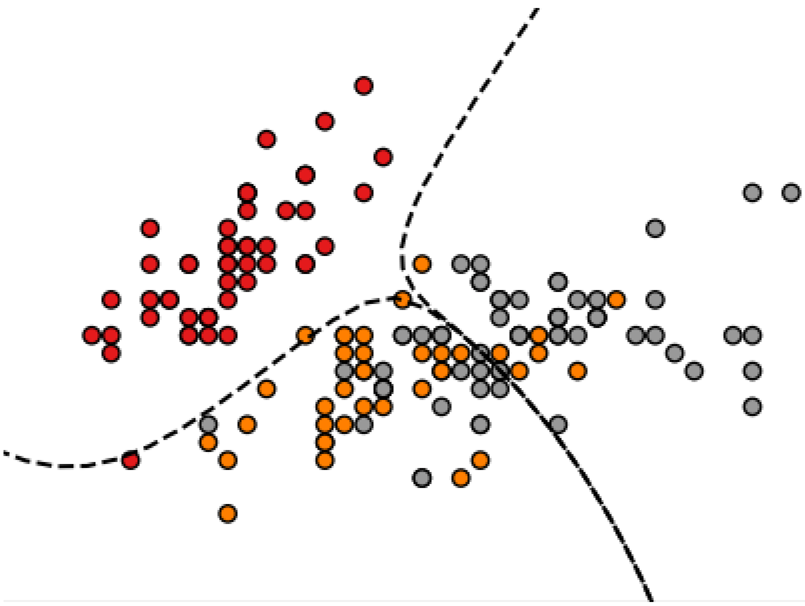


Bayesian Decision

Assignment 2 (applied machine learning)

By
Abdelmageed Ahmed Abdelmageed Ahmed
Mohamed Sayed Abdelwahab Hussien

Part 1

Setosa				Versicolour				Virginica			
SL	SP	PL	PW	SL	SW	PL	PW	SL	SW	PL	PW
5.1	3.4	1.5	0.2	4.9	2.4	3.3	1	6.5	3	5.8	2.2
5	3.4	1.5	0.2	5.7	3	4.2	1.2	7.7	2.6	6.9	2.3
4.8	3	1.4	0.1	5.4	3	4.5	1.5				
5	3	1.4	0.2	5.6	2.5	3.9	1.1				

$$Mean = \frac{sum\ of\ the\ terms}{number\ of\ terms} \rightarrow \mu = \frac{\sum X}{N}$$

$$\text{variance} = \sigma^2 = \frac{\sum (x_r - \mu)^2}{n}$$

Mean	4.975	3.2	1.45	0.175	5.4	2.725	3.975	1.2	7.1	2.8	6.35	2.25
Variance	0.011875	0.04	0.0025	0.001875	0.095	0.076875	0.196875	0.035	0.36	0.04	0.3025	0.0025

Prior Probability 0.333333 for each class

$$Likelihood = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}}$$

Test point 1

Sepal length=5.7, Sepal width=2.8, Petal length=4.5, Petal width=1.3

For Setosa

$$P(SL = 5.7|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.011875}} e^{\frac{-(5.7-4.975)^2}{2*0.011875}} = 8 \cdot 95 * 10^{-10}$$

$$P(SW = 2.8|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.04}} e^{\frac{-(2.8-3.2)^2}{2*0.04}} = 0.26995$$

$$P(PL = 4.5|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.0025}} e^{\frac{-(4.5-1.45)^2}{2*0.0025}} = 0$$

$$P(PW = 1.3|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.001875}} e^{\frac{-(1.3-0.175)^2}{2*0.001875}} = 0$$

P (Species = 0 | SL = 5.7, SW = 2.8, PL = 4.5, PW = 1.3) =

Prior Probability * P (SL = 5.7 | Species = 0) * P (SW = 2.8 | Species = 0) * P (PL = 4.5 | Species = 0) * (PW = 1.3 | Species = 0)

$$= \frac{2}{5} * 8 \cdot 95 * 10^{-10} * 0.26995 * 0 * 0 = 0$$

For versicolor

$$P(SL = 5.7|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.095}}e^{\frac{-(5.7-5.4)^2}{2*0.095}} = 0.806$$

$$P(SW = 2.8|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.07687}}e^{\frac{-(2.8-2.725)^2}{2*0.07687}} = 1.3872$$

$$P(PL = 4.5|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.196875}}e^{\frac{-(4.5-3.975)^2}{2\ 0*196875}} = 0.4464$$

$$P(PW = 1.3|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.035}}e^{\frac{-(1.3-1.2)^2}{2*0.035}} = 1.848$$

P (Species = 21| SL = 5.7, SW = 2.8, PL = 4.5, PW = 1.3) =

Prior Probability * P (SL = 5.7 | Species = 1) * P (SW = 2.8 | Species = 1) * P (PL = 4.5 | Species = 1) * (SW = 1.3 | Species = 1)

$= \frac{2}{5}$ **** 0.806 *1.3872 *0.4464 * 1.848 = 0.3688***

For Virginica

$$P(SL = 5.7|Species = 2) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.36}}e^{\frac{-(5.7-7.1)^2}{2*0.36}} = 0.0437$$

$$P(SW = 2.8|Species = 2) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.04}}e^{\frac{-(2.8-2.8)^2}{2*0.04}} = 1.9947$$

$$P(PL = 4.5|Species = 2) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.3025}}e^{\frac{-(4.5-6.35)^2}{2\ 0*3025}} = 0.0025$$

$$P(PW = 1.3|Species = 2) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi*0.0025}}e^{\frac{-(1.3-2.25)^2}{2*0.0025}} = 3.249 * 10^{-78}$$

P (Species = 2 | SL = 5.7, SW = 2.8, PL = 4.5, PW = 1.3) =

Prior Probability * P (SL = 5.7 | Species = 2) * P (SW = 2.8 | Species = 2) * P (PL = 4.5 | Species = 2) * (PW = 1.3 | Species = 2)

$= \frac{1}{5}$ **** 0.0437 * 1.9947 *0.0025 *3.249 *** $\mathbf{10^{-78}}$ ***= 1.416*** $\mathbf{10^{-82}}$

P (Species = 0 | SL = 5.7, SW = 2.8, PL = 4.5, PW = 1.3) = 0

P (Species = 1 | SL = 5.7, SW = 2.8, PL = 4.5, PW = 1.3) = 0.3688

P (Species = 2 | SL = 5.7, SW = 2.8, PL = 4.5, PW = 1.3) = 3.249 $\mathbf{10^{-82}}$

As Posterior probability of class 1 is the highest probability so the predicted class for Sepal length=5.7, Sepal width=2.8, Petal length=4.5, Petal width=1.3 is Class 1 (versicolor)

Test point 2

Sepal length=5.4, Sepal width=3.9, Petal length=1.3, Petal width=0.4

For Setosa

$$P(SL = 5.4|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.011875}}e^{\frac{-(5.4-4.975)^2}{2*0.011875}} = 1.822 * 10^{-3}$$

$$P(SW = 3.9|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.04}}e^{\frac{-(3.9-3.2)^2}{2*0.04}} = 4.363 * 10^{-3}$$

$$P(PL = 1.3|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.0025}}e^{\frac{-(1.3-1.45)^2}{2*0.0025}} = 0.0886$$

$$P(PW = 0.4|Species = 0) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.001875}}e^{\frac{-(0.4-0.175)^2}{2*0.001875}} = 1.263 * 10^{-5}$$

P (Species = 0 | SL = 5.4, SW = 3.9, PL = 1.3, PW = 0.4) =

Prior Probability * P (SL = 5.4 | Species = 0) * P (SW = 3.9 | Species = 0) * P (PL = 1.3 | Species = 0) * (PW = 0.4 | Species = 0)

$$= \frac{2}{5} * \mathbf{1.822 * 10^{-3} * 4.363 * 10^{-3} * 0.0886 * 1.263 * 10^{-5} = 3.675 * 10^{-12}}$$

For versicolor

$$P(SL = 5.4|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.095}}e^{\frac{-(5.4-5.4)^2}{2*0.095}} = 1.294$$

$$P(SW = 3.9|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.07687}}e^{\frac{-(3.9-2.725)^2}{2*0.07687}} = 1.811 * 10^{-4}$$

$$P(PL = 1.3|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.196875}}e^{\frac{-(1.3-3.975)^2}{2\,0*196875}} = 1.15 * 10^{-8}$$

$$P(PW = 0.4|Species = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.035}}e^{\frac{-(0.4-1.2)^2}{2*0.035}} = 2.28 * 10^{-4}$$

P (Species = 1 | SL = 5.4, SW = 3.9, PL = 1.3, PW = 0.4) =

Prior Probability * P (SL = 5.4 | Species = 1) * P (SW = 3.9 | Species = 1) * P (PL = 1.3 | Species = 1) * (PW = 0.4 | Species = 1)

$$= \frac{2}{5} * \mathbf{2.28 * 10^{-4} * 1.15 * 10^{-8} * 1.811 * 10^{-4} * 1.294 = 2.4577 * 10^{-16}}$$

For Virginica

$$P(SL = 5.4|Species = 2) \ = \ \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.36}} e^{\frac{-(5.4-7.1)^2}{2*0.36}} \ = \ 0.012$$

$$P(SW = 3.9|Species = 2) \ = \ \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.04}} e^{\frac{-(3.9-2.8)^2}{2*0.04}} \ = \ 5.38 * 10^{-7}$$

$$P(PL = 1.3|Species = 2) \ = \ \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.3025}} e^{\frac{-(1.3-6.35)^2}{2\ 0*3025}} = 3.579 * 10^{-19}$$

$$P(PW = 0.4|Species = 2) \ = \ \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(X-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi * 0.0025}} e^{\frac{-(0.4-2.25)^2}{2*0.0025}} = 0$$

P (Species = 2 | SL = 5.4, SW = 3.9, PL = 1.3, PW = 0.4) =

Prior Probability * P (SL = 5.4 | Species = 2) * P (SW = 3.9 | Species = 2) * P (PL = 1.3 | Species = 12) * (PW = 0.4 | Species = 12)

$$= \frac{1}{5} \ * \ 0.012 * 5.38 * 10^{-7} * 3.579 * 10^{-19} * 0 = 0$$

P (Species = 0 | SL = 5.4, SW = 3.9, PL = 1.3, PW = 0.4) = $\mathbf{3.675 * 10^{-12}}$

P (Species = 1 | SL = 5.4, SW = 3.9, PL = 1.3, PW = 0.4) = 2.4577 * $\mathbf{10^{-16}}$

P (Species = 2 | SL = 5.4, SW = 3.9, PL = 1.3, PW = 0.4) = 0

As Posterior probability of class 0 is the highest probability so the predicted class for Sepal length=5.4, Sepal width=3.9, Petal length=1.3, Petal width=0.4 is Class 0 (Setosa)

Part 2

1. ***Load dataset***
2. ***Drop the petal length and petal width features to form a 2D Iris dataset***

```
[ ] iris = datasets.load_iris()
    X, y = iris.data, iris.target
    X_new = np.delete(X,[2,3],1)
    X_train, X_test, y_train, y_test = train_test_split(X_new, y, random_state=0)
```

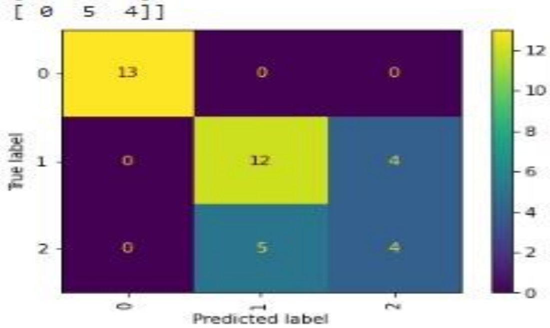
```
[ ] print(X.shape)
    print(X_new.shape)
```

```
(150, 4)
(150, 2)
```

3. ***Apply Naïve Bayes Classifier to get training and testing accuracy***

```
NB_model = GaussianNB()
NB_model.fit(X_train,y_train)
y_predic_train = NB_model.predict(X_train)
y_predic_test = NB_model.predict(X_test)
train_accuracy = accuracy_score(y_train, y_predic_train)
test_accuracy = accuracy_score(y_test, y_predic_test)
print("\n training accuracy :  "  ,round(train_accuracy,3),"\n")
print("\n testing accuracy :  "  ,round(test_accuracy,3),"\n")
print(classification_report(y_test, y_predic_test))
print('\nConfusion Matrix:\n')
print(confusion_matrix(y_test, y_predic_test))
plot_confusion_matrix(NB_model, X_test, y_test, xticks_rotation='vertical');
```

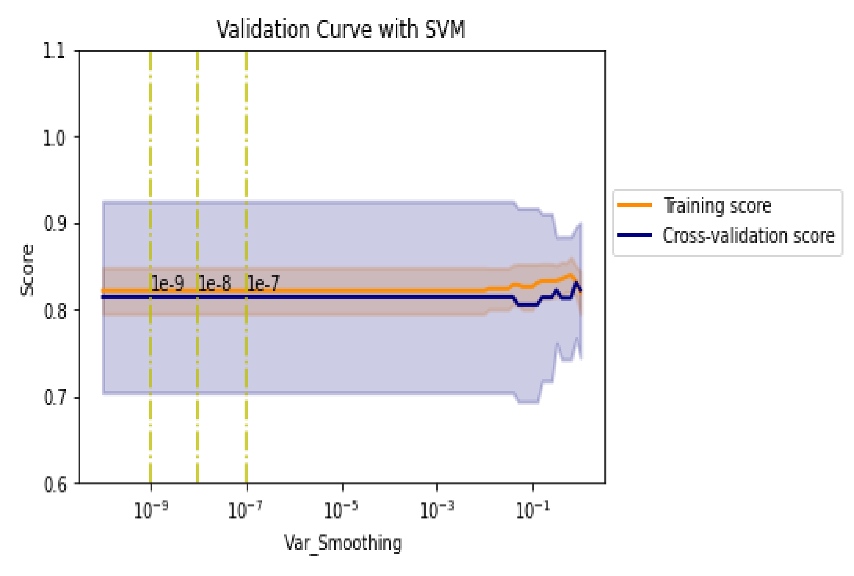
```
 training accuracy :   0.812

 testing accuracy :   0.763

              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       0.71      0.75      0.73        16
           2       0.50      0.44      0.47         9

    accuracy                           0.76        38
   macro avg       0.74      0.73      0.73        38
weighted avg       0.76      0.76      0.76        38


Confusion Matrix:

[[13  0  0]
 [ 0 12  4]
 [ 0  5  4]]
```

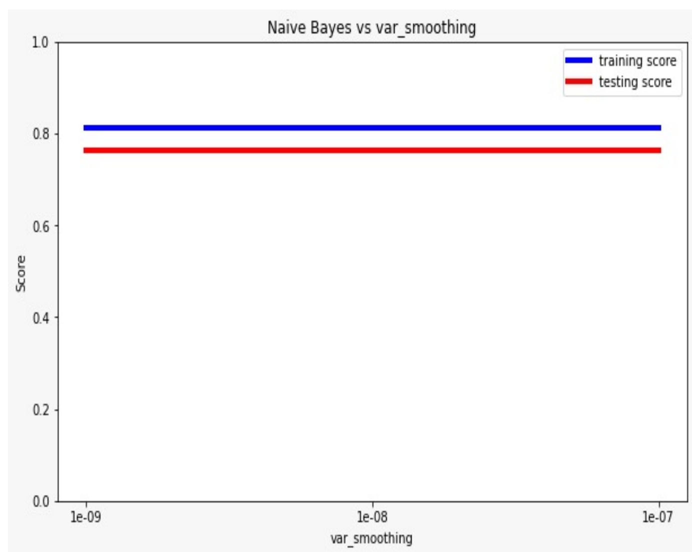


4. ***Tune hyperparameters of Naive Bayes Classifier (i.e., var_smoothing). Try var_smoothing as 1e-9, 1e-8, 1e-7. Plot accuracy vs var_smoothing curve for training and testing set***

```
[27] def build_NBmodel(smoothing, X_train, X_test, y_train, y_test):
         model = GaussianNB(var_smoothing = smoothing)
         model.fit(X_train,y_train)

         y_pred_train = model.predict(X_train)
         train_accuracy = accuracy_score(y_train ,y_pred_train)

         y_pred = model.predict(X_test)
         test_accuracy = accuracy_score(y_test, y_pred)

         print("train accuracy :  "  ,round(train_accuracy,3),"\n")
         print("testing accuracy :  "  ,round(test_accuracy,3),"\n")

         return train_accuracy ,test_accuracy
```

var_smoothing is a stability calculation to widen (or smooth) the curve and therefore account for more samples that are further away from the distribution mean.
As shown in the next figures the var_smoothing parameter doesn't affect the accuracy of the 3 models, they have the same training and testing accuracy. We found that the accuracy started to change when the smoothing variance equal 0.01.





5- Develop Risk-based Bayesian Decision Theory Classifier (RBDTC)

In fit function*: We used check_X_y sklearn method to check for consistent length, non-empty and nan values before fitting data to the model.*
In predict function*: we create function that enable the classifier to predict text format y*

```
class BayesianDecisionTheoryClassifier(BaseEstimator, ClassifierMixin):
    def __init__(self, estimator, utilityMat):
        self.estimator = estimator
        self.utilityMat = utilityMat

    def fit(self, X, y):
      # check_X_y method check for consistent length, X is checked
      #to be non-empty and check for nan values.
        cleaned_data= check_X_y(X,y)
        self.classes_ = np.unique(cleaned_data[1])
        self.estimator_ = clone(self.estimator).fit(cleaned_data[0],
                                                    cleaned_data[1])
        return self

    def predict_proba(self, X):
        check_is_fitted(self)
        prob = self.estimator_.predict_proba(X)
        probList = [(prob * self.utilityMat[index]).sum(axis=1).reshape((-1, 1))
                    for index, c in enumerate(self.classes_)]
        prob = np.hstack(probList)
        return prob

    def predict(self, X):
        pred = self.predict_proba(X).argmin(axis=1)
        # enable the classifier to predict text format y
        pred_txt = label_transform(pred)
        return self.classes_[pred], pred_txt
```

6. Apply Risk-based Bayesian Decision Theory Classifier which takes Naïve Bayes Classifier as base estimator and uses risk matrix

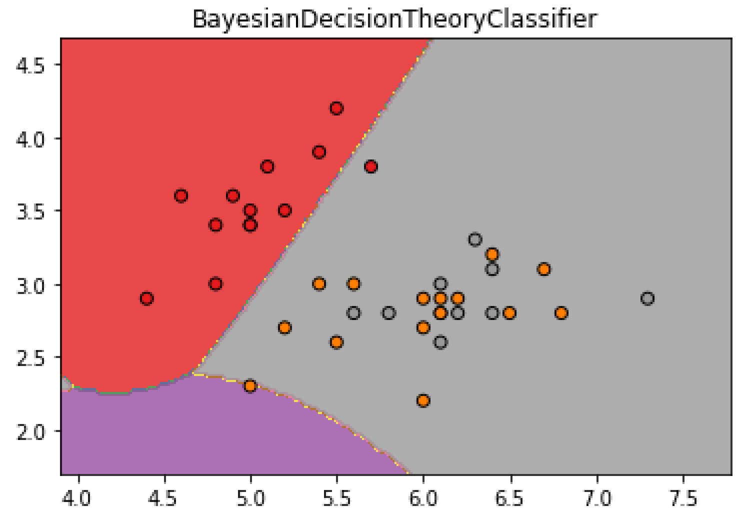
```
[144] utilityMat = np.array([
              [-10, -5, -5],
              [-5, -10, -5],
              [-5, -5, -100]])
      bdtc = BayesianDecisionTheoryClassifier(GaussianNB(), utilityMat)
      bdtc.fit(X_train, y_train)
      yPred, y_prd_txt = bdtc.predict(X_test)
```

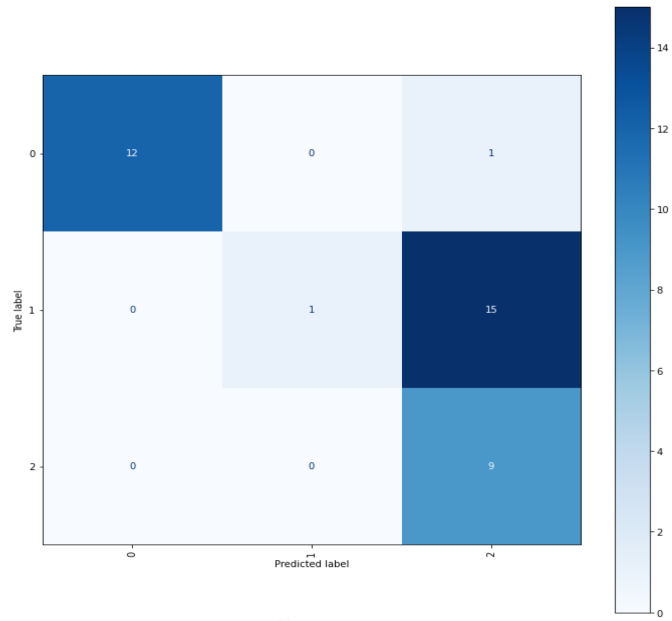
```
[172] print(yPred)
      print(y_prd_txt)
```

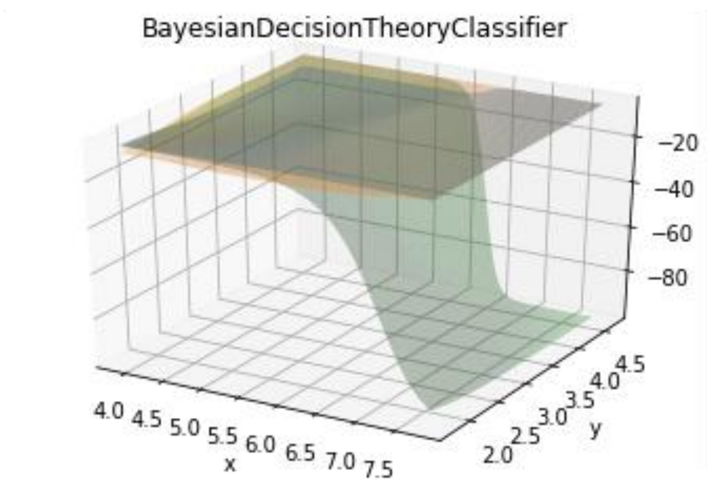
```
[2 2 0 2 0 2 0 2 2 2 2 2 2 2 2 0 2 2 0 0 2 2 0 0 2 0 0 2 1 0 2 2 0 2 2 2 2
 2]
['Iris-Virginica', 'Iris-Virginica', 'Iris-Setosa', 'Iris-Virginica', 'Iris-Setosa', 'Iris-Virginica',
```

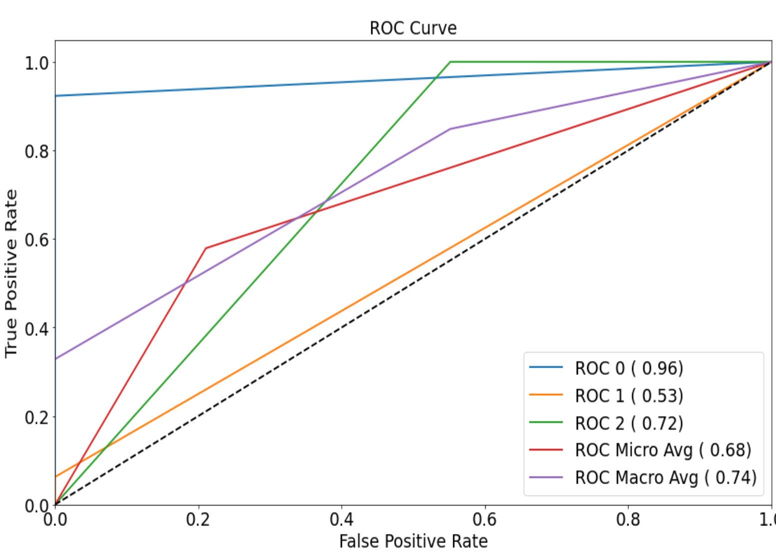
7 - Plot decision boundary and calculate precision, recall and accuracy for training and testing set.

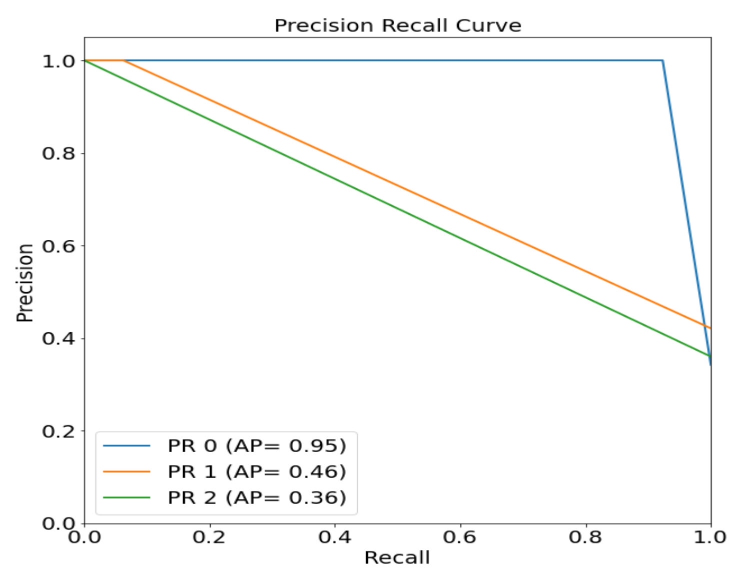
```
[209] plotDecisionBoundary(X_test, y_test, bdtc, 'BayesianDecisionTheoryClassifier', risk = True);
      plot3dProb(X_test, bdtc, 'BayesianDecisionTheoryClassifier');
      results(f'BayesianDecisionTheoryClassifier', y_test, yPred, folder='');
```







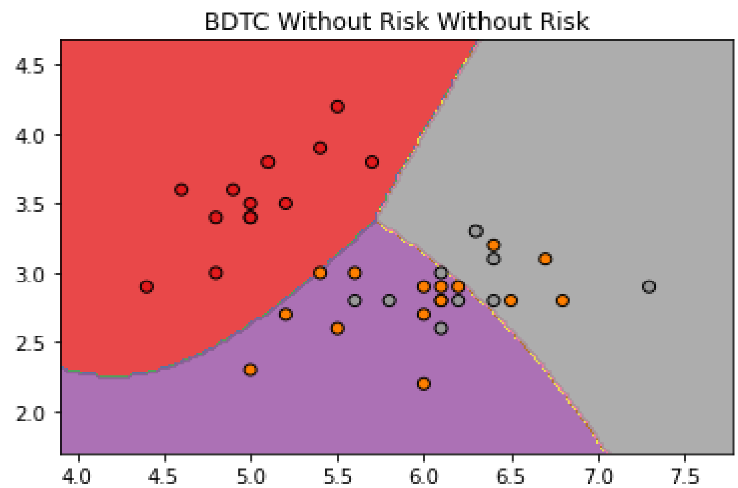


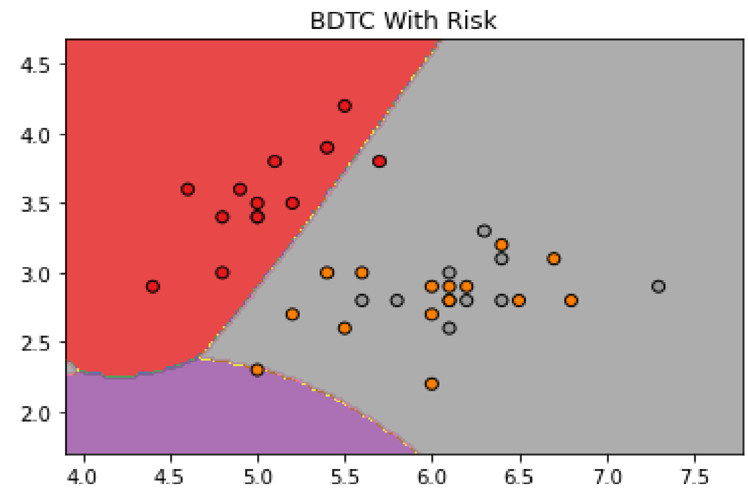


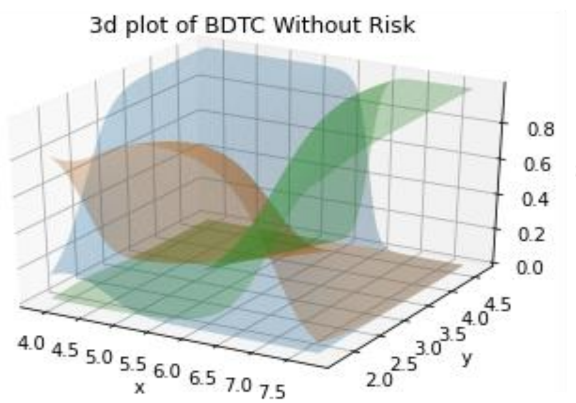
8 - Compare and analysis the performance between NB and RBDTC regarding to their decision boundary, precision, recall, and accuracy

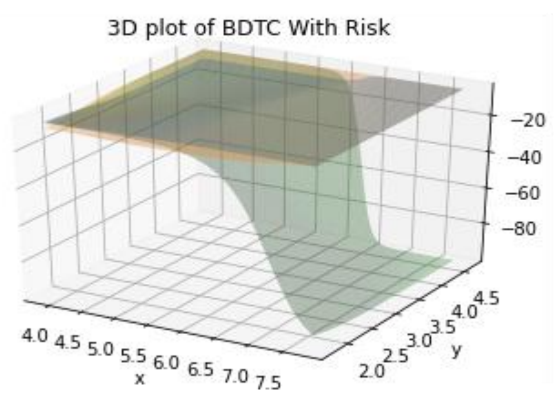
In the left figure (BDTC without risk): the classifier achieved 76% testing accuracy which the versicolor and virginica points are overlapped so the model confused in classify between these two classes,

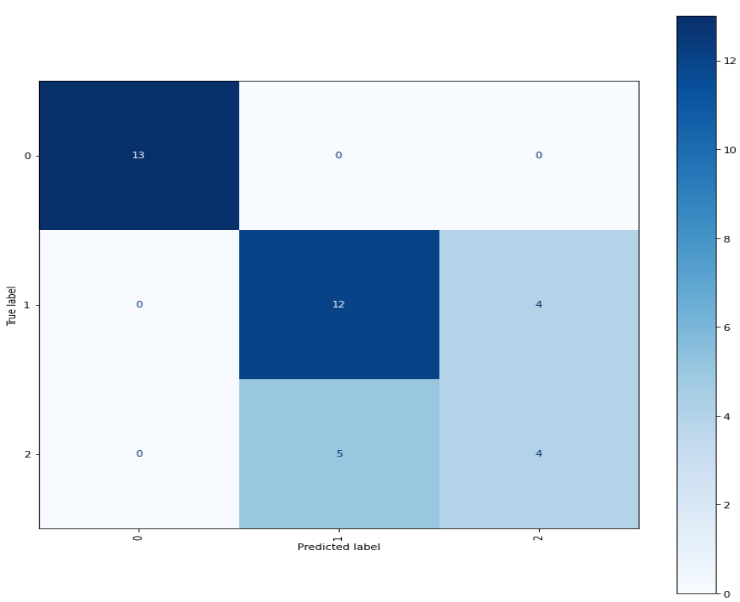
In the right figure (BDTC with risk): the classifier achieved 57% testing accuracy as we used weighted risk matrix that focus on minimize the risk of class 2 (iris_virginica) and maximize its probability which extend the decision region of class 2 (virginica).

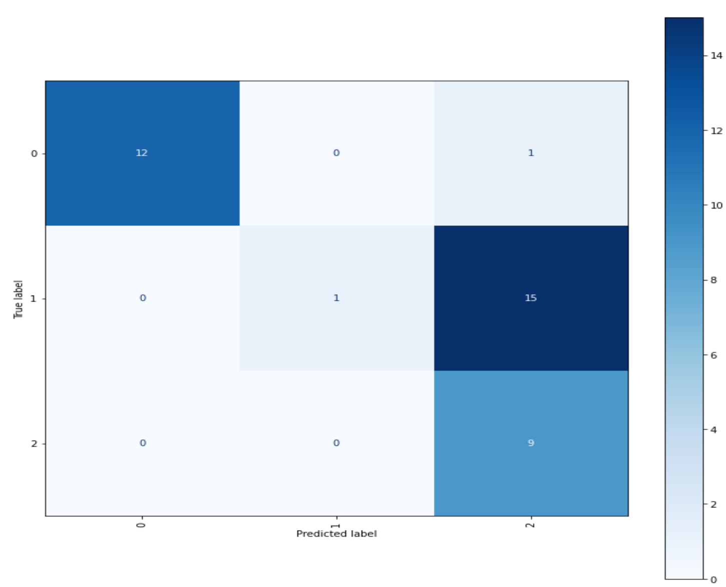




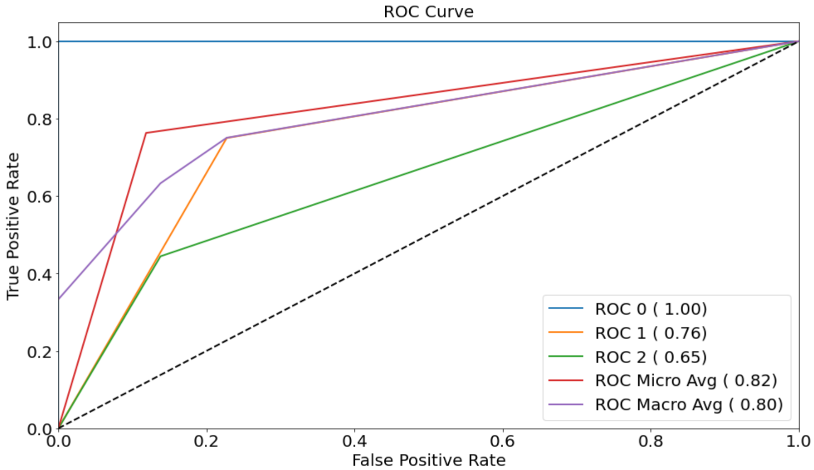


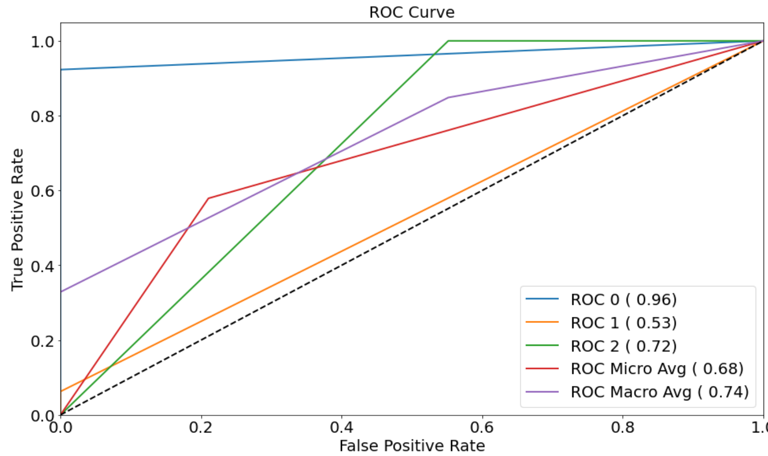




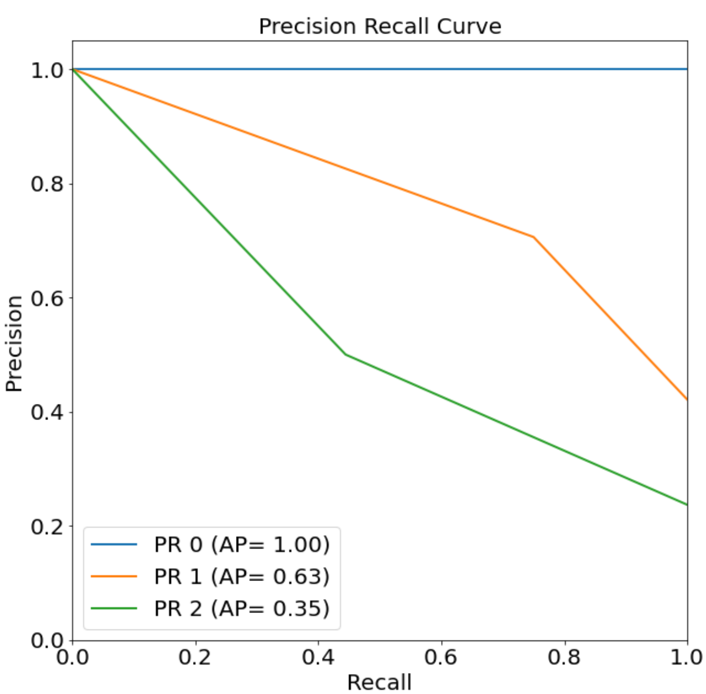


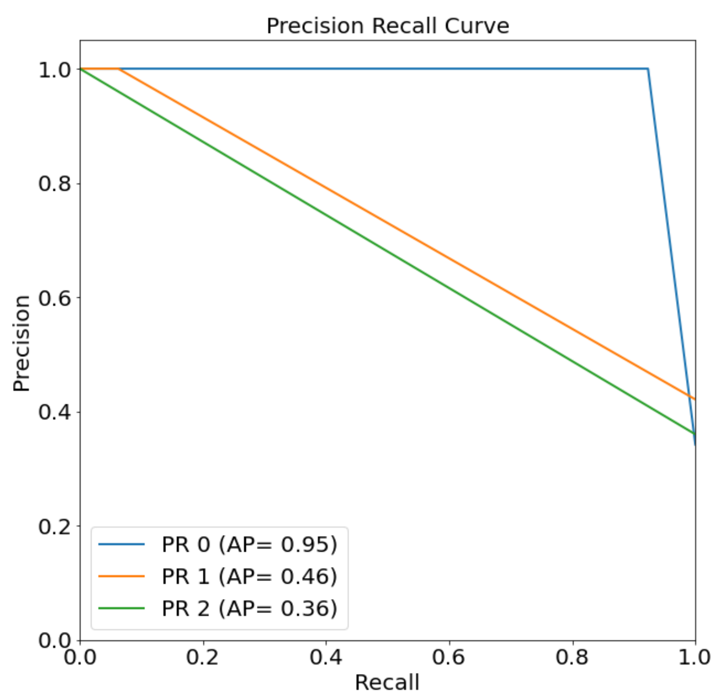
As shown in the left figure (BDTC without risk) we find that the class 2 has 0.65 ROC before adding the risk matrix to the naïve classifier, while it increased to 0.72 when minimize its risk by using risk matrix has minimum weight for class 2(virginica) to minimize its risk.





A macro-average will compute the metric independently for each class and then take the average hence treating all classes equally, whereas a micro-average will aggregate the contributions of all classes to compute the average metric, in our case we found that the ROC macro Avg decreased from 0.80 to 0.74 and also micro Avg decreased from 0.80 to 0.68 as risk matrix damages the overall model accuracy in order to increase accuracy for specific class.





Conclusion

By going through the assignment steps, we had learnt:

- *what is the math behind naive bayes, how to deals with continuous data (normally distributed) and calculate the Posterior probability for classes.*

- *how to implement Naïve bayes classifier and tune its hyperparameters as variance smoothing*

- *Developing risk-based Bayesian decision theory and how it affects the classification results based on risk matrix to focus on specific class.*

- *The risk matrix damages the overall model accuracy in order to increase accuracy for specific class when dealing with overlapped classes.*