### Importing preprocessed Bank Marketing csv

```
bankData =
readtable('BankMarketingForModelTraining.csv','PreserveVariableNames',t
rue)
```

### Splitting the Data into training and testing

```
cv = cvpartition(size(bankData,1),'HoldOut',0.3);
dataUpd = cv.test;
% Separate to training and test data
dataTrain = bankData(~dataUpd,:);


dataTest  = bankData(dataUpd,:);


XTrain= dataTrain(1:end, 1:end-1)
yTrain= dataTrain(1:end,end)


XTest= dataTest(1:end, 1:end-1)
yTest= dataTest(1:end,end)


code ref: https://uk.mathworks.com/help/stats/fitcensemble.html
```

### Training Model

```
finalModelRF=fitcensemble(XTrain,yTrain,'Method','Bag','OptimizeHyperpa
rameters',
{'NumLearningCycles','MaxNumSplits','MinLeafSize','NumVariablesToSample
'},'HyperparameterOptimizationOptions',struct('KFold',5,'Optimizer','gr
idsearch','MaxObjectiveEvaluations',150 ))
```

### Saving Model

```
save('finalModelRF.mat','finalModelRF',"yTest","XTest")
```

### Loading the Trained Random Forest Model

```
predRFLoad=load('finalModelRF.mat')
predRF= predict(predRFLoad.finalModelRF,predRFLoad.XTest)
```

### Plotting Confusion Matrix

```
cm = confusionchart(table2array(predRFLoad.yTest),predRF);
```

**Precision**

```
cmatrixRF= confusionmat(table2array(predRFLoad.yTest),predRF)


precisionRF =  diag(cmatrixRF)./sum(cmatrixRF,2)
overAllPrecRF= mean(precisionRF)
```

**Recall**

```


recallRF =  diag(cmatrixRF)./sum(cmatrixRF,1)';
overAllRecRF= mean(recallRF)
```

**F1 Score**

```
f1ScoreRF =
2*(overAllPrecRF.*overAllRecRF)./(overAllPrecRF+overAllRecRF)


[~,Scores] = predict(predRFLoad.finalModelRF,predRFLoad.XTest)
classNamesBankData=predRFLoad.finalModelRF.ClassNames


rocObj =
rocmetrics(table2array(predRFLoad.yTest),Scores,classNamesBankData);
plot(rocObj)



bankData =
readtable('BankMarketingForModelTraining.csv','PreserveVariableNames',t
rue)
```

**Splitting the Data into training and testing**

```
cv = cvpartition(size(bankData,1),'HoldOut',0.3);
dataUpd = cv.test;
% Separate to training and test data
dataTrain = bankData(~dataUpd,:);


dataTest  = bankData(dataUpd,:);


XTrain= dataTrain(1:end, 1:end-1)
yTrain= dataTrain(1:end,end)


XTest= dataTest(1:end, 1:end-1)
yTest= dataTest(1:end,end)
```

code ref: https://uk.mathworks.com/help/stats/fitcnb.html

```
finalModelNB= fitcnb(XTrain,yTrain,
'OptimizeHyperparameters','auto','HyperparameterOptimizationOptions',st
ruct('KFold',5))
```

### Saving Model

```
save('finalModelNB.mat','finalModelNB', "yTest","XTest")
```

### Loading the Trained Random Forest Model

```
predNBLoad=load('finalModelNB.mat')
predNB= predict(predNBLoad.finalModelNB,predNBLoad.XTest)
```

### Plotting Confusion Matrix

```
cm = confusionchart(table2array(predNBLoad.yTest),predNB);
```

### Precision

```
cmatrixNB= confusionmat(table2array(predNBLoad.yTest),predNB)


precisionNB =  diag(cmatrixNB)./sum(cmatrixNB,2)
overAllPrecNB= mean(precisionNB)
```

### Recall

```


recallNB =  diag(cmatrixNB)./sum(cmatrixNB,1)';
overAllRecNB= mean(recallNB)
```

### F1 Score

```
f1ScoreNB =
2*(overAllPrecNB.*overAllRecNB)./(overAllPrecNB+overAllRecNB)
```

```
[~,Scores] = predict(predNBLoad.finalModelNB,predNBLoad.XTest)
classNamesBankData=predNBLoad.finalModelNB.ClassNames


rocObj =
rocmetrics(table2array(predNBLoad.yTest),Scores,classNamesBankData);
plot(rocObj)
```