Maternity Health Risk Classification

POST-GRADUATE DIPLOMA IN

STATISTICAL METHODS AND ANALYTICS

<u>Submitted By:</u>

Anik Saha

(DST-23/24-022)

INDIAN STATISTICAL INSTITUTE

NORTH EAST CENTRE

INTRODUCTION:

In the realm of data-driven decision-making, predictive modeling plays a pivotal role in assessing and mitigating risks across diverse domains. My project centers on the classification of risk levels within a dataset, leveraging a range of machine learning algorithms to discern patterns and make informed predictions.

OBJECTIVE:

1. To predict whether a women is in maternity risk or not based on certain attributes like Age, Systolic BP, Diastolic BP etc. as given in dataset
2. To incorporate the dataset using different classification models – kNN, Naïve Bayes, Decision Tree and Ensemble Method and check accuracy for each model.
3. To select the model which gives the best accuracy for the given dataset to assess whether a women is in risk level or not.

Experiment Performed:

1. *Data collection:*

I am collected data for this project from open-source website>>> https://archive.ics.uci.edu/dataset/863/maternal+health+risk

2. *Data Overview:*

- ***Dataset has 1014 distinct data points.***
- ***6 explanatory variables and 1 binary response variable which determines maternity risk or not.***

Variables Table

Variable Name	Role	Type	Demographic	Description	Units	Missing Values
Age	Feature	Integer	Age	Any ages in years when a women during pregnant.		no
SystolicBP	Feature	Integer		Upper value of Blood Pressure in mmHg, another significant attribute during pregnancy.		no
DiastolicBP	Feature	Integer		Lower value of Blood Pressure in mmHg, another significant attribute during pregnancy.		no
BS	Feature	Integer		Blood glucose levels is in terms of a molar concentration	mmol/L	no
BodyTemp	Feature	Integer			F	no
HeartRate	Feature	Integer		A normal resting heart rate	bpm	no
RiskLevel	Target	Categorical		Predicted Risk Intensity Level during pregnancy considering the previous attribute.		no

Description of the variables

Splitting data:

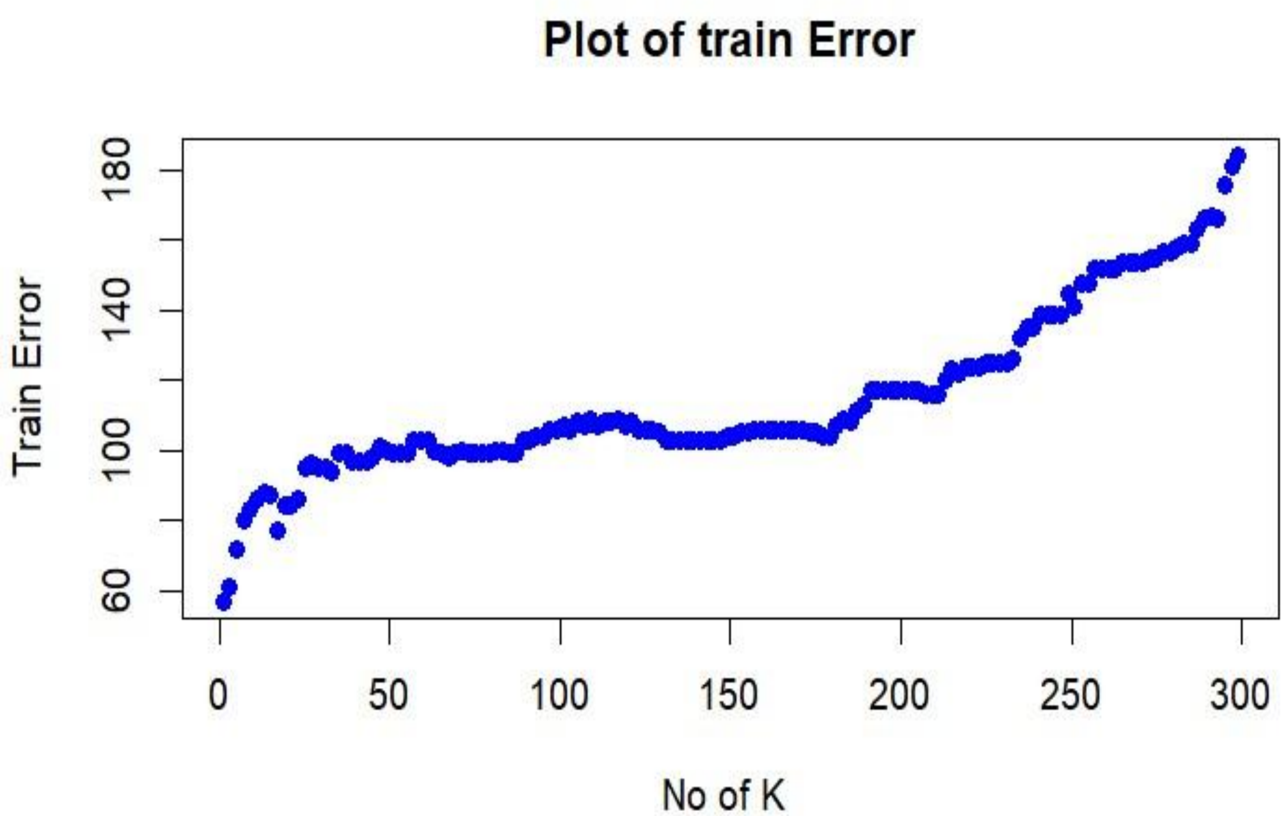
75% Training set,15% testing set & 10% Validation set.

Experimental Results:

Experimental setup in kNN:

For understanding the process of kNN algorithm with the dataset, I have built the algorithm in R programming language with using class library.

Error Rate Graph for selection of k-value:

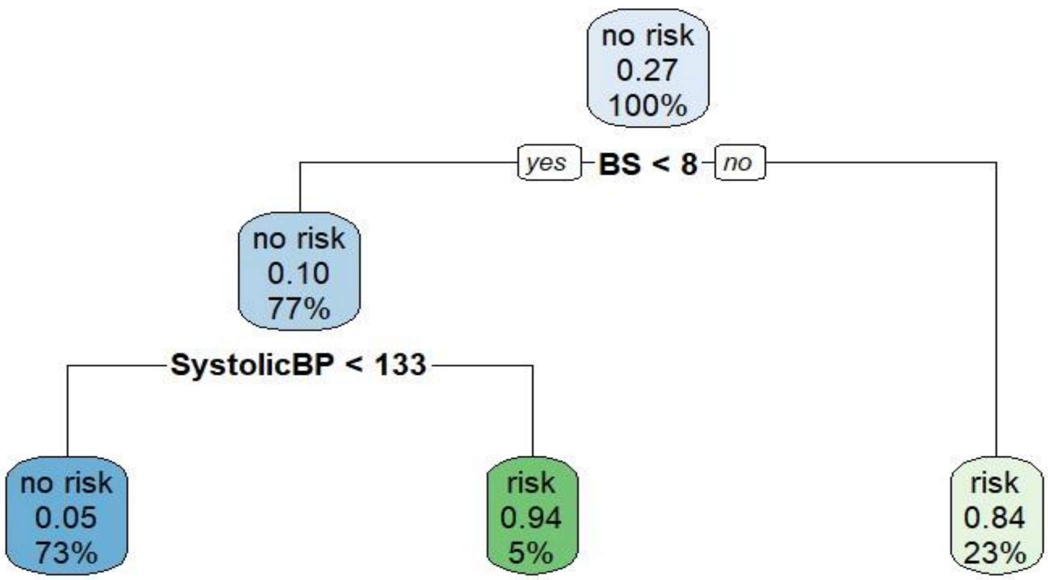


Plot of train error v/s different values of k

For the dataset the error rate to be found minimum at k =171.

Decision Tree:

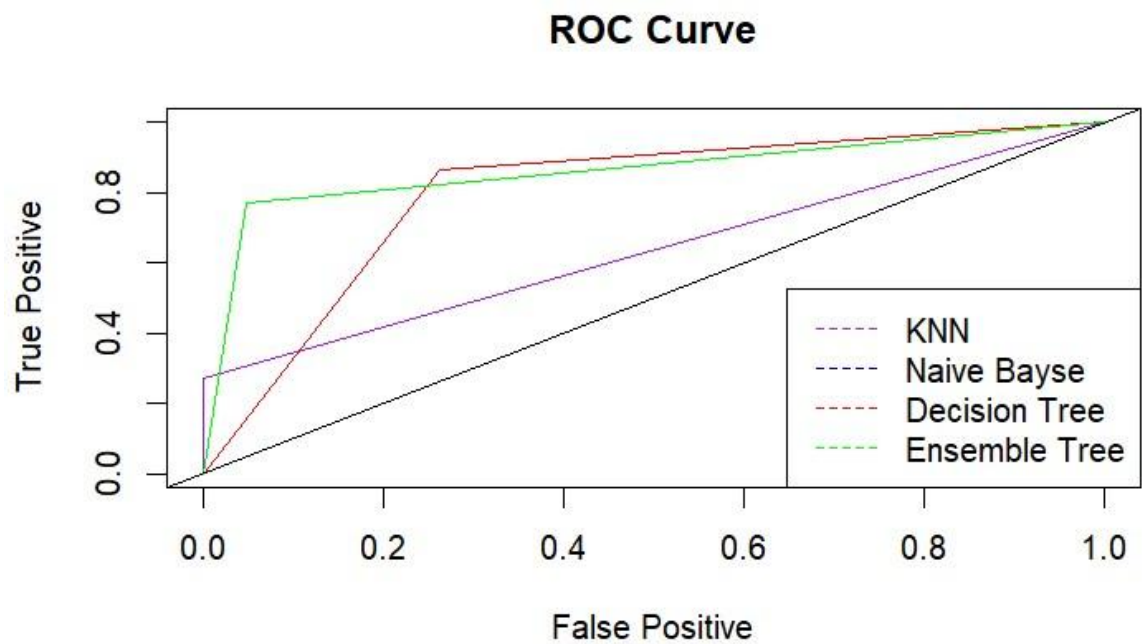
Given below are the result of Decision Tree for training data and test data:



Decision Tree

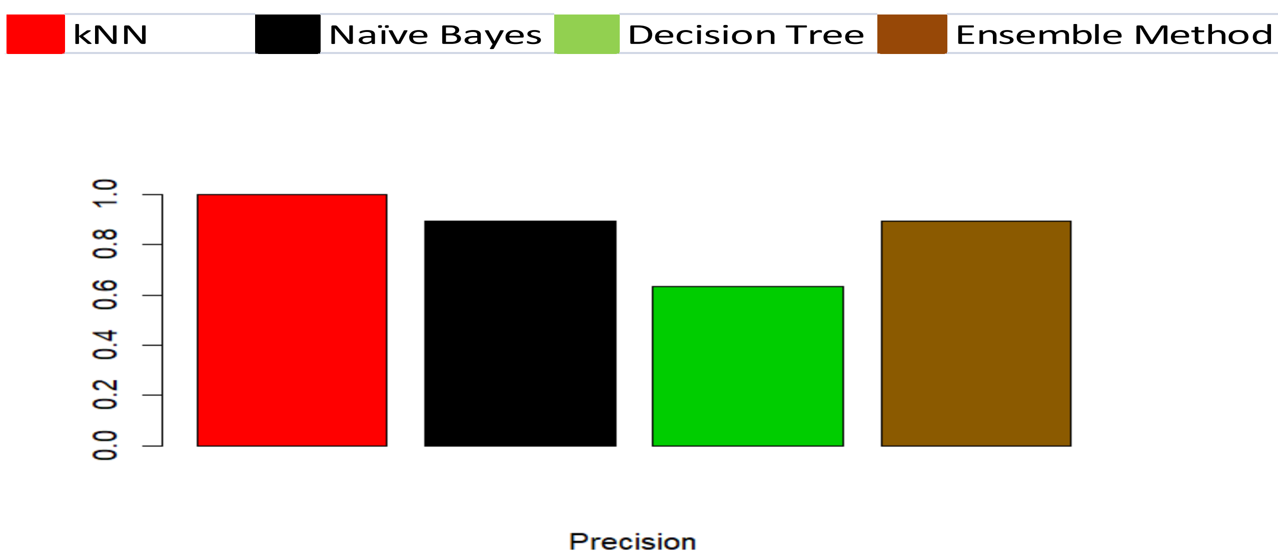
Performance Metric	*k-Nearest Neighbour (kNN)*	*Naïve Bayes*	*Decision Tree*	*Ensemble Method*
Confusion Matrix	6 0 16 42	17 2 5 40	19 11 3 31	17 2 5 40
Precision	1	0.8947368	.63334	0.8947368
Recall	.2727273	.7727273	.863636	.7727273
Accuracy(%)	75	89.0625	78.125	89.0625
F-measure	.42857	.829268	.730769	.829268
False Positive Rate	0	.047619	.2619048	.047619
Area Under The Curve(AUC)	.6364	.8626	.8009	.8626

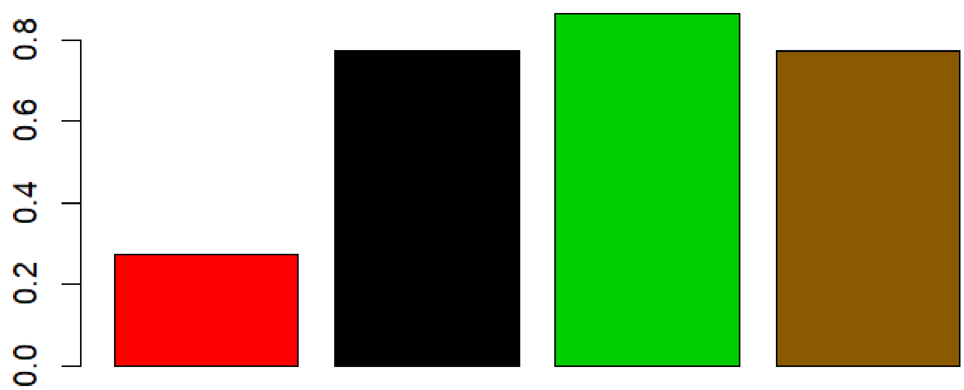
ROC Curve plot:



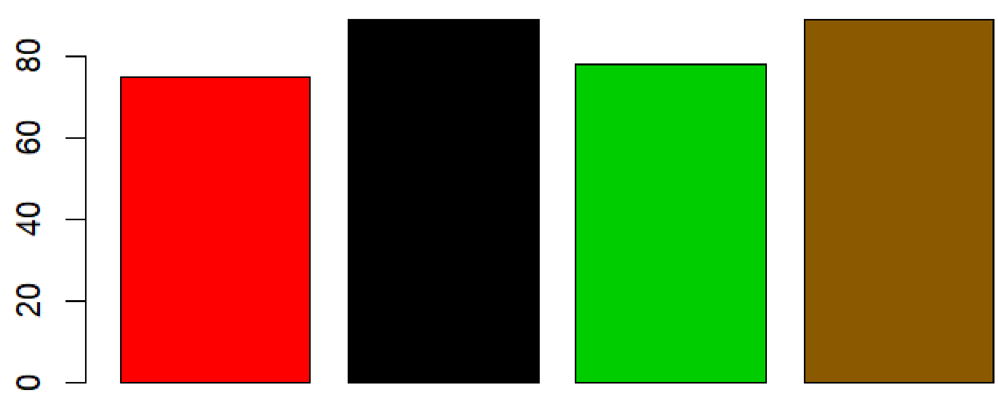
(Here AUC for Naïve bayes and Ensemble Method is same so curve of naïve bayes is not showing.)

Bar plot of five performance measures for all algorithm:

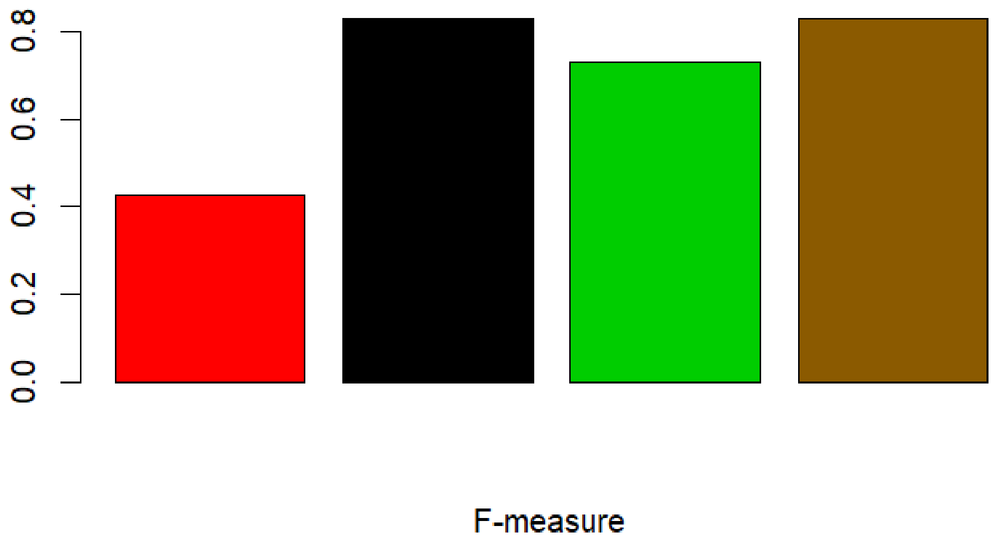




Recall



Accuracy


0.0
0.2
0.4
0.6
0.8
F-measure


0.00
0.10
0.20
False Positive Rate

Conclusion:

- **In Naïve Bayes and Ensemble Method accuracy is high. I am looking for there is any maternity health risk or not . So in this case I choose model in which Recall is higher . Here recall is higher in Decision Tree. I choose Decision Tree as my model for prediction.**

CODE:

```
mydata=read.csv("dataset1.csv")

# convert my 3 class as 2 class . I take high risk class as risk & low , mid risk class as not risk.

mydata$RiskLevel[mydata$RiskLevel=="high risk"]="risk"

mydata$RiskLevel[mydata$RiskLevel=="low risk" ]="no risk"

mydata$RiskLevel[mydata$RiskLevel=="mid risk" ]="no risk"


# training data


training_data=mydata[1:200,]

training_data=rbind(training_data,mydata[273:822,])

training_data=cbind(sl_no=1:750,training_data)

row.names(training_data)=1:750


# test data


test_data=mydata[201:250,]

test_data=rbind(test_data,mydata[823:972,])

row.names(test_data)=1:200
```

```
# validation data

validation_data=mydata[251:272,]

validation_data=rbind(validation_data,mydata[973:1014,])

row.names(validation_data)=1:64

#####knn

library(class)

te=c()

for (i in 1:150) {

  te[i]=sum((knn(train = training_data[2:7],test =
training_data[2:7],cl=training_data$RiskLevel,k=2*i+1))!=training_data$RiskLevel)

}

plot(y=te,x=2*(1:150)-1,type = "b",main="Plot of train Error",xlab="No of K ",ylab="Train
Error",col="blue",pch=16)

which(te==106)

k=86*2-1

# Finding the the predicted class by KNN:

K.N.N=knn(train = training_data[2:7],test = test_data[1:6],cl=training_data$RiskLevel,k=k)

confusion_matrix=table(test_data$RiskLevel,K.N.N)

#finding the accuracy of prediction class

Accuracy=(confusion_matrix[1,1]+confusion_matrix[2,2])*100/length(test_data$RiskLevel)

# for validation data
```

```
KNN_val=knn(training_data[2:7],validation_data[1:6],cl=training_data$RiskLevel,k=k)

t1=table(KNN_val,Actual=validation_data$RiskLevel)

accu_val_knn=(t1[1,1]+t1[2,2])*100/64

#ROC

library(pROC)

roc_knn=roc(c(0,1)[as.factor(validation_data$RiskLevel)],c(0,1)[as.factor(KNN_val)])

roc_knn$auc


#recall

recall_knn=t1[2,2]/(t1[2,2]+t1[1,2])

#precision

precision_knn=t1[2,2]/(t1[2,2]+t1[2,1])

#f1 score

f1_score_knn=2*recall_knn*precision_knn/(recall_knn+precision_knn)

#True positive rate

tpr_knn=recall_knn

#false positive rate

fpr_knn=t1[2,1]/(t1[2,1]+t1[1,1])

#Naive bayse classifier

age_1=dnorm(test_data$Age,mean(training_data$Age[which(training_data$RiskLevel=="risk")]
),

      sd(training_data$Age[which(training_data$RiskLevel=="risk")]))


age_2=dnorm(test_data$Age,mean(training_data$Age[which(training_data$RiskLevel=="no
risk")]),
```

```
        sd(training_data$Age[which(training_data$RiskLevel=="no risk")]))

systolicBP_1=dnorm(test_data$SystolicBP,mean(training_data$SystolicBP[which(training_data$
RiskLevel=="risk")]),
            sd(training_data$SystolicBP[which(training_data$RiskLevel=="risk")]))

systolicBP_2=dnorm(test_data$SystolicBP,mean(training_data$SystolicBP[which(training_data$
RiskLevel=="no risk")]),
            sd(training_data$SystolicBP[which(training_data$RiskLevel=="no risk")]))

dyastolicBP_1=dnorm(test_data$DiastolicBP,mean(training_data$DiastolicBP[which(training_d
ata$RiskLevel=="risk")]),
            sd(training_data$DiastolicBP[which(training_data$RiskLevel=="risk")]))

dyastolicBP_2=dnorm(test_data$DiastolicBP,mean(training_data$DiastolicBP[which(training_d
ata$RiskLevel=="no risk")]),
             sd(training_data$DiastolicBP[which(training_data$RiskLevel=="no risk")]))

BS_1=dnorm(test_data$BS,mean(training_data$BS[which(training_data$RiskLevel=="risk")]),
             sd(training_data$BS[which(training_data$RiskLevel=="risk")]))

BS_2=dnorm(test_data$BS,mean(training_data$BS[which(training_data$RiskLevel=="no
risk")]),
       sd(training_data$BS[which(training_data$RiskLevel=="no risk")]))
```

```
bodytemp_1=dnorm(test_data$BodyTemp,mean(training_data$BodyTemp[which(training_data$RiskLevel=="risk")]),
        sd(training_data$BodyTemp[which(training_data$RiskLevel=="risk")]))

bodytemp_2=dnorm(test_data$BodyTemp,mean(training_data$BodyTemp[which(training_data$RiskLevel=="no risk")]),
        sd(training_data$BodyTemp[which(training_data$RiskLevel=="no risk")]))

heartrate_1=dnorm(test_data$HeartRate,mean(training_data$HeartRate[which(training_data$RiskLevel=="risk")]),
        sd(training_data$HeartRate[which(training_data$RiskLevel=="risk")]))

heartrate_2=dnorm(test_data$HeartRate,mean(training_data$HeartRate[which(training_data$RiskLevel=="no risk")]),
         sd(training_data$HeartRate[which(training_data$RiskLevel=="no risk")]))

#prob of risk>>>>>>>>>>>>>>>

prob_risk=age_1*systolicBP_1*dyastolicBP_1*BS_1*bodytemp_1*heartrate_1

#prob of risk>>>>>>>>>>>>>>>

prob_no_risk=age_2*systolicBP_2*dyastolicBP_2*BS_2*bodytemp_2*heartrate_2

# create data frame of risks>>>>>>>>>
```

```
df=data.frame(prob_risk,prob_no_risk,Actual=test_data$RiskLevel)

class=c()

for (i in 1:nrow(df)) {

   if(df$prob_risk[i] > df$prob_no_risk[i]){

     class[i]="risk"

   }

  if(df$prob_risk[i]<df$prob_no_risk[i]){

    class[i]="no risk"

  }

}

df=cbind(df,Predict=class)

#Accuracy

sum(df$Actual==df$Predict)*100/length(df$Actual)


#for validation




#Naive bayse classifier

age_v1=dnorm(validation_data$Age,mean(training_data$Age[which(training_data$RiskLevel==
"risk")]),

           sd(training_data$Age[which(training_data$RiskLevel=="risk")]))


age_v2=dnorm(validation_data$Age,mean(training_data$Age[which(training_data$RiskLevel==
"no risk")]),

           sd(training_data$Age[which(training_data$RiskLevel=="no risk")]))
```

```
systolicBP_v1=dnorm(validation_data$SystolicBP,mean(training_data$SystolicBP[which(trainin
g_data$RiskLevel=="risk")]),

                sd(training_data$SystolicBP[which(training_data$RiskLevel=="risk")]))


systolicBP_v2=dnorm(validation_data$SystolicBP,mean(training_data$SystolicBP[which(trainin
g_data$RiskLevel=="no risk")]),

                sd(training_data$SystolicBP[which(training_data$RiskLevel=="no risk")]))


dyastolicBP_v1=dnorm(validation_data$DiastolicBP,mean(training_data$DiastolicBP[which(trai
ning_data$RiskLevel=="risk")]),

                 sd(training_data$DiastolicBP[which(training_data$RiskLevel=="risk")]))


dyastolicBP_v2=dnorm(validation_data$DiastolicBP,mean(training_data$DiastolicBP[which(trai
ning_data$RiskLevel=="no risk")]),

                 sd(training_data$DiastolicBP[which(training_data$RiskLevel=="no risk")]))


BS_v1=dnorm(validation_data$BS,mean(training_data$BS[which(training_data$RiskLevel=="ris
k")]),

          sd(training_data$BS[which(training_data$RiskLevel=="risk")]))


BS_v2=dnorm(validation_data$BS,mean(training_data$BS[which(training_data$RiskLevel=="no
risk")]),

          sd(training_data$BS[which(training_data$RiskLevel=="no risk")]))


bodytemp_v1=dnorm(validation_data$BodyTemp,mean(training_data$BodyTemp[which(traini
ng_data$RiskLevel=="risk")]),
```

```
sd(training_data$BodyTemp[which(training_data$RiskLevel=="risk")]))

bodytemp_v2=dnorm(validation_data$BodyTemp,mean(training_data$BodyTemp[which(training_data$RiskLevel=="no risk")]),
        sd(training_data$BodyTemp[which(training_data$RiskLevel=="no risk")]))

heartrate_v1=dnorm(validation_data$HeartRate,mean(training_data$HeartRate[which(training_data$RiskLevel=="risk")]),
        sd(training_data$HeartRate[which(training_data$RiskLevel=="risk")]))

heartrate_v2=dnorm(validation_data$HeartRate,mean(training_data$HeartRate[which(training_data$RiskLevel=="no risk")]),
        sd(training_data$HeartRate[which(training_data$RiskLevel=="no risk")]))

#prob of risk>>>>>>>>>>>>>>>

prob_risk_val=age_v1*systolicBP_v1*dyastolicBP_v1*BS_v1*bodytemp_v1*heartrate_v1

#prob of risk>>>>>>>>>>>>>>>

prob_no_risk_val=age_v2*systolicBP_v2*dyastolicBP_v2*BS_v2*bodytemp_v2*heartrate_v2

# create data frame of risks>>>>>>>>>

df_naive_val=data.frame(prob_risk_val,prob_no_risk_val,Actual=validation_data$RiskLevel)
```

```
class_val=c()

for (i in 1:nrow(df_naive_val)) {

  if(df_naive_val$prob_risk_val[i] > df_naive_val$prob_no_risk_val[i]){

    class_val[i]="risk"

  }

  else{

    class_val[i]="no risk"

  }

}

t2=table(class_val,Actual=validation_data$RiskLevel)

accu_nv_val=sum(df_naive_val$Actual==class_val)*100/64

#ROC Naive bayse


roc_nv=roc(c(0,1)[as.factor(validation_data$RiskLevel)],c(0,1)[as.factor(class_val)])

roc_nv$auc


#recall

recall_nv=t2[2,2]/(t2[2,2]+t2[1,2])

#precision

precision_nv=t2[2,2]/(t2[2,2]+t2[2,1])

#f1 score

f1_score_nv=2*recall_nv*precision_nv/(recall_nv+precision_nv)

#True positive rate

tpr_nv=recall_nv
```

```
#false positive rate

fpr_nv=t2[2,1]/(t2[2,1]+t2[1,1])


#training data

newtraining=training_data[,-1]


# Decision Tree

library(rpart)

library(rpart.plot)


# training the decision tree


dt_fit=rpart(RiskLevel~.,newtraining,method = "class")

rpart.plot(dt_fit)


# test the data>


predicted_class=predict(dt_fit,newdata = test_data,type = "class")


#confusion matrix


conf_mat_dt=table(test_data$RiskLevel,predicted_class)

Accuracy_dt=(conf_mat_dt[1,1]+conf_mat_dt[2,2])*100/length(test_data$RiskLevel)
```

```
# for validation data

pre_val=predict(dt_fit,newdata = validation_data,type = "class")

t=table(pre_val,Actual=validation_data$RiskLevel)
accu_val_dt=(t[1,1]+t[2,2])*100/64
#ROC Decision tree

roc_dt=roc(c(0,1)[as.factor(validation_data$RiskLevel)],c(0,1)[as.factor(pre_val)])
roc_dt$auc

#recall
recall_dt=t[2,2]/(t[2,2]+t[1,2])
#precision
precision_dt=t[2,2]/(t[2,2]+t[2,1])
#f1 score
f1_score_dt=2*recall_dt*precision_dt/(recall_dt+precision_dt)
#True positive rate
tpr_dt=recall_dt
#false positive rate
fpr_dt=t[2,1]/(t[2,1]+t[1,1])

# creay a data frame of predicted classes of validation data by three different method>>
```

```
df_val=data.frame(KNN=KNN_val,"Naive Bayse"=class_val,"Decision
tree"=pre_val,actual_class=validation_data$RiskLevel,final_class=rep(NA))

# by combaining 3 mehtods now my predicted classes are:

df_val$final_class[df_val$KNN=="risk" & df_val$Naive.Bayse=="risk" &
df_val$Decision.tree=="risk"]="risk"

df_val$final_class[df_val$KNN=="no risk" & df_val$Naive.Bayse=="risk" &
df_val$Decision.tree=="risk"]="risk"

df_val$final_class[df_val$KNN=="risk" & df_val$Naive.Bayse=="no risk" &
df_val$Decision.tree=="risk"]="risk"

df_val$final_class[df_val$KNN==" risk" & df_val$Naive.Bayse==" risk" &
df_val$Decision.tree=="no risk"]="risk"

df_val$final_class[df_val$KNN=="no risk" & df_val$Naive.Bayse=="no risk" &
df_val$Decision.tree=="no risk"]="no risk"

df_val$final_class[df_val$KNN==" risk" & df_val$Naive.Bayse=="no risk" &
df_val$Decision.tree=="no risk"]="no risk"

df_val$final_class[df_val$KNN=="no risk" & df_val$Naive.Bayse==" risk" &
df_val$Decision.tree=="no risk"]="no risk"
```

```
df_val$final_class[df_val$KNN=="no risk" & df_val$Naive.Bayse=="no risk" &
df_val$Decision.tree=="risk"]="no risk"


# Accuracy of combined methods:

t3=table(predict=df_val$final_class,Actual=validation_data$RiskLevel)

accu_en=(sum(df_val$actual_class==df_val$final_class))*100/64


#recall

recall_en=t3[2,2]/(t3[2,2]+t3[1,2])

#precision

precision_en=t3[2,2]/(t3[2,2]+t3[2,1])

#f1 score

f1_score_en=2*recall_en*precision_en/(recall_en+precision_en)

#True positive rate

tpr_en=recall_en

#false positive rate

fpr_en=t3[2,1]/(t3[2,1]+t3[1,1])

#roc of ensemble

library(pROC)

roc_en=roc(c(0,1)[as.factor(validation_data$RiskLevel)],c(0,1)[as.factor(df_val$final_class)])

auc(roc_en)

#plot of roc curve::

plot(y=roc_knn$sensitivities,x=1-roc_knn$specificities,main="ROC Curve",type =
"l",col="purple",ylab = "True Positive",xlab = "False Positive")

lines(y=roc_nv$sensitivities,x=1-roc_nv$specificities,col="blue")
```

```
lines(y=roc_dt$sensitivities,x=1-roc_dt$specificities,col="red")

lines(y=roc_en$sensitivities,x=1-roc_en$specificities,col="green")

abline(a=0,b=1)

legend(

  "bottomright",c("KNN","Naive Bayse","Decision Tree","Ensemble Tree"),col=c("purple","blue","red","green"),lty=c(2))
```

```
list("confusion matrix for knn"=t1,"confusion matrix for naive bayse"=t2,"confusion matrix for decision tree"=t,"confusion matrix for ensemble method"=t3,

     "Accuracy of KNN"=accu_val_knn, "Accuracy of NAive Bayes"=accu_nv_val,"Accuracy of Decision Tree"=accu_val_dt,"Accuracy of ensemble method"=accu_en,

     "recall of KNN"=recall_knn,"recall of naive bayse"=recall_nv,"recall of decision tree"=recall_dt,"recall of ensemble method"=recall_en,

     "precision of knn"=precision_knn,"precision of naive bayse"=precision_nv,"precision of decision tree"=precision_dt,"precision of enemble method"=precision_en,

     "F1 score of knn"=f1_score_knn,"F1 score of naive bayse"=f1_score_nv,"F1 score of decision tree"=f1_score_dt,"F1 score of ensemble method"=f1_score_en,

     "True positive rate of knn"=tpr_knn,"True positive rate of naive bayse"=tpr_nv,"True positive rate of decision tree"=tpr_dt,

     "True positive rate of ensemble method"=tpr_en,"False positive rate of KNN"=fpr_knn,"False positive rate of naive bayse"=fpr_nv,

     "False positive rate of decision tree"=fpr_dt,"False positive rate of ensemble method"=fpr_en,"AUC in knn"=auc(roc_knn),"AUC in Naive bayse"=auc(roc_nv),

     "AUC of decision tree"=roc_dt$auc,"AUC of Ensemble method"=roc_en$auc)
```

```
barplot(c(precision_knn,precision_nv,precision_dt,precision_en),xlab =
"Precision",col=c("red","black","green3","orange4"))

barplot(c(recall_knn,recall_nv,recall_dt,recall_en),xlab =
"Recall",col=c("red","black","green3","orange4"))

barplot(c(accu_val_knn,accu_nv_val,accu_val_dt,accu_en),xlab =
"Accuracy",col=c("red","black","green3","orange4"))

barplot(c(f1_score_knn,f1_score_nv,f1_score_dt,f1_score_en),xlab = "F-
measure",col=c("red","black","green3","orange4"))

barplot(c(fpr_knn,fpr_nv,fpr_dt,fpr_en),xlab = "False Positive
Rate",col=c("red","black","green3","orange4"))


############################################### Thank You
##############################################################
```