Task 3

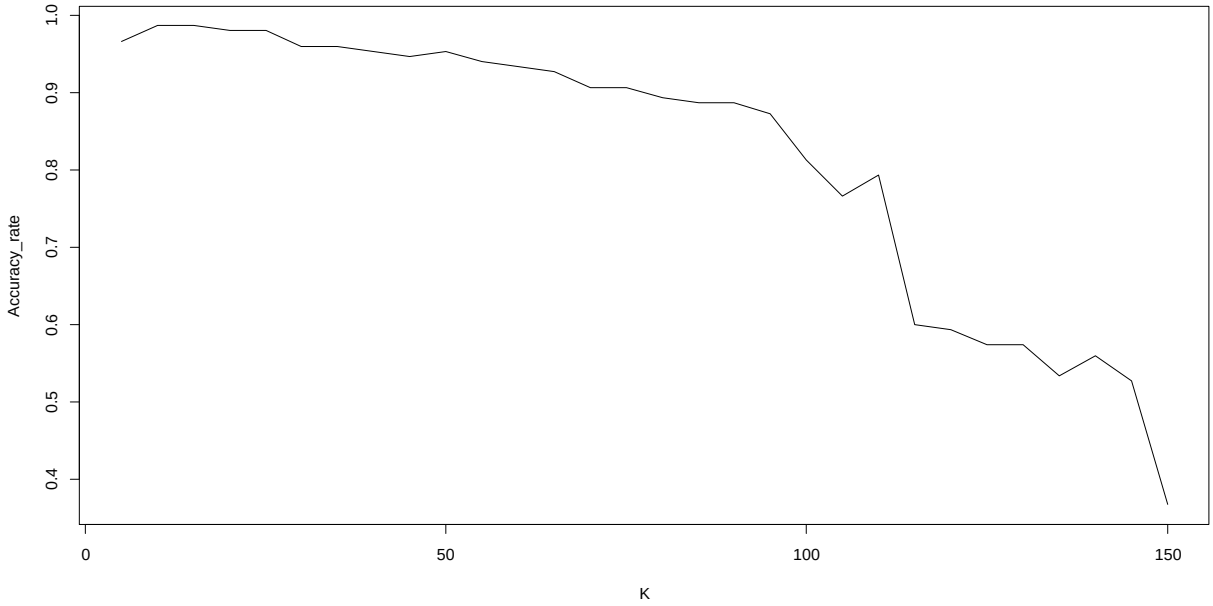
Problems :

1) The Iris flower dataset consists of three species: setosa, versicolor, and virginica. These species can be distinguished based on their measurements. Now, imagine that you have the measurements of Iris flowers categorized by their respective species. Your objective is to train a machine learning model that can learn from these measurements and accurately classify the Iris flowers into their respective species.
2) Use the Iris dataset to develop a model that can classify iris flowers into different species based on their sepal and petal measurements. This dataset is widely used for introductory classification tasks.

Classification Techniques

1 KNN

```
require(class)
> data=read.csv(file.choose())
> data
> X1=data$sepal_length
> X2=data$sepal_width
> X3=data$petal_length
> X4=data$petal_width
> Y=data$species
>train<-cbind(X1,X2,X3,X4)
```



confusion matrix

```
> test<-train
> actual=Y
> predicted<-knn(train,test,actual,k=opt_k)
> which(actual==predicted)
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  85  86  87  88  89  90  91
 [91]  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 108 109 110
[109] 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128
[127] 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146
[145] 147 148 149 150
> which(actual==predicted)  # accurately classified obs.
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
```

```
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  85  86  87  88  89  90  91
 [91]  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 108 109 110
[109] 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128
[127] 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146
[145] 147 148 149 150
> which(actual!=predicted)   #Misclassified obs.
[1]  84 107
> CM<-table(actual,predicted)        #Confusion Matrix and interest in 'No'
> #CM1<-CM[2:1,2:1]      #Confusion Matrix and interest in 'Yes'
> #CM<-CM1
> Accuracy_rate<-sum(diag(CM))/sum(CM)
> Accuracy_rate
[1] 0.9866667
> #Accuracy_rate<-sum(actual==predicted)/n
> Error_rate<-1-Accuracy_rate;Error_rate
[1] 0.01333333
> #Error_rate<-sum(actual!=predicted)/n
> Sensitivity<-CM[1,1]/sum(CM[1,])
> Sensitivity
[1] 1
> recall<-Sensitivity
> recall
[1] 1
> Specificity<-CM[2,2]/sum(CM[2,])
```

```
> Specificity
[1] 0.98
> Precision<-CM[1,1]/sum(CM[,1])
> Precision
[1] 1
> F1<-(2*Precision*recall)/(Precision+recall)
> F1
[1] 1
> beta<-2
> beta
[1] 2
> Fbeta<-((1+beta*beta)*Precision*recall)/(beta*beta*Precision+recall)
> Fbeta
[1] 1
```

Conclusion： The Accuracy rate of data using KNN Method is 98.6%

Decision Tree

```
> require(rattle)
> require(class)
> require(rpart.plot)
> require(RColorBrewer)
> require(tree)
> require(modeltools)
> require(party)
> require(rpart)
> data=read.csv(file.choose())
> X1=data$sepal_length
```

```
> X2=data$sepal_width
> X3=data$petal_length
> X4=data$petal_width
> Y=data$species
> Y1=as.factor(Y)
> test<-data.frame(X1=5.84, X2= 3.05, X3=3.76,X4=1.20)
> train<-data.frame(X1,X2,X3,X4)
> m<-tree(Y1~X1+X2+X3+X4)
> m
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 150 329.600 Iris-setosa ( 0.33333 0.33333 0.33333 )
   2) X3 < 2.45 50   0.000 Iris-setosa ( 1.00000 0.00000 0.00000 ) *
   3) X3 > 2.45 100 138.600 Iris-versicolor ( 0.00000 0.50000 0.50000 )
     6) X4 < 1.75 54  33.320 Iris-versicolor ( 0.00000 0.90741 0.09259 )
      12) X3 < 4.95 48   9.721 Iris-versicolor ( 0.00000 0.97917 0.02083 )
        24) X1 < 5.15 5   5.004 Iris-versicolor ( 0.00000 0.80000 0.20000 ) *
        25) X1 > 5.15 43   0.000 Iris-versicolor ( 0.00000 1.00000 0.00000 ) *
      13) X3 > 4.95 6   7.638 Iris-virginica ( 0.00000 0.33333 0.66667 ) *
     7) X4 > 1.75 46   9.635 Iris-virginica ( 0.00000 0.02174 0.97826 )
      14) X3 < 4.95 6   5.407 Iris-virginica ( 0.00000 0.16667 0.83333 ) *
      15) X3 > 4.95 40   0.000 Iris-virginica ( 0.00000 0.00000 1.00000 ) *
> summary(m)

Classification tree:
tree(formula = Y1 ~ X1 + X2 + X3 + X4)
```

Variables actually used in tree construction:

[1] "X3" "X4" "X1"

Number of terminal nodes: 6

Residual mean deviance: 0.1253 = 18.05 / 144

Misclassification error rate: 0.02667 = 4 / 150

> plot(m)

> text(m)

> predict(m,test)

Iris-setosa Iris-versicolor Iris-virginica

1 0 1 0

> predict(m,test,type="class")

[1] Iris-versicolor

Levels: Iris-setosa Iris-versicolor Iris-virginica

> rpart(m)

n= 150

node), split, n, loss, yval, (yprob)

* denotes terminal node

1) root 150 100 Iris-setosa (0.33333333 0.33333333 0.33333333)

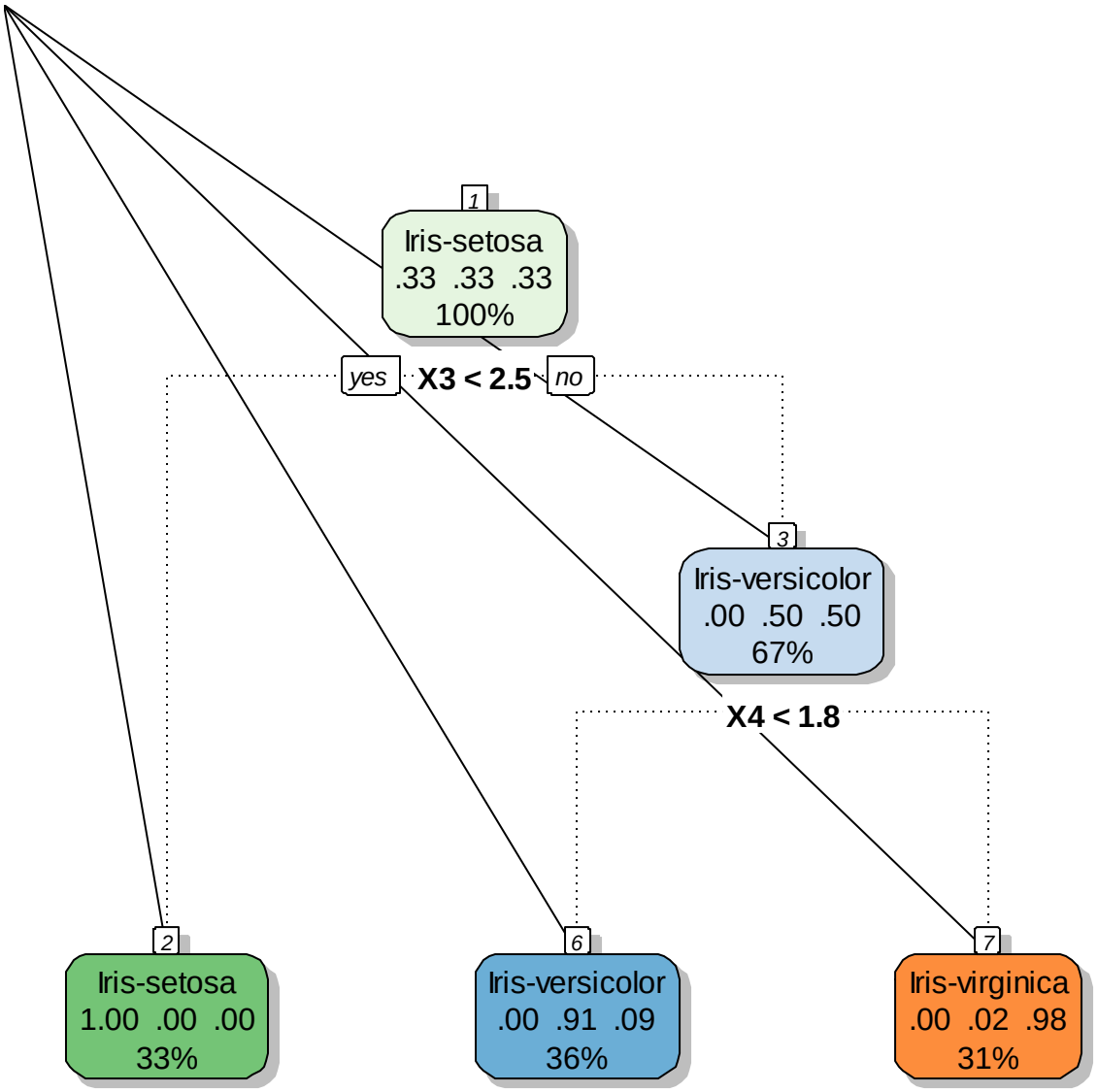
2) X3< 2.45 50 0 Iris-setosa (1.00000000 0.00000000 0.00000000) *

3) X3>=2.45 100 50 Iris-versicolor (0.00000000 0.50000000 0.50000000)

6) X4< 1.75 54 5 Iris-versicolor (0.00000000 0.90740741 0.09259259) *

7) X4>=1.75 46 1 Iris-virginica (0.00000000 0.02173913 0.97826087)

> fancyRpartPlot(rpart(m),caption=NULL)



Confusion Matrix :

```
> test<-train
> n<-nrow(test)
> predicted<-predict(m,test,type='class')
> #predicted<-predict(m,test,type='prob')
> actual<-Y1
> which(actual==predicted)  # accurately classified obs.
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
```

```
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  72  73
 [73]  74  75  76  77  79  80  81  82  83  85  86  87  88  89  90  91  92  93
 [91]  94  95  96  97  98  99 100 101 102 103 104 105 106 108 109 110 111 112
[109] 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
[127] 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148
[145] 149 150
> which(actual!=predicted)   #Misclassified obs.
[1]  71  78  84 107
> CM<-table(actual,predicted)
> CM       #Confusion Matrix and interest in 'No'
             predicted
actual          Iris-setosa Iris-versicolor Iris-virginica
 Iris-setosa            50             0            0
 Iris-versicolor         0            47            3
 Iris-virginica          0             1           49
> #CM1<-CM[2:1,2:1]     #Confusion Matrix and interest in 'Yes'
> #CM<-CM1
> Accuracy_rate<-sum(diag(CM))/sum(CM);Accuracy_rate
[1] 0.9733333
> #Accuracy_rate<-sum(actual==predicted)/n
> Error_rate<-1-Accuracy_rate; Error_rate
[1] 0.02666667
> #Error_rate<-sum(actual!=predicted)/n
> Sensitivity<-CM[1,1]/sum(CM[1,]);Sensitivity
[1] 1
```

```
> recall<-Sensitivity;recall
[1] 1
> Specificity<-CM[2,2]/sum(CM[2,]);Specificity
[1] 0.94
> Precision<-CM[1,1]/sum(CM[,1]);Precision
[1] 1
> F1<-(2*Precision*recall)/(Precision+recall)
> beta<-2
> Fbeta<-((1+beta*beta)*Precision*recall)/(beta*beta*Precision+recall)
> Fbeta
[1] 1
```

Conclusion : **The Accuracy rate of data Using Decision Tree Method is 97.33%**

Naïve Bayesian

```
> require(naivebayes)
> data=read.csv(file.choose())
> X1=data$sepal_length
> X2=data$sepal_width
> X3=data$petal_length
> X4=data$petal_width
> Y=data$species
> Y1=as.factor(Y)
> test<-data.frame(X1=5.84, X2= 3.05, X3=3.76,X4=1.20)
> train<-data.frame(X1,X2,X3,X4)
> data=cbind(train,test)
> m<-naive_bayes(Y1~.,data=train)
> predicted<-predict(m,data[150,],type="class")
```

[1] Iris-virginica

```
Levels: Iris-setosa Iris-versicolor Iris-virginica
> predicted<-predict(m,data[150,],type="prob")
> predicted
      Iris-setosa Iris-versicolor Iris-virginica
[1,] 5.948103e-283     0.004040394      0.9959596
> test<-train
> n<-nrow(test)
> predicted<-predict(m,test,type='class')
> #predicted<-predict(m,test,type='prob')
> actual<-Y1
> which(actual==predicted)  # accurately classified obs.
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  54  55
 [55]  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  72  73  74
 [73]  75  76  77  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93
 [91]  94  95  96  97  98  99 100 101 102 103 104 105 106 108 109 110 111 112
[109] 113 114 115 116 117 118 119 121 122 123 124 125 126 127 128 129 130 131
[127] 132 133 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
> which(actual!=predicted)   #Misclassified obs.
[1]  53  71  78 107 120 134
> CM<-table(actual,predicted)
> CM       #Confusion Matrix and interest in 'No'
             predicted
actual          Iris-setosa Iris-versicolor Iris-virginica
  Iris-setosa            50              0             0
```

```
 Iris-versicolor         0          47          3
 Iris-virginica          0          3          47
> #CM1<-CM[2:1,2:1]     #Confusion Matrix and interest in 'Yes'
> #CM<-CM1
> Accuracy_rate<-sum(diag(CM))/sum(CM);Accuracy_rate
[1] 0.96
> #Accuracy_rate<-sum(actual==predicted)/n
> Error_rate<-1-Accuracy_rate;Error_rate
[1] 0.04
> #Error_rate<-sum(actual!=predicted)/n
> Sensitivity<-CM[1,1]/sum(CM[1,]);Sensitivity
[1] 1
> recall<-Sensitivity ;recall
[1] 1
> Specificity<-CM[2,2]/sum(CM[2,]); Specificity
[1] 0.94
> Precision<-CM[1,1]/sum(CM[,1]);Precision
[1] 1
> F1<-(2*Precision*recall)/(Precision+recall);F1
[1] 1
> beta<-2
> Fbeta<-((1+beta*beta)*Precision*recall)/(beta*beta*Precision+recall)
> Fbeta
[1] 1
```

Comparison Test :

```
library(caret)
```

```
> data(iris)
> data1<-iris[1:150,]
> control <- trainControl(method="repeatedcv",number=10,repeats=15)
> set.seed(1)
> knn<-train(Species~., data=data1, method="knn", trControl=control)
> set.seed(1)
> tree1<-train(Species~., data=data1, method="C5.0Tree", trControl=control, verbose=FALSE)
> set.seed(1)
> nb1<-train(Species~., data=data1, method="naive_bayes", trControl=control)
> set.seed(1)
> #logistic_reg<-train(Species~., data=iris, method="glm",famil="binomial",trControl=control)
> logistic_reg<-train(X~., data=data, method="multinom",trControl=control, trace=FALSE)
> results <- resamples(list(KNN=knn,Decision_Tree=tree1,Naive_Bayes=nb1))
> summary(results)

Call:
summary.resamples(object = results)
```

Models: KNN, Decision_Tree, Naive_Bayes ,logistic Regression

Number of resamples: 150

Acuuracy :

	Min	1st Qu	Median	Mean	3rd Qu	Max	NA's
KNN	0.8667	0.9333	1	0.9707	1	1	0
Decision_Tree	0.7333	0.9333	0.933	0.9480	1	1	0
Naive_Bayes	0.800	0.933	1	0.9573	1	1	0
Logistic Regreesion	0.8666667	0.9333333	1	0.9737778	1	1	0

Conclusion : The above Table show That Accuracy of data is high using **KNN** method