data(iris)  
  
dataset <- iris  
  
View(dataset)  
  
  
write.csv(dataset, "iris\_09-07-2018.csv")  
  
dataset <- na.omit(dataset)  
  
  
library("caret")

## Loading required package: lattice

## Loading required package: ggplot2

#Cretaing training and validation dataset  
#We will split the loaded dataset into two, 80% of which   
#we will use to train our models and 20% that we will hold back as a validation dataset.  
  
  
# use the remaining 80% of data to training and testing the models  
  
  
train\_data <- createDataPartition(dataset$Species, p=0.80, list=FALSE)  
  
# select 20% of the data for validation  
test\_data <- dataset[-train\_data,]  
  
#added 80% data into project\_iris  
dataset <- dataset[train\_data,]  
  
dim(dataset)

## [1] 120 5

#types to attributes  
  
# list types for each attribute  
  
sapply(dataset, class)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species   
## "numeric" "numeric" "numeric" "numeric" "factor"

#Peek at the Data  
  
head(dataset)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 8 5.0 3.4 1.5 0.2 setosa  
## 9 4.4 2.9 1.4 0.2 setosa

#Levels of the Class  
levels(dataset$Species)

## [1] "setosa" "versicolor" "virginica"

#class distribution  
  
percentage <- prop.table(table(dataset$Species)) \* 100  
cbind(freq=table(dataset$species), percentage=percentage)

## percentage  
## setosa 33.33333  
## versicolor 33.33333  
## virginica 33.33333

#Now we can look at the interactions between the variables.  
  
#We can also look at box and whisker plots of each input variable again,  
#but this time broken down into separate plots for each class.   
#This can help to tease out obvious linear separations between the classes.  
  
  
x <- dataset[,1:4]  
y <- dataset[,5]  
  
  
  
featurePlot(x=x, y=y, plot="box")

#Boxplot  
boxplot(x, dataset,varwidth = TRUE,notch = T, main=names(dataset))

# density plots for each attribute by class value  
scales <- list(x=list(relation="free"), y=list(relation="free"))  
featurePlot(x=x, y=y, plot="density", scales=scales)

#Apply k-means clustering algorithm  
  
#Process the data  
iris.new<- iris[,c(1,2,3,4)]  
iris.class<- iris[,"Species"]  
head(iris.new)

## Sepal.Length Sepal.Width Petal.Length Petal.Width  
## 1 5.1 3.5 1.4 0.2  
## 2 4.9 3.0 1.4 0.2  
## 3 4.7 3.2 1.3 0.2  
## 4 4.6 3.1 1.5 0.2  
## 5 5.0 3.6 1.4 0.2  
## 6 5.4 3.9 1.7 0.4

result<- kmeans(iris.new,3) #aplly k-means algorithm with no. of centroids(k)=3  
result$size # gives no. of records in each cluster

## [1] 50 62 38

# gives value of cluster center datapoint value(3 centers for k=3)  
result$centers

## Sepal.Length Sepal.Width Petal.Length Petal.Width  
## 1 5.006000 3.428000 1.462000 0.246000  
## 2 5.901613 2.748387 4.393548 1.433871  
## 3 6.850000 3.073684 5.742105 2.071053

#gives cluster vector showing the custer where each record falls  
result$cluster

## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
## [71] 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3  
## [106] 3 2 3 3 3 3 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 3 2 3 3 3 3 2 3  
## [141] 3 3 2 3 3 3 2 3 3 2

#Verify results of clustering  
par(mfrow=c(2,2), mar=c(5,4,2,2))  
  
# Plot to see how Sepal.Length and Sepal.Width data points have been distributed in clusters  
plot(iris.new[c(1,2)], col=result$cluster)  
  
# Plot to see how Sepal.Length and Sepal.Width   
#data points have been distributed originally as per "class" attribute in dataset  
plot(iris.new[c(1,2)], col=iris.class)  
  
# Plot to see how Petal.Length and Petal.Width data points have been distributed in clusters   
plot(iris.new[c(3,4)], col=result$cluster)  
  
plot(iris.new[c(3,4)], col=iris.class)

#Build Models  
#We don't know which algorithms would work on this dataset or what configurations   
#to use. We get an idea from the plots that some of the classes are partially   
#linearly separable in some dimensions, so we are expecting generally good results.  
  
#We have evaluate 3 different algorithms:  
#Linear Discriminant Analysis (LDA) Classification, Support Vector Machines (SVM) with a linear kernel and Random Forest (RF)  
  
#This is a good mixture of simple linear (LDA) and   
#complex nonlinear methods (SVM, RF).   
#We reset the random number seed before reach run to ensure   
#that the evaluation of each algorithm is performed using exactly   
#the same data splits. It ensures the results are directly comparable.  
  
  
# Linear discriminant Analysis(LDA)  
  
# to find the colration between the variables and present scatter plot graph between them  
  
library(psych)

##   
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

pairs.panels(dataset[1:4],  
 gap = 0,  
 bg = c("red", "green", "blue")[dataset$species],  
 pch =21)

library(MASS)  
  
#split the data into two (traning\_data , testing\_data)  
z <- sample(2, nrow(iris),  
 replace = T,  
 prob = c(0.8,0.2))  
training\_data <- iris[z==1,]  
testing\_data <- iris[z==2,]  
  
linear <- lda(Species~.,training\_data)  
linear

## Call:  
## lda(Species ~ ., data = training\_data)  
##   
## Prior probabilities of groups:  
## setosa versicolor virginica   
## 0.3589744 0.3333333 0.3076923   
##   
## Group means:  
## Sepal.Length Sepal.Width Petal.Length Petal.Width  
## setosa 4.969048 3.416667 1.461905 0.2428571  
## versicolor 5.956410 2.769231 4.230769 1.3051282  
## virginica 6.586111 2.963889 5.575000 2.0444444  
##   
## Coefficients of linear discriminants:  
## LD1 LD2  
## Sepal.Length 0.9174611 -0.1486767  
## Sepal.Width 1.5857848 2.1200071  
## Petal.Length -2.0788293 -1.0718324  
## Petal.Width -3.3802429 3.3001883  
##   
## Proportion of trace:  
## LD1 LD2   
## 0.9911 0.0089

#First discriminant function is a linear combination of the four variables:  
#1.252207(Sepel.Length)  
#1.115823(Sepel.Width)

# Percentage seperations achived by the first discriminant function is 99.37%  
  
  
  
#to check other outputs  
  
attributes(linear)

## $names  
## [1] "prior" "counts" "means" "scaling" "lev" "svd" "N"   
## [8] "call" "terms" "xlevels"  
##   
## $class  
## [1] "lda"

linear$scaling

## LD1 LD2  
## Sepal.Length 0.9174611 -0.1486767  
## Sepal.Width 1.5857848 2.1200071  
## Petal.Length -2.0788293 -1.0718324  
## Petal.Width -3.3802429 3.3001883

linear$xlevels

## named list()

linear$counts

## setosa versicolor virginica   
## 42 39 36

#histogram  
  
p<- predict(linear, training\_data)  
p

## $class  
## [1] setosa setosa setosa setosa setosa setosa   
## [7] setosa setosa setosa setosa setosa setosa   
## [13] setosa setosa setosa setosa setosa setosa   
## [19] setosa setosa setosa setosa setosa setosa   
## [25] setosa setosa setosa setosa setosa setosa   
## [31] setosa setosa setosa setosa setosa setosa   
## [37] setosa setosa setosa setosa setosa setosa   
## [43] versicolor versicolor versicolor versicolor versicolor versicolor  
## [49] versicolor versicolor versicolor versicolor versicolor versicolor  
## [55] versicolor versicolor versicolor versicolor versicolor versicolor  
## [61] versicolor versicolor versicolor versicolor versicolor versicolor  
## [67] virginica versicolor versicolor versicolor versicolor versicolor  
## [73] versicolor versicolor versicolor versicolor versicolor versicolor  
## [79] versicolor versicolor versicolor virginica virginica virginica   
## [85] virginica virginica virginica virginica virginica virginica   
## [91] virginica virginica virginica virginica virginica virginica   
## [97] virginica virginica virginica virginica virginica virginica   
## [103] virginica virginica virginica virginica virginica virginica   
## [109] virginica virginica virginica virginica virginica virginica   
## [115] virginica virginica virginica   
## Levels: setosa versicolor virginica  
##   
## $posterior  
## setosa versicolor virginica  
## 1 1.000000e+00 1.251992e-22 8.907755e-46  
## 3 1.000000e+00 8.089397e-20 5.086954e-42  
## 4 1.000000e+00 6.638640e-17 4.570149e-38  
## 5 1.000000e+00 5.322410e-23 3.803940e-46  
## 6 1.000000e+00 1.640808e-21 2.935221e-43  
## 7 1.000000e+00 8.783454e-19 5.425380e-40  
## 8 1.000000e+00 1.292189e-20 4.637444e-43  
## 9 1.000000e+00 1.406734e-15 2.410265e-36  
## 10 1.000000e+00 2.885342e-19 8.125004e-42  
## 12 1.000000e+00 5.669659e-19 1.030990e-40  
## 13 1.000000e+00 4.710254e-19 1.420397e-41  
## 14 1.000000e+00 6.702074e-20 1.646484e-42  
## 15 1.000000e+00 9.738758e-31 9.291565e-57  
## 16 1.000000e+00 3.886372e-28 5.081464e-52  
## 17 1.000000e+00 4.104905e-25 3.309728e-48  
## 18 1.000000e+00 2.222260e-21 1.083564e-43  
## 19 1.000000e+00 3.940627e-23 4.386070e-46  
## 20 1.000000e+00 1.047286e-22 3.149143e-45  
## 22 1.000000e+00 1.027183e-20 3.219734e-42  
## 23 1.000000e+00 4.054510e-25 7.118894e-49  
## 24 1.000000e+00 1.074673e-14 5.820981e-34  
## 25 1.000000e+00 2.850184e-16 5.298354e-37  
## 26 1.000000e+00 9.579053e-17 3.993986e-38  
## 27 1.000000e+00 3.237064e-17 1.184168e-37  
## 28 1.000000e+00 4.237835e-22 4.282758e-45  
## 29 1.000000e+00 2.945065e-22 2.085945e-45  
## 30 1.000000e+00 4.066606e-17 2.614232e-38  
## 31 1.000000e+00 9.565891e-17 6.121795e-38  
## 32 1.000000e+00 1.336989e-19 4.134507e-41  
## 33 1.000000e+00 8.387138e-28 9.985936e-53  
## 36 1.000000e+00 7.848542e-22 6.374925e-45  
## 38 1.000000e+00 7.043867e-24 1.122417e-47  
## 39 1.000000e+00 3.201723e-17 1.661733e-38  
## 40 1.000000e+00 5.500849e-21 1.292028e-43  
## 42 1.000000e+00 3.805461e-11 1.668901e-30  
## 43 1.000000e+00 1.048588e-18 2.352222e-40  
## 45 1.000000e+00 7.430414e-18 3.397247e-38  
## 46 1.000000e+00 1.483990e-16 2.101755e-37  
## 47 1.000000e+00 4.691486e-23 4.467538e-46  
## 48 1.000000e+00 1.510953e-18 3.150843e-40  
## 49 1.000000e+00 5.908387e-24 1.689016e-47  
## 50 1.000000e+00 8.980009e-21 2.258697e-43  
## 51 1.041015e-18 9.999891e-01 1.087689e-05  
## 53 7.140874e-23 9.991846e-01 8.153758e-04  
## 54 2.112035e-23 9.998238e-01 1.761583e-04  
## 55 6.982386e-24 9.984711e-01 1.528888e-03  
## 57 4.595838e-23 9.935123e-01 6.487709e-03  
## 59 2.859793e-20 9.999860e-01 1.402767e-05  
## 61 3.111010e-19 9.999997e-01 3.331905e-07  
## 62 5.077618e-21 9.996206e-01 3.794385e-04  
## 63 1.529146e-18 9.999999e-01 9.997688e-08  
## 64 2.827964e-24 9.982654e-01 1.734632e-03  
## 65 5.646316e-15 9.999996e-01 4.196269e-07  
## 66 7.306297e-18 9.999942e-01 5.773253e-06  
## 67 7.688648e-25 9.863467e-01 1.365332e-02  
## 68 1.795418e-16 9.999999e-01 6.221352e-08  
## 70 3.795578e-18 9.999995e-01 4.889401e-07  
## 72 1.828527e-17 9.999983e-01 1.700416e-06  
## 74 1.615104e-22 9.999437e-01 5.627340e-05  
## 77 2.544162e-23 9.997051e-01 2.949403e-04  
## 78 7.222677e-28 7.719688e-01 2.280312e-01  
## 79 4.278989e-24 9.961519e-01 3.848075e-03  
## 80 3.495519e-12 1.000000e+00 1.383587e-09  
## 81 2.325042e-18 9.999995e-01 5.235941e-07  
## 82 3.281429e-16 1.000000e+00 3.520291e-08  
## 83 3.602930e-17 9.999994e-01 6.203308e-07  
## 84 4.237543e-33 1.351148e-01 8.648852e-01  
## 86 1.242682e-21 9.967593e-01 3.240720e-03  
## 87 8.184929e-22 9.995957e-01 4.043066e-04  
## 88 4.899315e-24 9.998684e-01 1.315622e-04  
## 89 9.816482e-19 9.999867e-01 1.328383e-05  
## 90 6.449463e-22 9.999239e-01 7.614524e-05  
## 92 1.244020e-22 9.994739e-01 5.260869e-04  
## 93 8.200250e-19 9.999980e-01 2.047850e-06  
## 94 3.318542e-15 1.000000e+00 2.009972e-08  
## 95 7.316619e-22 9.998985e-01 1.014528e-04  
## 96 5.147677e-18 9.999972e-01 2.753263e-06  
## 97 5.248283e-20 9.999713e-01 2.869996e-05  
## 98 4.721377e-19 9.999925e-01 7.479295e-06  
## 99 6.741322e-12 1.000000e+00 3.811517e-09  
## 100 7.552141e-20 9.999799e-01 2.011470e-05  
## 101 1.009229e-55 1.937494e-10 1.000000e+00  
## 102 2.112482e-40 2.078519e-04 9.997921e-01  
## 103 1.768983e-44 7.829998e-06 9.999922e-01  
## 104 7.063637e-40 5.698091e-04 9.994302e-01  
## 105 1.173713e-48 1.948640e-07 9.999998e-01  
## 106 2.312204e-50 2.875106e-07 9.999997e-01  
## 107 1.170806e-35 9.622148e-03 9.903779e-01  
## 108 5.502166e-43 1.743185e-04 9.998257e-01  
## 110 3.433312e-49 1.426162e-08 1.000000e+00  
## 111 5.565080e-34 4.779747e-03 9.952203e-01  
## 113 3.392610e-41 4.869741e-05 9.999513e-01  
## 114 1.182120e-43 1.862311e-05 9.999814e-01  
## 119 2.186798e-61 1.709231e-10 1.000000e+00  
## 120 1.907526e-34 2.220110e-01 7.779890e-01  
## 122 3.374494e-40 9.308690e-05 9.999069e-01  
## 123 8.280835e-51 5.995562e-07 9.999994e-01  
## 125 1.884681e-41 2.381304e-05 9.999762e-01  
## 126 4.659444e-37 4.088421e-03 9.959116e-01  
## 127 1.644048e-31 1.077437e-01 8.922563e-01  
## 128 1.938247e-31 7.770382e-02 9.222962e-01  
## 129 1.676803e-46 1.779329e-06 9.999982e-01  
## 130 2.095681e-32 2.818783e-01 7.181217e-01  
## 131 5.752553e-43 1.203593e-04 9.998796e-01  
## 136 8.583635e-48 4.503977e-07 9.999995e-01  
## 137 9.150940e-48 4.481044e-08 1.000000e+00  
## 138 3.079159e-36 4.355313e-03 9.956447e-01  
## 140 1.765870e-38 2.455876e-04 9.997544e-01  
## 141 2.557813e-48 6.940106e-08 9.999999e-01  
## 143 2.112482e-40 2.078519e-04 9.997921e-01  
## 144 1.826488e-48 1.081312e-07 9.999999e-01  
## 145 8.608022e-50 1.079577e-08 1.000000e+00  
## 146 3.282912e-42 6.937421e-06 9.999931e-01  
## 147 3.069673e-38 1.621483e-03 9.983785e-01  
## 148 4.582390e-37 9.555276e-04 9.990445e-01  
## 149 9.235682e-44 9.466906e-07 9.999991e-01  
## 150 5.436083e-35 7.603071e-03 9.923969e-01  
##   
## $x  
## LD1 LD2  
## 1 7.950822348 0.305651046  
## 3 7.315985398 -0.163697151  
## 4 6.649894951 -0.575196673  
## 5 8.017654721 0.532519428  
## 6 7.560677233 1.447538794  
## 7 6.995489033 0.498007524  
## 8 7.492614829 0.001334767  
## 9 6.357128699 -0.862279505  
## 10 7.263157567 -0.949818516  
## 12 7.101239683 -0.076113131  
## 13 7.220715905 -1.039768311  
## 14 7.385634148 -0.643880223  
## 15 9.801703373 1.475947372  
## 16 9.044573823 2.677305809  
## 17 8.392208948 1.876271762  
## 18 7.612798058 0.635669872  
## 19 8.015361368 0.860916241  
## 20 7.880650573 1.164488759  
## 22 7.384047802 1.282506876  
## 23 8.482202001 1.020723086  
## 24 5.995943729 0.550156379  
## 25 6.477590897 -0.397662857  
## 26 6.650417975 -0.953851314  
## 27 6.608683320 0.554189177  
## 28 7.834685528 0.183600131  
## 29 7.883989975 0.078782664  
## 30 6.692336612 -0.485246878  
## 31 6.625504239 -0.712115260  
## 32 7.183550684 0.601901729  
## 33 9.124180706 1.125585563  
## 36 7.799106653 -0.101116926  
## 38 8.263932902 0.217368274  
## 39 6.723590109 -0.543095554  
## 40 7.584360938 -0.013532906  
## 42 5.367262559 -1.711949368  
## 43 7.040747072 -0.119094134  
## 45 6.711094568 1.065774617  
## 46 6.544667325 -0.379730659  
## 47 8.010791934 0.727286691  
## 48 7.016356361 -0.256012721  
## 49 8.243588599 0.592733878  
## 50 7.541919277 -0.103482701  
## 51 -1.698165161 -0.189657935  
## 53 -2.702279898 -0.271138630  
## 54 -2.708358331 -1.454385367  
## 55 -2.921350991 -0.526120343  
## 57 -2.857858020 0.786454134  
## 59 -1.994977821 -0.989024958  
## 61 -1.589336805 -2.470189402  
## 62 -2.323138965 0.415820079  
## 63 -1.394133399 -2.730780917  
## 64 -2.999615583 -0.691851012  
## 65 -0.833609620 0.231484187  
## 66 -1.508333182 -0.035505901  
## 67 -3.222026078 0.138873370  
## 68 -0.992616139 -1.748225266  
## 70 -1.415523751 -1.598106030  
## 72 -1.364989273 -0.483587853  
## 74 -2.482145484 -1.563889374  
## 77 -2.723854232 -1.115108671  
## 78 -3.928282106 0.099450415  
## 79 -3.013620124 -0.132598029  
## 80 0.004356844 -1.302258851  
## 81 -1.457965413 -1.688055826  
## 82 -0.912058194 -1.910891410  
## 83 -1.252898861 -0.873821130  
## 84 -4.916098949 -0.869680075  
## 86 -2.558752008 1.257424345  
## 87 -2.470006258 -0.027036801  
## 88 -2.805921177 -2.002059715  
## 89 -1.714445782 -0.092431314  
## 90 -2.391201369 -1.030383948  
## 92 -2.633154173 -0.372667060  
## 93 -1.619360271 -1.193005081  
## 94 -0.697835503 -1.619820789  
## 95 -2.398064155 -0.835616685  
## 96 -1.492558313 -0.544501055  
## 97 -1.989161084 -0.426482938  
## 98 -1.738313469 -0.608004542  
## 99 -0.003307936 -0.559118489  
## 100 -1.939856636 -0.531300406  
## 101 -8.602554705 2.363241422  
## 102 -6.113664037 0.150111748  
## 103 -6.784341190 0.395405851  
## 104 -6.039166885 -0.366160231  
## 105 -7.464959204 0.921813954  
## 106 -7.780791148 -0.429215206  
## 107 -5.333189825 -0.157018819  
## 108 -6.576886299 -1.265119650  
## 110 -7.508987212 2.758251258  
## 111 -5.016573160 1.436060416  
## 113 -6.228047801 0.868741837  
## 114 -6.652708470 0.178180069  
## 119 -9.623056331 -0.953597791  
## 120 -5.163084137 -2.152519208  
## 122 -6.060836206 0.936233113  
## 123 -7.876060641 -1.305286366  
## 125 -6.259824324 1.305245154  
## 126 -5.569248178 -0.292700122  
## 127 -4.626428045 0.294172668  
## 128 -4.608900119 0.625858518  
## 129 -7.120072127 0.397027865  
## 130 -4.794590703 -1.162372710  
## 131 -6.565977104 -0.947602722  
## 136 -7.325678976 0.751870985  
## 137 -7.274420219 2.673956274  
## 138 -5.422380885 0.150156758  
## 140 -5.769840283 1.173058116  
## 141 -7.383171228 1.978483456  
## 143 -6.113664037 0.150111748  
## 144 -7.418471134 1.524047940  
## 145 -7.611921484 2.625320459  
## 146 -6.372193704 1.865196888  
## 147 -5.764207527 -0.241044791  
## 148 -5.541613051 0.904875754  
## 149 -6.612376180 2.573171605  
## 150 -5.208158194 0.441227379

#histogram1 is based on LD1  
ldahist(data = p$x[,1],g = training\_data$Species)

#histogram2 is based on LD2  
  
ldahist(data = p$x[,2],g = training\_data$Species)

#bi-plot  
#install.packages("devtools")  
  
library(devtools)

## Warning: package 'devtools' was built under R version 3.5.1

#install\_github("fawda123/ggord")  
library(ggord)  
  
ggord(linear, training\_data$Species,ylim=c(-10,10))

#partitioning plot  
  
#install.packages("CRAN")  
  
#install.packages("klaR")  
library(klaR)

## Warning: package 'klaR' was built under R version 3.5.1

partimat(Species~.,data = training\_data, method = "lda")

#Partition plot on the bases of Quadratic Discriminent Analysis(QDA)  
partimat(Species~.,data = training\_data, method = "qda")

# confusion matrix and accuracy with training data  
p1 <- predict(linear, training\_data)$class  
p1

## [1] setosa setosa setosa setosa setosa setosa   
## [7] setosa setosa setosa setosa setosa setosa   
## [13] setosa setosa setosa setosa setosa setosa   
## [19] setosa setosa setosa setosa setosa setosa   
## [25] setosa setosa setosa setosa setosa setosa   
## [31] setosa setosa setosa setosa setosa setosa   
## [37] setosa setosa setosa setosa setosa setosa   
## [43] versicolor versicolor versicolor versicolor versicolor versicolor  
## [49] versicolor versicolor versicolor versicolor versicolor versicolor  
## [55] versicolor versicolor versicolor versicolor versicolor versicolor  
## [61] versicolor versicolor versicolor versicolor versicolor versicolor  
## [67] virginica versicolor versicolor versicolor versicolor versicolor  
## [73] versicolor versicolor versicolor versicolor versicolor versicolor  
## [79] versicolor versicolor versicolor virginica virginica virginica   
## [85] virginica virginica virginica virginica virginica virginica   
## [91] virginica virginica virginica virginica virginica virginica   
## [97] virginica virginica virginica virginica virginica virginica   
## [103] virginica virginica virginica virginica virginica virginica   
## [109] virginica virginica virginica virginica virginica virginica   
## [115] virginica virginica virginica   
## Levels: setosa versicolor virginica

tab <- table(Predicted = p1, Actual = training\_data$Species)  
tab

## Actual  
## Predicted setosa versicolor virginica  
## setosa 42 0 0  
## versicolor 0 38 0  
## virginica 0 1 36

#Check Accuracy  
sum(diag(tab)/sum(tab))

## [1] 0.991453

#confusion matrix and accuracy with testing data  
  
p2 <- predict(linear, testing\_data)$class  
tab1 <- table(predicted = p2, Actual = testing\_data$Species)  
tab1

## Actual  
## predicted setosa versicolor virginica  
## setosa 8 0 0  
## versicolor 0 10 1  
## virginica 0 1 13

sum(diag(tab1)/sum(tab1))

## [1] 0.9393939

#Analysis by Using Support vector Machine  
  
#set.seed(7)  
#fit.svm <- train(Species~., data=iris, method="svmRadial", metric=metric, trControl=control)  
  
#fit.svm  
  
  
#Analysis by Using Random Forest  
#set.seed(7)  
#fit.rf <- train(Species~., data=iris, method="rf", metric=metric, trControl=control)  
  
#fit.rf

NA