IRIS FLOWER CLASSIFICATION

SAGNIK SAMANTA

CODSOFT TASK2

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.

Import the Dataset

```
IRIS=read.csv("C:/Users/shrey/Desktop/Datasets/IRIS_Data.csv",sep=",",header=T)
head(IRIS)
```

```
##   sepal_length sepal_width petal_length petal_width     species
## 1          5.1         3.5          1.4         0.2 Iris-setosa
## 2          4.9         3.0          1.4         0.2 Iris-setosa
## 3          4.7         3.2          1.3         0.2 Iris-setosa
## 4          4.6         3.1          1.5         0.2 Iris-setosa
## 5          5.0         3.6          1.4         0.2 Iris-setosa
## 6          5.4         3.9          1.7         0.4 Iris-setosa
```

Dimention of the Dataset

```
dim(IRIS)
```

```
[1] 150   5
```

Hence IRIS dataset has 150 number of rows and 5 number of columns.

Column Names

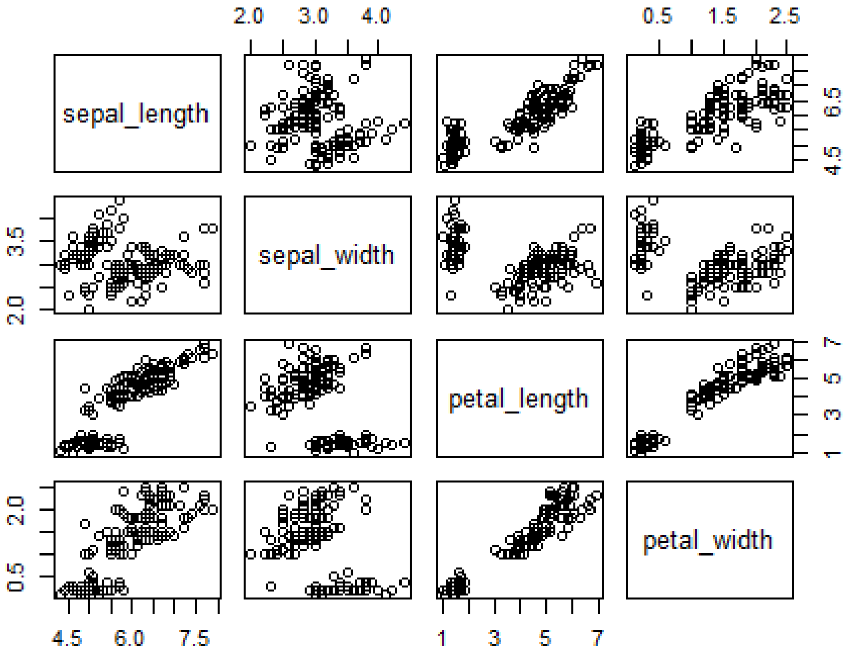
```
names(IRIS)
```

```
[1] "sepal_length" "sepal_width"  "petal_length" "petal_width"  "species"
```

Data type

```
str(IRIS)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ sepal_length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ sepal_width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ petal_length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ petal_width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ species     : chr  "Iris-setosa" "Iris-setosa" "Iris-setosa" "Iris-
setosa" ...

Visualization
pairs(IRIS[,1:4])
```



```
Discriminant Analysis

library(MASS)

library(ggplot2)
```

```
attach(IRIS)

scale each predictor variable
IRIS[1:4]=scale(IRIS[1:4])
find mean of each predictor variable
apply(IRIS[1:4], 2, mean)

 sepal_length   sepal_width  petal_length   petal_width 
-4.484318e-16  3.827274e-16  1.031799e-17 -1.581504e-16 

find standard deviation of each predictor variable
apply(IRIS[1:4], 2, sd)
```

```
sepal_length  sepal_width petal_length  petal_width 
           1            1            1            1 

Splitting the dataset into train and test dataset
set.seed(1)

Use 75% of dataset as training set and remaining 25% as testing set
sample=sample(c(TRUE, FALSE), nrow(IRIS), replace=TRUE, prob=c(0.75,0.25))
train=IRIS[sample, ]
test=IRIS[!sample, ]

fit LDA model
model=lda(species ~ ., data=train)

view model output
model

## Call:
## lda(species ~ ., data = train)
## 
## Prior probabilities of groups:
##     Iris-setosa Iris-versicolor  Iris-virginica 
##       0.3217391       0.3130435       0.3652174 
## 
## Group means:
##                 sepal_length sepal_width petal_length petal_width
## Iris-setosa       -1.0347565   0.8166807   -1.2939042  -1.2519295
## Iris-versicolor    0.1891958  -0.6050202    0.3351431   0.2128574
## Iris-virginica     0.9454041  -0.1794524    1.0300968   1.1217758
## 
## Coefficients of linear discriminants:
##                     LD1        LD2
## sepal_length  0.7658350  0.3865457
## sepal_width   0.5948438  0.7285488
## petal_length -4.1071869 -2.5628395
## petal_width  -2.0633305  2.5941262
## 
## Proportion of trace:
##    LD1    LD2 
## 0.9922 0.0078

Prior probabilities of group: These represent the proportions of each Species
in the training set.
Group means: These display the mean values for each predictor variable for
each Species Groups.
Coefficients of linear discriminants: These display the linear combination of
predictor variables that are used to form the decision rule of the LDA model
Proportion of trace: These display the percentage separation achieved by each
linear discriminant function.
Based on the training dataset, 32.17391% belongs to Iris-setosa group,
```

31.30435% belongs to Iris-versicolor groups and 36.52174% belongs to Iris-virginica groups.

use LDA model to make predictions on test data

```
predicted=predict(model, test)
```

```
names(predicted)
```

```
[1] "class"     "posterior" "x"
```

class: The predicted class
posterior: The posterior probability that an observation belongs to each class
x: The linear discriminants

view predicted class for first six observations in test set

```
head(predicted$class)
```

```
[1] Iris-setosa Iris-setosa Iris-setosa Iris-setosa Iris-setosa Iris-setosa
Levels: Iris-setosa Iris-versicolor Iris-virginica
```

view posterior probabilities for first six observations in test set

```
head(predicted$posterior)
```

```
##    Iris-setosa Iris-versicolor Iris-virginica
## 4            1    1.152468e-17   3.093008e-36
## 6            1    5.378758e-22   6.786877e-41
## 7            1    1.247472e-19   2.502455e-38
## 15           1    2.115233e-31   7.728632e-54
## 18           1    2.673118e-22   8.317657e-42
## 20           1    3.418176e-23   8.089241e-43
```

view linear discriminants for first six observations in test set

```
head(predicted$x)
```

```
##          LD1        LD2
## 4   7.229447 -0.6750038
## 6   8.060548  1.4319795
## 7   7.603426  0.3142424
## 15 10.272300  1.8331449
## 18  8.203039  0.7156705
## 20  8.381827  1.0744958
```

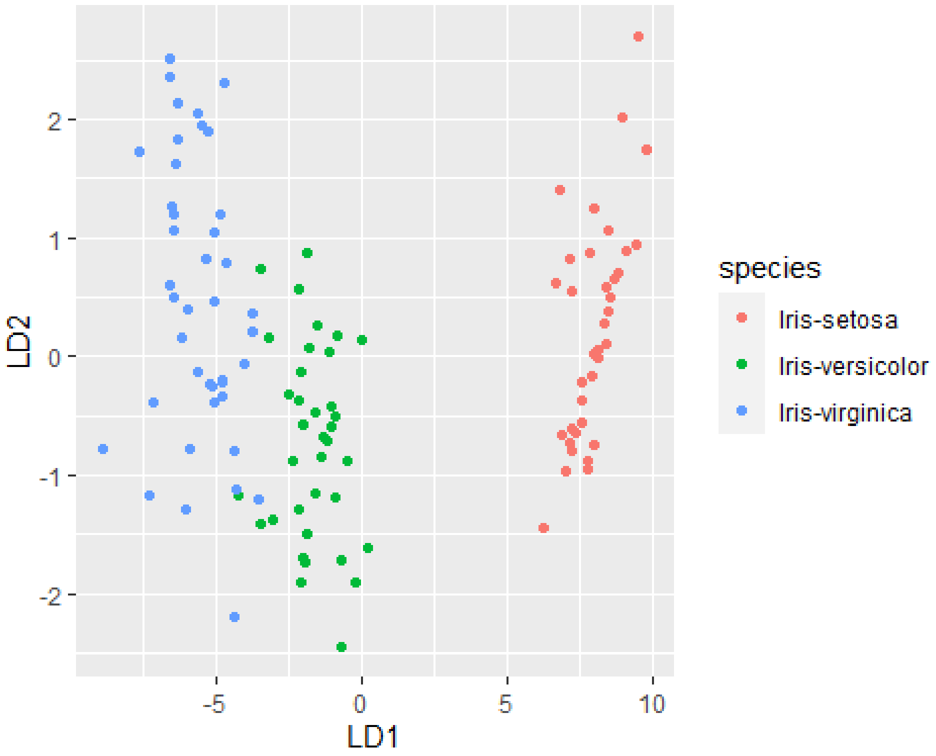
find accuracy of model

```
mean(predicted$class==test$species)
```

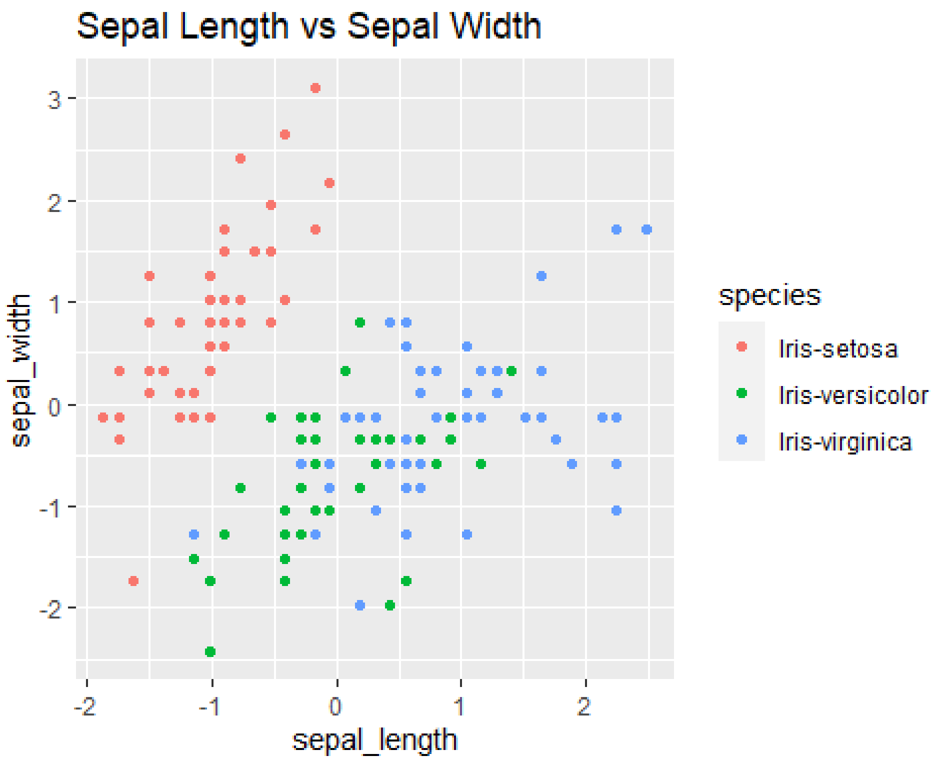
```
[1] 1
```

It turns out that the model correctly predicted the Species for 100% of the observations in our test dataset.

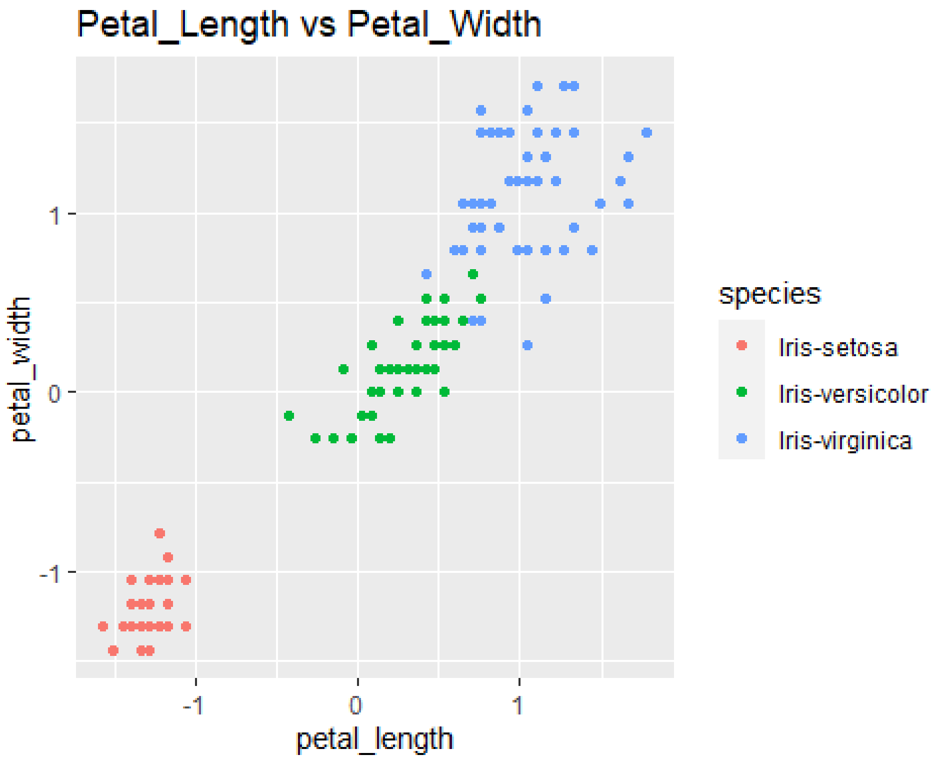
```
define data to plot
lda_plot=cbind(train, predict(model)$x)

Create plot
ggplot(lda_plot, aes(LD1, LD2)) +
  geom_point(aes(color = species))
```



```
Support Vector Machine
install.packages(c("tidyverse","e1071"))
library(tidyverse)

library(e1071)

ggplot(IRIS, aes(x = sepal_length, y = sepal_width, colour = species)) +
  geom_point() +
  labs(title = 'Sepal Length vs Sepal Width')
```



```
ggplot(IRIS, aes(x = petal_length, y = petal_width, colour = species)) +
  geom_point() +
  labs(title = 'Petal_Length vs Petal_Width')
```



```
Splitting the dataset into train and test dataset
set.seed(2)

Use 75% of dataset as training set and remaining 25% as testing set
sample1=sample(c(TRUE, FALSE), nrow(IRIS), replace=TRUE, prob=c(0.75,0.25))
IRIS$species=as.factor(IRIS$species)
train=IRIS[sample1, ]
test=IRIS[!sample1,]

attach(IRIS)
```

```
## The following objects are masked from IRIS (pos = 13):
##
##     petal_length, petal_width, sepal_length, sepal_width, species
```

```
train1=subset(IRIS, select = -species)
test1=species
model=svm(train1, test1)

print(model)
```

```
##
## Call:
## svm.default(x = train1, y = test1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  51
```

```
summary(model)
```

```
##
## Call:
## svm.default(x = train1, y = test1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  51
##
##  ( 8 22 21 )
##
##
## Number of Classes:  3
```

```
##
## Levels:
##  Iris-setosa Iris-versicolor Iris-virginica

test with train data
pred= predict(model, train1)

Check accuracy
tab=table(pred,test1)
tab

                 test1
pred              Iris-setosa Iris-versicolor Iris-virginica
  Iris-setosa              50               0              0
  Iris-versicolor           0              48              2
  Iris-virginica            0               2             48

accuracy=sum(diag(tab)/nrow(IRIS))
accuracy

[1] 0.9733333
```