Experiment -2

1. Use decision tree for dimension reduction of the given data set and find the classification techniques with efficiency.
2. Use Random Forest for dimension reduction of the given data set and find the classification techniques with efficiency.
3. Use LDA (Linear Discriminant Analysis) for solving any multi class classification problem.

Aim: Understanding and implementing the concepts of decision tree, random forest and LDA for dimensionality reduction perspectives.

Description:

- The **decision tree models** have shown excellent results for both classification and dimensionality reduction, and provide easily interpretable tree structure that show variable importance.
- **Decision Tree Ensembles**, also referred to as **random forests**, are useful for feature selection in addition to being effective classifiers.
- One approach to dimensionality reduction is to generate a large and carefully constructed set of trees against a target attribute and then use each attribute's usage statistics to find the most informative subset of features.
- **Linear Discriminant Analysis**, or LDA for short, is a predictive modeling algorithm for multi-class classification. It can also be used as a dimensionality reduction technique, providing a projection of a training dataset that best separates the examples by their assigned class.

Algorithm:

1. First, take absolute values of correlation matrix (Use abs(Corrmatrix) function in R)
2. Replace all diagonal values (1s) in the matrix with NAs (Use diag(Corrmatrix) <- NA)
3. Compute mean of each column.

4. Calculate rank of the mean values (calculated in Step 3) on descending order.
5. Reorder correlation matrix based on the Rank.
6. Now, checks if the matrix(i,j) > cutoff, then calculates the following steps -
    i). Mean value of the row of matrix(i ,).
    ii). Mean of the row of matrix(-j ,).

Program:

```
Corrmatrix <- structure(c(1, 0.82, 0.54, 0.36, 0.85, 0.82, 1, 0.01, 0.74, 0.36,
                          0.54, 0.01, 1, 0.65, 0.91, 0.36, 0.74, 0.65, 1, 0.36,
                          0.85, 0.36, 0.91, 0.36, 1),
                        .Dim = c(5L, 5L))
Corrmatrix

library(caret)
findCorrelation(Corrmatrix, cutoff = .6, verbose = TRUE, names = F)


## R Code : Removing Redundant Variables

# load required libraries
library(caret)
library(corrplot)
library(plyr)

# load required dataset
dat <- read.csv("H:\\JGi Classes\\Dimentionality Reduction and Model Validation VII Sem
2018-22\\Lab Programs\\pml-training.csv")
```

```
# Set seed
set.seed(227)

# Remove variables having high missing percentage (50%)
dat1 <- dat[, colMeans(is.na(dat)) <= .5]
dim(dat1)

# Remove Zero and Near Zero-Variance Predictors
nzv <- nearZeroVar(dat1)
dat2 <- dat1[, -nzv]
dim(dat2)

# Identifying numeric variables
numericData <- dat2[sapply(dat2, is.numeric)]

# Calculate correlation matrix
descrCor <- cor(numericData)

# Print correlation matrix and look at max correlation
print(descrCor)
summary(descrCor[upper.tri(descrCor)])

# Check Correlation Plot
corrplot(descrCor, order = "FPC", method = "color", type = "lower", tl.cex = 0.7, tl.col = rgb(0,
0, 0))

# find attributes that are highly corrected
highlyCorrelated <- findCorrelation(descrCor, cutoff=0.7)

# print indexes of highly correlated attributes
print(highlyCorrelated)
```

```
# Indentifying Variable Names of Highly Correlated Variables
highlyCorCol <- colnames(numericData)[highlyCorrelated]

# Print highly correlated attributes
highlyCorCol

# Remove highly correlated variables and create a new dataset
dat3 <- dat2[, -which(colnames(dat2) %in% highlyCorCol)]
dim(dat3)


## R Code : Feature Selection with Random Forest

library(randomForest)

#Train Random Forest
rf <-randomForest(classe~.,data=dat3, importance=TRUE,ntree=1000)

#Evaluate variable importance
imp = importance(rf, type=1)
imp <- data.frame(predictors=rownames(imp),imp)

# Order the predictor levels by importance
imp.sort <- arrange(imp,desc(MeanDecreaseAccuracy))
imp.sort$predictors <- factor(imp.sort$predictors,levels=imp.sort$predictors)

# Select the top 20 predictors
imp.20 <- imp.sort[1:20,]
print(imp.20)
```

```
# Plot Important Variables
varImpPlot(rf, type=1)

# Subset data with 20 independent and 1 dependent variables
dat4 = cbind(classe = dat3$classe, dat3[,c(imp.20$predictors)])
dim(dat4)
colnames(dat4)

## Implementation of LDA

# Load Library

library(klaR)
library(psych)
library(MASS)
library(ggord)
library(devtools)

## Getting Data
data("iris")
str(iris)

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

## Data Partition

set.seed(123)
ind <- sample(2, nrow(iris),
```

```
            replace = TRUE,
            prob = c(0.6, 0.4))
training <- iris[ind==1,]
testing <- iris[ind==2,]

## Linear discriminant analysis
linear <- lda(Species~., training)
linear

attributes(linear)

## Histogram
## Stacked histogram for discriminant function values.

p <- predict(linear, training)
ldahist(data = p$x[,1], g = training$Species)

ldahist(data = p$x[,2], g = training$Species)



## Partition plot
# It provides the classification of each and every combination in the training dataset.

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



## Confusion matrix and accuracy – training data
p1 <- predict(linear, training)$class
tab <- table(Predicted = p1, Actual = training$Species)
tab
```

```
## Confusion matrix and accuracy – testing data
p2 <- predict(linear, testing)$class
tab1 <- table(Predicted = p2, Actual = testing$Species)
tab1
```

Output:

```
> # Indentifying Variable Names of Highly Correlated Variables
> highlyCorCol <- colnames(numericData)[highlyCorrelated]
> # Print highly correlated attributes
> highlyCorCol
 [1] "accel_belt_z"      "roll_belt"         "accel_belt_y"
 [4] "accel_arm_y"       "total_accel_belt"  "yaw_belt"
 [7] "accel_dumbbell_z"  "accel_belt_x"      "magnet_belt_x"
[10] "magnet_dumbbell_x" "accel_dumbbell_y"  "magnet_dumbbell_y"
[13] "magnet_dumbbell_z" "accel_arm_x"       "accel_dumbbell_x"
[16] "accel_arm_z"       "magnet_arm_y"      "magnet_belt_y"
[19] "accel_forearm_y"   "gyros_forearm_z"   "gyros_forearm_y"
[22] "gyros_dumbbell_z"  "gyros_arm_x"
> # Remove highly correlated variables and create a new dataset
> dat3 <- dat2[, -which(colnames(dat2) %in% highlyCorCol)]
> dim(dat3)
[1] 19622    36


> dim(dat4)
[1] 19622    21
> colnames(dat4)
 [1] "classe"              "X"
 [3] "user_name"           "raw_timestamp_part_1"
 [5] "raw_timestamp_part_2" "cvtd_timestamp"
 [7] "num_window"          "pitch_belt"
 [9] "gyros_belt_x"        "gyros_belt_y"
[11] "gyros_belt_z"        "magnet_belt_z"
[13] "roll_arm"            "pitch_arm"
[15] "yaw_arm"             "total_accel_arm"
[17] "gyros_arm_y"         "gyros_arm_z"
[19] "magnet_arm_x"        "magnet_arm_z"
[21] "roll_dumbbell"


> ## Linear discriminant analysis
> linear <- lda(Species~., training)
> linear
Call:
lda(Species ~ ., data = training)

Prior probabilities of groups:
    setosa versicolor  virginica
 0.3370787  0.3370787  0.3258427

Group means:
```

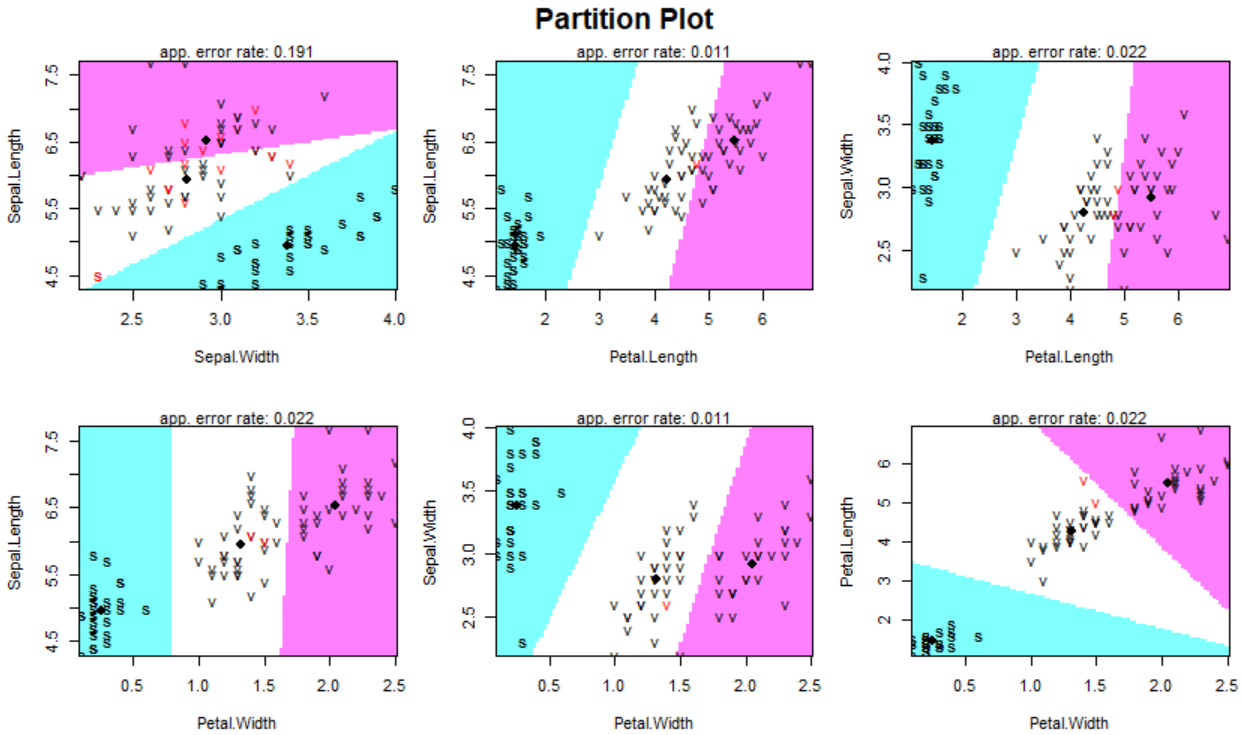
```
           Sepal.Length Sepal.Width Petal.Length Petal.Width
setosa         4.946667    3.380000     1.443333    0.250000
versicolor     5.943333    2.803333     4.240000    1.316667
virginica      6.527586    2.920690     5.489655    2.048276

Coefficients of linear discriminants:
                    LD1         LD2
Sepal.Length  0.3629008  0.05215114
Sepal.Width   2.2276982  1.47580354
Petal.Length -1.7854533 -1.60918547
Petal.Width  -3.9745504  4.10534268

Proportion of trace:
   LD1    LD2
0.9932 0.0068
> attributes(linear)
$names
 [1] "prior"   "counts"  "means"   "scaling" "lev"     "svd"
 [7] "N"       "call"    "terms"   "xlevels"

$class
[1] "lda"
```

```
> partimat(Species~., data = training, method = "lda")
```




```
> ## Confusion matrix and accuracy – training data
> p1 <- predict(linear, training)$class
> tab <- table(Predicted = p1, Actual = training$Species)
> tab
```

```
          Actual
Predicted    setosa versicolor virginica
  setosa         30          0         0
  versicolor      0         30         0
  virginica       0          0        29
```

```
> ## Confusion matrix and accuracy – testing data
> p2 <- predict(linear, testing)$class
> tab1 <- table(Predicted = p2, Actual = testing$Species)
> tab1
          Actual
Predicted    setosa versicolor virginica
  setosa         20          0         0
  versicolor      0         19         1
  virginica       0          1        20
```