SAPIENZA
UNIVERSITÀ DI ROMA

MACHINE LEARNING
FOREST COVER PREDICTION

Aakash Nandrajog
1734002

Professor Paola Velardi

Contents

1. Introduction

The study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. Each observation is a 30m x 30m patch. The task is to predict an integer classification for the forest cover type. The seven types are:

1 - Spruce/Fir
2 - Lodgepole Pine
3 - Ponderosa Pine
4 - Cottonwood/Willow
5 - Aspen
6 - Douglas-fir
7 - Krummholz

The training set (15120 observations) contains both features and the Cover_Type.

The software used for solving this problem is be Rstudio and the programming language used is R.

```
> R.version
                _
platform       x86_64-pc-linux-gnu
arch           x86_64
os             linux-gnu
system         x86_64, linux-gnu
status
major          3
minor          3.3
year           2017
month          03
day            06
svn rev        72310
language       R
version.string R version 3.3.3 (2017-03-06)
nickname       Another Canoe
```

2. Summary of the Data Set

Following is a description of the covariates of the dataset.

Cover_Type for every row in the test set (565892 observations).

Data Fields

Elevation - Elevation in meters
Aspect - Aspect in degrees azimuth
Slope - Slope in degrees
Horizontal_Distance_To_Hydrology - Horz Dist to nearest surface water features
Vertical_Distance_To_Hydrology - Vert Dist to nearest surface water features
Horizontal_Distance_To_Roadways - Horz Dist to nearest roadway
Hillshade_9am (0 to 255 index) - Hillshade index at 9am, summer solstice
Hillshade_Noon (0 to 255 index) - Hillshade index at noon, summer solstice
Hillshade_3pm (0 to 255 index) - Hillshade index at 3pm, summer solstice
Horizontal_Distance_To_Fire_Points - Horz Dist to nearest wildfire ignition points
Wilderness_Area (4 binary columns, 0 = absence or 1 = presence) - Wilderness area designation
Soil_Type (40 binary columns, 0 = absence or 1 = presence) - Soil Type designation
Cover_Type (7 types, integers 1 to 7) - Forest Cover Type designation

The wilderness areas are:

1 - Rawah Wilderness Area
2 - Neota Wilderness Area
3 - Comanche Peak Wilderness Area
4 - Cache la Poudre Wilderness Area

The soil types are:

1 Cathedral family - Rock outcrop complex, extremely stony.
2 Vanet - Ratake families complex, very stony.
3 Haploborolis - Rock outcrop complex, rubbly.
4 Ratake family - Rock outcrop complex, rubbly.
5 Vanet family - Rock outcrop complex complex, rubbly.
6 Vanet - Wetmore families - Rock outcrop complex, stony.
7 Gothic family.
8 Supervisor - Limber families complex.
9 Troutville family, very stony.
10 Bullwark - Catamount families - Rock outcrop complex, rubbly.
11 Bullwark - Catamount families - Rock land complex, rubbly.
12 Legault family - Rock land complex, stony.
13 Catamount family - Rock land - Bullwark family complex, rubbly.
14 Pachic Argiborolis - Aquolis complex.
15 unspecified in the USFS Soil and ELU Survey.
16 Cryaquolis - Cryoborolis complex.
17 Gateview family - Cryaquolis complex.
18 Rogert family, very stony.
19 Typic Cryaquolis - Borohemists complex.
20 Typic Cryaquepts - Typic Cryaquolls complex.
21 Typic Cryaquolls - Leighcan family, till substratum complex.
22 Leighcan family, till substratum, extremely bouldery.
23 Leighcan family, till substratum - Typic Cryaquolls complex.
24 Leighcan family, extremely stony.
25 Leighcan family, warm, extremely stony.
26 Granile - Catamount families complex, very stony.
27 Leighcan family, warm - Rock outcrop complex, extremely stony.

28 Leighcan family - Rock outcrop complex, extremely stony.
29 Como - Legault families complex, extremely stony.
30 Como family - Rock land - Legault family complex, extremely stony.
31 Leighcan - Catamount families complex, extremely stony.
32 Catamount family - Rock outcrop - Leighcan family complex, extremely stony.
33 Leighcan - Catamount families - Rock outcrop complex, extremely stony.
34 Cryorthents - Rock land complex, extremely stony.
35 Cryumbrepts - Rock outcrop - Cryaquepts complex.
36 Bross family - Rock land - Cryumbrepts complex, extremely stony.
37 Rock outcrop - Cryumbrepts - Cryorthents complex, extremely stony.
38 Leighcan - Moran families - Cryaquolls complex, extremely stony.
39 Moran family - Cryorthents - Leighcan family complex, extremely stony.
40 Moran family - Cryorthents - Rock land complex, extremely stony.

In R, the summary() command explains some features of the dataset.

```
Console ~/Regression_SL/
> summary(data_forest)
      Id          Elevation        Aspect          Slope      Horizontal_Distance_To_Hydrology
 Min.   :    1   Min.   :1863   Min.   :  0.0   Min.   : 0.0   Min.   :   0.0
 1st Qu.: 3781   1st Qu.:2376   1st Qu.: 65.0   1st Qu.:10.0   1st Qu.:  67.0
 Median : 7560   Median :2752   Median :126.0   Median :15.0   Median : 180.0
 Mean   : 7560   Mean   :2749   Mean   :156.7   Mean   :16.5   Mean   : 227.2
 3rd Qu.:11340   3rd Qu.:3104   3rd Qu.:261.0   3rd Qu.:22.0   3rd Qu.: 330.0
 Max.   :15120   Max.   :3849   Max.   :360.0   Max.   :52.0   Max.   :1343.0
 Vertical_Distance_To_Hydrology Horizontal_Distance_To_Roadways Hillshade_9am   Hillshade_Noon
 Min.   :-146.00                Min.   :   0                    Min.   :  0.0   Min.   : 99
 1st Qu.:   5.00                1st Qu.: 764                    1st Qu.:196.0   1st Qu.:207
 Median :  32.00                Median :1316                    Median :220.0   Median :223
 Mean   :  51.08                Mean   :1714                    Mean   :212.7   Mean   :219
 3rd Qu.:  79.00                3rd Qu.:2270                    3rd Qu.:235.0   3rd Qu.:235
 Max.   : 554.00                Max.   :6890                    Max.   :254.0   Max.   :254
 Hillshade_3pm   Horizontal_Distance_To_Fire_Points Wilderness_Area1 Wilderness_Area2 Wilderness_Area3
 Min.   :  0.0   Min.   :   0                       Min.   :0.0000   Min.   :0.000    Min.   :0.0000
 1st Qu.:106.0   1st Qu.: 730                       1st Qu.:0.0000   1st Qu.:0.000    1st Qu.:0.0000
 Median :138.0   Median :1256                       Median :0.0000   Median :0.000    Median :0.0000
 Mean   :135.1   Mean   :1511                       Mean   :0.2379   Mean   :0.033    Mean   :0.4199
 3rd Qu.:167.0   3rd Qu.:1988                       3rd Qu.:0.0000   3rd Qu.:0.000    3rd Qu.:1.0000
 Max.   :248.0   Max.   :6993                       Max.   :1.0000   Max.   :1.000    Max.   :1.0000
 Wilderness_Area4   Soil_Type1        Soil_Type2       Soil_Type3        Soil_Type4        Soil_Type5
 Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000
 Median :0.0000   Median :0.00000   Median :0.0000   Median :0.00000   Median :0.00000   Median :0.00000
 Mean   :0.3092   Mean   :0.02348   Mean   :0.0412   Mean   :0.06362   Mean   :0.05575   Mean   :0.01091
 3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000
 Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
   Soil_Type6        Soil_Type7   Soil_Type8         Soil_Type9         Soil_Type10      Soil_Type11
 Min.   :0.00000   Min.   :0    Min.   :0.00e+00   Min.   :0.0000000   Min.   :0.0000   Min.   :0.00000
 1st Qu.:0.00000   1st Qu.:0    1st Qu.:0.00e+00   1st Qu.:0.0000000   1st Qu.:0.0000   1st Qu.:0.00000
 Median :0.00000   Median :0    Median :0.00e+00   Median :0.0000000   Median :0.0000   Median :0.00000
 Mean   :0.04299   Mean   :0    Mean   :6.61e-05   Mean   :0.0006614   Mean   :0.1417   Mean   :0.02685
 3rd Qu.:0.00000   3rd Qu.:0    3rd Qu.:0.00e+00   3rd Qu.:0.0000000   3rd Qu.:0.0000   3rd Qu.:0.00000
 Max.   :1.00000   Max.   :0    Max.   :1.00e+00   Max.   :1.0000000   Max.   :1.0000   Max.   :1.00000
  Soil_Type12       Soil_Type13       Soil_Type14       Soil_Type15  Soil_Type16       Soil_Type17
 Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0    Min.   :0.00000   Min.   :0.00000
 1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0    1st Qu.:0.00000   1st Qu.:0.00000
 Median :0.00000   Median :0.00000   Median :0.00000   Median :0    Median :0.00000   Median :0.00000
 Mean   :0.01501   Mean   :0.03148   Mean   :0.01118   Mean   :0    Mean   :0.00754   Mean   :0.04048
 3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0    3rd Qu.:0.00000   3rd Qu.:0.00000
 Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :0    Max.   :1.00000   Max.   :1.00000
```

3. Processing the Data

The dataset was available in .csv format and is present in the working directory of R Studio.

It can be read in R, through **read.csv**

data_forest <- read.csv(file = "train.csv", header=TRUE, sep=",")

The dimensions of the dataset is simply ***dim****(data_forest)*

The names of all the columns can be displayed by ***colnames****(data_forest)*

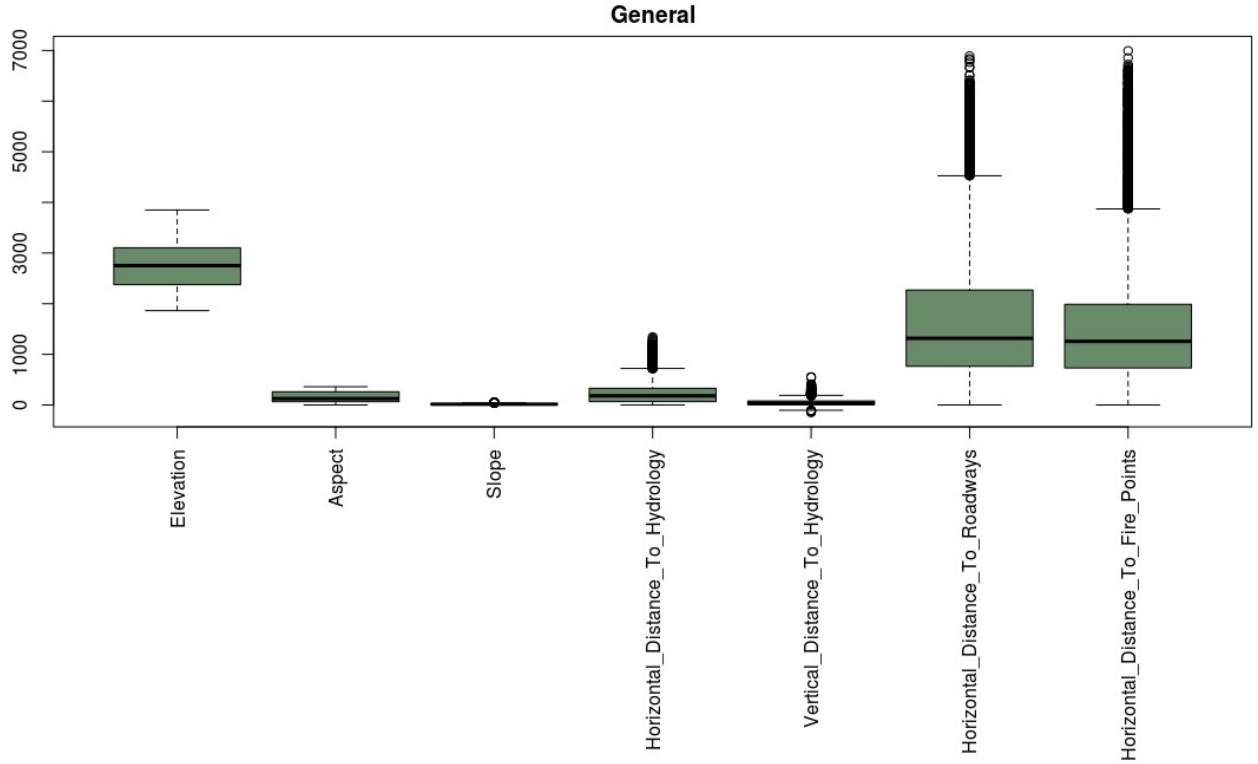
```
> colnames(data_forest)
 [1] "Id"                                  "Elevation"                           "Aspect"
 [4] "Slope"                               "Horizontal_Distance_To_Hydrology"    "Vertical_Distance_To_Hydrology"
 [7] "Horizontal_Distance_To_Roadways"     "Hillshade_9am"                       "Hillshade_Noon"
[10] "Hillshade_3pm"                       "Horizontal_Distance_To_Fire_Points"  "Wilderness_Area1"
[13] "Wilderness_Area2"                    "Wilderness_Area3"                    "Wilderness_Area4"
[16] "Soil_Type1"                          "Soil_Type2"                          "Soil_Type3"
[19] "Soil_Type4"                          "Soil_Type5"                          "Soil_Type6"
[22] "Soil_Type7"                          "Soil_Type8"                          "Soil_Type9"
[25] "Soil_Type10"                         "Soil_Type11"                         "Soil_Type12"
[28] "Soil_Type13"                         "Soil_Type14"                         "Soil_Type15"
[31] "Soil_Type16"                         "Soil_Type17"                         "Soil_Type18"
[34] "Soil_Type19"                         "Soil_Type20"                         "Soil_Type21"
[37] "Soil_Type22"                         "Soil_Type23"                         "Soil_Type24"
[40] "Soil_Type25"                         "Soil_Type26"                         "Soil_Type27"
[43] "Soil_Type28"                         "Soil_Type29"                         "Soil_Type30"
[46] "Soil_Type31"                         "Soil_Type32"                         "Soil_Type33"
[49] "Soil_Type34"                         "Soil_Type35"                         "Soil_Type36"
[52] "Soil_Type37"                         "Soil_Type38"                         "Soil_Type39"
[55] "Soil_Type40"                         "Cover_Type"
> dim(data_forest)
[1] 15120    56
```

Scale the non-categorical covariates and convert the categorical to as.factor()

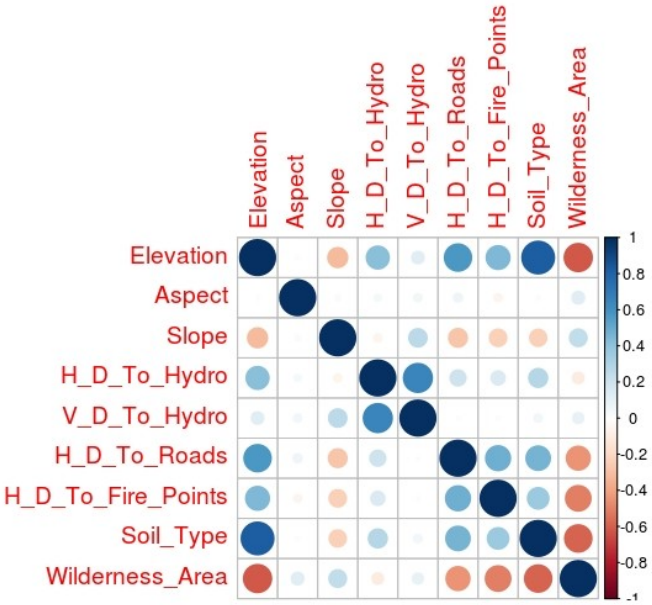
```
#Scale variables
for( i in 1:12){
  data_forest[,i] = scale(data_forest[,i])
}
for( i in 12:55){
  data_forest[,i]  =as.factor(data_forest[,i])
}
```

Some plots generated :

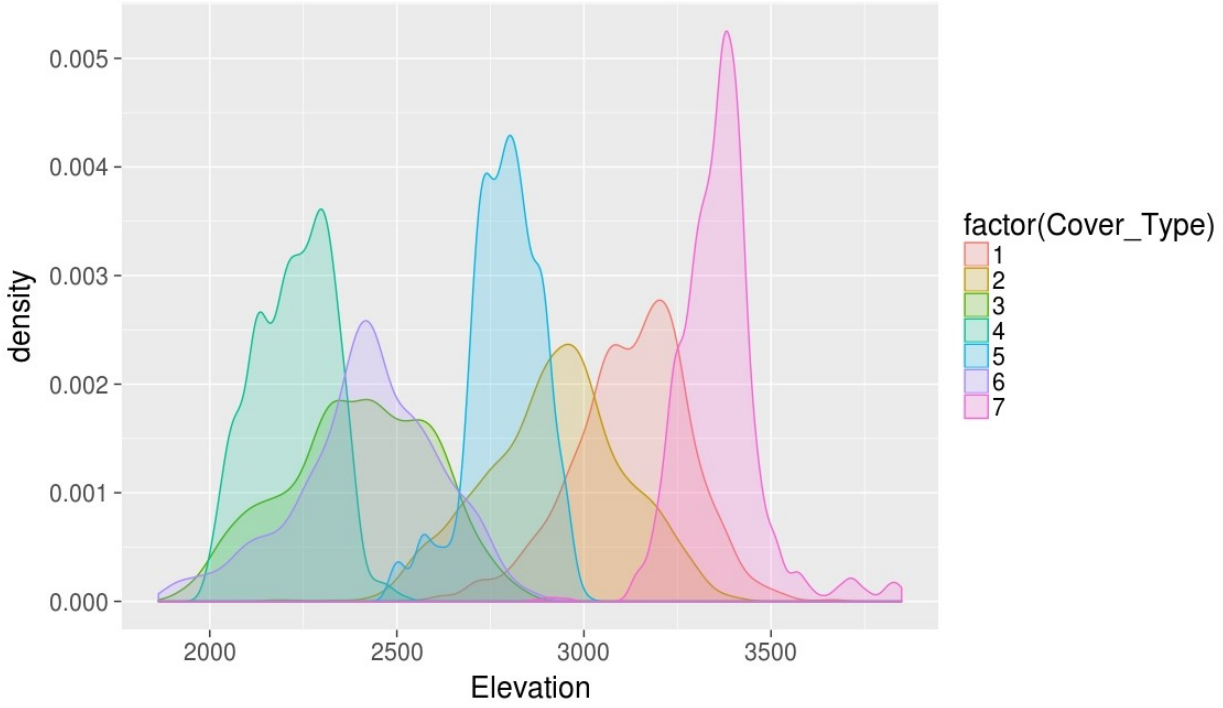
Box Plot :



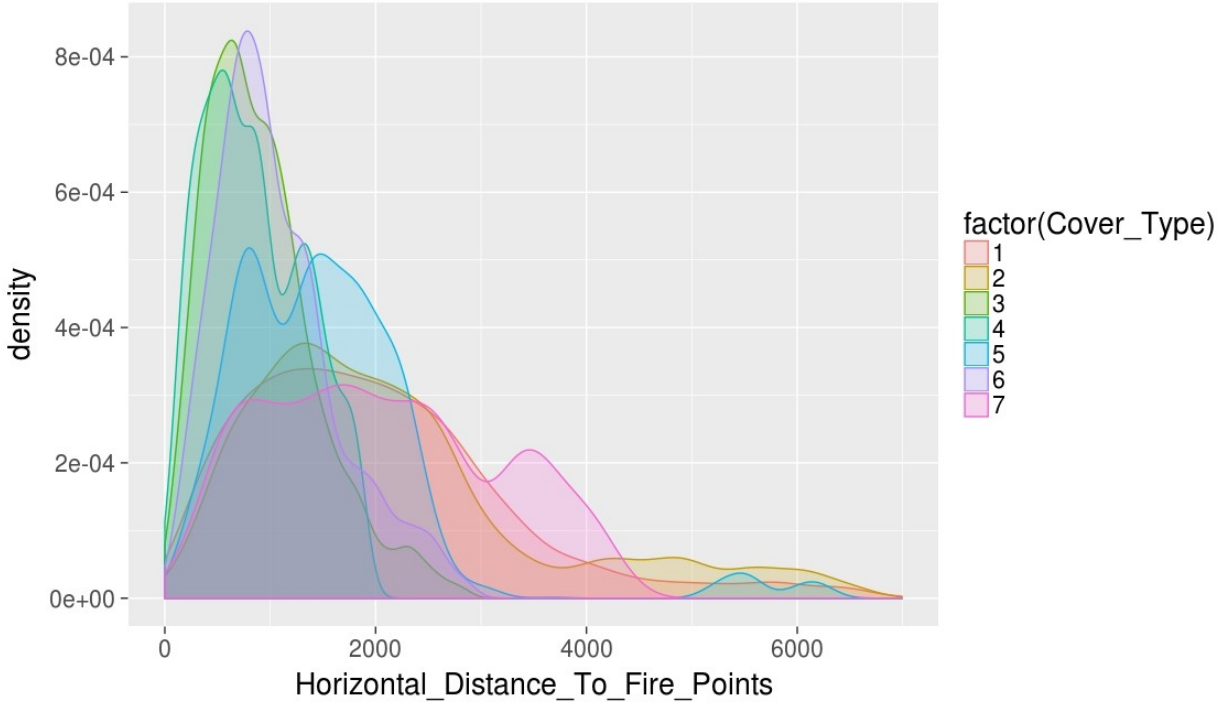
Correlation Plot:



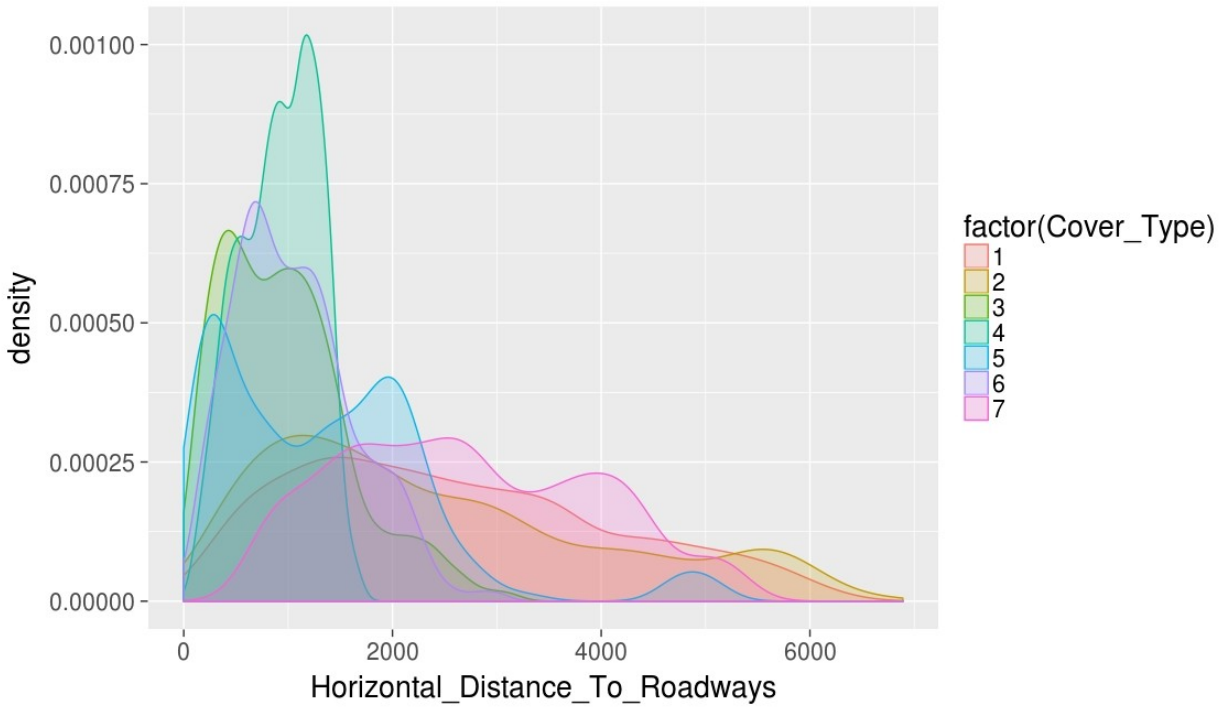
Density Plot of Elevation Covariate:



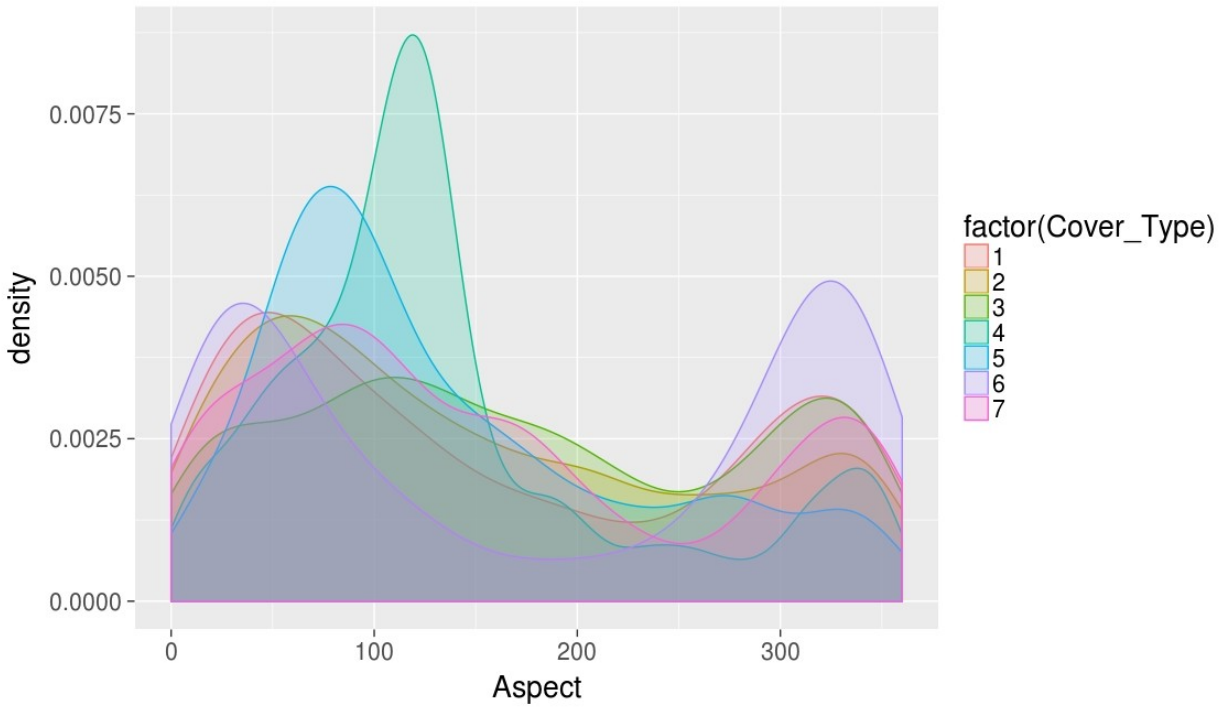
Density Plot of 2nd Covariate (Horizontal_distance):



Density Plot of 3rd Covariate :



Density Plot of 4th Covariate (Aspect) :



Script to generate these plots :

```
forest <- data_forest
forest$Id<- NULL
soil<- forest[ ,c(15:54)]
area<- forest[,c(11:14)]
forest<- forest[,c(-15:-54, -11:-14)]
Newfactor <- factor(apply(soil, 1, function(x) which(x == 1)), labels = c(1:38))
forest$Soil_Type<- as.integer(Newfactor)
Newfactor2 <- factor(apply(area, 1, function(x) which(x == 1)), labels = c(1:4))
forest$Wilderness_Area<- as.integer(Newfactor2)
forest<- forest[ ,c(1:10,12,13,11)]
head(forest)
forestTrain<-forest

boxplot(forest[,c(-7,-8,-9,-11,-12,-13)], las=3, par(mar = c(15, 4, 2, 2)),
col="darkseagreen4",main="General")
theme_set(theme_gray(base_size = 20))


g1<- ggplot(forest, aes(Elevation, color = factor(Cover_Type), fill =
factor(Cover_Type))) + geom_density(alpha = 0.2)
g2<- ggplot(forest, aes(Aspect, color = factor(Cover_Type), fill =
factor(Cover_Type))) + geom_density(alpha = 0.2)
g3<- ggplot(forest, aes(Horizontal_Distance_To_Roadways, color =
factor(Cover_Type), fill = factor(Cover_Type))) + geom_density(alpha = 0.2)
g4<- ggplot(forest, aes(Horizontal_Distance_To_Fire_Points, color =
factor(Cover_Type), fill = factor(Cover_Type))) + geom_density(alpha = 0.2)
grid.arrange(g1, g2,g3,g4, ncol=2,nrow=2)
```

4. Generating Train and Test sets

In R, one can generate Train and Test sets by either manually selecting 60 % of observations as Train set and rest 40% as Test set or by using th ***Caret*** library.

But one should prefer to use the ***Caret*** library to avoid ambiguity with the dataframe column names.

```
#create train and test sets
library(caret)

set.seed(107)
inTrain = createDataPartition(y = data_forest$Cover_Type,
                              p = .75,
                              list = FALSE)
head(inTrain)
training = data_forest[ inTrain, ]
testing = data_forest[-inTrain, ]

```

Lets look at some results.

head(inTrain)

```
> head(inTrain)
     Resample1
[1,]         1
[2,]         2
[3,]         3
[4,]         4
[5,]         5
[6,]         6
>
```

head(training)

```
> head(training)
  Id Elevation Aspect Slope Horizontal_Distance_To_Hydrology Vertical_Distance_To_Hydrology Horizontal_Distance_To_Roadways
1  1      2596     51     3                             258                             0                             510
2  2      2590     56     2                             212                            -6                             390
3  3      2804    139     9                             268                            65                            3180
4  4      2785    155    18                             242                           118                            3090
5  5      2595     45     2                             153                            -1                             391
6  6      2579    132     6                             300                           -15                              67
  Hillshade_9am Hillshade_Noon Hillshade_3pm Horizontal_Distance_To_Fire_Points Wilderness_Area1 Wilderness_Area2
1           221            232           148                              6279               1               0
2           220            235           151                              6225               1               0
3           234            238           135                              6121               1               0
4           238            238           122                              6211               1               0
5           220            234           150                              6172               1               0
6           230            237           140                              6031               1               0
  Wilderness_Area3 Wilderness_Area4 Soil_Type1 Soil_Type2 Soil_Type3 Soil_Type4 Soil_Type5 Soil_Type6 Soil_Type7 Soil_Type8
1                0                0          0          0          0          0          0          0          0          0
2                0                0          0          0          0          0          0          0          0          0
3                0                0          0          0          0          0          0          0          0          0
4                0                0          0          0          0          0          0          0          0          0
5                0                0          0          0          0          0          0          0          0          0
6                0                0          0          0          0          0          0          0          0          0
  Soil_Type9 Soil_Type10 Soil_Type11 Soil_Type12 Soil_Type13 Soil_Type14 Soil_Type15 Soil_Type16 Soil_Type17 Soil_Type18
1          0           0           0           0           0           0           0           0           0           0
2          0           0           0           0           0           0           0           0           0           0
3          0           0           0           1           0           0           0           0           0           0
4          0           0           0           0           0           0           0           0           0           0
5          0           0           0           0           0           0           0           0           0           0
6          0           0           0           0           0           0           0           0           0           0
```

Dimensions of the training and test data :

```
> dim(training)
[1] 11340    56
> dim(testing)
[1] 3780   56
>
```

The story until now....

additive_models.R | additive_models_SpAM.R | hw2.R | forest_cover_prediction.R* | train.csv

Source on Save | Run | Source

```r

library(caret)
library(randomForest)
library(ranger)
library(e1071)
library(ggplot2)
library(gridExtra)
library(corrplot)

# Forest cover prediction ------------------------------------------------

# Some plots -------------------------------------------------------------
data_forest <- read.csv(file = "train.csv", header=TRUE, sep=",")
summary(data_forest)

forest <- data_forest
forest$Id<- NULL
soil<- forest[ ,c(15:54)]
area<- forest[,c(11:14)]
forest<- forest[,c(-15:-54, -11:-14)]
Newfactor <- factor(apply(soil, 1, function(x) which(x == 1)), labels = c(1:38))
forest$Soil_Type<- as.integer(Newfactor)
Newfactor2 <- factor(apply(area, 1, function(x) which(x == 1)), labels = c(1:4))
forest$Wilderness_Area<- as.integer(Newfactor2)
forest<- forest[ ,c(1:10,12,13,11)]
head(forest)
forestTrain<-forest

png(filename="Box_plot.png")
boxplot(forest[,c(-7,-8,-9,-11,-12,-13)], las=3, par(mar = c(15, 4, 2, 2)), col="darkseagreen4",main="General")
theme_set(theme_gray(base_size = 20))

dev.off()
g1<- ggplot(forest, aes(Elevation, color = factor(Cover_Type), fill = factor(Cover_Type))) + geom_density(alpha = 0.2)
g2<- ggplot(forest, aes(Aspect, color = factor(Cover_Type), fill = factor(Cover_Type))) + geom_density(alpha = 0.2)
g3<- ggplot(forest, aes(Horizontal_Distance_To_Roadways, color = factor(Cover_Type), fill = factor(Cover_Type))) + geom_density(alpha = 0.2)
g4<- ggplot(forest, aes(Horizontal_Distance_To_Fire_Points, color = factor(Cover_Type), fill = factor(Cover_Type))) + geom_density(alpha = 0.2)
grid.arrange(g1, g2,g3,g4, ncol=2,nrow=2)

cor<- forest[,c(-9,-8,-7,-13)]
names(cor)<- c("Elevation", "Aspect","Slope","H_D_To_Hydro","V_D_To_Hydro","H_D_To_Roads", "H_D_To_Fire_Points" ,"Soil_Type","Wilderness_Area" )
#Correlation between variables
```

additive_models.R | additive_models_SpAM.R | hw2.R | forest_cover_prediction.R* | train.csv

Source on Save | Run | Source

```r
cor<- forest[,c(-9,-8,-7,-13)]
names(cor)<- c("Elevation", "Aspect","Slope","H_D_To_Hydro","V_D_To_Hydro","H_D_To_Roads", "H_D_To_Fire_Points" ,"Soil_Type","Wilderness_Area" )
#Correlation between variables
m<- cor(cor)
corrplot(m, method = "circle", tl.cex=1.2, mar = c(2, 2, 2, 2))



names = colnames(data_forest)[12:56]

#Scale variables
for( i in 1:12){
  data_forest[,i] = scale(data_forest[,i])
}
for( i in 12:55){
  data_forest[,i]  =as.factor(data_forest[,i])
}


#create train and test sets

set.seed(107)
inTrain = createDataPartition(y = data_forest$Cover_Type,
                              p = .75,
                              list = FALSE)
head(inTrain)
training = data_forest[ inTrain, ]
testing = data_forest[-inTrain, ]
```

5. Fitting the SVM Model on the dataset

In R, the SVM model is available under the ***e1071*** library.

library(e1071)

Following is the command to fit the SVM model.

svmfit=svm(as.factor(Cover_Type)~ ., data = training, kernel="radial", cost=10,scale=FALSE, cross = 10)

additive_models.R | additive_models_SpAM.R | hw2.R | forest_cover_prediction.R* | train.csv

Source on Save | Run | Source

```
# SVM ------------------------------------------------------------------

svmfit=svm(as.factor(Cover_Type)~ ., data = training, kernel="radial", cost=10,scale=FALSE, cross = 10)
yhat.rf = predict(svmfit,newdata=testing[,-56])
round( mean( yhat.rf != testing$Cover_Type )*100, 3)
confusionMatrix(yhat.rf,testing$Cover_Type)

svm_tune <- tune(svm, as.factor(Cover_Type)~ ., data = training, kernel="radial", ranges=list(cost=2^(-1:5)
                                                                                              , gamma=c(.5,1,2)))

bestmod=svm_tune$best.model
summary(bestmod)
bestmod$cost
bestmod$kernel
bestmod$degree
bestmod$gamma
```

One can perform Generalized Cross validation as described in the code by using **tune** function.

summary(svmfit)

```
> summary(svmfit)

Call:
svm(formula = as.factor(Cover_Type) ~ ., data = training, kernel = "radial", cost = 10,
    cross = 10, scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  10
      gamma:  0.01818182

Number of Support Vectors:  6548

 ( 841 1285 1188 440 1187 1120 487 )


Number of Classes:  7

Levels:
 1 2 3 4 5 6 7

10-fold cross-validation on training data:

Total Accuracy: 77.48677
Single Accuracies:
 76.54321 77.77778 77.95414 76.80776 77.60141 78.65961 77.60141 78.30688 76.89594 76.71958
```

6. Results of SVM

Predicting on the test set :

yhat.rf = predict(svmfit,newdata=testing[,-56])

Miss classification error and Confusion matrix of the test predictions :

round(mean(yhat.rf != testing$Cover_Type)*100, 3)
confusionMatrix(yhat.rf,testing$Cover_Type,mode = "prec_recall")

```
> #Miss classification error
> round( mean( yhat.rf != testing$Cover_Type )*100, 3)
[1] 23.069
> # Confusion matrix
> confusionMatrix(yhat.rf,testing$Cover_Type,mode = "prec_recall")
Confusion Matrix and Statistics

          Reference
Prediction   1   2   3   4   5   6   7
         1 394 113   0   0   4   0  43
         2  87 307   4   0  30  17   0
         3   0  12 357  14  17 103   0
         4   0   0  63 507   0  37   0
         5  17  89  13   0 482   7   1
         6   5  20 107  15  18 365   0
         7  34   2   0   0   0   0 496

Overall Statistics

               Accuracy : 0.7693
                 95% CI : (0.7555, 0.7827)
    No Information Rate : 0.1458
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7309
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6 Class: 7
Precision              0.7112  0.68989  0.70974   0.8353   0.7915  0.68868   0.9323
Recall                 0.7337  0.56538  0.65625   0.9459   0.8748  0.68998   0.9185
F1                     0.7223  0.62146  0.68195   0.8871   0.8310  0.68933   0.9254
Prevalence             0.1421  0.14365  0.14392   0.1418   0.1458  0.13995   0.1429
Detection Rate         0.1042  0.08122  0.09444   0.1341   0.1275  0.09656   0.1312
Detection Prevalence   0.1466  0.11772  0.13307   0.1606   0.1611  0.14021   0.1407
Balanced Accuracy      0.8422  0.76137  0.80557   0.9575   0.9177  0.81961   0.9537
>
```

7. Fitting RandomForests on the dataset

In R, RandomForests are available under the package name **randomForest.** However, this is a bit slow and takes up some time if the dataset is huge. So there is another implementation of randomForest which is extensively fast. It goes by the name of **ranger.**

library(ranger)

?ranger

Description

Ranger is a fast implementation of Random Forest (Breiman 2001) or recursive partitioning, particularly suited for high dimensional data. Classification, regression, and survival forests are supported. Classification and regression forests are implemented as in the original Random Forest (Breiman 2001), survival forests as in Random Survival Forests (Ishwaran et al. 2008).

The following line fits the model using ranger :

```
rang3 <- ranger(as.factor(Cover_Type) ~ .,
            write.forest=TRUE,
            data=training, num.trees = 2000, importance = 'impurity',
        classification = T)
```

summary(rang3)

```
> # Summary of Ranger object
> summary(rang3)
                         Length Class         Mode
predictions              11340  factor        numeric
num.trees                    1  -none-        numeric
num.independent.variables    1  -none-        numeric
mtry                         1  -none-        numeric
min.node.size                1  -none-        numeric
variable.importance         55  -none-        numeric
prediction.error             1  -none-        numeric
forest                      10  ranger.forest list
confusion.matrix            49  table         numeric
splitrule                    1  -none-        character
treetype                     1  -none-        character
call                         7  -none-        call
importance.mode              1  -none-        character
num.samples                  1  -none-        numeric
>
```

To perform cross-validation , **caret** package has **train()** function which can train as ranger object.

Other options are by using another function **holdoutRF()** or **csrf()** packages.

forest_cover_prediction.R

Source on Save Run Source

```r
# Something with Ranger ---------------------------------------------------
#Ranger is a fast implementation of Random Forest (Breiman 2001) or recursive partitioning,
#particularly suited for high dimensional data. Classification, regression, and survival
#forests are supported. Classification and regression forests are implemented as in the
#original Random Forest (Breiman 2001), survival forests as in Random Survival Forests (Ishwaran et al. 2008).

library(ranger)
rang3 <- ranger(as.factor(Cover_Type) ~ .,
                write.forest=TRUE,
                data=training, num.trees = 2000, importance = 'impurity', classification = T)

summary(rang3)

#Cross validation
rang3.rf.tune = csrf(
  as.factor(Cover_Type) ~ .,
  training_data = training,
  test_data = training,
  params1 = list(num.trees = 1000, mtry=4),
  params2 = list(num.trees = 500, mtry=8)
)

rang3.cv <- holdoutRF(as.factor(Cover_Type) ~ .,
                      write.forest=TRUE,
                      data=training)
#Train set predictions
yhat.tr <- rang3$predictions
#Variable importance
var_imp <- rang3$variable.importance
plot(var_imp, type = 'h')

# Predictions on Test set
yhat.rf.ranger= predict(rang3,data=testing[,-56])

#Miss classification error on Test set
round( mean( yhat.rf.ranger$predictions != testing$Cover_Type )*100, 3)

#Confusion matrix , precision, recall, F-1 score measures

confusionMatrix(yhat.rf.ranger$predictions,testing$Cover_Type, mode = "prec_recall")
```

8. Results of RandomForest

Predicting on the train set :

yhat.tr <- rang3$predictions

Predicting on test set :
yhat.rf.ranger= predict(rang3,data=testing[,-56])

Miss classification error and Confusion matrix of the test predictions :

round(mean(yhat.rf.ranger$predictions != testing$Cover_Type)*100, 3)

confusionMatrix(yhat.rf.ranger$predictions,testing$Cover_Type, mode = "prec_recall")

Train set predictions :

```
> #Train set predictions
> yhat.tr <- rang3$predictions
> print(yhat.tr)
    [1] 5 2 2 5 5 5 5 5 5 5 5 5 5 5 5 5 2 2 5 5 5 5 2 5 5 2 2 5 5 1 2 5 5 5 5 5 2 5 5 2 5 5 5 5 2 2 5 5 2 2 2 2 2 2 2 2
   [60] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2
  [119] 2 2 2 2 2 1 2 1 2 2 2 2 1 1 2 1 2 1 1 2 2 1 2 2 1 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2 2 1 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2
  [178] 2 2 2 1 1 2 2 2 1 2 2 2 2 1 1 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2
  [237] 1 2 1 1 2 2 1 1 2 2 2 2 2 1 1 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 5 5 5 2 5 5 5 5 5 5 2 2 2
  [296] 5 5 1 2 2 5 5 5 2 2 5 5 2 2 5 5 5 5 5 1 2 1 2 5 2 2 2 2 1 2 2 2 2 2 2 2 5 2 2 2 2 2 2 2 2 1 5 2 2 1 1 2 2 2 2 1 2 2
  [355] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
  [414] 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 5 2 5 1 1 5 5 1 2 5 5 5 5 5 2 2 2 2 2 5 5 2 5 5 2 5
  [473] 2 1 5 2 2 5 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 1 2 2 5 2 2 2 2 2 2 1 1 2 2 2 2 5 2 5 5 5 5 2 2 1 1 2 2 2 2 2 2
  [532] 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 1 2
  [591] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 5 2 2 5 5 2 2 5 5 2 2 2 5 5 2 2
```

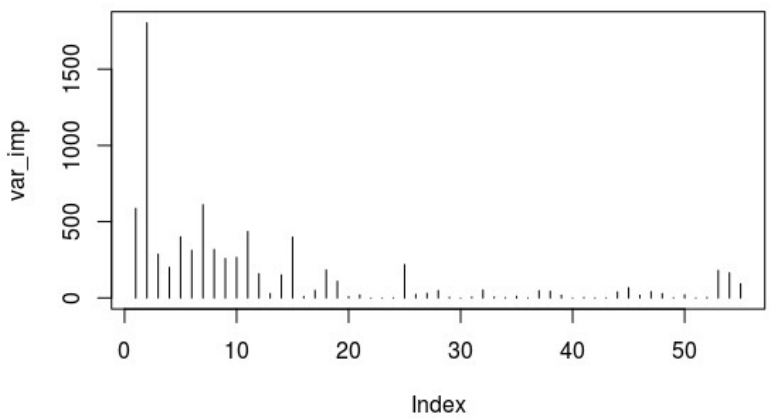
Miss classification error:

```
> # Predictions on Test set
> yhat.rf.ranger= predict(rang3,data=testing[,-56])
>
> #Miss classification error on Test set
> round( mean( yhat.rf.ranger$predictions != testing$Cover_Type )*100, 3)
[1] 16.455
>
```

Confusion Matrix:

```
> #Confusion matrix , precision, recall, F-1 score measures
>
> confusionMatrix(yhat.rf.ranger$predictions,testing$Cover_Type, mode = "prec_recall")
Confusion Matrix and Statistics

          Reference
Prediction   1   2   3   4   5   6   7
         1 414  88   0   0   1   0  22
         2  72 361   2   0  16   6   0
         3   0  15 400   9  11  58   0
         4   0   0  43 516   0  16   0
         5  14  56   8   0 510   9   1
         6   4  20  91  11  13 440   0
         7  33   3   0   0   0   0 517

Overall Statistics

               Accuracy : 0.8354
                 95% CI : (0.8232, 0.8471)
    No Information Rate : 0.1458
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.808
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6 Class: 7
Precision              0.7886   0.7899   0.8114   0.8974   0.8528   0.7599   0.9349
Recall                 0.7709   0.6648   0.7353   0.9627   0.9256   0.8318   0.9574
F1                     0.7797   0.7220   0.7715   0.9289   0.8877   0.7942   0.9460
Prevalence             0.1421   0.1437   0.1439   0.1418   0.1458   0.1399   0.1429
Detection Rate         0.1095   0.0955   0.1058   0.1365   0.1349   0.1164   0.1368
Detection Prevalence   0.1389   0.1209   0.1304   0.1521   0.1582   0.1532   0.1463
Balanced Accuracy      0.8684   0.8176   0.8533   0.9722   0.9492   0.8945   0.9731
>
```

Variable Importance :



9. Conclusion

Random Forests perform better than the SVM classifier both in terms of predictions and computational time.

Accuracy of Random Forests: 83.45 %

Accuracy of SVM: 76.93 %