Exercise 2 R Markdown

Santanu Mukherjee

06/18/2022

R Markdown

Q1

1. The UC Irvine Machine Learning Repository6 contains a data set related to glass identification. The data consist of 214 glass samples labeled as one of seven class categories. There are nine predictors, including the refractive index and percentages of eight elements: Na, Mg, Al, Si, K, Ca, Ba, and Fe. The data can be accessed via:

```
library(mlbench)
data(Glass)
str(Glass)
```

```
## 'data.frame':    214 obs. of  10 variables:
##  $ RI  : num  1.52 1.52 1.52 1.52 1.52 ...
##  $ Na  : num  13.6 13.9 13.5 13.2 13.3 ...
##  $ Mg  : num  4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...
##  $ Al  : num  1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...
##  $ Si  : num  71.8 72.7 73 72.6 73.1 ...
##  $ K   : num  0.06 0.48 0.39 0.57 0.55 0.64 0.58 0.57 0.56 0.57 ...
##  $ Ca  : num  8.75 7.83 7.78 8.22 8.07 8.07 8.17 8.24 8.3 8.4 ...
##  $ Ba  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Fe  : num  0 0 0 0 0 0.26 0 0 0 0.11 ...
##  $ Type: Factor w/ 6 levels "1","2","3","5",..: 1 1 1 1 1 1 1 1 1 1 ...
```

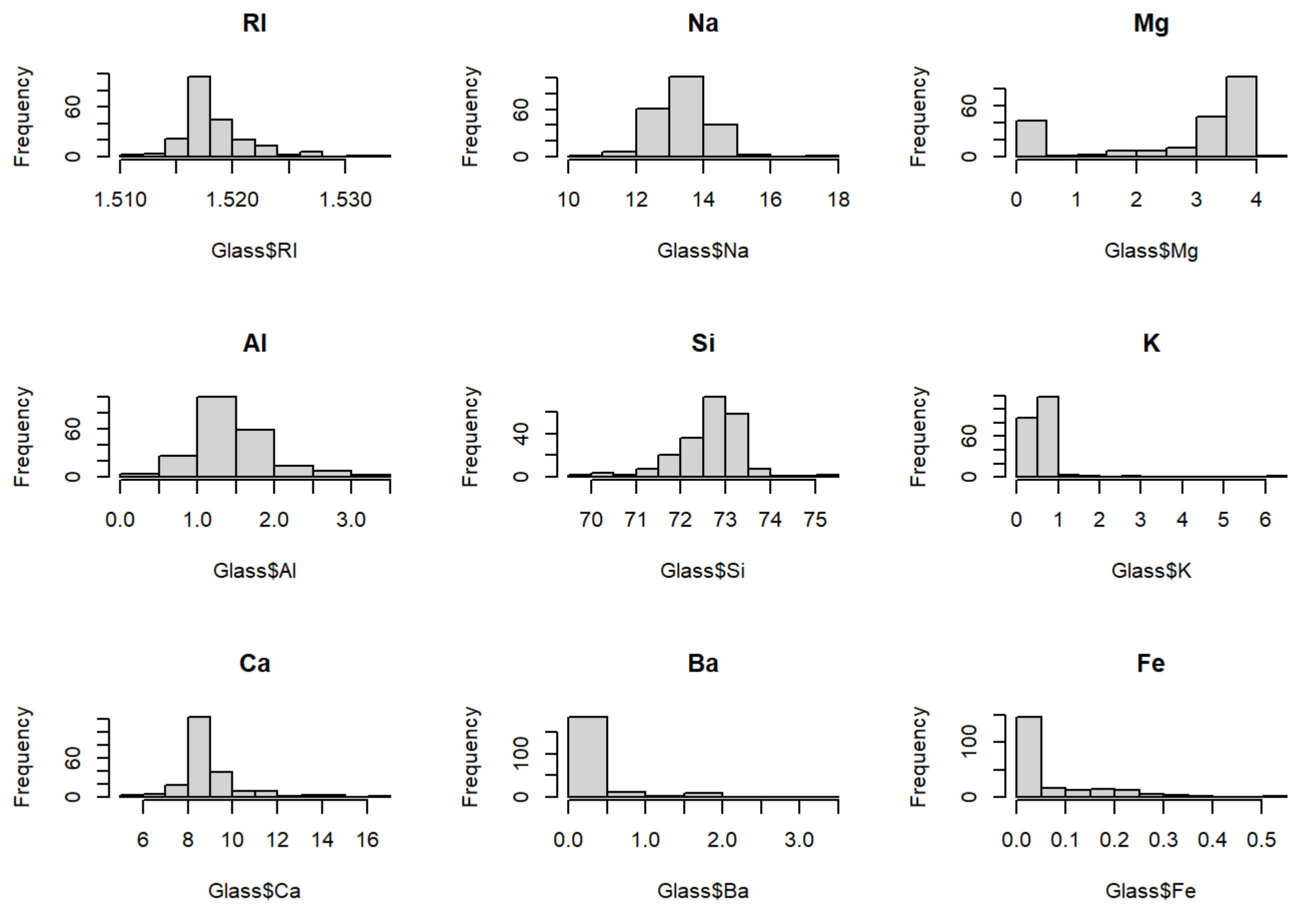
Q 1.1a

a. Using visualizations, explore the predictor variables to understand their distributions as well as the relationships between predictors.

Answer 1.1 (a)

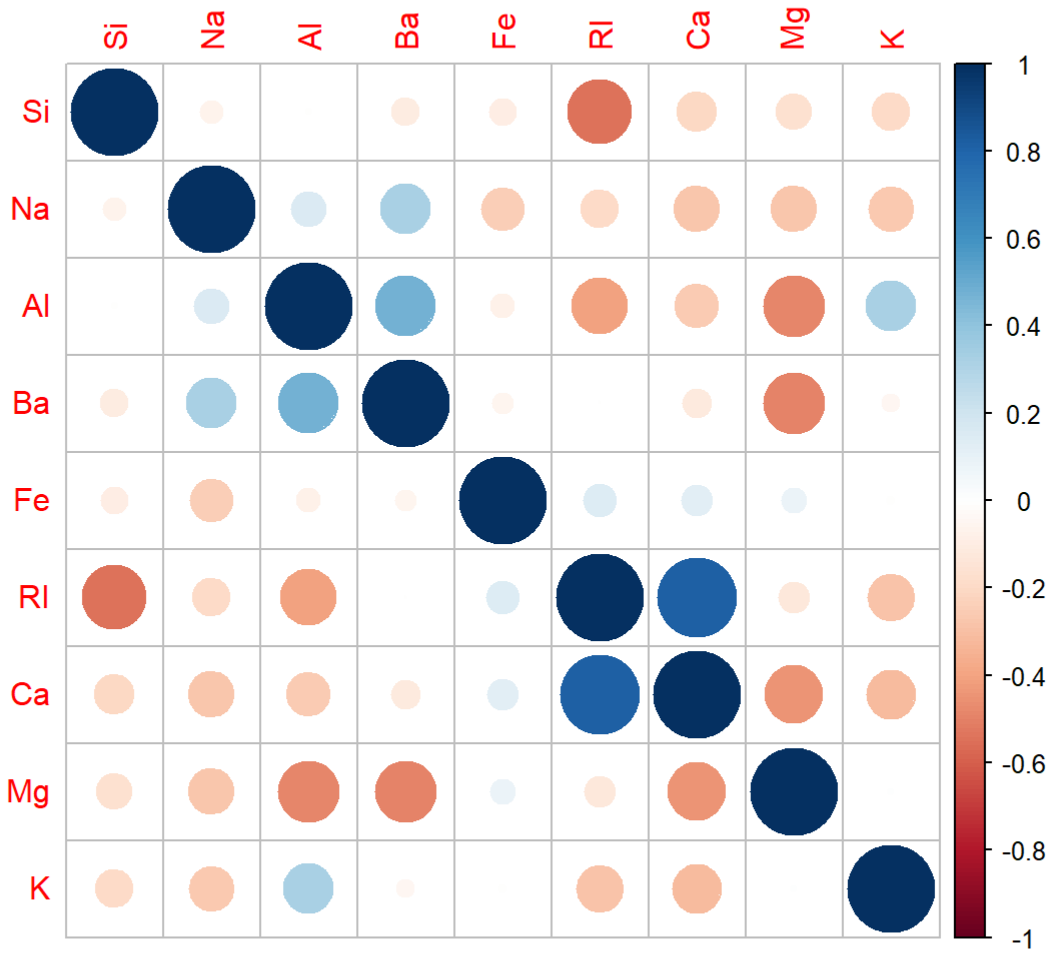
The histograms are used to visualize the distribution of each predictor and the correlation matrix plot to see the relationships between predictors.

```
par(mfrow=c(3,3))
# Histograms
hist(Glass$RI, main="RI")
hist(Glass$Na, main="Na")
hist(Glass$Mg, main="Mg")
hist(Glass$Al, main="Al")
hist(Glass$Si, main="Si")
hist(Glass$K, main="K")
hist(Glass$Ca, main="Ca")
hist(Glass$Ba, main="Ba")
hist(Glass$Fe, main="Fe")
```

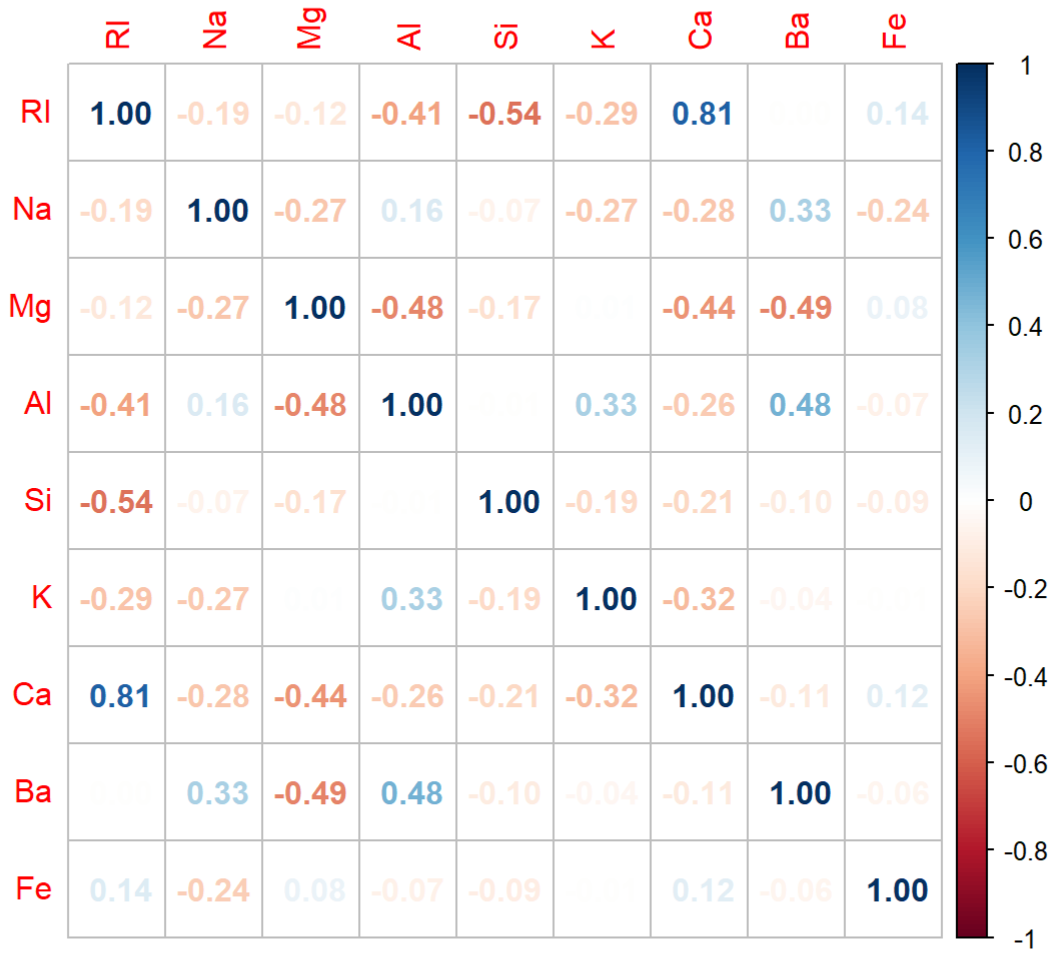


From the above histograms, we can see that some predictors are heavily skewed, such as Mg, K, Ba, and Fe. We will further confirm our visualizations about the skewness of each predictor in part (b). In addition, we use the correlation matrix to check the relationship between predictors shown below.

```
#Correlation plot
Cor = round(cor(Glass[,1:9]), 4)
library(corrplot)
corrplot(Cor, order = "hclust")
```



```
corrplot(Cor, method="number")
```



```
highCorr <- findCorrelation(Cor, cutoff = .75)
head(highCorr)
```

```
## [1] 7
```

It can be observed that the predictors Ca and RI are highly correlated with a correlation value of 0.81. By using $findCorrelation$ in R, we found out that the predictor Ca may be removed to reduce pair-wise correlation with a threshold of 0.75, as recommended by the textbook.

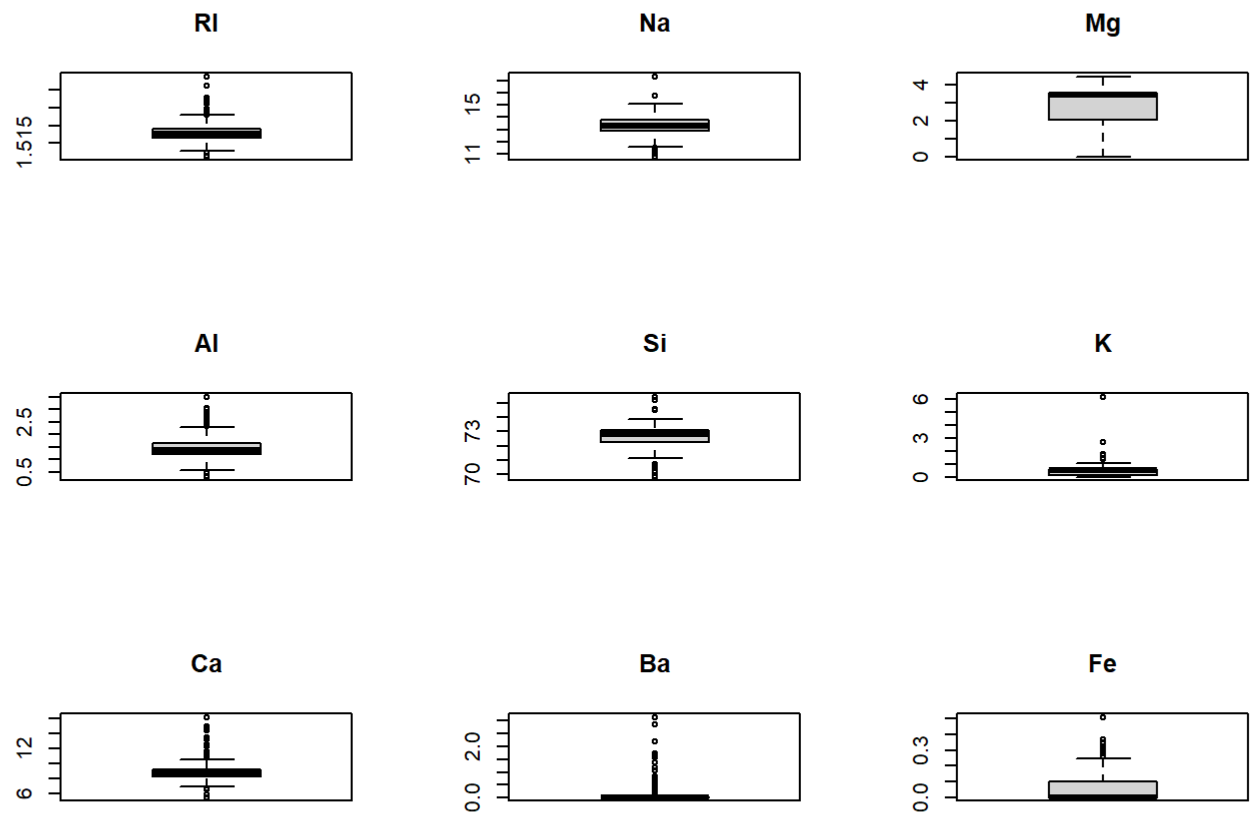
Q 1.1b

b. Do there appear to be any outliers in the data? Are any predictors skewed? Show all work !

Answer 1.1 (b)

b. By drawing the boxplot of each predictor shown below, we can draw some conclusions as follows: (i) there are several predictors having a couple of potential outliers, such as RI, Ca, Ba (ii) we also observe that several predictors are heavily skewed, such as Mg, K, Ba, and Fe, which matches the findings from the histogram. So, it should be beneficial to employ the data transformation technique to resolve the outlier issues in the predictors.

```
# Boxplot
par(mfrow=c(3,3))
boxplot(Glass$RI, main="RI")
boxplot(Glass$Na, main="Na")
boxplot(Glass$Mg, main="Mg")
boxplot(Glass$Al, main="Al")
boxplot(Glass$Si, main="Si")
boxplot(Glass$K, main="K")
boxplot(Glass$Ca, main="Ca")
boxplot(Glass$Ba, main="Ba")
boxplot(Glass$Fe, main="Fe")
```



In addition, we calculated the skewness of each predictor in the following table. The value Skewness can be used as a guide to prioritize the transformation of the predictor. We here consider the predictor to be nearly symmetric if the value falls between -0.5 and 0.5, moderately skewed if the absolute value falls between 0.5 and 1, and heavily skewed, otherwise.

Q 1.1c

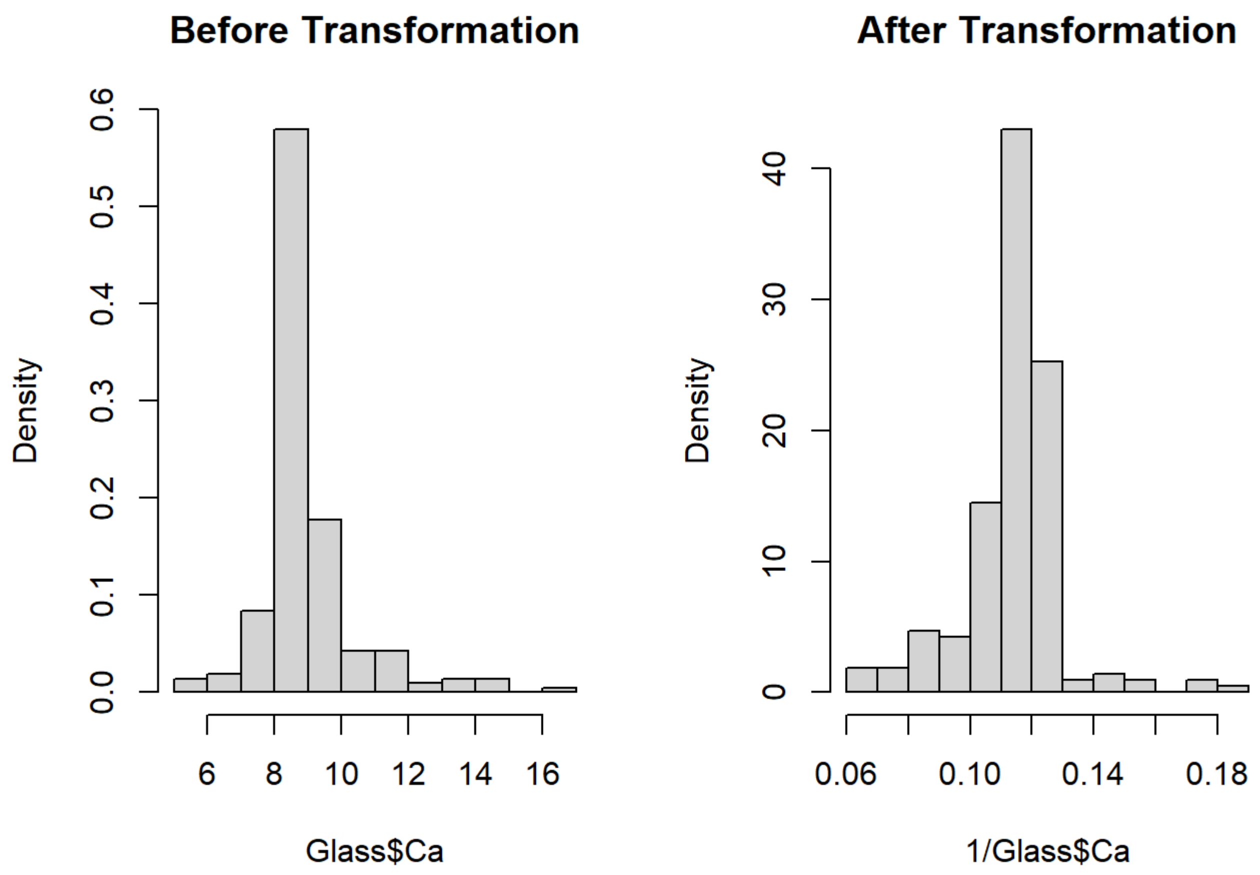
c. Are there any relevant transformations of one or more predictors that might improve the classification model? Show all work !

Answer 1.1 (c)

```
# Skewness
skewValue = apply(Glass[,1:9], 2, skewness)
skewValue
```

```
##         RI         Na         Mg         Al         Si          K         Ca
##  1.6027151  0.4478343 -1.1364523  0.8946104 -0.7202392  6.4600889  2.0184463
##         Ba         Fe
##  3.3686800  1.7298107
```

```
par(mfrow=c(1,2))
hist(Glass$Ca, prob=T,main="Before Transformation")
hist(1/Glass$Ca, prob=T,main="After Transformation")
```



We observe from this table that the data transformation techniques, such as the Box-Cox transformation, may be required to resolve skewness.

Q 1.1d

d. Fit SVM model (You may refer to Chapter 4 material for details) using the following codes:

Answer 1.1 (d)

```
# SVM

library(kernlab)
set.seed(231)
sigDist <- sigest(Type~ ., data = Glass, frac = 1)
sigDist
```

```
##        90%        50%        10%
## 0.03407935 0.11297847 0.62767315
```

```
svmTuneGrid <- data.frame(sigma = as.vector(sigDist)[1], C = 2^(-2:10))
svmTuneGrid
```

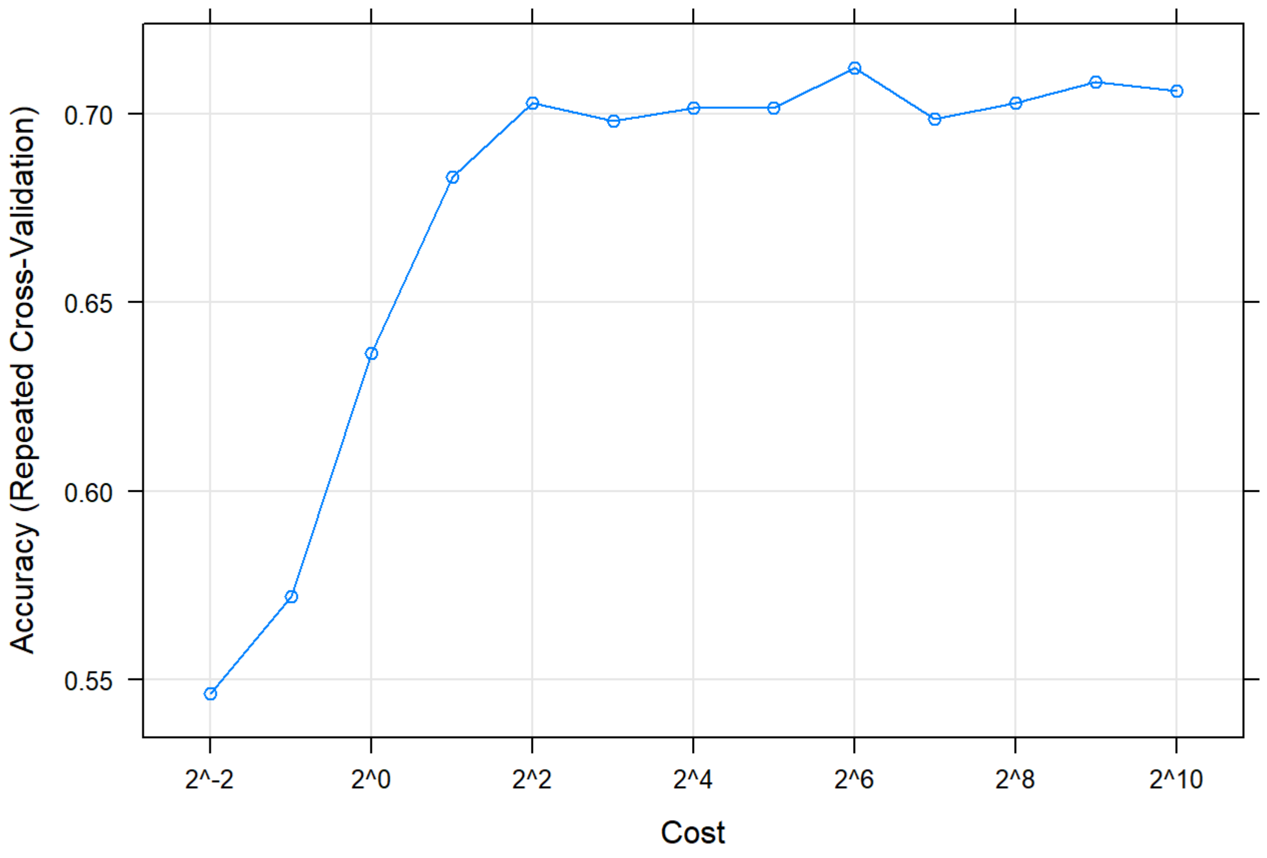
```
##         sigma       C
## 1  0.03407935    0.25
## 2  0.03407935    0.50
## 3  0.03407935    1.00
## 4  0.03407935    2.00
## 5  0.03407935    4.00
## 6  0.03407935    8.00
## 7  0.03407935   16.00
## 8  0.03407935   32.00
## 9  0.03407935   64.00
## 10 0.03407935  128.00
## 11 0.03407935  256.00
## 12 0.03407935  512.00
## 13 0.03407935 1024.00
```

```
set.seed(1056)
svmFit <- train(Type~ .,
data = Glass,
method = "svmRadial",
preProc = c("center", "scale"),
tuneGrid = svmTuneGrid,
trControl = trainControl(method = "repeatedcv",
repeats = 5))

svmFit
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 214 samples
##   9 predictor
##   6 classes: '1', '2', '3', '5', '6', '7'
##
## Pre-processing: centered (9), scaled (9)
## Resampling: Cross-Validated (10 fold, repeated 5 times)
## Summary of sample sizes: 194, 191, 194, 192, 193, 194, ...
## Resampling results across tuning parameters:
##
##   C        Accuracy   Kappa
##      0.25  0.5462532  0.3286847
##      0.50  0.5721161  0.3689913
##      1.00  0.6364625  0.4706664
##      2.00  0.6832063  0.5506030
##      4.00  0.7028410  0.5801977
##      8.00  0.6980384  0.5744221
##     16.00  0.7015910  0.5810250
##     32.00  0.7014216  0.5834227
##     64.00  0.7121792  0.6028075
##    128.00  0.6985407  0.5863168
##    256.00  0.7029157  0.5932771
##    512.00  0.7084049  0.6017276
##   1024.00  0.7059784  0.5981677
##
## Tuning parameter 'sigma' was held constant at a value of 0.03407935
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.03407935 and C = 64.
```

```
plot(svmFit, scales = list(x = list(log = 2)))
```



The above code is creating a **Radial SVM Classification** model and also perform $10-fold\,CV$ on it **repeated 5 times** and tune it.

The above diagram is a line plot of the average performance. The $scales$ argument is actually an argument to plot that converts the x-axis to $log-2$ units.