Assignment 1

Question 2.1

Describe a situation or problem from your job, everyday life, current events, etc., for which a classification model would be appropriate. List some (up to 5) predictors that you might use.

Answer: I am presently working in identifying vibration related problems in drilling a geothermal well. For this analysis drilling data is analyzed to predict if their individual or combination of the signatures of the predictor variables fall under the problematic zone. Typically, we are looking to identify three issues which can cause productivity loss (stick-slip, whirl and bit bounce). All the three issues can be classified as problems and rest as normal. The predictors used to identify these issues are as follows:

- Lateral Vibration
- Tangential Vibration
- Axial Vibration
- Revolution per minute (RPM)
- Torque
- Weight on bit (WOB)

Question 2.2

1. Using the support vector machine function ksvm contained in the R package kernlab, find a good classifier for this data. Show the equation of your classifier, and how well it classifies the data points in the full data set.

Answer: The framework below explains the logic of how the problem was perceived and tackled. Remarks are provided to understand the code. For this question no exploratory data analysis was done as the data is clean. Also, the the model was evaluated without test/validation split as would have been done otherwise.

#Loading and Visualizing the data

```
#Importing the libraries
library(kernlab)
library(ggplot2)
library(dplyr)

#Reading the text file
credit_data<-read.table('credit_card_data-headers.txt', sep="", header = TRUE)

#Viewing the data

head(credit_data)
```

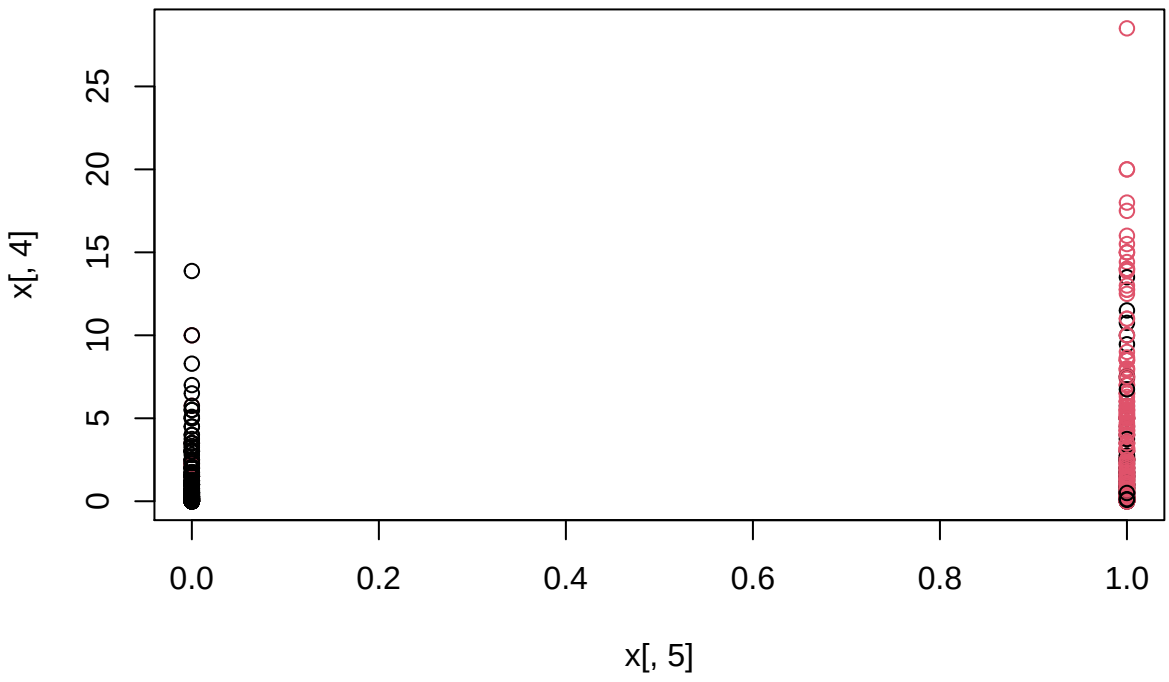
```
##   A1    A2    A3   A8 A9 A10 A11 A12 A14 A15 R1
## 1  1 30.83 0.000 1.25  1   0   1   1 202   0  1
## 2  0 58.67 4.460 3.04  1   0   6   1  43 560  1
```

```
## 3  0 24.50 0.500 1.50  1   1   0   1 280 824  1
## 4  1 27.83 1.540 3.75  1   0   5   0 100   3  1
## 5  1 20.17 5.625 1.71  1   1   0   1 120   0  1
## 6  1 32.08 4.000 2.50  1   1   0   0 360   0  1
```

```
#Summarizing the data
str(credit_data)
```

```
## 'data.frame':    654 obs. of  11 variables:
##  $ A1 : int  1 0 0 1 1 1 1 0 1 1 ...
##  $ A2 : num  30.8 58.7 24.5 27.8 20.2 ...
##  $ A3 : num  0 4.46 0.5 1.54 5.62 ...
##  $ A8 : num  1.25 3.04 1.5 3.75 1.71 ...
##  $ A9 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ A10: int  0 0 1 0 1 1 1 1 1 1 ...
##  $ A11: int  1 6 0 5 0 0 0 0 0 0 ...
##  $ A12: int  1 1 1 0 1 0 0 1 1 0 ...
##  $ A14: int  202 43 280 100 120 360 164 80 180 52 ...
##  $ A15: int  0 560 824 3 0 0 31285 1349 314 1442 ...
##  $ R1 : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
#Defining the training features (predictors variables) and target lable
#(response variable)

x<-credit_data[,1:10]
y<-credit_data[,11]

#Plotting combination of two features to find some insights
#Multiple plots were drawn with the the present plot telling that features
#4 and 5 could be most important as it can separate the data nicely.
#Just an observation.

plot(x[,5],x[,4], col=as.factor(y))
```

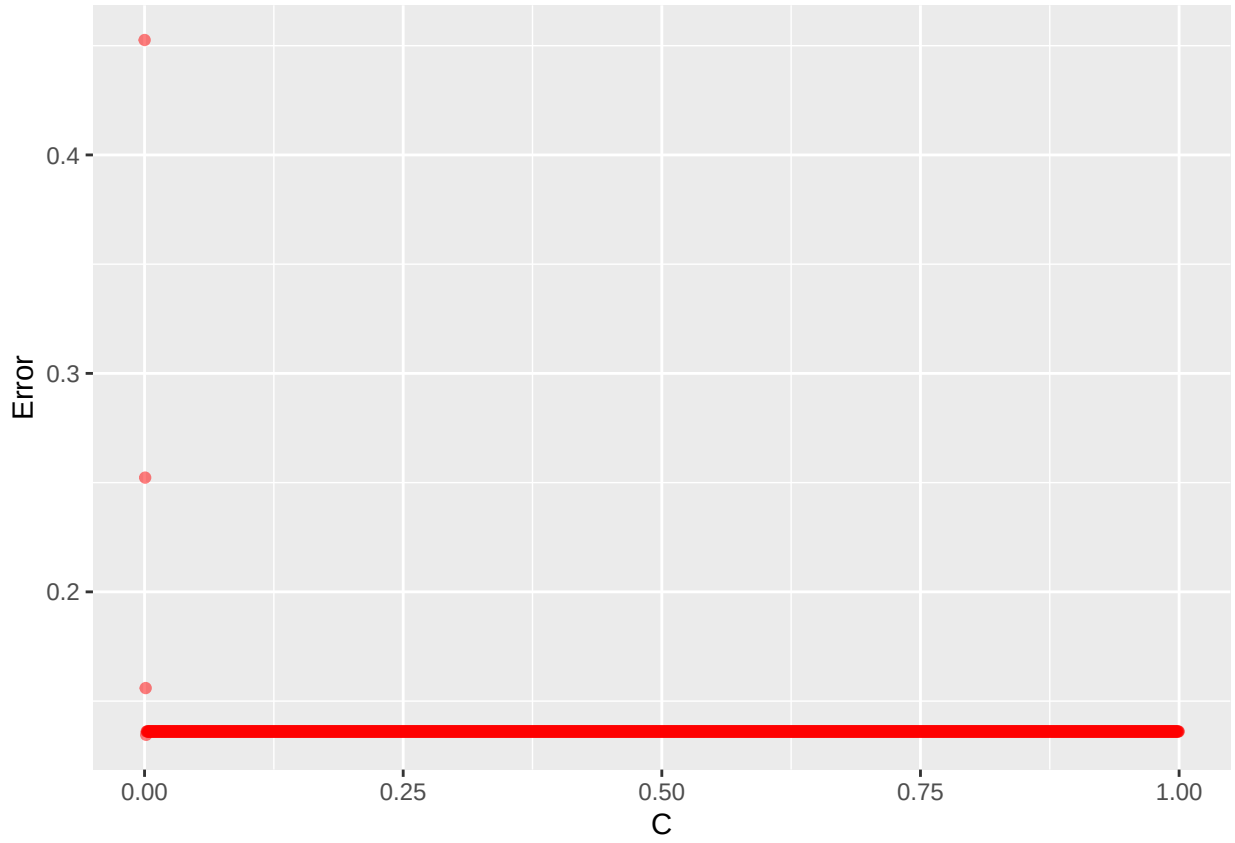


Modelling

```
#Setting seed for reproducibility

set.seed(42)

#Creating SVM Model and the model summary to do find a range for C by
#trial and run

svm_model<-ksvm(x=as.matrix(x), y=as.factor(y),type="C-svc",
                kernel="vanilladot", C=0.01, scaled=TRUE, cross=5)

#After doing several trials and runs it was decided to test the following
#range of C (.0001 to 1)

#Predictions
pred<-predict(svm_model, x[,1:10])

#Percentage of model predictions matching actual classifications
sum(pred == y)/nrow(credit_data)

#Now our task is to find the best value of C, within the decided range of
#C and then using a loop to use them we will also collect the training errors
#for each C and thus decide on the best C
```

```
err<-list() #Creating an empty error list
regu<-as.list(seq(from = 0.0001, to = 1, by = 0.0005))

#Now running the loop to calculate the model error using different C
for (i in 1:length(regu)){

  set.seed(42)
  test_model<-ksvm(x=as.matrix(x), y=as.factor(y),
                   type="C-svc",kernel="vanilladot",
                   C=regu[[i]], scaled=TRUE, cross=5)
  err[[i]]=test_model@error

}
```

```
#To plot using ggplot C and error are put in a dataframe

dat<-data.frame(C=unlist(regu), Error=unlist(err))
ggplot(data=dat,aes(x = C, y =Error)) + geom_point(alpha=0.5,color = "red")
```




```
#Sorting the dataframe based on least error and then using
#the corresponding C for the best model

dat_sort<-dat[order(dat$Error),]
```

```r
best_c<-dat_sort[1,1]
message('Best C: ',best_c)
```

```
## Best C: 0.0016
```

```r
#Model based on best C
best_model<-ksvm(x=as.matrix(x), y=as.factor(y),
                 type="C-svc",kernel="vanilladot",
                 C=best_c, scaled=TRUE, cross=5)
```

```
##  Setting default kernel parameters
```

```r
#Predictions
pred_new<-predict(best_model, x[,1:10])

#Percentage of model predictions matching actual classifications
sum(pred_new == y)/nrow(credit_data)
```

```
## [1] 0.8654434
```

```r
#Finding the coefficient and intercept
coff<-colSums(best_model@xmatrix[[1]]*best_model@coef[[1]])
intercept<-best_model@b
```

Equation of the Classifier

$$\begin{aligned} y = A1 \times -0.002061415 + A2 \times 0.011215933 + \\ A3 \times 30.023437737 + A8 \times 0.104176966 + A9 \times 0.507725910 + \\ A10 \times -0.230008684 + A11 \times 0.169276410 + \\ A12 \times -0.003537710 + A14 \times -0.019113050 + \\ A15 \times 0.102202961 + 0.0878633 \end{aligned}$$

Question 2.2.2

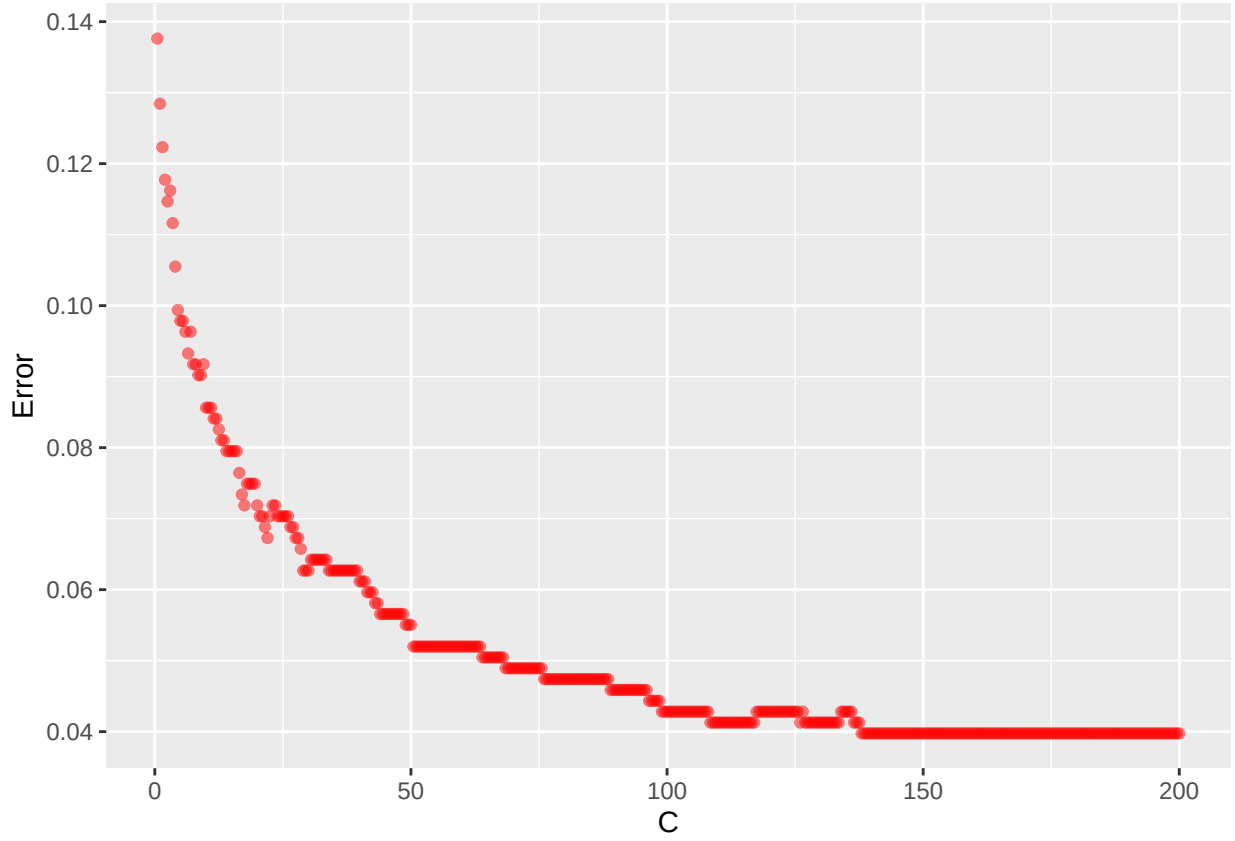
2. You are welcome, but not required, to try other (nonlinear) kernels as well; we're not covering them in this course, but they can sometimes be useful and might provide better predictions than vanilladot.

Answer: The same logic was used for this part and radial kernal was found to the best

```r
err_non<-list()
regu_non<-as.list(seq(from = 0.5, to = 200, by = 0.5))


#Now running a loop to evaluate the error and using different C
for (i in 1:length(regu_non)){

  set.seed(42)
  test_model<-ksvm(x=as.matrix(x), y=as.factor(y),
```

```
                    type="C-svc",kernel="rbfdot",
                    C=regu_non[[i]], scaled=TRUE, cross=5)
  err_non[[i]]=test_model@error

}
```

```
#
#To plot using ggplot C and error are put in a dataframe
dat_non<-data.frame(C=unlist(regu_non), Error=unlist(err_non))

#Plotting using ggplot
ggplot(data=dat_non,aes(x = C, y =Error)) + geom_point(alpha=0.5,color = "red")
```




```
#Sorting the dataframe based on least error

dat_sort_non<-dat_non[order(dat_non$Error),]

best_non_c<-dat_sort_non[1,1]
message('Best C: ',best_non_c)
```

```
## Best C: 138
```

```
#Model based on best C
best_model_non<-ksvm(x=as.matrix(x), y=as.factor(y),
```

```
                    type="C-svc",kernel="rbfdot",
                C=best_non_c, scaled=TRUE, cross=5)

#Predictions
pred_non_new<-predict(best_model_non, x[,1:10])

#Percentage of model predictions matching actual classifications
sum(pred_non_new == y)/nrow(credit_data)
```

```
## [1] 0.9571865
```

```
coff_non<-colSums(best_model@xmatrix[[1]]*best_model@coef[[1]])
intercept_non<-best_model@b
```

Equation of the Classifier

$$\begin{aligned} y = A1 \times -19.211189 + A2 \times -28.936618 + \\ A3 \times -5.782988 + A8 \times 68.184956 + A9 \times 56.828218 + \\ A10 \times -36.371359 + A11 \times 36.373165 + \\ A12 \times -30.170859 + A14 \times -55.210125 + \\ A15 \times 64.531905 + -0.5415526 \end{aligned}$$

Question 2.2.3

Using the k-nearest-neighbors classification function kknn contained in the R kknn package, suggest a good value of k, and show how well it classifies that data points in the full data set.

Answer:

```
library(kknn)

#Reading the text file
credit_data<-read.table('credit_card_data-headers.txt', sep="", header = TRUE)

#Viewing the data
head(credit_data)
```

```
##   A1    A2    A3   A8 A9 A10 A11 A12 A14 A15 R1
## 1  1 30.83 0.000 1.25  1   0   1   1 202   0  1
## 2  0 58.67 4.460 3.04  1   0   6   1  43 560  1
## 3  0 24.50 0.500 1.50  1   1   0   1 280 824  1
## 4  1 27.83 1.540 3.75  1   0   5   0 100   3  1
## 5  1 20.17 5.625 1.71  1   1   0   1 120   0  1
## 6  1 32.08 4.000 2.50  1   1   0   0 360   0  1
```

```
#Summerizing the data
str(credit_data)
```

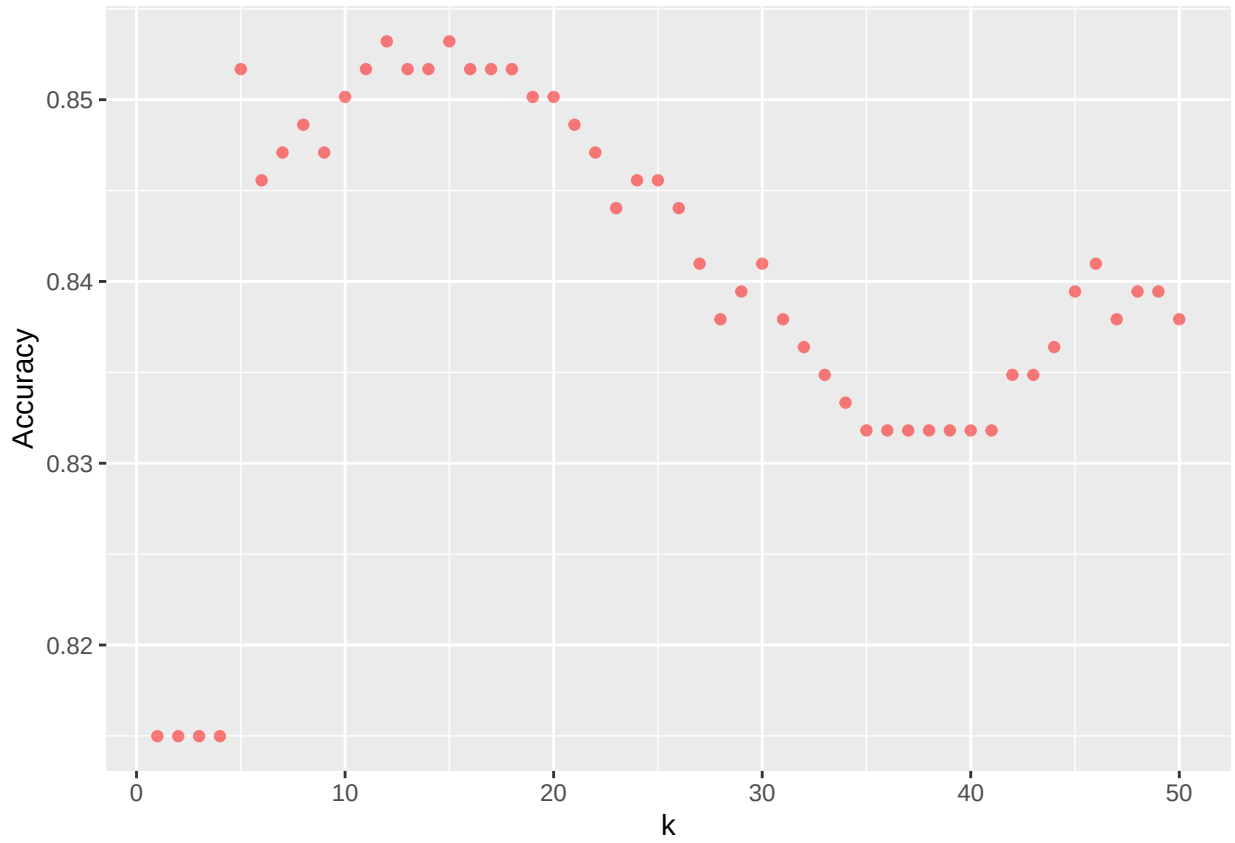
```
## 'data.frame':    654 obs. of  11 variables:
##  $ A1 : int  1 0 0 1 1 1 1 0 1 1 ...
##  $ A2 : num  30.8 58.7 24.5 27.8 20.2 ...
##  $ A3 : num  0 4.46 0.5 1.54 5.62 ...
##  $ A8 : num  1.25 3.04 1.5 3.75 1.71 ...
##  $ A9 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ A10: int  0 0 1 0 1 1 1 1 1 1 ...
##  $ A11: int  1 6 0 5 0 0 0 0 0 0 ...
##  $ A12: int  1 1 1 0 1 0 0 1 1 0 ...
##  $ A14: int  202 43 280 100 120 360 164 80 180 52 ...
##  $ A15: int  0 560 824 3 0 0 31285 1349 314 1442 ...
##  $ R1 : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
k<-as.list(seq(from = 1, to = 50, by =1))  #k is number of nearest neighbors
```

```
#Now to find the best k two loops were run. The first loop runs the different
#k values and the second loop uses the k from the first loop to run the model
#The strength of the modl is evaluated using the accuracy calculated for each
#model

pre= list()
accu<-list()

for (j in 1:length(k)){
  for (i in 1:nrow(credit_data)){
    model_knn = kknn(R1~.,
                     credit_data[-i,],
                     credit_data[i,],
                     k = j,
                     scale = TRUE)
    pred_knn = fitted(model_knn)
    pre[[i]]<-ifelse(pred_knn>0.5,1,0)
  }
  accu[[j]]=sum(pre == credit_data[,11]) / nrow(credit_data)
}
```

Finding Best k

```
#To plot using ggplot k and accuracy are put in a dataframe
dat<-data.frame(k=unlist(k), Accuracy=unlist(accu))
ggplot(data=dat,aes(x = k, y =Accuracy)) + geom_point(alpha=0.5,color = "red")
```




```
#Sorting the dataframe based on least error
dat_sort<-dat[order(-dat$Accuracy),] #Sorts the dataframe in descending order

best_k<-dat_sort[1,1]
message('Best k: ',best_k)
```

```
## Best k: 12
```

```
message('Accuracy of the model using the best k: ', dat_sort[1,2])
```

```
## Accuracy of the model using the best k: 0.853211009174312
```