Dylan Rivera
ISYE 6501
Homework #1

Question 2.1

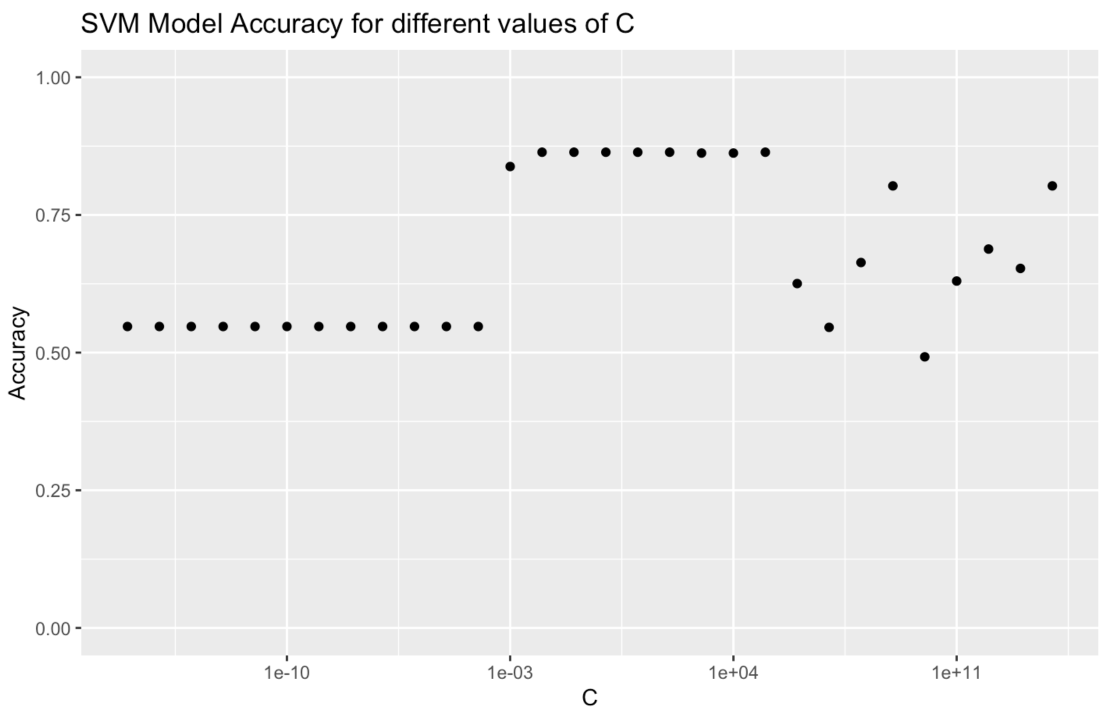
A project I've worked on at my job involved assigning inspection priorities for different metallic storage tanks and pressure vessels in a large ammonia production facility. More specifically, deciding which assets were potentially close to failure and should be inspected/replaced at the next planned facility shutdown, and which assets were deemed in good enough condition to wait until a later time.

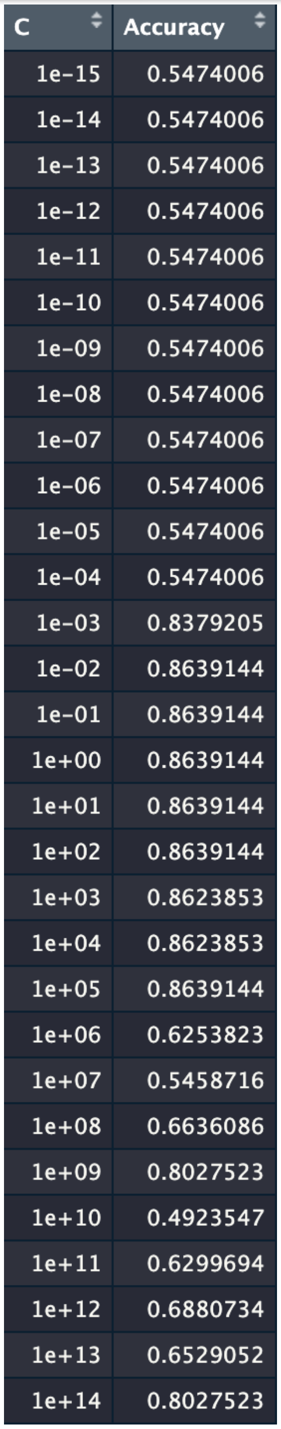
Some predictors one might use to make this decision include:

- Corrosion rate
- Material of construction
- Date of the last inspection
- The process fluid (and whether it's toxic or flammable)

Question 2.2

I have attached a plot showing the ksvm model accuracy for different values of C, ranging from 1e-15 to 1e14, based on the given credit card application dataset. Below the plot is a table showing the exact values.





C	Accuracy
1e-15	0.5474006
1e-14	0.5474006
1e-13	0.5474006
1e-12	0.5474006
1e-11	0.5474006
1e-10	0.5474006
1e-09	0.5474006
1e-08	0.5474006
1e-07	0.5474006
1e-06	0.5474006
1e-05	0.5474006
1e-04	0.5474006
1e-03	0.8379205
1e-02	0.8639144
1e-01	0.8639144
1e+00	0.8639144
1e+01	0.8639144
1e+02	0.8639144
1e+03	0.8623853
1e+04	0.8623853
1e+05	0.8639144
1e+06	0.6253823
1e+07	0.5458716
1e+08	0.6636086
1e+09	0.8027523
1e+10	0.4923547
1e+11	0.6299694
1e+12	0.6880734
1e+13	0.6529052
1e+14	0.8027523

Taking C = 1, the full equation of the classifier is:

$$\begin{gathered}\boldsymbol{-0.0011A1 - 0.0009A2 - 0.0016A3 + 0.0029A8 + 1.0047A9 - 0.003A10 - 0.0002A11} \\ \boldsymbol{- 0.0006A12 - 0.0013A14 + 0.1064A15 + 0.0815 = 0}\end{gathered}$$

The primary observation from the results is that values of C between 0.001 and 100000 all give a fairly high accuracy for the model (83-86%). One interesting observation is that at lower extreme values of C, the model accuracy stays constant, while at higher extremes, there is much more fluctuation in the accuracy as the value of C changes.

In terms of the classifier, most of the coefficients (the values of a) are close to zero. From what I understand, this means that they do not have a strong influence on the “direction” of the classifier, and can technically be excluded from the equation. In fact, the predictor A9 is by far the one with the most weight. I did not attempt any of the other kernels, but maybe a non-linear kernel may help to create a model that takes into account more of the predictors.

CODE

```
# install and load packages
install.packages("kernlab")
library(kernlab)
library(ggplot2)

path <- "~/Downloads/hw1/data 2.2/"
# load dataset
ccData <- read.csv(file.path(path,"credit_card_data-headers.txt"), sep="")

# initialize empty data frame
table = data.frame(C = numeric(), Accuracy = numeric())

# set initial value of C
C <- 1e-15

# call ksvm, loop through different values of C
while (C <= 1e15) {

  # call ksvm function
  model <- ksvm(as.matrix(ccData[,1:10]), as.factor(ccData[,11]), type="C-svc", kernel="vanilladot", C = C, scaled=TRUE)

  # calculate a1…am
  a <- colSums(model@xmatrix[[1]] * model@coef[[1]])
  a

  # calculate a0
  a0 <- -(model@b)
  a0

  # see what the model predicts
  pred <- predict(model,ccData[,1:10])
  pred

  # see what fraction of the model’s predictions match the actual classification
  accuracy <- sum(pred == ccData[,11]) / nrow(ccData)

  # add value of C and model's accuracy into table
  row <- c(C, accuracy)
  table <- rbind(table, row)
  colnames(table) <- c("C", "Accuracy")

  # iterate to next value of C
  C = C * 10
}

# plot graph of accuracy vs C value
plot <- ggplot(table, aes(C, Accuracy))
plot + geom_point() + scale_x_log10() + ylim(0,1) + labs(title = "SVM Model Accuracy for different values of C")
```

For the k-nearest neighbors portion, I tested several values of k, and, given the continuous nature of the results, implemented a rough estimate of the accuracy of the model with the assumption that values below 0.5 would be rounded down to 0 while values 0.5 or greater would be rounded up to 1.

I was not able to create the code to iterate over a large range of k, but I found that a k value of 10 gave a relatively good classification accuracy of 85%.

Something I would test further is changing the threshold of classification (for example, rounding everything below 0.7 down to 0, instead of 0.5). Many real-world data problems are often not weighted evenly between the two decisions. I also would be curious how changing the distance parameter would affect the results.

CODE

```
# install packages
install.packages("kknn")
library(kknn)

path <- "~/Downloads/hw1/data 2.2/"
# load dataset
ccData <- read.csv(file.path(path,"credit_card_data-headers.txt"), sep="")

# initialize empty results table
results <- data.frame(Prediction = numeric(), Actual = numeric())

# call kknn
for (i in 1:654){
  model_kknn <- kknn(R1~A1+A2+A3+A8+A9+A10+A11+A12+A14+A15, ccData[-i,], ccData[i,], k = 10, distance = 2, kernel = "optimal", scale
= TRUE)

  # compare prediction with actual value
  prediction <- fitted.values(model_kknn)
  actual <- ccData[i,11]

  # round down or up
  if (prediction < 0.5){
    prediction = 0
  } else {
    prediction = 1
  }

  # add results into table
  row <- c(prediction, actual)
  results <- rbind(results, row)
  colnames(results) <- c("Prediction", "Actual")

  # iterate to next row
  i = i+1
}

  # rough estimate of accuracy based on rounding up or down
  accuracy <- sum(results[,1]==results[,2]) / nrow(results)
```