ISYE6501
Bryson Cook

HW1
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.

I am currently training for a half marathon in March, but not doing a very good job of it. I would think that classification model would be good to help me predict how much I need to train to finish the half marathon under my goal. Predictors would include total number of training runs as well as the distance, pace, and elevation change of each of those runs. If I used past race finish times and their associated training runs, I could create a classification model to determine whether my probably finish time will meet my goal.

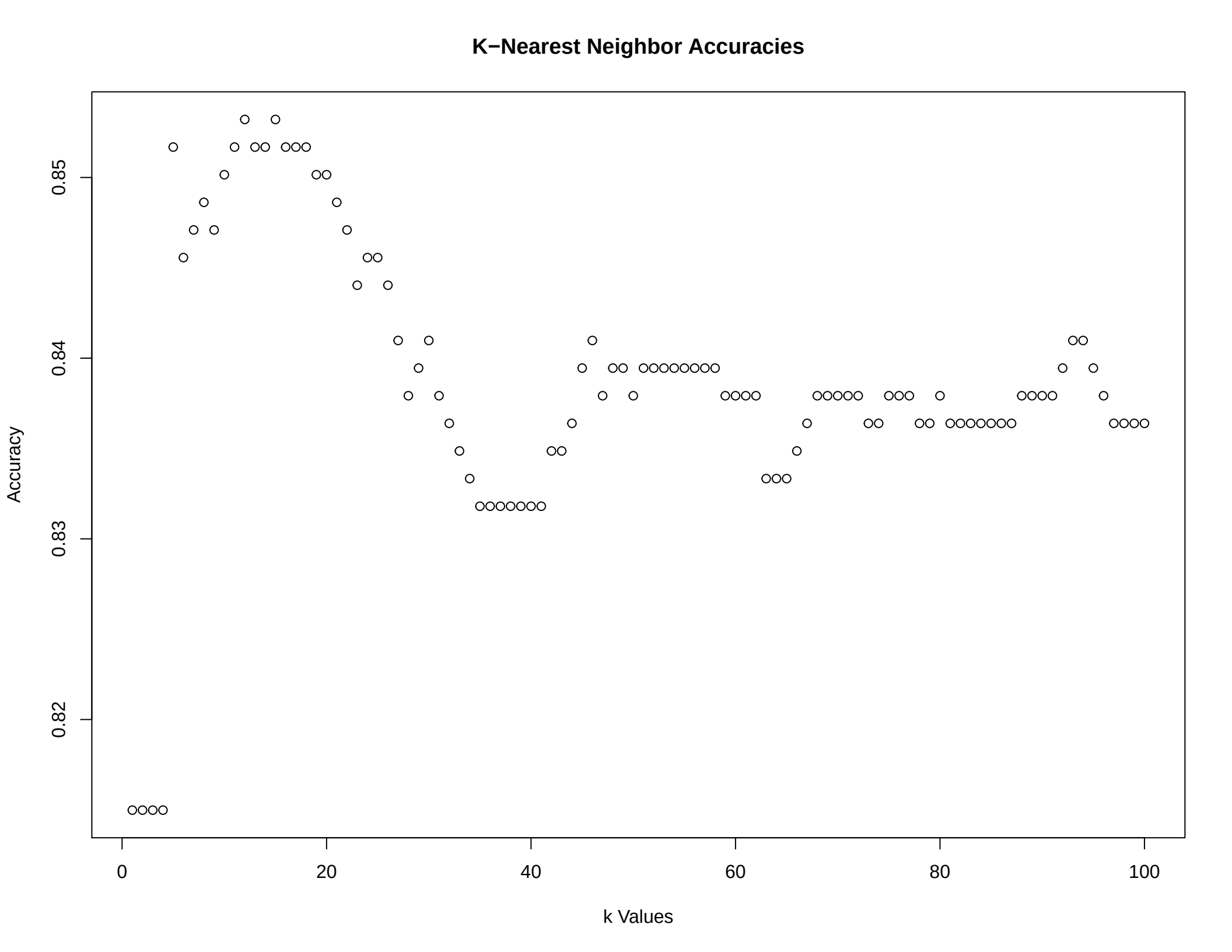
```r
#Bryson Cook
#ISYE6501, Spring 2018
#Homework 1

#Part 2.2.1a: SVM

install.packages("kknn")
install.packages("kernlab")

library(kknn)
library(kernlab)


mydata = as.matrix(read.csv("credit_card_data.csv",header=FALSE))

model = ksvm(mydata[,1:10], mydata[,11],type="C-svc",
kernel="vanilladot",C=100,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,mydata[,1:10])
pred

# see what fraction of the model's predictions match the actual classification
sum(pred == mydata[,11]) / nrow(mydata)


#Part 2.2.1b: Vary C values to view accuracy.  C varies on a logarithmic scale.

Cvals = seq(-10,10)
ans = matrix(, nrow = length(Cvals), ncol = 2)
y = 0
for (x in Cvals){
    model = ksvm(mydata[,1:10], mydata[,11],type="C-svc",
    kernel="vanilladot",C=1*10^x,scaled=TRUE)
    pred = predict(model,mydata[,1:10])
    accuracy = sum(pred == mydata[,11]) / nrow(mydata)
    y=y+1
    ans[y,1]= 1*10^x
    ans[y,2] = accuracy
    }

#Prints C values then the accuracy:
plot(log10(ans[,1]), ans[,2], main="KSVM Accuracies",
     xlab="Log(C Value)", ylab="Accuracy")

# As C increases, the max accuracy is found between 1*10^-3 and 100, and also at
100,000.  As C increases,
# we are decreasing the margin, but error should also get smaller, though only up to
# a certain point (C = 1*10^5 apparently).


################################################################################
#Part 3, K-Nearest Neighbor

mydata = data.frame(read.csv("credit_card_data.csv",header=FALSE))
steps = seq(1,100)
answery = matrix(, nrow = length(steps), ncol = 2)
z=1

```

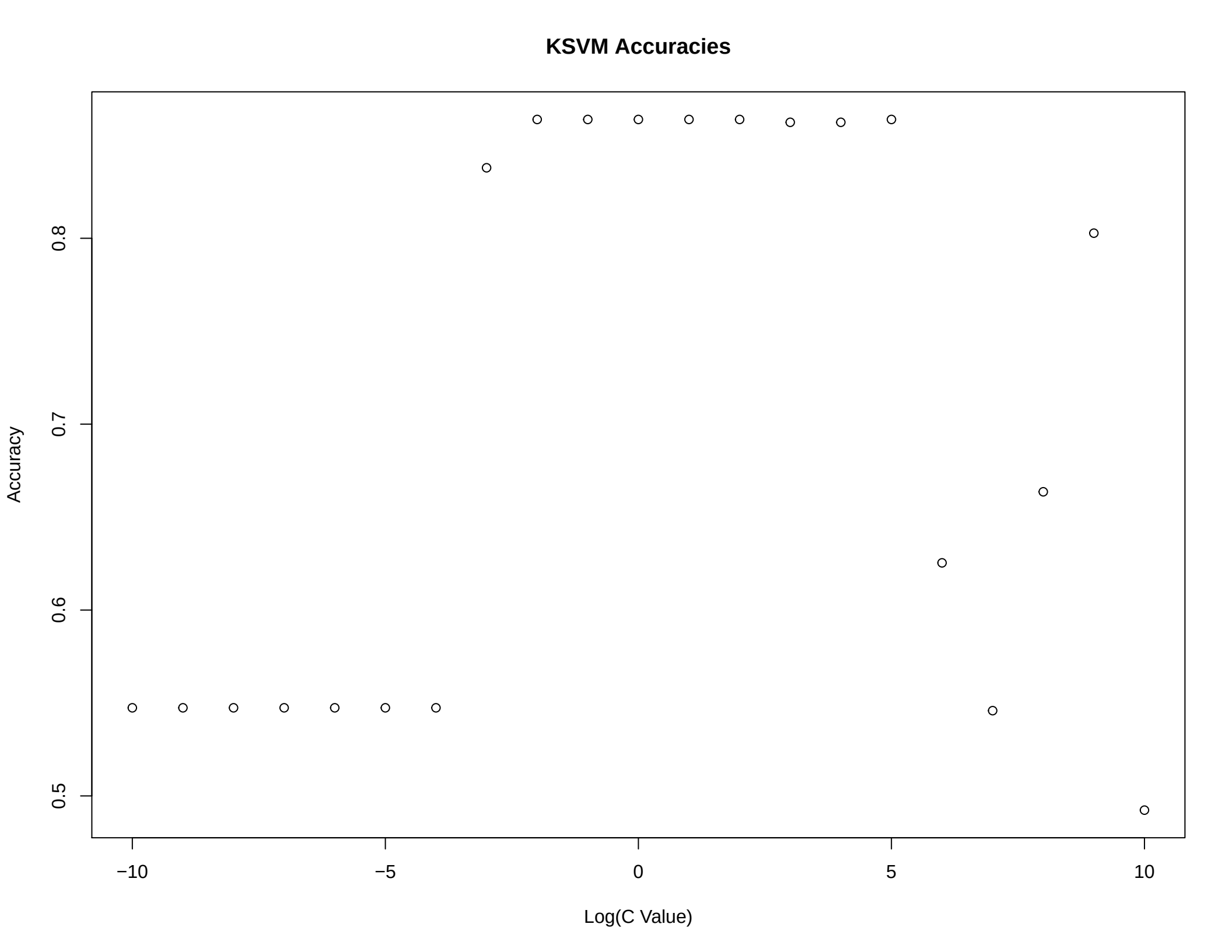
```
for (y in steps){
  answerx = matrix(, nrow(mydata), ncol = 1)
  #answerx = matrix(, nrow=100, ncol = 1)
  #for (x in 1:100){
  for (x in 1:nrow(mydata)){
    knn = kknn(V11~., mydata[-x,], mydata[x,], k = y, scale=TRUE)
    class = round(fitted(knn),0)
    match = class == mydata[x,11]
    answerx[x,1] = match
    total = sum(answerx)
  }
  answery[z,1] = y
  accuracy = total / nrow(mydata)
  answery[z,2] = accuracy
  z = z+1
}
#The max accuracy,0.8532110 occurs at a k-values of 12 and 15
plot(answery[,1], answery[,2], main="K-Nearest Neighbor Accuracies",
     xlab="k Values", ylab="Accuracy")

```

```
> #Bryson Cook
> #ISYE6501, Spring 2018
> #Homework 1
>
> #Part 2.2.1a: SVM
>
> install.packages("kknn")
Installing package into ‘C:/Users/212450481/Documents/R/win-library/3.4’
(as ‘lib’ is unspecified)
also installing the dependencies ‘irlba’, ‘pkgconfig’, ‘igraph’

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.4/irlba_2.3.2.zip'
Content type 'application/zip' length 279988 bytes (273 KB)
downloaded 273 KB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.4/pkgconfig_2.0.1.zip'
Content type 'application/zip' length 20040 bytes (19 KB)
downloaded 19 KB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.4/igraph_1.1.2.zip'
Content type 'application/zip' length 8246053 bytes (7.9 MB)
downloaded 7.9 MB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.4/kknn_1.3.1.zip'
Content type 'application/zip' length 334837 bytes (326 KB)
downloaded 326 KB

package ‘irlba’ successfully unpacked and MD5 sums checked
package ‘pkgconfig’ successfully unpacked and MD5 sums checked
package ‘igraph’ successfully unpacked and MD5 sums checked
package ‘kknn’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\212450481\AppData\Local\Temp\Rtmpq0ayYb\downloaded_packages
> install.packages("kernlab")
Installing package into ‘C:/Users/212450481/Documents/R/win-library/3.4’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.4/kernlab_0.9-25.zip'
Content type 'application/zip' length 2218659 bytes (2.1 MB)
downloaded 2.1 MB

package ‘kernlab’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\212450481\AppData\Local\Temp\Rtmpq0ayYb\downloaded_packages
>
> library(kknn)
> library(kernlab)
>
>
> mydata = as.matrix(read.csv("credit_card_data.csv",header=FALSE))
>
> model = ksvm(mydata[,1:10], mydata[,11],type="C-svc",
kernel="vanilladot",C=100,scaled=TRUE)
 Setting default kernel parameters
> # calculate a1-am
> a = colSums(model@xmatrix[[1]] * model@coef[[1]])
> a
           V1            V2            V3            V4            V5            V6
-0.0010065348 -0.0011729048 -0.0016261967  0.0030064203  1.0049405641 -0.0028259432
           V7            V8            V9           V10
 0.0002600295 -0.0005349551 -0.0012283758  0.1063633995
>
> # calculate a0
> a0 = -model@b
> a0
```

```
[1] 0.08158492
>
> # see what the model predicts
> pred = predict(model,mydata[,1:10])
> pred
  [1] 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1
 [44] 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
 1 1
 [87] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1
[130] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
1 1
[173] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1
[216] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0
0 0
[259] 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0
[302] 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0
[345] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0
[388] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0
[431] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
1 1
[474] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1
[517] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1
1 1
[560] 1 1 1 1 1 0 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0
[603] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0
[646] 0 0 0 0 0 0 0 0 0
>
> # see what fraction of the model's predictions match the actual classification
> sum(pred == mydata[,11]) / nrow(mydata)
[1] 0.8639144
>
>
> #Part 2.2.1b: Vary C values to view accuracy.  C varies on a logarithmic scale.
>
> Cvals = seq(-10,10)
> ans = matrix(, nrow = length(Cvals), ncol = 2)
> y = 0
> for (x in Cvals){
+   model = ksvm(mydata[,1:10], mydata[,11],type="C-svc",
kernel="vanilladot",C=1*10^x,scaled=TRUE)
+   pred = predict(model,mydata[,1:10])
+   accuracy = sum(pred == mydata[,11]) / nrow(mydata)
+   y=y+1
+   ans[y,1]= 1*10^x
+   ans[y,2] = accuracy
+   }
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
```

```
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
 Setting default kernel parameters
>
> #Prints C values then the accuracy:
> plot(log10(ans[,1]), ans[,2], main="KSVM Accuracies",
+      xlab="Log(C Value)", ylab="Accuracy")
>
> # As C increases, the max accuracy is found between 1*10^-3 and 100, and also at 100,000.  As C increases,
> # we are decreasing the margin, but error should also get smaller, though only up to
> # a certain point (C = 1*10^5 apparently).
>
>
> ###############################################################################
> #Part 3, K-Nearest Neighbor
>
> mydata = data.frame(read.csv("credit_card_data.csv",header=FALSE))
> steps = seq(1,100)
> answery = matrix(, nrow = length(steps), ncol = 2)
> z=1
>
> for (y in steps){
+   answerx = matrix(, nrow(mydata), ncol = 1)
+   #answerx = matrix(, nrow=100, ncol = 1)
+   #for (x in 1:100){
+   for (x in 1:nrow(mydata)){
+     knn = kknn(V11~., mydata[-x,], mydata[x,], k = y, scale=TRUE)
+     class = round(fitted(knn),0)
+     match = class == mydata[x,11]
+     answerx[x,1] = match
+     total = sum(answerx)
+   }
+   answery[z,1] = y
+   accuracy = total / nrow(mydata)
+   answery[z,2] = accuracy
+   z = z+1
+ }
> #The max accuracy,0.8532110 occurs at a k-values of 12 and 15
> plot(answery[,1], answery[,2], main="K-Nearest Neighbor Accuracies",
+      xlab="k Values", ylab="Accuracy")
```


K−Nearest Neighbor Accuracies
Accuracy
0.82
0.83
0.84
0.85
0
20
40
60
80
100
k Values


KSVM Accuracies
Accuracy
0.5
0.6
0.7
0.8
−10
−5
0
5
10
Log(C Value)