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 2.1

At my work, we analyzed various predictors that can affect the manufacturing yields. Whether the manufactured products meet the design specification will be the response (Pass/NG in the factor of 0 and 1). Predictors including but not limited to the following:
- Material vendors
- Material molding ID
- Material date code
- Assembly machine ID
- Temperature
- Humidity

Question 2.2

The files `credit_card_data.txt` (without headers) and `credit_card_data-headers.txt` (with headers) contain a dataset with 654 data points, 6 continuous and 4 binary predictor variables.

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.
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`.
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. Don't forget to scale the data (`scale=TRUE` in `kknn`).

Answer 2.2.1

```
# Clear global environment
> rm(list = ls())

# Install and load package "kernlab"
> install.packages("kernlab")
> library(kernlab)

# Load and preview dataset
> df1 <- read.table("/Users/…/credit_card_data-headers.txt", stringsAsFactors = F, header = T)
> head(df1)
  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
```

Using ksvm to fit the model. List model outputs. Result suggests model consists of approximately 13.6% training error.

```
> model1 <- ksvm(as.matrix(df1[,1:10]), as.factor(df1[,11]), type = "C-svc", kernel = "vanilladot",
C = 100, scaled = T)
 Setting default kernel parameters
> model1
Support Vector Machine object of class "ksvm"

SV type: C-svc  (classification)
 parameter : cost C = 100

Linear (vanilla) kernel function.

Number of Support Vectors : 189

Objective Function Value : -17887.92
Training error : 0.136086
```

Calculate and preview a_1 to a_m

```
> a <- colSums(model1@xmatrix[[1]] * model1@coef[[1]])
> a
         A1          A2          A3          A8          A9
-0.0010065348 -0.0011729048 -0.0016261967  0.0030064203  1.0049405641
        A10         A11         A12         A14         A15
-0.0028259432  0.0002600295 -0.0005349551 -0.0012283758  0.1063633995
```

Calculate and preview a_0

```
> a0 <- model1@b
> head(a0)
[1] -0.08158492
```

Store model's prediction into variable pred

```
> pred <- predict(model1, df1[,1:10])
```

See what fraction of the model's predictions match the classification. Results suggest model:

$$-0.001007a_1 - 0.001173a_2 - 0.001626a_3 + 0.003006a_8 + 1.004941a_9 - 0.002826a_{10} + 0.000260a_{11} - 0.000535a_{12} - 0.001228a_{14} + 0.106363a_{15} - 0.08158492 = 0$$

predicts approximately 86.4% of the actual classification. Extra notes: After trying ksvm models with C set to 10 and 1000, the predicted result for 10 is the same and the result for 1000 predicts about 86.2% of the classification.

```
> sum(pred == df1[,11]) / nrow(df1)
[1] 0.8639144
```

Answer 2.2.3

```
# Clear global environment
> rm(list = ls())

# Install and load package “kknn”
> install.packages(“kknn”)
> library(kknn)

# Load dataset
> df1 <- read.table("/Users/…/data 2.2/credit_card_data-headers.txt", stringsAsFactors = F, header = T)

## Set variable x to be zero vector
> x <- rep(0,nrow(df1))

# Using kknn to build a function with scaled dataset and target value k set to k. Afterwards, calculate fitted model predicted results.
> val1 <- function(k){
+     for (i in 1:nrow(df1)){
+         model1 <- kknn(R1~., df1[-i,], df1[i,], k=k, scale = T)
+         x[i] <- round(fitted(model1))
+     }
+     sum(x == df1[,11]) / nrow(df1)
+ }

# Set val2 as zero vector first, then trying different k values to validate model predictions. After trying k values ranging from 1 to 30, there is a clear trend that model has highest level of prediction when k is at 12 and 15 which predicts 85.32% of the data points, afterwards the accuracy starts to go downhill, so no point to further increase the range of k.
> val2 <- rep(0,30)
> for (k in 1:30){
+     val2[k] = val1(k)
+ }
> val2
[1] 0.8149847 0.8149847 0.8149847 0.8149847 0.8516820 0.8455657 0.8470948
[8] 0.8486239 0.8470948 0.8501529 0.8516820 0.8532110 0.8516820 0.8516820
[15] 0.8532110 0.8516820 0.8516820 0.8516820 0.8501529 0.8501529 0.8486239
```

[22] 0.8470948 0.8440367 0.8455657 0.8455657 0.8440367 0.8409786 0.8379205
[29] 0.8394495 0.8409786