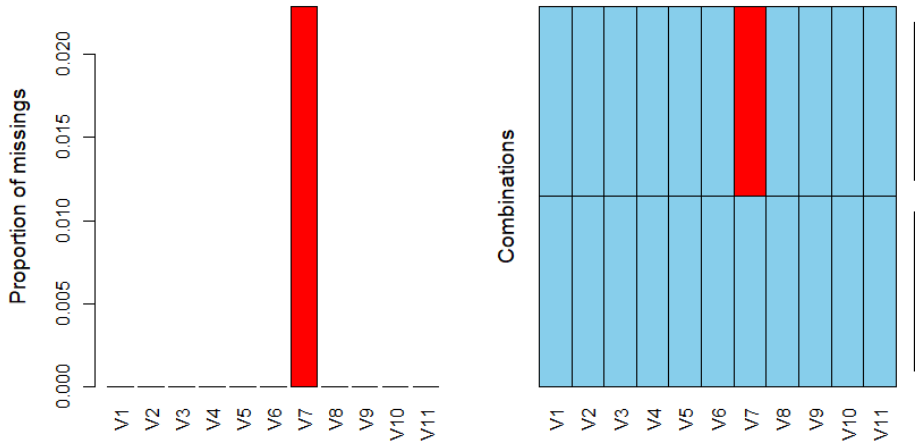
Homework 6

Question 14.1 The breast cancer data set breast-cancer-wisconsin.data.txt from http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/ (description at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+1. Use the mean/mode imputation method to impute values for the missing data.
2. Use regression to impute values for the missing data.
3. Use regression with perturbation to impute values for the missing data.
4. (Optional) Compare the results and quality of classification models (e.g., SVM, KNN) build using (1) the data sets from questions 1,2,3; (2) the data that remains after data points with missing values are removed; and (3) the data set when a binary variable is introduced to indicate missing values.

Answer: First, lets load the data and see where and how many missing values there are in the dataset.

```
data <- read.csv("http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data", header=FALSE, na.strings="?")
library(VIM)
aggr(data)
```



As we can see above in the picture, there are 16 missing values in the "V7" column. For the mean and mode imputation, we'll just take the mean and mode of column 7, and put them or impute them in the missing value spot. For mean, we get "3.544656", and for mode we get "1". For regression imputation, we first need to know what are some good predictor variables for V7.

```
Call:
lm(formula = V7 ~ ., data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-7.5771 -0.4427 -0.2088  0.8940  8.6145

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.039e+00  3.487e-01 -11.582  < 2e-16 ***
V1          -1.656e-07  1.240e-07  -1.335  0.18223
V2           1.825e-02  3.960e-02   0.461  0.64499
V3          -1.594e-01  6.731e-02  -2.369  0.01813 *
V4           1.863e-01  6.548e-02   2.844  0.00459 **
V5           2.194e-01  4.124e-02   5.320 1.42e-07 ***
V6           1.872e-02  5.520e-02   0.339  0.73457
V8           1.505e-01  5.327e-02   2.825  0.00487 **
V9          -8.724e-02  3.967e-02  -2.199  0.02821 *
V10         -6.365e-02  5.215e-02  -1.220  0.22272
V11          2.495e+00  1.784e-01  13.990  < 2e-16 ***
```

As we can see, that V5, and V11 are good predictors for the dependent variable V7.Let's see:

```
Call:
lm(formula = V7 ~ V5 + V11, data = data)

Residuals:
   Min     1Q Median     3Q    Max
-7.631 -0.268 -0.268  1.232  8.732

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.27630    0.25722 -16.625  < 2e-16 ***
V5           0.22739    0.03819   5.954 4.19e-09 ***
V11          2.65845    0.11460  23.198  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.022 on 680 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.6928,    Adjusted R-squared:  0.6919
F-statistic: 766.9 on 2 and 680 DF,  p-value: < 2.2e-16
```

So now we have a linear formula that best predicts V7, which looks like this V7=-4.27630+0.22739(V5)+2.65845(V11). Next we'll impute the predict points into our missing values (NA) in V7.

```
Ind<- function(t)
{
  x<-dim(length(t))
  x[which(!is.na(t))]=1
  x[which(is.na(t))]=0
  return(x)
}

data$I<-Ind(data$V7)

for (i in 1:nrow(data)) {
  if(data$I[i]==0)
  {
    data$V7[i]= -4.27630 + 2.65845*data$V11[i]+ 0.22739*data$V5[i]
  }

}
```

12	1036172	2	1	1	1	2	1.00000	2	1	1	2	1
13	1041801	5	3	3	3	2	3.00000	4	4	1	4	1
14	1043999	1	1	1	1	2	3.00000	3	1	1	2	1
15	1044572	8	7	5	10	7	9.00000	5	5	4	4	1
16	1047630	7	4	6	4	6	1.00000	4	3	1	4	1
17	1048672	4	1	1	1	2	1.00000	2	1	1	2	1
18	1049815	4	1	1	1	2	1.00000	3	1	1	2	1
19	1050670	10	7	7	6	4	10.00000	4	1	2	4	1
20	1050718	6	1	1	1	2	1.00000	3	1	1	2	1
21	1054590	7	3	2	10	5	10.00000	5	4	4	4	1
22	1054593	10	5	5	3	6	7.00000	7	10	1	4	1
23	1056784	3	1	1	1	2	1.00000	2	1	1	2	1
24	1057013	8	4	5	1	2	6.58489	7	3	1	4	0
25	1059552	1	1	1	1	2	1.00000	3	1	1	2	1
26	1065726	5	2	3	4	2	7.00000	3	6	1	4	1
27	1066373	3	2	1	1	1	1.00000	2	1	1	2	1
28	1066979	5	1	1	1	2	1.00000	2	1	1	2	1
29	1067444	2	1	1	1	2	1.00000	2	1	1	2	1
30	1070935	1	1	3	1	2	1.00000	1	1	1	2	1
31	1070935	3	1	1	1	1	1.00000	2	1	1	2	1
32	1071760	2	1	1	1	2	1.00000	3	1	1	2	1
33	1072179	10	7	7	3	8	5.00000	7	4	3	4	1
34	1074610	2	1	1	2	2	1.00000	3	1	1	2	1
35	1075123	3	1	2	1	2	1.00000	2	1	1	2	1
36	1079304	2	1	1	1	2	1.00000	2	1	1	2	1
37	1080185	10	10	10	8	6	1.00000	8	9	1	4	1
38	1081791	6	2	1	1	1	1.00000	7	1	1	2	1
39	1084584	5	4	4	9	2	10.00000	5	6	1	4	1
40	1091262	2	5	3	3	6	7.00000	7	5	1	4	1
41	1096800	6	6	6	9	6	3.08711	7	8	1	2	0
42	1099510	10	4	3	1	3	3.00000	6	5	2	4	1
43	1100524	6	10	10	2	8	10.00000	7	3	3	4	1
44	1102573	5	6	5	6	10	1.00000	3	1	1	4	1
45	1103608	10	10	10	4	8	1.00000	8	10	1	4	1
46	1103722	1	1	1	1	2	1.00000	2	1	2	2	1
47	1105257	3	7	7	4	4	9.00000	4	8	1	4	1
48	1105524	1	1	1	1	2	1.00000	2	1	1	2	1

Question 15.1 Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?

Answer: A company is interested in boxing the maximum amount of their

good into a container before shipping it. The constraint would be the size or the dimension of such container. And what we are maximizing is the amount of good in that container without "breaking" or deforming any of the object in the container. This is certainly an optimization problem one can in encounter in the real world.