ISYE 6501 HW Wk6 Solution

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

# Set seed
set.seed(25)

# library
# Cross Validation
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2

## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
library(MASS)
library(glmnet)
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-18
```

```
library(kknn)
```

```
##
## Attaching package: 'kknn'

## The following object is masked from 'package:caret':
##
##     contr.dummy
```

Question 14.1

The breast cancer data set breast-cancer-wisconsin.data.txt has missing values. 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.

```r
# load txt data to dataframe
bcw <- read.table("breast-cancer-wisconsin.data.txt", stringsAsFactors = FALSE, header = FALSE, sep = "

# Explore data
head(bcw)
```

```
##        V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11
## 1 1000025  5  1  1  1  2  1  3  1   1   2
## 2 1002945  5  4  4  5  7 10  3  2   1   2
## 3 1015425  3  1  1  1  2  2  3  1   1   2
## 4 1016277  6  8  8  1  3  4  3  7   1   2
## 5 1017023  4  1  1  3  2  1  3  1   1   2
## 6 1017122  8 10 10  8  7 10  9  7   1   4
```

```r
str(bcw)
```

```
## 'data.frame':    699 obs. of  11 variables:
##  $ V1 : int  1000025 1002945 1015425 1016277 1017023 1017122 1018099 1018561 1033078 1033078 ...
##  $ V2 : int  5 5 3 6 4 8 1 2 2 4 ...
##  $ V3 : int  1 4 1 8 1 10 1 1 1 2 ...
##  $ V4 : int  1 4 1 8 1 10 1 2 1 1 ...
##  $ V5 : int  1 5 1 1 3 8 1 1 1 1 ...
##  $ V6 : int  2 7 2 3 2 7 2 2 2 2 ...
##  $ V7 : chr  "1" "10" "2" "4" ...
##  $ V8 : int  3 3 3 3 3 9 3 3 1 2 ...
##  $ V9 : int  1 2 1 7 1 7 1 1 1 1 ...
##  $ V10: int  1 1 1 1 1 1 1 1 5 1 ...
##  $ V11: int  2 2 2 2 2 4 2 2 2 2 ...
```

```r
# show the column with missing data
# Show the value distribution of each variable

for (i in seq(2, 11)){
  x = paste("V",i, sep="")
  print(x)
  print(table(bcw[,i]))
}
```

```
## [1] "V2"
##
##   1   2   3   4   5   6   7   8   9  10
## 145  50 108  80 130  34  23  46  14  69
## [1] "V3"
##
##   1   2   3   4   5   6   7   8   9  10
## 384  45  52  40  30  27  19  29   6  67
## [1] "V4"
##
##   1   2   3   4   5   6   7   8   9  10
## 353  59  56  44  34  30  30  28   7  58
## [1] "V5"
##
```

```
##   1   2   3   4   5   6   7   8   9  10 
## 407  58  58  33  23  22  13  25   5  55 
## [1] "V6"
## 
##   1   2   3   4   5   6   7   8   9  10 
##  47 386  72  48  39  41  12  21   2  31 
## [1] "V7"
## 
##   ?   1  10   2   3   4   5   6   7   8   9 
##  16 402 132  30  28  19  30   4   8  21   9 
## [1] "V8"
## 
##   1   2   3   4   5   6   7   8   9  10 
## 152 166 165  40  34  10  73  28  11  20 
## [1] "V9"
## 
##   1   2   3   4   5   6   7   8   9  10 
## 443  36  44  18  19  22  16  24  16  61 
## [1] "V10"
## 
##   1   2   3   4   5   6   7   8  10 
## 579  35  33  12   6   3   9   8  14 
## [1] "V11"
## 
##   2   4 
## 458 241
```

From above we are able to see the value distribution for all the variables, and we find that there is missing value showing as "?" in V7. For next step, need to work on the missing value of V7.

```r
# Save the rows with missing data
# return the row numbers where V7 has missing value "?"
impute_me <- which(bcw$V7 == "?")
impute_me
```

```
##  [1]  24  41 140 146 159 165 236 250 276 293 295 298 316 322 412 618
```

```r
# As mentioned in the coures videos, the estimated missing dat should not be over 5% of the entire data
# Check the missing data pct to make sure we are estimating the missing data within reasonable range
length(impute_me)/nrow(bcw)
```

```
## [1] 0.02288984
```

The mssing value pct is about 2%. So we are good to go ahead to work on those missing values.

Create two dataset to split the data has no missing value and the data has missing value. Save for later analysis

```r
bcw_clean <- bcw[-impute_me,]
bcw_missing <- bcw[impute_me,]
```

```r
# talbe showing response variable from original, with/without missing data
prop.table(table(bcw$V11))
```

```
##
##         2         4
## 0.6552217 0.3447783
```

```r
prop.table(table(bcw_clean$V11))
```

```
##
##         2         4
## 0.6500732 0.3499268
```

```r
prop.table(table(bcw_missing$V11))
```

```
##
##     2     4
## 0.875 0.125
```

For the entire data and without missing data, we have 65% of 2 and 35% of 4. It is very similar that the data without missing data has the same pct to the entire data set.

1. Use mode value for the imputation

```r
# Mode
# Calculate the missing value using mode
# find the mode in table
table(bcw[,7])
```

```
##
##    ?   1  10   2   3   4   5   6   7   8   9
##   16 402 132  30  28  19  30   4   8  21   9
```

There are 402 as 1. So to use mode value for imputation, we should use 1 for those missing data.

```r
# Mean
# Calculate the missing data using mean of V7
bcw_mean <- sum(as.numeric(bcw_clean$V7))/nrow(bcw_clean)
as.integer(bcw_mean)
```

```
## [1] 3
```

The mean of V7 is around 3.54. Since V7 is categorical data from 1 to 10, we can use 3 for imputation to fill in those missing V7 data.

2. Build regression model for imputation

In this method, use the no missing value dataset bcw_clean as data and V7 as response, build a lm model using the rest of the variables to forecast V7. Then apply the regression model to the missing data set to estimate the missing V7.

```r
# Build linear regression model using V7 as response with bcw_clean
# remove column V1 and V11
# convert V7 to integer from string
# buld bacis lm model with all factors
bcw_clean_1 <- bcw_clean[,2:10]
bcw_clean_1$V7 <- as.integer(bcw_clean_1$V7)
lm_model <- lm(V7~., data = bcw_clean_1)
summary(lm_model)
```

```
##
## Call:
## lm(formula = V7 ~ ., data = bcw_clean_1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7316 -0.9426 -0.3002  0.6725  8.6998
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.616652   0.194975  -3.163  0.00163 **
## V2           0.230156   0.041691   5.521 4.83e-08 ***
## V3          -0.067980   0.076170  -0.892  0.37246
## V4           0.340442   0.073420   4.637 4.25e-06 ***
## V5           0.339705   0.045919   7.398 4.13e-13 ***
## V6           0.090392   0.062541   1.445  0.14883
## V8           0.320577   0.059047   5.429 7.91e-08 ***
## V9           0.007293   0.044486   0.164  0.86983
## V10         -0.075230   0.059331  -1.268  0.20524
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.274 on 674 degrees of freedom
## Multiple R-squared:  0.615,  Adjusted R-squared:  0.6104
## F-statistic: 134.6 on 8 and 674 DF,  p-value: < 2.2e-16
```

From the result, we can see there is opportuntiy to improve the model using the stepwise regression methods learned last week. Perform stepwise regression to select variables as well as the 10 fold cross-validation

```r
# create 10 fold cross-validation
train_control <- trainControl(method='cv', number=10)

# method "lmStepAIC" is the stepwise regression method in library 'Caret'
# use trace = 0 to jump directly to the stepwise result
step.model <- train(V7~., data=bcw_clean_1,method="lmStepAIC",
                    scope = list(lower=V7~1,upper=V7~.),
                    direction='both',
                    trControl = train_control,
                    trace=0)
summary(step.model)
```

```
##
## Call:
```

```
## lm(formula = .outcome ~ V2 + V4 + V5 + V8, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8115 -0.9531 -0.3111  0.6678  8.6889
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.53601    0.17514  -3.060   0.0023 **
## V2           0.22617    0.04121   5.488 5.75e-08 ***
## V4           0.31729    0.05086   6.239 7.76e-10 ***
## V5           0.33227    0.04431   7.499 2.03e-13 ***
## V8           0.32378    0.05606   5.775 1.17e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.274 on 678 degrees of freedom
## Multiple R-squared:  0.6129, Adjusted R-squared:  0.6107
## F-statistic: 268.4 on 4 and 678 DF,  p-value: < 2.2e-16
```

From the above result, we will only use variable V2, V4, V5 and V8 to do V7 regression. The adjusted R-squre is 0.6107. It is simpler and it is able to prevent overfit.

```r
# using the step.model to predict those missing V7 value
V7_predict <- predict(step.model, bcw_missing)
V7_predict <- as.integer(V7_predict)
V7_predict
```

```
##  [1] 5 7 0 1 0 2 2 1 2 6 0 2 5 1 0 0
```

Above is the predict missing V7. As covered in the office hour, it is good to see that there is no forecast V7 that over 10. We need to convert those 0 to 1 because V7 is factor data from 1 to 10.

```r
V7_predict[V7_predict <1] <- 1
V7_predict
```

```
##  [1] 5 7 1 1 1 2 2 1 2 6 1 2 5 1 1 1
```

Above is the final missing value for V7.

3. regression with perturbation – add some random "noise"

```r
# generate random value in normal distribution from 0-1
perturb <- rnorm(length(impute_me),0,1)

# add the random value to predict V7
V7_perturb <- round(V7_predict+perturb)
V7_perturb
```

```
##  [1] 5 6 1 1 0 2 3 2 1 6 2 1 5 2 1 1
```

```r
# for value less than 1, need to convert to 1
V7_perturb[V7_perturb <1] <- 1
V7_perturb
```

```
##  [1] 5 6 1 1 1 2 3 2 1 6 2 1 5 2 1 1
```

4. (optional) Use KNN model

```r
# first, to build the dataset from question 1-3

# for dataset 1, replace all ? to 1
bcw_1 <- bcw
bcw_1[bcw_1$V7=="?",]$V7 <- 1
bcw_1$V7 <- as.integer(bcw_1$V7)
table(bcw_1$V7)
```

```
##
##   1   2   3   4   5   6   7   8   9  10
## 418  30  28  19  30   4   8  21   9 132
```

```r
# dataset 2, repalce ? with lm model
bcw_2 <- bcw
bcw_2[bcw_2$V7=="?",]$V7 <- V7_predict
bcw_2$V7 <- as.integer(bcw_2$V7)
table(bcw_2$V7)
```

```
##
##   1   2   3   4   5   6   7   8   9  10
## 410  34  28  19  32   5   9  21   9 132
```

```r
# dataset 3, repalce ? with lm model with perturbation
bcw_3 <- bcw
bcw_3[bcw_3$V7=="?",]$V7 <- V7_perturb
bcw_3$V7 <- as.integer(bcw_3$V7)
table(bcw_3$V7)
```

```
##
##   1   2   3   4   5   6   7   8   9  10
## 409  34  29  19  32   6   8  21   9 132
```

```r
# dataset 4, remove those missing values
bcw_4 <- bcw_clean
bcw_4$V7 <- as.integer(bcw_4$V7)
```

```r
# dataset 5, create one more binary variable V12: 0 when V7 is ? otherwise 1
bcw_5 <- bcw
bcw_5$V12 <- 0
bcw_5[bcw_5$V7=="?",]$V12 <- 0
bcw_5[bcw_5$V7!="?",]$V12 <- 1
```

Create training and validate row numbers

```r
# split data into training, validation as 70% and 30%
train_sample <- sample(nrow(bcw), size=nrow(bcw)*0.7)

# because dataset 4 has less rows, need to generate a new set of random rows for training and validatio
train_sample_4 <- sample(nrow(bcw_4), size=nrow(bcw_4)*0.7)
```

Build KNN model with all 5 models

```r
# test on KNN model for all 5 models
# default accruacy
correct_ac = rep(0, 25)

# 1-5 to build KNN model with dataset 1 - mode
for (x in seq(5)){
  model_knn <- kknn(V11~V2+V3+V4+V5+V6+V7+V8+V9+V10, bcw_1[train_sample,],bcw_1[-train_sample,], k=x)
  predict_v <- as.integer(fitted(model_knn)+0.5) # to fit the result to 2 or 4 to compare with V11
  correct_ac[x] = sum(predict_v==bcw_1[-train_sample,11])/nrow(bcw_1[-train_sample,])
}

# 6-10 to build KNN model with dataset 2 - linear regression
for (x in seq(5)){
  model_knn <- kknn(V11~V2+V3+V4+V5+V6+V7+V8+V9+V10, bcw_2[train_sample,],bcw_2[-train_sample,], k=x)
  predict_v <- as.integer(fitted(model_knn)+0.5) # to fit the result to 2 or 4 to compare with V11
  correct_ac[x+5] = sum(predict_v==bcw_2[-train_sample,11])/nrow(bcw_2[-train_sample,])
}

# 11-15 to build KNN model with dataset 3 - perturbation
for (x in seq(5)){
  model_knn <- kknn(V11~V2+V3+V4+V5+V6+V7+V8+V9+V10, bcw_3[train_sample,],bcw_3[-train_sample,], k=x)
  predict_v <- as.integer(fitted(model_knn)+0.5) # to fit the result to 2 or 4 to compare with V11
  correct_ac[x+10] = sum(predict_v==bcw_3[-train_sample,11])/nrow(bcw_3[-train_sample,])
}

# 16-20 to build KNN model with dataset 4 - remove missing value
for (x in seq(5)){
  model_knn <- kknn(V11~V2+V3+V4+V5+V6+V7+V8+V9+V10, bcw_4[train_sample_4,],bcw_4[-train_sample_4,], k=
  predict_v <- as.integer(fitted(model_knn)+0.5) # to fit the result to 2 or 4 to compare with V11
  correct_ac[x+15] = sum(predict_v==bcw_4[-train_sample_4,11])/nrow(bcw_4[-train_sample_4,])
}

# 21-25 to build KNN model with dataset 5 - binary variable
for (x in seq(5)){
  model_knn <- kknn(V11~V2+V3+V4+V5+V6+V12+V8+V9+V10, bcw_5[train_sample,],bcw_5[-train_sample,], k=x)
  predict_v <- as.integer(fitted(model_knn)+0.5) # to fit the result to 2 or 4 to compare with V11
  correct_ac[x+20] = sum(predict_v==bcw_5[-train_sample,11])/nrow(bcw_5[-train_sample,])
}

# show the predict% for k from 1-25
correct_ac
```

```
##  [1] 0.9666667 0.9666667 0.9285714 0.9285714 0.9285714 0.9619048 0.9619048
##  [8] 0.9190476 0.9190476 0.9190476 0.9619048 0.9619048 0.9238095 0.9238095
## [15] 0.9238095 0.9756098 0.9756098 0.9560976 0.9560976 0.9560976 0.9523810
```

```
## [22] 0.9523810 0.9142857 0.9142857 0.9142857
```

From the result, it turns out the difference using different method of missing data is not that big. Overall, when using KNN model, K=1 and K=2 will have generally better results

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?

In the work, we need to do optimization for global sourcing decision. We need to make the demand of quantity, certain user preference of preferred suppliers, certain constraints of the total count of suppliers, supply available in different location, to achieve the goal of minimize total spend.

For certain category, Data: 1. list of the items: total demand, location 2. list of suppliers, and the portfolio items from suppliers including max supply by location, unit price 3. constraint: prefered suppliers has higher priority 4. constraint: total suppliers count limit 5. constraint: volume discount threshold from suppliers 6. Object: achieve the optimized minimum spend