Assignment 2

Statistics and Data Science 365/565

Due: Wednesday February 12th (11:59pm)

%

Problem 1: Spam, wonderful spam! (35 points)

Problem 1 Part 1

```
library(pacman)
p_load(dplyr,Hmisc,GGally,FNN,stringr,MASS,car,knitr,polynom)
df_test <- read.csv("spam_test.csv")
df_train <- read.csv("spam_train.csv")

X_impute_full <- df_train%>%filter(is.na(capital_run_length_average))
X_train_full <- df_train%>%filter(!is.na(capital_run_length_average))
y_train <- X_train_full$capital_run_length_average%>%scale()
X_train <- X_train_full%>%dplyr::select(-c(spam,capital_run_length_average))%>%scale()
X_impute <- X_impute_full%>%dplyr::select(-c(spam,capital_run_length_average))%>%
  scale(center=attr(X_train, "scaled:center"), scale=attr(X_train, "scaled:scale"))

X_impute_full_test <- df_test%>%filter(is.na(capital_run_length_average))
X_test_full <- df_test%>%filter(!is.na(capital_run_length_average))
y_test <- X_test_full$capital_run_length_average%>%scale()
X_test <- X_test_full%>%dplyr::select(-c(capital_run_length_average))%>%scale()

X_impute_test <- X_impute_full_test%>%dplyr::select(-c(capital_run_length_average))%>%
  scale(center=attr(X_test, "scaled:center"), scale=attr(X_test, "scaled:scale"))

X_impute_full$capital_run_length_average<-knn.reg(X_train, X_impute, y_train, k = 15)$pred
df_train_imputed <- rbind(X_train_full,X_impute_full)

X_impute_full_test$capital_run_length_average<-knn.reg(X_test, X_impute_test, y_test, k = 15)$pred
df_test_imputed <- rbind(X_test_full,X_impute_full_test)
```

Problem 1 Part 2

```
start_time <- Sys.time()
#split train and validation by 0.8 0.2
train.split <- function(xtrain, ytrain, sample.p = .8){
  nrow <- dim(xtrain)[1]
  split <- sample(nrow,sample.p*nrow)
  return(list(xtrain = xtrain[split,], ytrain = ytrain[split],
              xval = xtrain[-split,], yval = ytrain[-split]))
}
#calculate euclid distance
euclid <- function(y1,y2){
  return(sqrt(sum((y1-y2)^2)))
}
#get the mode of a vector
```

```
getmode <- function(v){
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

knnclass <- function(xtrain, xtest, ytrain){
  df <- train.split(xtrain, ytrain)
  xtrain <- df$xtrain
  ytrain <- df$ytrain
  xval <- df$xval
  yval <- df$yval

  # scale
  xtrain <- scale(xtrain)
  xval = scale(xval, center=attr(xtrain, "scaled:center"),
                             scale=attr(xtrain, "scaled:scale"))
  # calculate the classification result for different k each observation of val output has dimension k =
  knn_matrix <- data.frame(sapply(1:length(yval), function(j){
    x <- xval[j,]
    ecludian <- data.frame(distance = sapply(1:length(ytrain),
                                             function(i){euclid(x,xtrain[i,])}))
    ecludian$label <- ytrain
    ecludian <- ecludian%>%arrange(distance)
    knn<- c()
    for(k in 1:sqrt(dim(xtrain)[1])){
      knn_k <- ecludian$label[1:k]
      result <- getmode(knn_k)
      knn <- cbind(knn,result)
    }
    knn
   }))
  # calculate the classification accuracy for each k
   acur <- apply(knn_matrix, 1, function(x){sum(x==yval)/length(yval)})
   k_optimal <- which.max(acur)

   #predict result
   xtest <- scale(xtest, center=attr(xtrain, "scaled:center"),
                              scale=attr(xtrain, "scaled:scale"))
   ypred <- apply(xtest,1,function(x){
     ecludian <- data.frame(distance =sapply(1:length(ytrain),
                                             function(i){euclid(x,xtrain[i,])}),
                            label = ytrain)%>%arrange(distance)
   knn_k <- getmode(ecludian$label[1:k_optimal])
   knn_k})
   return(ypred)
}

xtrain1 <- df_train_imputed%>%
  dplyr::select(-c('capital_run_length_average','spam'))
ytrain <- df_train_imputed[,58]
ypred <- knnclass(xtrain1,df_test_imputed%>%
                    dplyr::select(-c('capital_run_length_average')),ytrain)
xtrain2 <- df_train_imputed[,-58]
ypred2 <- knnclass(xtrain2,df_test_imputed,ytrain)
```

```
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 52.96677 secs
```

Problem 1 Part 3

```
logit1 <- glm(spam ~ .,data=df_train_imputed,family="binomial")
ypred_log <- ifelse(predict(logit1, df_test_imputed, type = "response")>.5,1,0)
logit2 <- glm(spam ~ .,data=df_train_imputed%>%
                dplyr::select(-'capital_run_length_average'),family="binomial")
ypred_log2 <- ifelse(predict(logit2, df_test_imputed%>%
                               dplyr::select(-'capital_run_length_average'),
                             type = "response")>.5,1,0)

result <- data.frame(capital_run_length_average = df_test_imputed$capital_run_length_average,
                     knn_pred1 = ypred, knn_pred2  = ypred2,
                     logm_pred1 = ypred_log, logm_pred2 = ypred_log2)

write.csv(result, "assn2_cs2628_results.csv")
kable(head(result,10))
```

capital_run_length_average	knn_pred1	knn_pred2	logm_pred1	logm_pred2
1.729	1	1	1	1
1.312	1	1	1	1
5.659	1	1	1	1
1.320	0	1	1	1
4.857	1	1	1	1
4.000	1	1	1	1
3.100	1	1	1	1
4.000	1	1	1	1
2.145	1	1	1	1
1.468	1	1	1	1

Problem 2: Cross validation (20 points)

Problem 2 Part a.

```
set.seed(1)
x = rnorm(100)
y = x - 2*x^2 + rnorm(100)
```

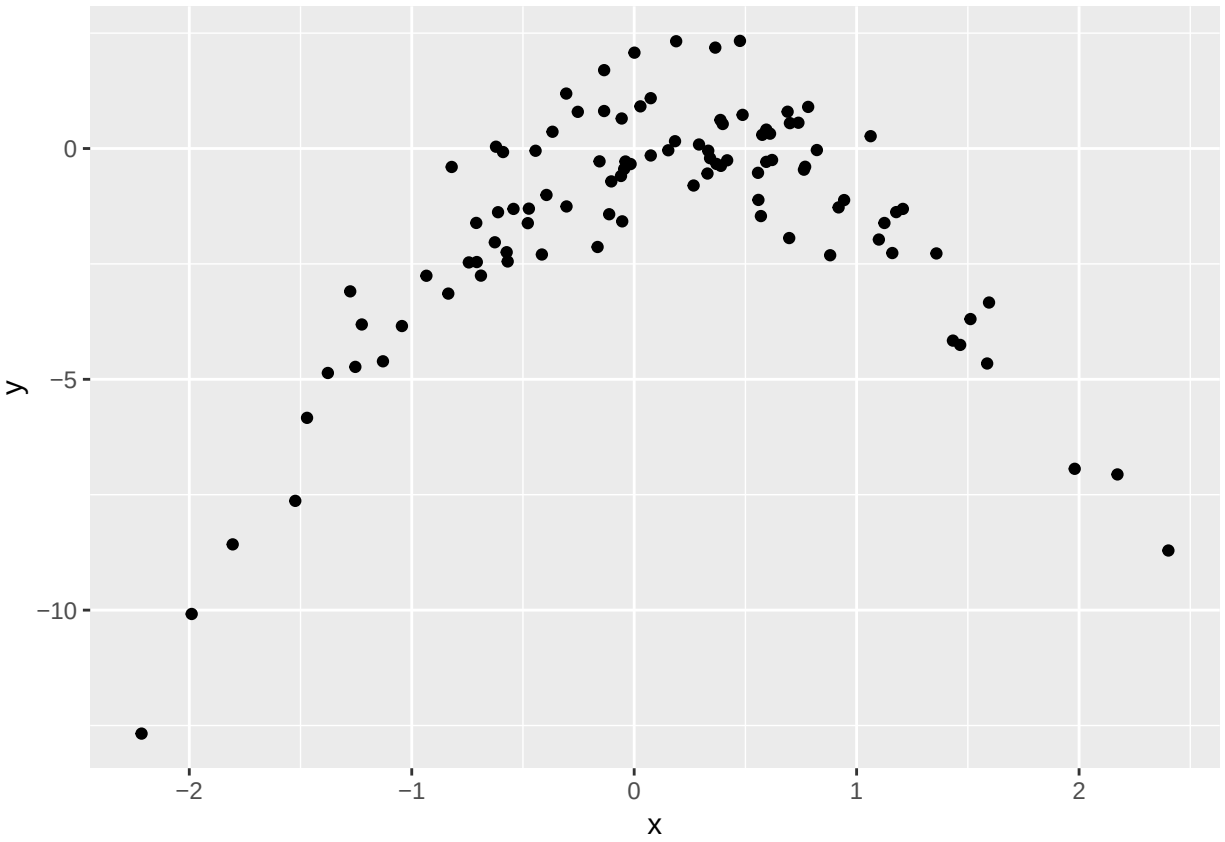
for this question, $n = 100, p = 2$, the model that generate the model is as follows

$$Y = X\beta + \epsilon$$

where $X = [1, x, x^2]^T, \epsilon \sim N(0, 1)$

Problem 2 Part b.

```
data <-  data.frame(y = y, x=x)
ggplot(data, aes(x= x,y =y))+geom_point()
```



find : X and Y are not linear related, however, they have non linear dependency, we might need to fit a polynomial regession model

Problem 2 Part c.

```
data$x2 <- data$x^2
data$x3 <- data$x^3
data$x4 <- data$x^4
set.seed(88)
loocv<-c()
for(i in 1:4){
  designedX <- as.matrix(cbind(1, data[,(1:i)+1]))
  hatMatrix <- designedX%*%solve(t(designedX)%*%designedX)%*%t(designedX)
  diagH <- diag(hatMatrix)
  ypred <- lm(y~.,data = data[,1:(i+1)])$fitted
  y <- data$y
  loocv[i] <- sum(((y-ypred)/(1-diagH))^2)/length(y)
}
loocv
```

```
## [1] 7.2881616 0.9374236 0.9566218 0.9539049
```

Problem 2 Part d.

```
set.seed(777)
x = rnorm(100)
y = x - 2*x^2 + rnorm(100)
data1 <-  data.frame(y = y, x=x)
data1$x2 <- data$x^2
```

```
data1$x3 <- data$x^3
data1$x4 <- data$x^4

loocv2<-c()
for(i in 1:4){
  designedX <- as.matrix(cbind(1, data1[,(1:i)+1]))
  hatMatrix <- designedX%*%solve(t(designedX)%*%designedX)%*%t(designedX)
  diagH <- diag(hatMatrix)
  ypred <- lm(y~.,data = data1[,1:(i+1)])$fitted
  y <- data$y
  loocv2[i] <- sum(((y-ypred)/(1-diagH))^2)/length(y)
}
loocv2
```

```
## [1]  9.120398 11.314950 14.508276 20.216015
```

the result would be same the result determineded only by the underlying original data which will prefer the polymial degree 2 model

Problem 2 Part e.

the second model have the smallest loocv this result is the same as my expectation, since only the polynomial regression with the same or similar degree as the polynomial that data generated will be a best fit. Model 1 is a underfit, while Model 3 and Model 4 are overfits.

	loocv
Model 1	7.2881616
Model 2	0.9374236
Model 3	0.9566218
Model 4	0.9539049

Problem 2 Part f.

The four models and their summaries are as follows, in the first 2 models all the variables are significant and the second model gives a better adjust R^2, which means a better fit. In the second and the third model, the higher order terms are all non significant, which is a evidence for the model is overfitting and not appropriate.

```
model1 <- lm(y~.,data = data[,1:2])
summary(model1)
```

```
##
## Call:
## lm(formula = y ~ ., data = data[, 1:2])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5161 -0.6800  0.6812  1.5491  3.8183
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.6254     0.2619  -6.205 1.31e-08 ***
## x             0.6925     0.2909   2.380   0.0192 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.6 on 98 degrees of freedom
## Multiple R-squared:  0.05465,    Adjusted R-squared:  0.045
## F-statistic: 5.665 on 1 and 98 DF,  p-value: 0.01924
```

```
model2 <- lm(y~.,data = data[,1:3])

summary(model2)
```

```
##
## Call:
## lm(formula = y ~ ., data = data[, 1:3])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9650 -0.6254 -0.1288  0.5803  2.2700
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.05672    0.11766   0.482    0.631
## x            1.01716    0.10798   9.420  2.4e-15 ***
## x2          -2.11892    0.08477 -24.997  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.958 on 97 degrees of freedom
## Multiple R-squared:  0.873,  Adjusted R-squared:  0.8704
## F-statistic: 333.3 on 2 and 97 DF,  p-value: < 2.2e-16
```

```
model3 <- lm(y~.,data = data[,1:4])
summary(model3)
```

```
##
## Call:
## lm(formula = y ~ ., data = data[, 1:4])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9765 -0.6302 -0.1227  0.5545  2.2843
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.06151    0.11950   0.515    0.608
## x            0.97528    0.18728   5.208 1.09e-06 ***
## x2          -2.12379    0.08700 -24.411  < 2e-16 ***
## x3           0.01764    0.06429   0.274    0.784
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9626 on 96 degrees of freedom
## Multiple R-squared:  0.8731, Adjusted R-squared:  0.8691
## F-statistic: 220.1 on 3 and 96 DF,  p-value: < 2.2e-16
```

```
model4 <- lm(y~.,data = data[,1:5])
summary(model4)
```

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

```
## lm(formula = y ~ ., data = data[, 1:5])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0550 -0.6212 -0.1567  0.5952  2.2267
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.156703   0.139462   1.124    0.264
## x            1.030826   0.191337   5.387 5.17e-07 ***
## x2          -2.409898   0.234855 -10.261  < 2e-16 ***
## x3          -0.009133   0.067229  -0.136    0.892
## x4           0.069785   0.053240   1.311    0.193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9591 on 95 degrees of freedom
## Multiple R-squared:  0.8753, Adjusted R-squared:  0.8701
## F-statistic: 166.7 on 4 and 95 DF,  p-value: < 2.2e-16
```