```
knn_1x = function (training_x,training_y,testing_x,k){
+   predicted_y = list()
+   for (i in testing_x){
+     # distance from each training x to target x
+     distance = abs(training_x - i)
+     # pick out the nearest k neighbors
+     indices = order(distance)[1:k]
+     # take the mean of the nearest y values
+     predicted_y = append(predicted_y,mean(training_y[indices]))
+   }
+   return (predicted_y)
+ }
> # Generate Data
> set.seed(111)
> Data_a_i = rnorm(100,mean = 1,sd = 1)
> Data_a_ii = rnorm(100,mean = 2.5,sd = 1)
> Data_b_i = rnorm(100,mean = 1,sd = 1)
> Data_b_ii = rnorm(100,mean = 4, sd = 1)
> y1 = rep(1,100)
> y2 = rep(0,100)
> Data_a_all = c(Data_a_i,Data_a_ii)
> Data_b_all = c(Data_b_i,Data_b_ii)
> y_all = c(y1,y2)
> # loocv with knn
> loocv_knn = function(x,y,k){
+   errors = list()
+   for (i in 1:length(x)){
+     training_x = x[-i]
+     training_y = y[-i]
+     testing_x = x[i]
+     true_y = y[i]
+     output = knn_1x(training_x,training_y,testing_x,k)[[1]]
+     errors = append(errors,round(output)-true_y)
+   }
+   return(errors)
+ }
> #loocv with ls
> loocv_ls = function(x,y){
+   errors = list()
+   for (i in 1:length(x)){
+     training_x = x[-i]
+     training_y = y[-i]
+     df = data.frame(x = training_x,y = training_y)
+     testing_x = x[i]
+     true_y = y[i]
+     lm1 = lm(y ~ x, data = df)
+     output = predict(lm1, newdata =  data.frame(x = testing_x))
+     errors = append(errors,round(output) - true_y)
+   }
+   return(errors)
+ }
> # error of prediction
> knn5_a = loocv_knn(Data_a_all,y_all,5)
> knn10_a = loocv_knn(Data_a_all,y_all,10)
> knn15_a = loocv_knn(Data_a_all,y_all,15)
> ls_a = loocv_ls(Data_a_all,y_all)
> knn5_b = loocv_knn(Data_b_all,y_all,5)
> knn10_b = loocv_knn(Data_b_all,y_all,10)
> knn15_b = loocv_knn(Data_b_all,y_all,15)
> ls_b = loocv_ls(Data_b_all,y_all)
> # calculate accuracy by count number of 0s, more 0s more accurate
> accuracy_a = c(sum(unlist(knn5_a) == 0),sum(unlist(knn10_a) == 0),sum(unlist(knn15_a)
== 0), sum(unlist(ls_a)==0))
> accuracy_b = c(sum(unlist(knn5_b) == 0),sum(unlist(knn10_b) == 0),sum(unlist(knn15_b)
== 0), sum(unlist(ls_b)==0))
> accuracy_a
[1] 157 159 153 156
> accuracy_b
```

```
[1] 181 182 184 184
> #================
> # Q5 --------
> diabetes = read.csv("/Users/vanris/Downloads/diabetes_F24.txt",sep = "", header =
TRUE)
> head(diabetes)
           age         sex         BMI           bp           S1          S2
S3           S4           S5           S6          y
1  0.038075906  0.05068012  0.06169621  0.021872350 -0.044223498 -0.03482076
0.043400846 -0.002592262  0.019908421 -0.017646125  -1.133484
2 -0.001882017 -0.04464164 -0.05147406 -0.026327830 -0.008448724 -0.01916334
-0.074411564 -0.039493383 -0.068329744 -0.092204050 -77.133484
3  0.085298906  0.05068012  0.04445121 -0.005670611 -0.045599451 -0.03419447
0.032355932 -0.002592262  0.002863771 -0.025930339 -11.133484
4 -0.089062939 -0.04464164 -0.01159501 -0.036656450  0.012190569  0.02499059
0.036037570  0.034308859  0.022692023 -0.009361911  53.866516
5  0.005383060 -0.04464164 -0.03638469  0.021872350  0.003934852  0.01559614
-0.008142084 -0.002592262 -0.031991445 -0.046640874 -17.133484
6 -0.092695478 -0.04464164 -0.04069594 -0.019442090 -0.068990650 -0.07928784
-0.041276824 -0.076394504 -0.041180385 -0.096346157 -55.133484
> #subset_selection
> x = as.matrix(diabetes[,c("age","sex","BMI","bp","S1","S2","S3","S4","S5","S6")])
> y = diabetes$y
> best_subset = leaps(x, y, nbest = 1)
> best_subset
$which
       1     2    3     4     5     6     7     8     9     A
1  FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
2  FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
3  FALSE FALSE TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE
4  FALSE FALSE TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE
5  FALSE  TRUE TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE
6  FALSE  TRUE TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE
7  FALSE  TRUE TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE
8  FALSE  TRUE TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
9  FALSE  TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
10  TRUE  TRUE TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE

$label
 [1] "(Intercept)" "1"           "2"           "3"           "4"           "5"
"6"           "7"           "8"           "9"           "A"

$size
 [1]  2  3  4  5  6  7  8  9 10 11

$Cp
 [1] 148.352561  47.072229  30.663634  21.998461   9.148045   5.560162   6.303221
7.248522   9.028080  11.000000

> model2 = lm(y ~ diabetes$sex + diabetes$BMI + diabetes$bp + diabetes$S1 + diabetes$S2
+ diabetes$S5)
> plot(x = c(1:10), y = best_subset$Cp)
> # Ridge model use CV to find best parameter lambda
> cv_ridge = cv.glmnet(x, y, alpha = 0)
> best_lambda_ridge = cv_ridge$lambda.min
> final_ridge_model <- glmnet(x, y, alpha = 0, lambda = best_lambda_ridge)
> ridge_coefficients <- coef(ridge_model, s = best_lambda_ridge)
> cv_ridge

Call:  cv.glmnet(x = x, y = y, alpha = 0)

Measure: Mean-Squared Error

    Lambda Index Measure    SE Nonzero
min   5.97    97    3018 195.2      10
1se  50.73    74    3192 173.7      10
> best_lambda_ridge
[1] 5.96989
> final_ridge_model
```

```
Call:  glmnet(x = x, y = y, alpha = 0, lambda = best_lambda_ridge) 

  Df  %Dev Lambda
1 10 51.34   5.97
> ridge_coefficients
11 x 1 sparse Matrix of class "dgCMatrix"
                       s1
(Intercept) -2.780757e-07
age         -4.183413e-01
sex         -2.133004e+02
BMI          4.976621e+02
bp           3.060026e+02
S1          -9.846695e+01
S2          -6.328231e+01
S3           1.859616e+02
S4           1.145028e+02
S5           4.575334e+02
S6           8.355353e+01
> cv_lasso <- cv.glmnet(x, y, alpha = 1)
> best_lambda_lasso <- cv_lasso$lambda.min
> final_lasso_model <- glmnet(x, y, alpha = 1, lambda = best_lambda_lasso)
> lasso_coefficients <- coef(final_lasso_model, s = best_lambda_lasso)
> cv_lasso

Call:  cv.glmnet(x = x, y = y, alpha = 1) 

Measure: Mean-Squared Error 

    Lambda Index Measure    SE Nonzero
min  0.907    43    3026 185.3       8
1se  7.710    20    3209 120.1       4
> best_lambda_lasso
[1] 0.9073702
> final_lasso_model

Call:  glmnet(x = x, y = y, alpha = 1, lambda = best_lambda_lasso) 

  Df  %Dev Lambda
1  8 51.36 0.9074
> lasso_coefficients
11 x 1 sparse Matrix of class "dgCMatrix"
                       s1
(Intercept) -2.673843e-07
age          .           
sex         -1.996266e+02
BMI          5.224097e+02
bp           2.982379e+02
S1          -1.078234e+02
S2           .           
S3           2.217713e+02
S4           4.267403e+00
S5           5.153914e+02
S6           5.535848e+01
```