We create a function for performing Knn classification. First define distance function:

```r
mydist <- function(x,y) {
    # Compute euclidean distance
    sqrt(sum( (x - y)^2 ))
}
```

Next we create main Knn function:

```r
myknn <- function(X.train, y.train, X.test, k) {
    # compute distance between test points and train points
    dist.mat <- apply(X.test,1,function(y){
        dist.vec <- apply(X.train,1,function(x){
            x <- as.vector(unlist(x))
            y <- as.vector(unlist(y))
            mydist(x,y)
        )
        dist.vec
    })
    # order training indicies according to distances
    ordered.dist.inds <- apply(dist.mat,2,function(x){
        order(x)
    })
    # get training classes (votes) for ordered points
    vote.mat <- apply(ordered.dist.inds,2,function(x){
        y.train[x]
    })
    # get majority vote for k closest points
    preds <- sapply(1:ncol(vote.mat), function(ind){
        vec <- factor(vote.mat[1:k,ind],levels=levels(y.train))
        t <- table(vec)
        t.max <- max(t)
        res <- as.numeric(names(t[t==t.max]))
        # break ties at random
        ifelse(length(res) == 1, res, sample(res,size=1))
    })
}
```

Next we read in test and training data for 3 and 8 digits:

```r
# read in training data
eight.train.X <- read.table('./train.8',sep=',')
eight.train.y <- rep(8,times=nrow(eight.train.X))
three.train.X <- read.table('./train.3',sep=',')
three.train.y <- rep(3,times=nrow(three.train.X))
# combine training data
X.train <- rbind.data.frame(
    eight.train.X,
    three.train.X
)
y.train <- as.factor(
    c(}
eight.train.y,
three.train.y
)
)
# readin test data
test <- read.table("./zip.test")
y.test <- test[,1]
X.test <- test[,-1]
# subset test data
is.three.or.eight <- (y.test == 3) | (y.test == 8)
y.test <- as.factor(y.test[is.three.or.eight])
X.test <- X.test[is.three.or.eight,]

Next, over a range of k values, fit our knn model on the training data and predict on the test set.

k.vec <- 1:5
error.vec <- c()
for(k in k.vec) {  
  # fit and get predictions
  preds <- myknn(X.train, y.train, X.test, k)
  # calculate confusion matrix and miss classification error
  con.mat <- table(preds, y.test)
  err <- 1 - (sum(diag(con.mat)) / sum(con.mat))
  error.vec <- c(error.vec, err)
}

Finally plot error rate as a function of k:

suppressMessages(library(ggplot2))
qplot(k.vec, error.vec, geom='line')
A value of $k = 3, 4$ seem to perform well

**Stat 444/640 - Regression**

**One**

Here we consider the regression model

$$ y = \alpha 1 + X \beta + \epsilon $$

We consider the effects of centering $X, y$ on our estimates. Let $H = I - 1 1^T$ be the centering
matrix. Note we can write our OLS estimates as
\[
\hat{\beta} = \left[X^T H X \right]^{-1} X^T H y \\
\hat{\alpha} = \frac{1}{n} 1^T \left( y - X \hat{\beta} \right) \\
= \frac{1}{n} 1^T \left( y - X \left[X^T H X \right]^{-1} X^T H y \right)
\]

Centering \( X, y \) yields the following:
\[
\hat{\beta} = \left[(H X)^T H (H X)\right]^{-1} (H X)^T H (H y) \\
= \left[X^T H X \right]^{-1} X^T H y
\]
so that our OLS estimate of \( \beta \) remains unchanged. Substituting for \( y \) in the above yields
\[
\hat{\beta} = \beta + \left[X^T H X \right]^{-1} X^T H \epsilon
\]
so that \( E[\hat{\beta}] = \beta \). Making similar replacements for \( \hat{\alpha} \) yields
\[
\hat{\alpha} = \frac{1}{n} 1^T \left( H y - H X \left[X^T H X \right]^{-1} X^T H y \right) \\
= \frac{1}{n} 1^T \left( I - H X \left[X^T H X \right]^{-1} X^T \right) H (\alpha 1 + X \beta + \epsilon) \\
= \frac{1}{n} 1^T \left( I - H X \left[X^T H X \right]^{-1} X^T \right) H X \beta + \frac{1}{n} 1^T \left( I - H X \left[X^T H X \right]^{-1} X^T \right) H \epsilon \\
= \frac{1}{n} 1^T \left( I - H X \left[X^T H X \right]^{-1} X^T \right) H \epsilon
\]
so that \( E[\hat{\alpha}] = 0 \)

Two

The conclusion that cholesterol is an unimportant predictor of cardiac arrest is not correct. Variables are deemed significant or insignificant only in the context of the current model. More accurately we could state that in the presence of the blood pressure variable (and others), cholesterol does contribute additional predictive value.

Three

Recall that our OLS estimate is given by
\[
\hat{\beta} = (X^T X)^{-1} X^T y
\]
In the case orthogonal design we have \( X^T X = I_p \) so that
\[
\hat{\beta} = (X^T X)^{-1} X^T y \\
= X^T y
\]

Regression - STAT 640

Question 1

If \( X^T X \) is invertible and \( y \) is in the column space of \( X \) then \( \hat{y} = X (X^T X)^{-1} X^T y = y \) and \( RSS = 0 \).
Question 2
We want to prove that \( \hat{\beta}_{OLS} \) is the Best Linear Unbiased Estimator. In other words we want to show that \( \hat{\beta}_{OLS} \) has a smaller mean squared error than all other unbiased linear estimators.

Let \( y = X\beta^* + \epsilon \) and \( \hat{\beta}_{linear} = Hy \) be a linear estimator. Consider the difference between the two hat matrices

\[
H - (X^T X)^{-1}X^T = D \Rightarrow H = D + (X^T X)^{-1}X^T
\]

Because \( \hat{\beta}_{linear} \) is unbiased

\[
E[\beta^* - \hat{\beta}_{linear}] = 0 \\
E[\beta^* - Hy] = 0 \\
E[\beta^* - (D + (X^T X)^{-1}X^T)y] = 0 \\
E[\beta^* - Dy] - E[(X^T X)^{-1}X^T y] = 0 \\
E[\beta^* - DX\beta^* - De] - \beta^* = 0 \\
E[\beta^* - DX\beta^* - DE[\epsilon] - \beta^* = 0 \\
\beta^* - DX\beta^* - \beta^* = 0 \\
(I^* - DX)\beta^* = \beta^* \\
DX\beta^* = 0
\]

Since the estimator \( \hat{\beta}_{linear} \) is unbiased \( MSE(\hat{\beta}_{linear}) = var(\hat{\beta}_{linear}) \)

\[
MSE(\hat{\beta}_{linear}) = var(\hat{\beta}_{linear}) \\
= H \text{var}(y)H^T \\
= \sigma^2(D + (X^T X)^{-1}X^T)(D^T + X(X^T X)^{-1}) \\
= \sigma^2[D D^T + (X^T X)^{-1}X^T D^T + DX(X^T X)^{-1} + (X^T X)^{-1}X^T X(X^T X)^{-1}] \\
= \sigma^2[D D^T + (X^T X)^{-1}] \\
= \sigma^2 D D^T + \text{var}(\hat{\beta}_{OLS})
\]

Therefore because \( DD^T \) is positive semidefinite and \( \text{var}(\hat{\beta}_{linear}) \geq \text{var}(\hat{\beta}_{OLS}) \)

Question 3
There are many algorithms that will solve this problem. We will use Proximal Gradient Descent and show that this problem satisfies the conditions necessary for convergence. The Proximal Gradient Descent algorithm is guaranteed to converge to the global minimum if we satisfy the following conditions (Nesterov 2007)

1. The constraint set \( \{ \beta : \beta \geq 0 \} \) is convex and \( \| y - X\beta \| \) is convex.
2. The differentiable function \( \| y - X\beta \| \) has a Lipschitz continuous gradient.
3. There exists a point \( x^* \) that minimizes the objective function.

We want to solve

\[
\hat{\beta} = \text{argmin}_\beta \| y - X\beta \| \\
\text{subject to} \ \beta \geq 0
\]
First we show that the constraint set and the objective function are convex, guaranteeing that any local minimum of the optimization problem is a global minimum. If we let \( f(\beta) = \| y - X\beta \| \) then the hessian of \( f \)

\[
\nabla^2 f(\beta) = X^T X
\]

is positive semi definite making it a convex function. The constraint set \( \{ \beta : \beta \geq 0 \} \) is also convex since any convex combination of non-negative vectors is still a non-negative vector. If \( x, y \geq 0 \) and \( t \in [0, 1] \) then \( tx + (1 - t)y \geq 0 \) making \( \{ \beta : \beta \geq 0 \} \) a convex set.

We say a function has a Lipschitz continuous gradient with constant \( L \) if

\[
\| \nabla f(\beta) - \nabla f(\eta) \| \leq L \| \beta - \eta \|
\]

For our problem

\[
\| \nabla f(\beta) - \nabla f(\eta) \|^2 = \| X^T y + X^T X\beta + X^T y - X^T X\eta \|^2
\]

\[
= \| X^T X(\beta - \eta) \|^2
\]

\[
= \| UD^2 U^T (\beta - \eta) \|^2
\]

\[
= (\beta - \eta)^T UD^2 U^T (\beta - \eta)
\]

\[
= \sum_{i=1}^{p} d_i^4 (\beta_i - \eta_i)^2
\]

\[
\leq \sum_{i=1}^{p} (\max_j d_j)^4 (\beta_i - \eta_i)^2
\]

\[
= \max_i (\lambda_i^2) \| \beta - \eta \|^2
\]

where \( \{ \lambda_i \} \) are the eigenvalues of \( X^T X \). So the Lipschitz constant is \( L = \max_i (\lambda_i^2) \). Existence of a solution follows if the constraint set is closed and convex, and our objective function is continuous on the set. Here the set of non-negative vectors is closed as its complement is open. The function \( \| y - X\beta \| \) is quadratic and Lipschitz continuous on \( \mathbb{R}^n_+ \). Therefore there exists a solution.