36-669 HW3

Question 1

```
snp_data <- read.csv("https://raw.githubusercontent.com/xuranw/469_public/master/data/snp_data.csv", ro
```

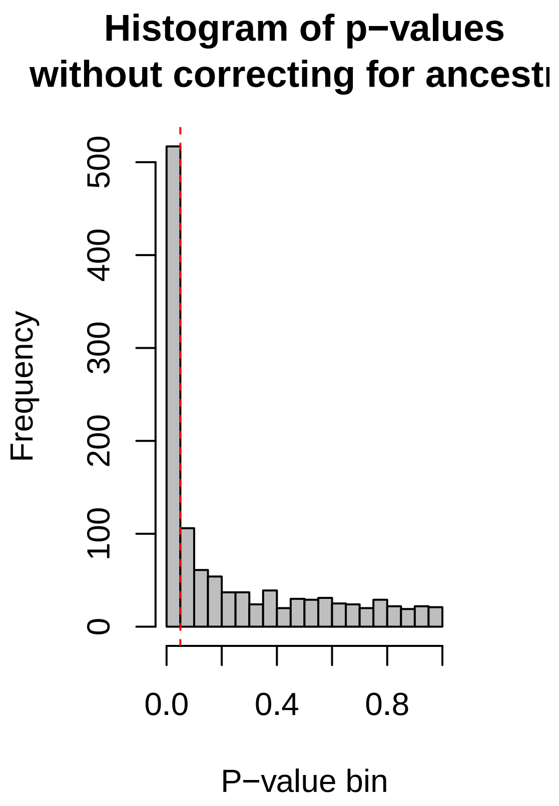
1.A

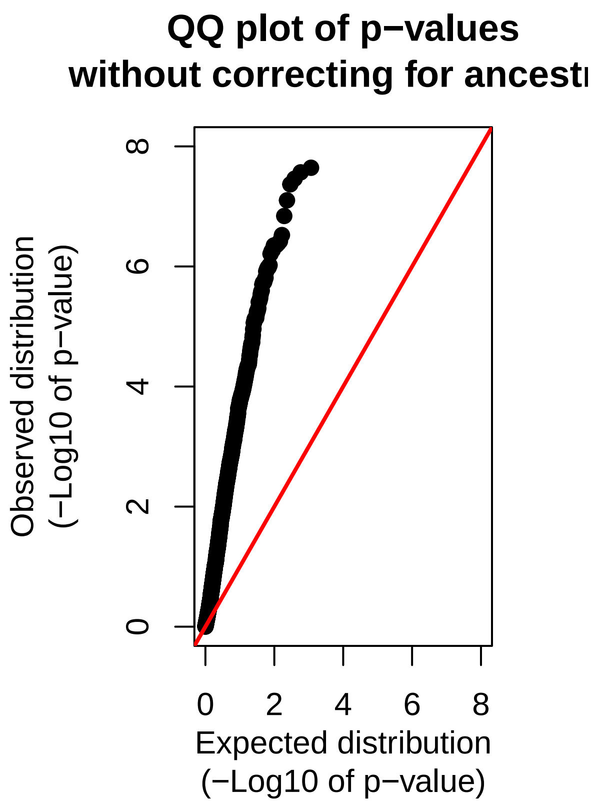
```
# storing the p-values for each of the logistic simple linear regressions of heart disease on the SNPs
collect <- list()
for (i in names(snp_data)[-1]) {
  collect[[i]] <- stats::glm(snp_data[,1] ~ get(i), data = snp_data, family = 'binomial')
}

p_vec <- rep(0, length(collect))
i <- 1L
for (i in seq_along(p_vec)) {
  p_vec[i] <- summary(collect[[i]])$coefficients[2,4]
}

par(mfrow = c(1,2), mar = c(5, 6, 4, 2.5))
hist(p_vec, breaks = 20, col = "gray", xlab = "P-value bin",
     main = "Histogram of p-values\nwithout correcting for ancestry")
abline(v = 0.05, col = "red", lty = 2)

log_p <- sort(-log(p_vec, base = 10), decreasing = F)
expected_p <- sort(-log(seq(0, 1, length.out = length(p_vec)), base = 10), decreasing = F)
plot(expected_p, log_p, xlim = c(0,8), ylim = c(0,8), pch = 19,
     ylab = "Observed distribution\n(-Log10 of p-value)",
     xlab = "Expected distribution\n(-Log10 of p-value)",
     main = "QQ plot of p-values\nwithout correcting for ancestry")
lines(c(-1,10), c(-1,10), col = "red", lwd = 2)
```



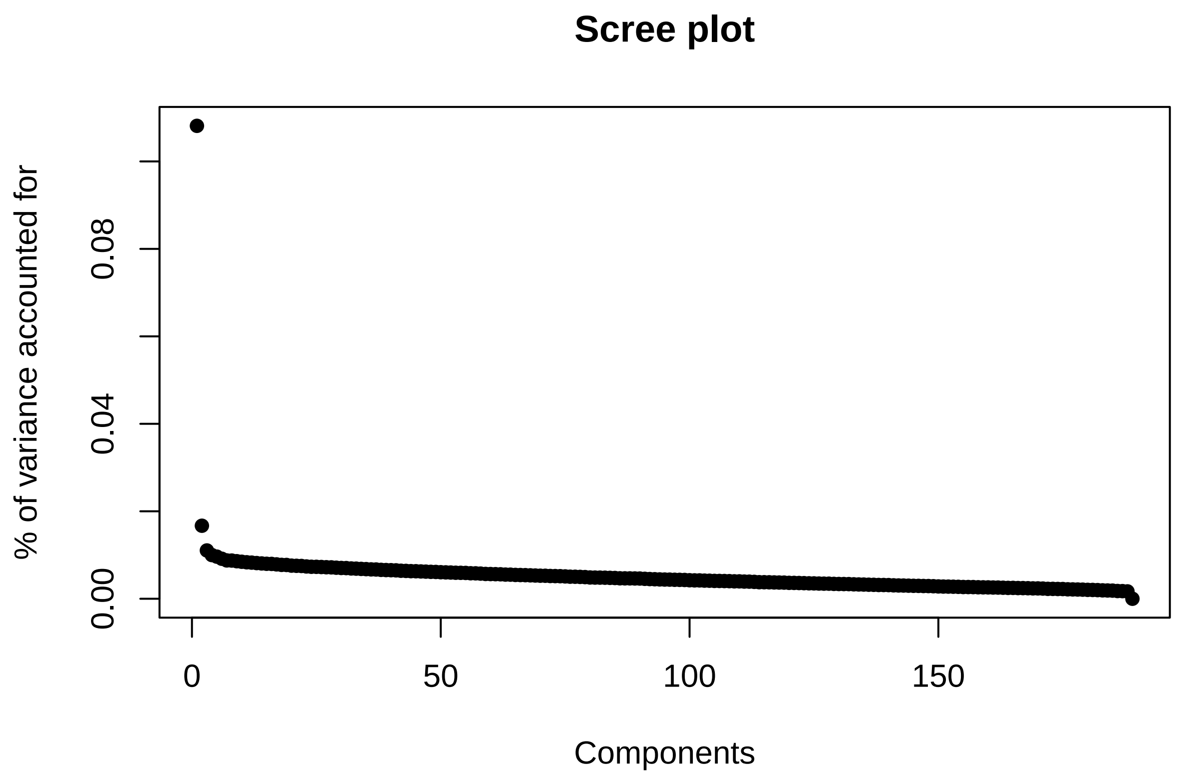


```r
par(mfrow = c(1,1))
```

The results of this analysis suggest that when we don't control for anything in the logistic regression, there can be many SNPs that seem important for predicting heart disease due to the extremely low p-values. We would expect a uniform distribution based on the histogram and that many of the points on the QQ plot would align with the red line drawn on the plot, indicating that the majority of the SNPs are not useful for prediction. Therefore, the results are misleading since the lack of controls suggest omitted variable bias is present in the results and can cause us to incorrectly conclude that many of these SNPs are important in predicting heart disease.

1.B

```r
snp_data_2 <- snp_data[,-1]
pca_res <- stats::prcomp(snp_data_2, center = T, scale. = T)
pca_res_plot <- (pca_res$sdev)^2/sum(pca_res$sdev^2)
plot(pca_res_plot, pch = 16, ylab = "% of variance accounted for", xlab = "Components", main = "Scree pl
```



```r
head(pca_res_plot,2)
```

```
## [1] 0.10813262 0.01669471
```

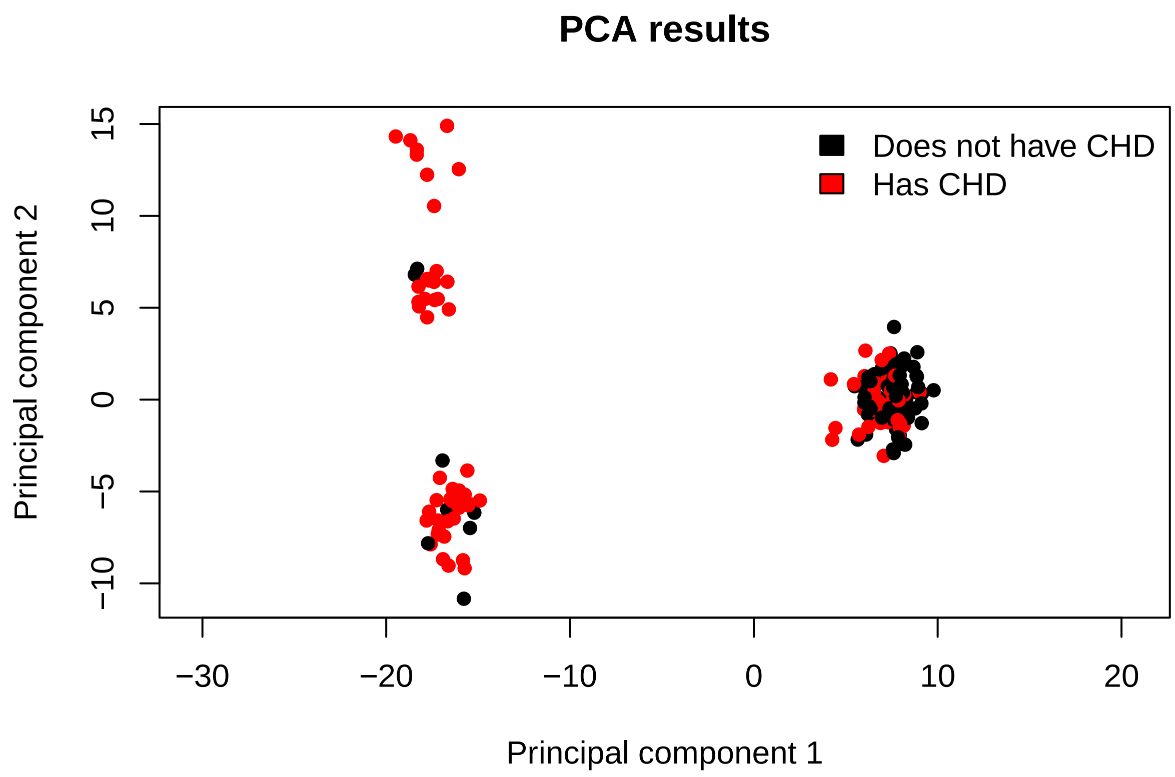
```r
pca_res_plot[1] + pca_res_plot[2]
```

```
## [1] 0.1248273
```

Based on the Scree plot, we see that the first two principle components account for the most percentage of the variance in heart disease in comparison to the others (approximately 12.48%), so these would be reasonable to use for the analysis.

1.C

```r
# plotting the PCAs
heart_disease <- snp_data[,1]
pca1 <- pca_res$x[,1]
pca2 <- pca_res$x[,2]
plot(pca1, pca2, asp = T, xlab = "Principal component 1", ylab = "Principal component 2", pch = 16, col
legend("topright", legend = c("Does not have CHD", "Has CHD"), fill = c("black", "red"), bty = "n")
```



Since the data tend to group together based on different values for the PCA, it is reasonable to assume that there is an underlying relationship the data (ancestry) that implies that the subjects in the data are not all the same but rather group together based on their ancestry. This suggests that our initial analysis suffers from omitted variable bias since we don't account for the differences between the groups in the logistic regressions that are performed with the SNPs. We will likely see much less predictive power in many of the SNPs after controlling for ancestry.

1.D

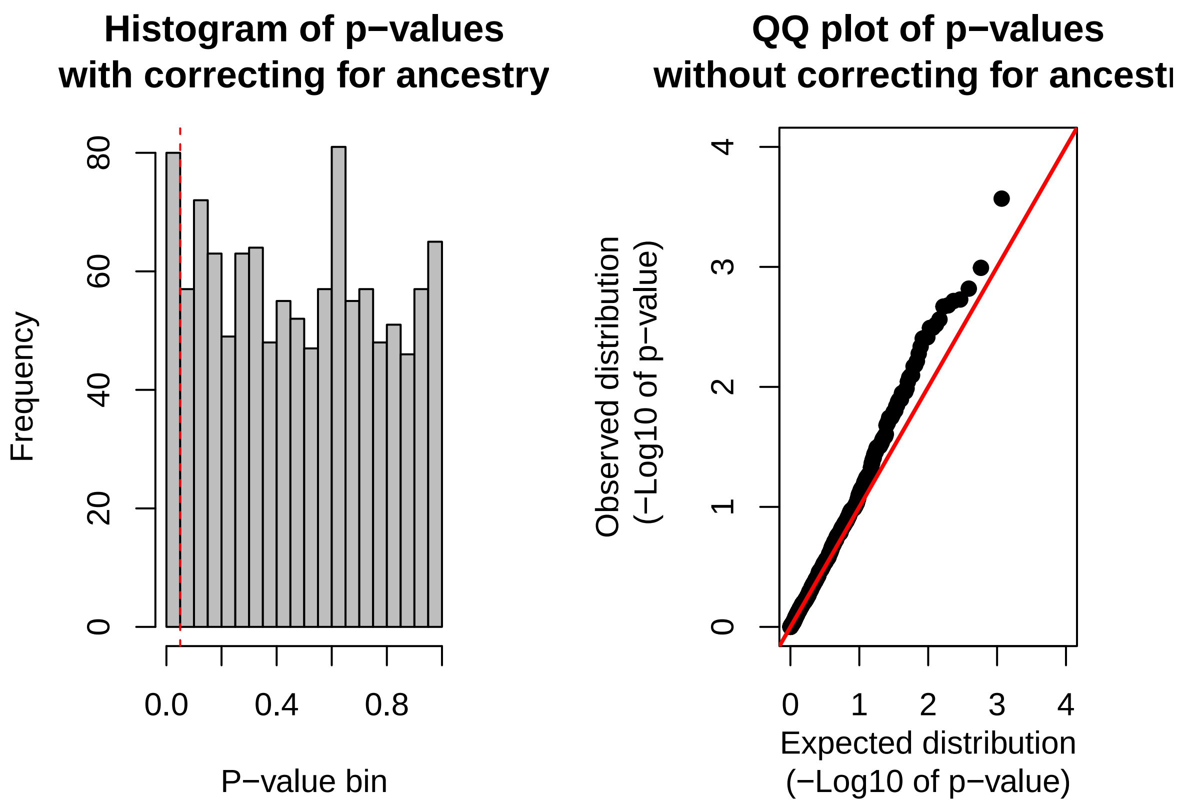
```
collect2 <- list()
i <- 1L
for (i in names(snp_data)[-1]) {
  collect2[[i]] <- stats::glm(snp_data[,1] ~ get(i) + pca1 + pca2, data = snp_data, family = 'binomial'
}

p_vec2 <- rep(0, length(collect2))
i <- 1L
for (i in seq_along(p_vec2)) {
  p_vec2[i] <- summary(collect2[[i]])$coefficients[2,4]
}

par(mfrow = c(1,2), mar = c(5, 6, 4, 2.5))
hist(p_vec2, breaks = 20, col = "gray", xlab = "P-value bin",
     main = "Histogram of p-values\nwith correcting for ancestry")
```

```
abline(v = 0.05, col = "red", lty = 2)

log_p2 <- sort(-log(p_vec2, base = 10), decreasing = F)
expected_p2 <- sort(-log(seq(0, 1, length.out = length(p_vec2)), base = 10), decreasing = F)
plot(expected_p2, log_p2, xlim = c(0,4), ylim = c(0,4), pch = 19,
     ylab = "Observed distribution\n (-Log10 of p-value)",
     xlab = "Expected distribution\n(-Log10 of p-value)",
     main = "QQ plot of p-values\nwithout correcting for ancestry")
lines(c(-1,5), c(-1,5), col = "red", lwd = 2)
```



```
par(mfrow = c(1,1))
```

After controlling for ancestry in the logistic regression model for each SNP, we see that many of the SNPS no longer are statistically significant (p-values less than 0.05). The histogram and QQ Plot demonstrate this, since the histogram more closely resembles a uniform distribution and the QQ plot shows that the majority of the observed p-values are much more closely aligned with their expected p-values. After controlling for ancestry, it is much more straightforward to identify important SNPs since they will still have low p-values after including the PCAs in the regression. These SNPs are more important in predicting heart disease in an individual and should be further investigated.

Question 2

```r
# load in the data and functions
source("https://raw.githubusercontent.com/xuranw/469_public/master/hw3/hw3_functions.R")
set.seed(10)
dat <- generate_data()
first_pc <- stats::prcomp(dat)$x[,1]
```

2.A

```r
# compute the first principal component of dat with svd and pca
# approximately equal vectors for pca and svd for first vector

# eigen vector pc1
eigen_pc <- as.vector(dat %*% eigen(stats::cov(dat), symmetric = F)$vectors[,1])
# svd pc1
svd_pc <- as.vector(dat %*% svd(dat)$v[,1])

# comparison

sum(abs(eigen_pc - first_pc)) <= 1e-6
```

```
## [1] TRUE
```

```r
sum(abs(svd_pc - first_pc)) <= 1e-6
```

```
## [1] TRUE
```

```r
# first eigenvalue
eigen(stats::cov(dat))$values[1]
```

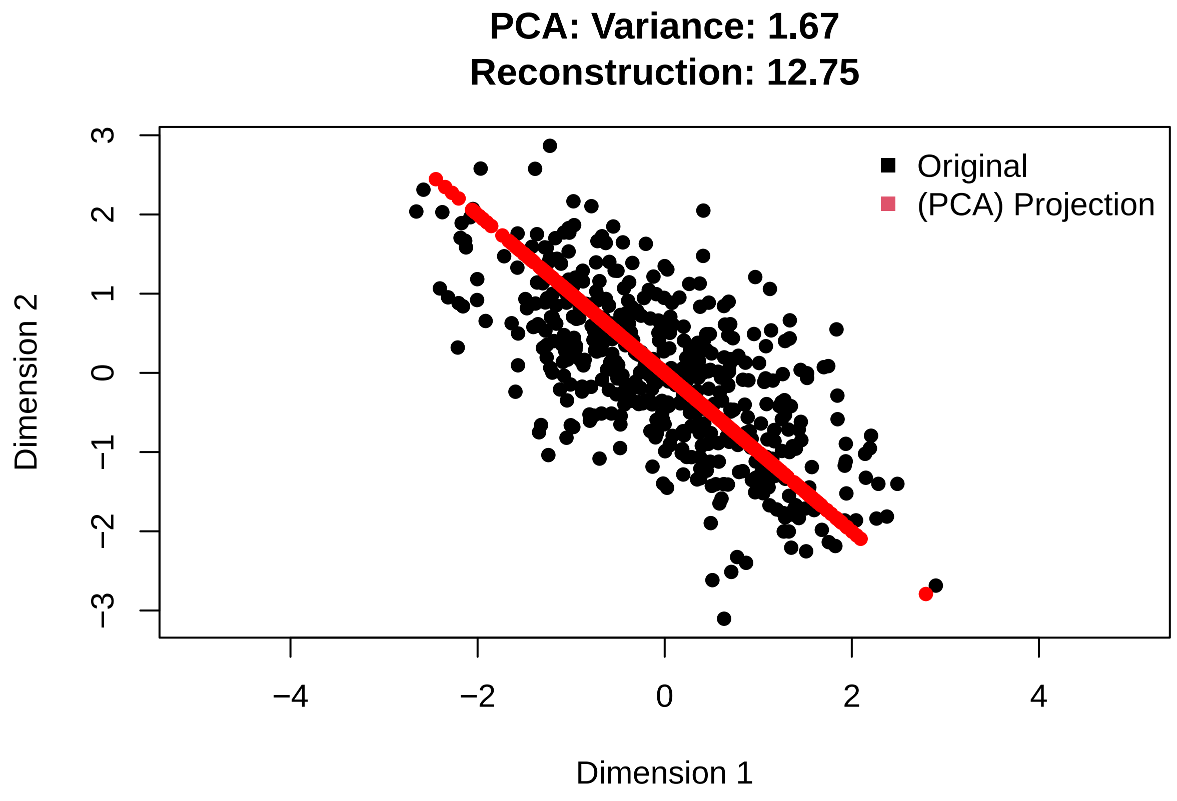
```
## [1] 1.674155
```

2.B

```r
# first function
compute_variance <- function() {
  eigen_res1 <- eigen(stats::cov(dat), symmetric = F)
  sum(diag(var(dat %*% eigen_res1$vectors[,1] %*% t(eigen_res1$vectors[,1]))))
}

# second function
compute_reconstruction <- function() {
  eigen_res2 <- eigen(stats::cov(dat), symmetric = F)
  sqrt(sum((dat - dat %*% eigen_res2$vectors[,1] %*% t(eigen_res2$vectors[,1]))^2))
}
```

```
plot(dat, main = "PCA: Variance: 1.67\nReconstruction: 12.75",
     xlab = "Dimension 1", ylab = "Dimension 2", pch = 16, xlim = c(-5,5))
eigen_res <- eigen(stats::cov(dat), symmetric = F)$vectors[,1]
plot_vec <- dat %*% eigen_res %*% t(eigen_res)
points(plot_vec, col = "red", pch = 16)
legend('topright', c('Original', '(PCA) Projection'), pch = c(15,15),
       col = c(1,2), bty = "n")
```



We see that the variance along the first principle component is the same as the first eigenvalue at 1.67.

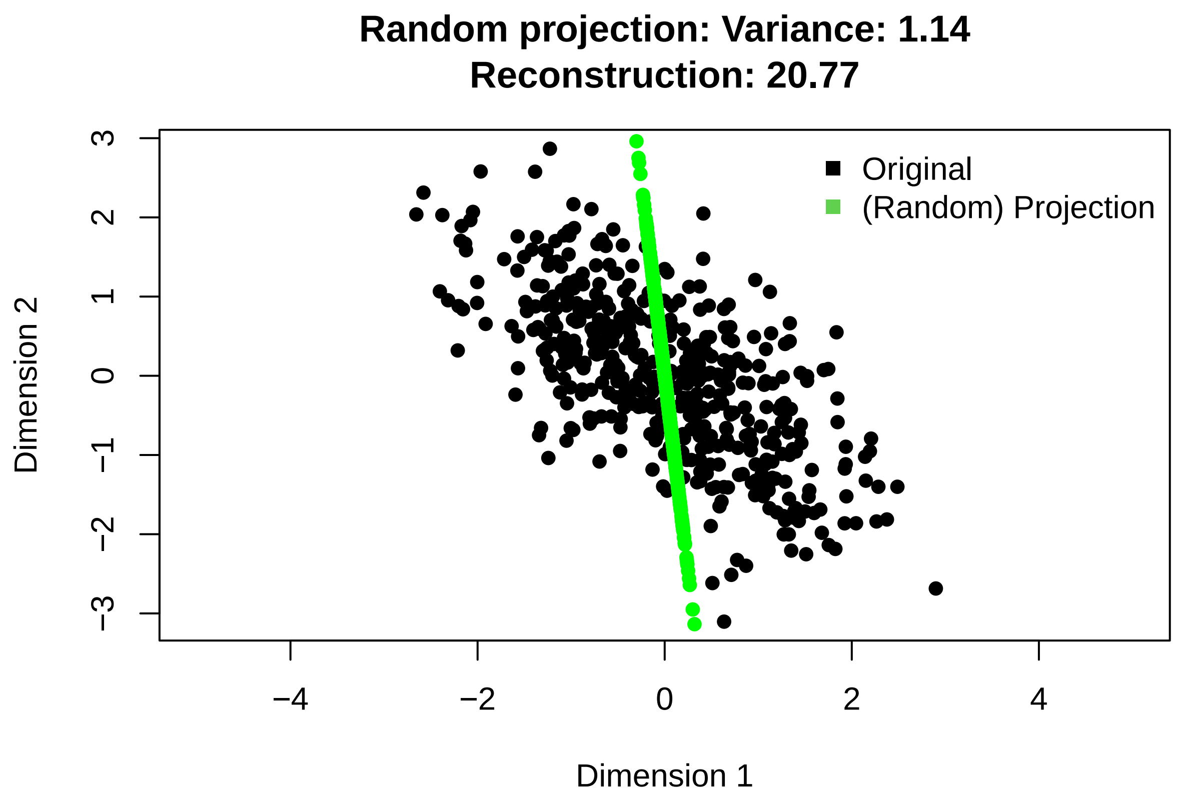
2.C

```
compute_variance <- function() {
  set.seed(10)
  unit_res1 <- generate_random_direction()
  sum(diag(var(dat %*% unit_res1 %*% t(unit_res1))))
}

# second function
compute_reconstruction <- function() {
  set.seed(10)
  unit_res2 <- generate_random_direction()
  sqrt(sum((dat - dat %*% unit_res2 %*% t(unit_res2))^2))
```

```r
}

plot(dat, main = "Random projection: Variance: 1.14\nReconstruction: 20.77",
     xlab = "Dimension 1", ylab = "Dimension 2", pch = 16, xlim = c(-5,5))
set.seed(10)
unit_res <- generate_random_direction()
plot_vec <- dat %*% unit_res %*% t(unit_res)
points(plot_vec, col = "green", pch = 16)
legend('topright', c('Original', '(Random) Projection'), pch = c(15,15),
       col = c(1,3), bty = "n")
```



Comparing the results to part 2.B, we see that PCA generates data that is a much better fit to the data than the random unit vector. The PCA also captures more variance in the response variable than the randomly generated unit vector. Coupling this with the smaller reconstruction error for the PCA projection, this suggests that PCA is a better fit for predicting the data.

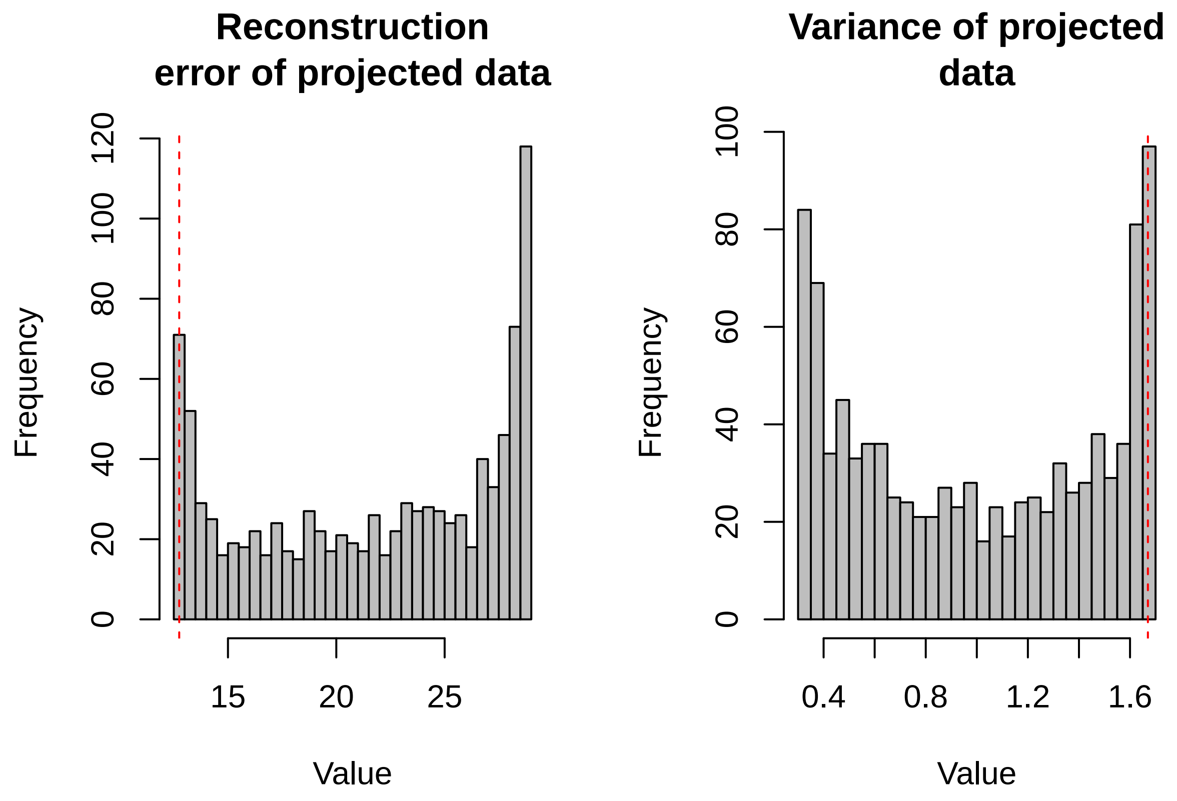
2.D

```r
# updated first function for simulation
compute_variance_sim <- function() {
  unit_res3 <- generate_random_direction()
  sum(diag(var(dat %*% unit_res3 %*% t(unit_res3))))
}
```

```r
# updated second function for simulation
compute_reconstruction_sim <- function() {
  unit_res4 <- generate_random_direction()
  sqrt(sum((dat - dat %*% unit_res4 %*% t(unit_res4))^2))
}

var_vec <- rep(0, 1000)
recon_vec <- rep(0, 1000)
i <- 1L
for (i in seq_along(var_vec)) {
  var_vec[i] <- compute_variance_sim()
  recon_vec[i] <- compute_reconstruction_sim()
}

par(mfrow = c(1,2))
hist(recon_vec, breaks = 30, col = "gray",
     xlab = "Value",
     main = "Reconstruction\nerror of projected data")
abline(v = 12.75, col = "red", lty = 2)
hist(var_vec, breaks = 30, col = "gray",
     xlab = "Value",
     main = "Variance of projected\ndata")
abline(v = 1.67, col = "red", lty = 2)
```



```r
par(mfrow = c(1,1))
```

Based on the reconstruction error histogram, we see that the PCA error (12.75 from Figure 5) is relatively low in comparison with the randomly generated unit vectors in the model (i.e. 20.77 for a random unit vector from Figure 6), suggesting that the PCA is very effective at being used to reconstruct the original data compared to random vectors. The variance histogram shows that the variance of the of the vector formed by projecting the data (X) into a 1-dimensional space is relatively large (1.67 from Figure 5) compared to the randomly generated unit vectors(i.e. 1.14 from a random unit vector from Figure 6). This implies that the PCA vector captures more of the variance in the data (relative to other random vectors) and is therefore useful for predicting in heart disease. The results of the simulation illustrate that the PCA finds the directions that maximize the explained variance and the lower-dimensional space that minimizes the error.