Predicting Heart Disease Using Logistic Regression and k-Nearest Neighbors

Anirudh Divecha
Department of Statistics, University of California, Los Angeles (UCLA)
adivecha@g.ucla.edu

December 10, 2024

1 Introduction and Objective

This report examines various patient data from certain parts of the world in order to determine key factors of heart disease and investigates whether heart disease can be predicted/prevented based off of various features of the data. According to the WHO, heart disease is the number one cause of death around the world [1]. As such, it is vital to conduct analysis on relevant datasets so that heart disease can be better understood and prevented. In this paper, a sophisticated machine learning method such as k-Nearest Neighbors is compared to a simple logistic regression model to see which model produces better prediction results.

2 Data Description and Preprocessing Steps

After investigating various sources for a dataset, I eventually settled on a Kaggle dataset that was actually derived from a very popular heart disease dataset from the UCI Machine Learning Repository [2]. This dataset from UCI has been widely cited and this Kaggle dataset parses and converts the UCI dataset into a more readable format for data analysis. In the dataset each row represents a patient. There are 15 predictors in the dataset, which are described and listed below, and 1 response variable (num) which simply indicates whether or not the patient has heart disease.

- **Source:** Kaggle [3].
- **Variables:**
  - id (Unique ID for each patient)
  - **age** (Age of the patient in years)
  - origin (Place of study)
  - **sex** (Male/Female)
  - **cp** (Chest pain type: typical angina, atypical angina, non-anginal, asymptomatic)
  - **trestbps** (Resting blood pressure in mm Hg on admission to the hospital)

[1] "The top 10 causes of death" *WHO*, 7 Aug. 2024. Web. https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death.
[2] UCI Machine Learning Repository, Heart Disease Dataset, https://archive.ics.uci.edu/dataset/45/heart+disease
[3] Kaggle, Heart Disease Data Set, https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data

- **chol** (Serum cholesterol in mg/dl)
- **fbs** (Fasting blood sugar > 120 mg/dl: True/False)
- **restecg** (Resting electrocardiographic results: normal, STT abnormality, LV hypertrophy)
- **thalach** (Maximum heart rate achieved at exercise)
- exang (Exercise-induced angina: True/False)
- oldpeak (ST depression induced by exercise relative to rest)
- slope (Slope of the peak exercise ST segment)
- ca (Number of major vessels colored by fluoroscopy)
- **thal** (Thalassemia: Normal, fixed defect, reversible defect)
- **num** (The predicted attribute, 0 for no heart disease, 1-4 indicates has some sort of heart disease)

After obtaining the dataset, data cleaning was performed. First, unnecessary predictors were removed along with those that were not well understood or were too medically specific. The variables removed included id, origin, exang, oldpeak, slope, and ca. Exang, oldpeak, slope, and ca, in particular, were too complex to understand fully and were therefore excluded. The nine predictors that remained are bolded in the list above. NaN values were also removed, reducing the dataset to approximately 300 usable observations. The final step involved converting categorical predictor variables into numeric codes (1, 2, 3, etc.). Additionally, the predicted attribute 'num,' which contained values from 1 to 4 indicating the degree of heart disease, was transformed into a binary classification. Any value greater than or equal to 1 was converted to 1, indicating the presence of heart disease. This conversion process for the categorical variables is outlined in the code and described below:

- **sex** (Male = 0, Female = 1)
- **cp** (asymptomatic = 0, non-anginal = 1, atypical angina = 2, typical angina = 3)
- **fbs** (True = 1, False = 0)
- **restecg** (normal = 0, LV hypertrophy = 1, STT abnormality = 2)
- **thal** (normal = 0, reversible defect = 1, fixed defect = 2)
- **num** (no heart disease = 0, any value greater than or equal to 1 was converted to 1)

3 Exploratory Data Analysis

A quick summary of the predictors is shown in Table 1 of Appendix A. A quick check of the response variable, num, shows that it is pretty balanced as well (check Figure 1 of Appendix A). These quick summary statistics confirm that this dataset can be used.

Quick analysis was conducted to identify any differences between men and women among some of the continuous predictor variables. Kernel density estimation (KDE) graphs were generated using C to visualize these differences for variables such as cholesterol and resting blood pressure (trestbps), which were considered most relevant. Figure 2 in Appendix A shows that the mean cholesterol levels for both men and women are around 220-230 mg/dl, with both groups having similar mean levels. Figure 3 in Appendix A suggests that women have slightly higher resting blood pressure than men. Overall, the plots indicate that there are no significant

differences between genders for the continuous predictors of heart disease. Any potential gender influences will be explored further in the regression analysis.

Additionally, the dataset allows for exploration of genetic diseases and its potential influence on resting blood pressure. The only genetic disease that is a predictor in this dataset is thal (or thalassemia). According to Mayo Clinic, thalassemia is an inherited blood disorder that causes hemoglobin levels to be much lower than normal [4]. As such, a box plot of resting blood pressure and thal was generated as shown in Figure 4 in Appendix A. The box plot shows that having a fixed genetic defect at level 2 does have a higher median blood pressure but overall the box plot does not appear to show significant differences at each level for thalassemia. A normal person (level 0) appears to have the same median blood pressure as a person with a reversible defect (level 1).

After exploring these potential relationships in the dataset, regression and machine learning methods could now be used to predict whether an individual has heart disease.

4 Methods and Results

In order to see which models would be the best predictor of heart disease for this dataset, logistic regression and k-nearest neighbors (kNN) were used and compared. Since the response variable is binary (0 or 1), usual linear regression cannot be used and thus this predictor problem became to a certain degree a classification problem.

For both methods, the dataset was split into training and testing data for evaluation of accuracy. Due to only 300 observations in the data set, a 90-10 split was used so that the model could train on more observations.

4.1 Logistic Regression

First, a model was fitted using num as the response variable and age, sex, cp, trestbps, chol, fbs, restecg, thalch, and thal as the predictors in order to see which predictors were most significant in determining heart disease. From the regression summary (see Figure 5 of Appendix A), it appears that age, cholesterol, and fasting blood sugar were not significant predictors. Even though increasing age, cholesterol, and high blood sugar increases the log odds of having heart disease, these seemed to be insignificant in comparison to other factors in the data. In fact, at the 0.1% level, sex, cp1, thalch, and thal1 are bigger predictors of heart disease. It can be noted that being male tends to increase the log odds of having heart disease (sex1 in the regression summary is female and thus a negative log odds coefficient). Additionally, having non-anginal chest pain (cp1), which is chest pain in people without heart disease, decreases the log odds of having heart disease which makes sense. Having a lower max heart rate achieved (thalch) also increases the chances of having heart disease as healthier people tend to be able to pump blood at a much faster rate. Lastly, having some sort of thalassemia defect (thal1 and thal2) in comparison to not having thalassemia seems to increase the log odds of having heart disease. This makes sense because if the body is not able to produce as much hemoglobin to carry oxygen throughout the body, the heart must work harder which would increase the potential for heart disease.

In order to get the best model for prediction, the logistic regression model was refitted with only the significant predictors (see Figure 6 of Appendix A). This model showed that the most

[4] Mayo Clinic Staff. "Thalassemia." Mayo Clinic, 17 Nov. 2021. https://www.mayoclinic.org/diseases-conditions/thalassemia/symptoms-causes/syc-20354995

significant at 0.1% were cp1, thalch, and thal1. Using the test data, the model was tested and found to have around 75.86% test accuracy.

4.2 k-Nearest Neighbors

In order to see if a sophisticated machine learning model can outperform a logistic regression model in terms of prediction, a kNN model was built and tested. For the kNN model, the caret package was used and an optimal k value was determined and used. The confusion matrix and statistics for the model are shown in Figure 7 of Appendix A. The results indicate that the kNN model actually performs worse in terms of accuracy as it had a rate of 68.97%.

4.3 ROC Curves

In order to determine which model performs better, logistic regression or kNN, ROC curves were generated using the pROC library in R. The ROC curves plot (see Figure 8 of Appendix A) indicates that the logistic regression model is performing better than the kNN model since curves closer to the left corner of the plot indicate better performance on ROC curves. In order to determine if there is a statistical difference between the two models, one must compare the Area Under the Curve (AUC) of the ROC curves. Statistical tests such as DeLong's test can determine statistical difference between AUC values and thus differences between model performance [5].

The Delong's test output (see Figure 9 of Appendix A) for the ROC curve indicates that the logistic regression model has better performance at discriminating heart disease from no heart disease but since the p-value is 0.4346, which is greater than 0.05, there is no statistically significant difference between the AUCs of the two models. In summary, while logistic regression slightly outperforms kNN in terms of AUC, the difference is not statistically significant.

5 Discussion

The results of this analysis provide meaningful insights into the predictability of heart disease based on the dataset's features. The logistic regression model achieved a test accuracy of 75.86%, slightly outperforming the k-Nearest Neighbors (kNN) model's accuracy of 68.97%. This suggests that for this dataset, logistic regression is better suited for classifying heart disease. However, the statistical comparison of the models' ROC curves using DeLong's test showed no significant difference in their AUC values, implying that while logistic regression appears more effective, the difference is not enough to claim statistical significance.

Key predictors identified in the logistic regression model, such as chest pain type, maximum heart rate achieved, and thalassemia status, align with established medical understanding of heart disease risk factors. Chest pain type, in particular, was most likely a key predictor because this dataset originates from hospital patients, where chest pain often indicates an acute heart problem. However, while it is a strong signal for diagnosing heart disease in clinical settings, it is less applicable as a long-term preventative measure. Additionally, many of these predictors are beyond patients' control, which may explain why healthcare professionals place greater emphasis on factors patients can control such as cholesterol levels and blood sugar. Although these were not significant predictors in this study, they remain a focal point in healthcare advice due to their controllability and potential to reduce the risk of heart disease over time. These findings

[5] "roc.test: Compare two ROC curves." https://www.rdocumentation.org/packages/pROC/versions/1.18.5/topics/roc.test

reinforce the potential utility of logistic regression as a simpler yet interpretable method for clinicians and researchers when analyzing similar datasets.

5.1 Limitations

Several limitations should be acknowledged. First, the dataset contains only 300 observations, which limits the generalizability of the findings. Additionally, some potentially important predictors were removed during preprocessing due to complexity or lack of domain expertise. Including these features and consulting medical professionals may have improved model performance. Moreover, because the data is derived from hospital patient records, it may not fully capture factors relevant for long-term preventative measures against heart disease. Finally, the kNN model's lower performance may be attributed to the relatively small dataset size, as kNN and many sophiticated machine learning models often perform better with larger datasets.

Lastly, while logistic regression was found to slightly outperform kNN, the lack of statistical significance in the AUC comparison suggests that both models have similar discriminative ability for this dataset. Future work could explore additional machine learning methods, such as Random Forests or Gradient Boosting, to determine whether they offer substantial improvements over these models.

6 Conclusion

Overall, this study highlights the predictive capabilities of logistic regression in identifying heart disease from patient data, with kNN offering a comparable, but slightly less accurate, alternative. This study also was able to identify significant predictors of heart disease which can be further studied by healthcare experts. These results can guide future research in developing robust, interpretable models for heart disease prediction.

7 Appendix

A Tables and Figures

	id	age	trestbps	chol	thalch	oldpeak	ca	num
count	920.00	920.00	861.00	890.00	865.00	858.00	309.00	920.00
mean	460.50	53.51	132.13	199.13	137.55	0.88	0.68	1.00
std	265.73	9.42	19.07	110.78	25.93	1.09	0.94	1.14
min	1.00	28.00	0.00	0.00	60.00	-2.60	0.00	0.00
25%	230.75	47.00	120.00	175.00	120.00	0.00	0.00	0.00
50%	460.50	54.00	130.00	223.00	140.00	0.50	0.00	1.00
75%	690.25	60.00	140.00	268.00	157.00	1.50	1.00	2.00
max	920.00	77.00	200.00	603.00	202.00	6.20	3.00	4.00

Table 1: Summary Statistics of Dataset

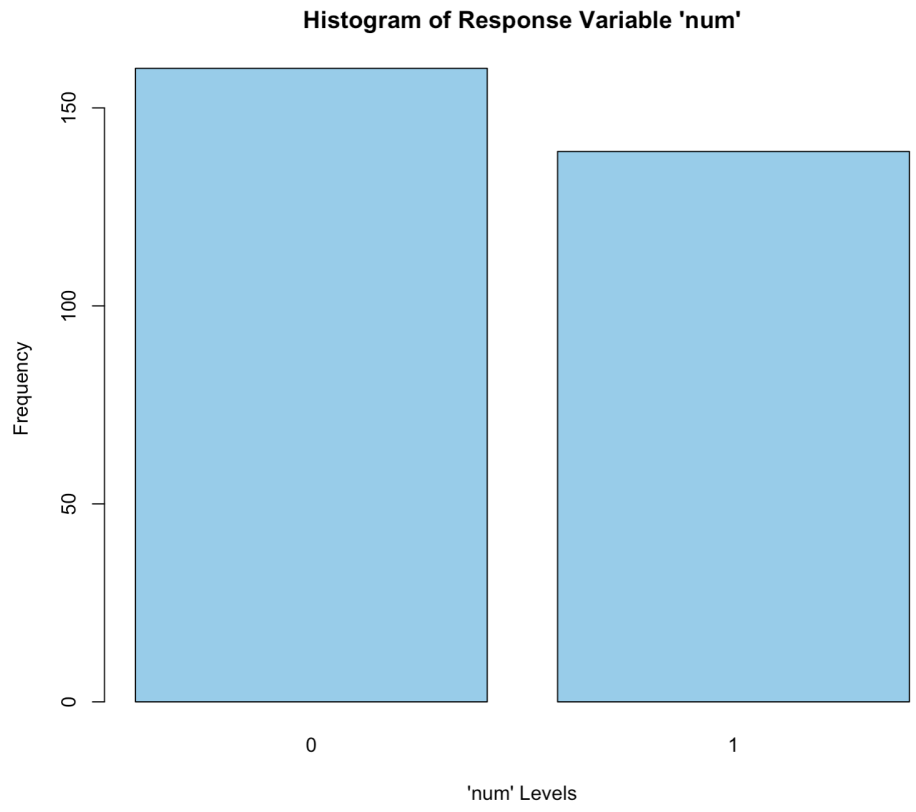

Figure 1

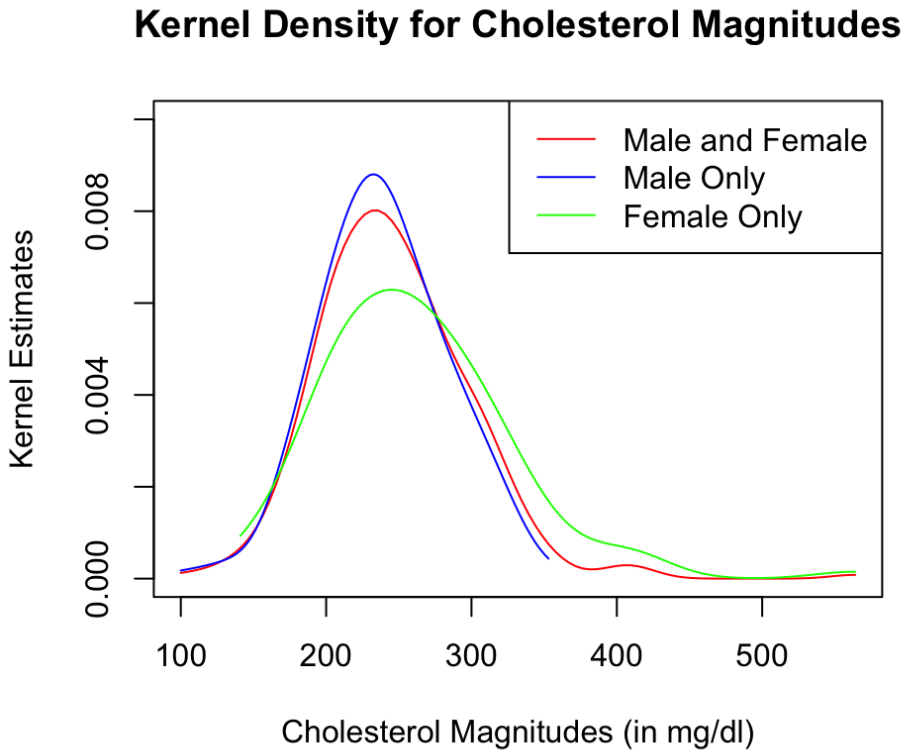

Figure 2

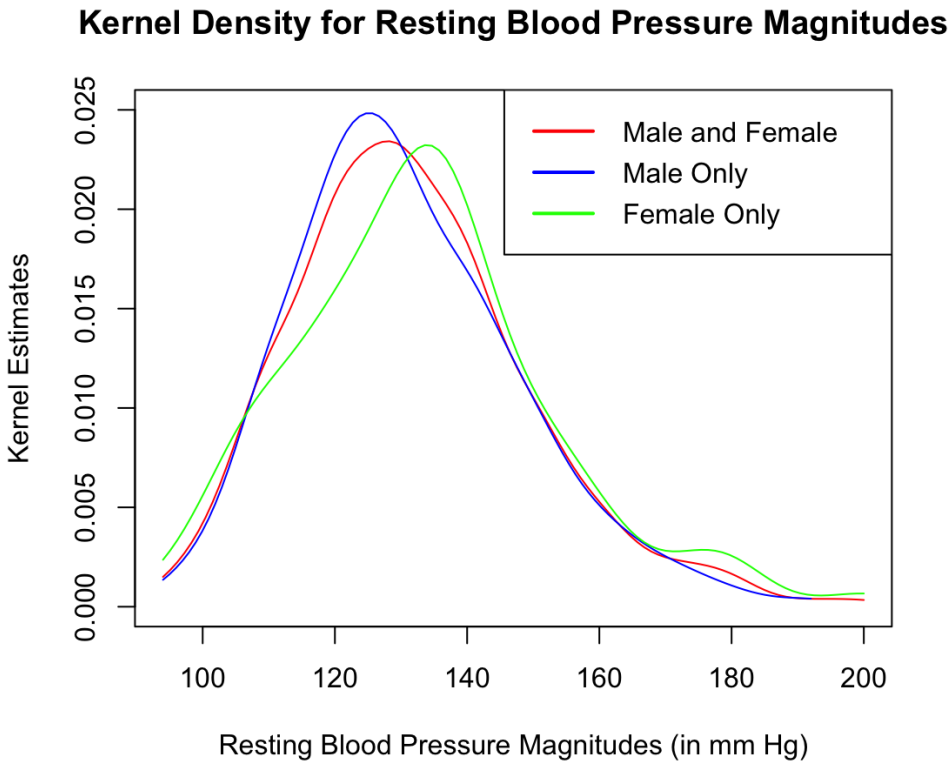

Figure 3

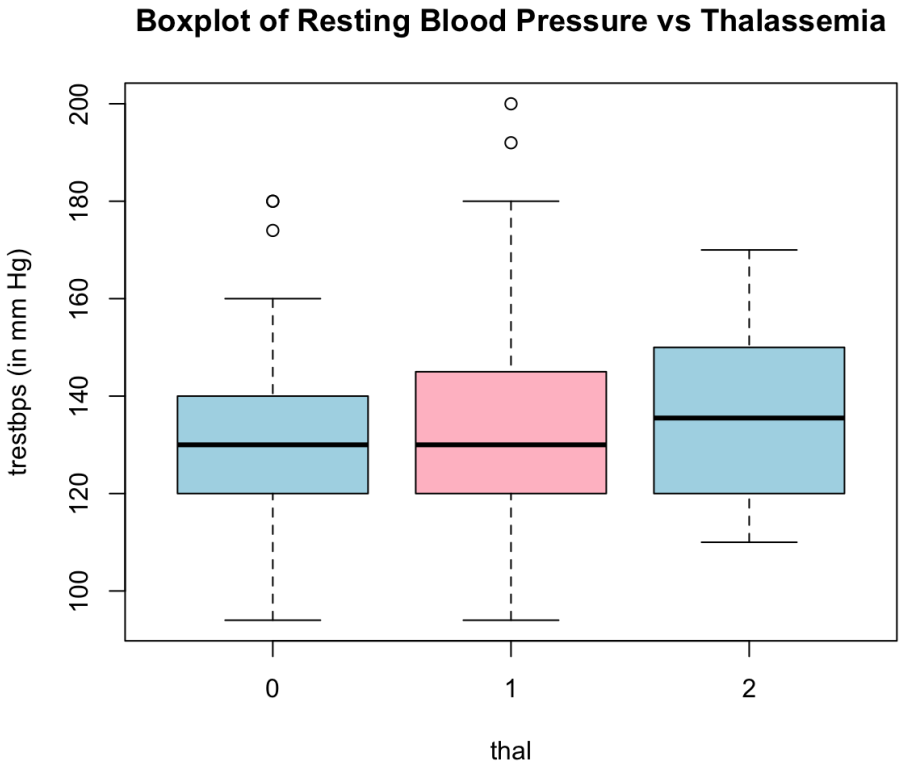
Boxplot of Resting Blood Pressure vs Thalassemia
trestbps (in mm Hg)
100
120
140
160
180
200
0
1
2
thal

Figure 4

```
Call:
glm(formula = train$num ~ age + sex + cp + trestbps + chol +
    fbs + restecg + thalch + thal, family = "binomial", data = train)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.238444   2.269166   0.105 0.916312
age          0.014314   0.023272   0.615 0.538494
sex1        -1.591229   0.479660  -3.317 0.000909 ***
cp1         -1.791715   0.423575  -4.230 2.34e-05 ***
cp2         -1.722054   0.564366  -3.051 0.002278 **
cp3         -2.147200   0.703593  -3.052 0.002275 **
trestbps     0.020490   0.010265   1.996 0.045931 *
chol         0.005864   0.003589   1.634 0.102301
fbs1         0.041146   0.495997   0.083 0.933887
restecg1     0.735015   0.371581   1.978 0.047921 *
restecg2     0.532584   1.935902   0.275 0.783233
thalch      -0.034647   0.009738  -3.558 0.000374 ***
thal1        1.782417   0.392735   4.538 5.67e-06 ***
thal2        0.313458   0.745860   0.420 0.674293
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 370.96  on 269  degrees of freedom
Residual deviance: 208.11  on 256  degrees of freedom
AIC: 236.11
```

Figure 5

```
Call:
glm(formula = train$num ~ sex + cp + trestbps + restecg + thalch +
    thal, family = "binomial", data = train)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.099508   1.722172   1.219  0.22280
sex1        -1.289129   0.439530  -2.933  0.00336 **
cp1         -1.780584   0.410851  -4.334 1.46e-05 ***
cp2         -1.664741   0.555239  -2.998  0.00272 **
cp3         -2.176427   0.685915  -3.173  0.00151 **
trestbps     0.023141   0.009912   2.335  0.01956 *
restecg1     0.876302   0.363656   2.410  0.01597 *
restecg2     0.520410   1.832427   0.284  0.77641
thalch      -0.035562   0.008837  -4.024 5.72e-05 ***
thal1        1.877224   0.391757   4.792 1.65e-06 ***
thal2        0.332619   0.736625   0.452  0.65160
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 370.96  on 269  degrees of freedom
Residual deviance: 211.63  on 259  degrees of freedom
AIC: 233.63
```

Figure 6

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 10  9
         1  0 10

               Accuracy : 0.6897
                 95% CI : (0.4917, 0.8472)
    No Information Rate : 0.6552
    P-Value [Acc > NIR] : 0.430578

                  Kappa : 0.4338

 Mcnemar's Test P-Value : 0.007661

            Sensitivity : 1.0000
            Specificity : 0.5263
         Pos Pred Value : 0.5263
         Neg Pred Value : 1.0000
             Prevalence : 0.3448
         Detection Rate : 0.3448
   Detection Prevalence : 0.6552
      Balanced Accuracy : 0.7632

       'Positive' Class : 0
```

Figure 7

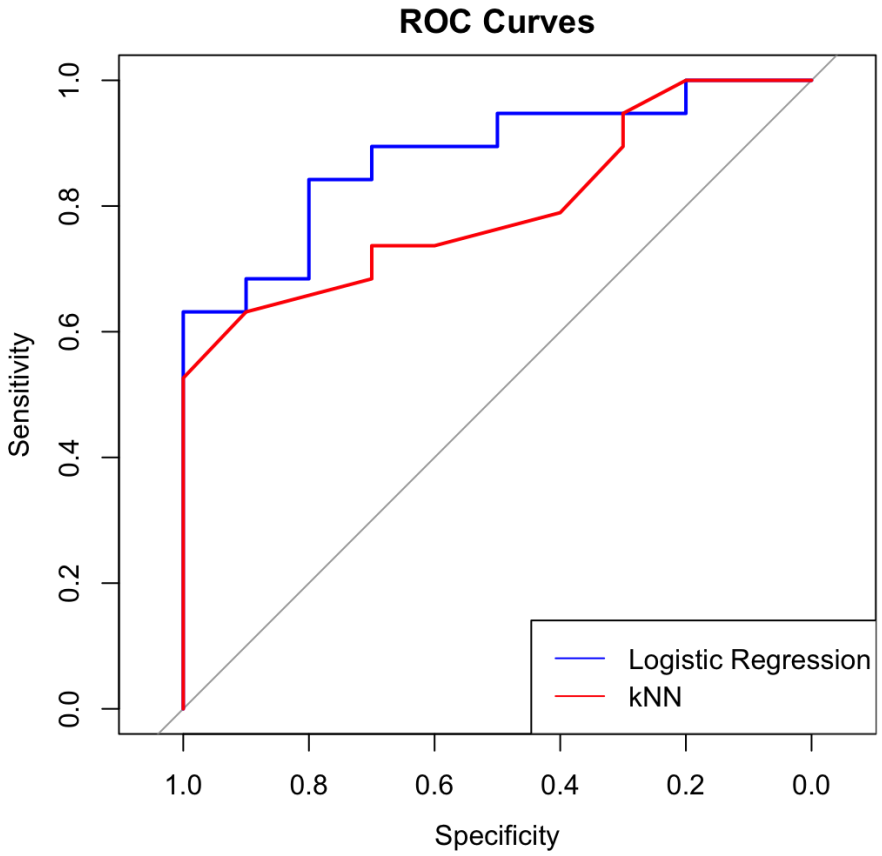

Figure 8

```
        DeLong's test for two ROC curves

data:  log_reg_roc and knn_roc
D = 0.78729, df = 52.746, p-value = 0.4346
alternative hypothesis: true difference in AUC is not equal to 0
sample estimates:
AUC of roc1 AUC of roc2
  0.8789474   0.7973684
```

Figure 9

STAT 202A Final Project Code

Anirudh Divecha

2024-12-10

Python Code Used for Data Cleaning

```
import pandas as pd
df = pd.read_csv("heart_disease_uci.csv")
df

df.describe()

df = df.dropna()
df.isnull().sum()

# drop columns I wont use
df = df.drop(columns=['id', 'dataset', 'exang', 'oldpeak', 'slope', 'ca'])
df

# Print all unique values
df['num'].unique()

# Get a frequency count of all values
df['num'].value_counts()

# replacing categorical data with numbers
df['sex'].replace(['Male', 'Female'],[0, 1], inplace=True)
df['cp'].replace(['asymptomatic', 'non-anginal', 'atypical angina', 'typical angina' ],[0, 1, 2, 3], inp
df['fbs'].replace(['False', 'True'],[0, 1], inplace=True)
df['restecg'].replace(['normal', 'lv hypertrophy', 'st-t abnormality'],[0, 1, 2], inplace=True)
df['thal'].replace(['normal', 'reversable defect', 'fixed defect'],[0, 1, 2], inplace=True)

df['fbs'] = df['fbs'].astype(int)
df['num'] = df['num'].apply(lambda x: 1 if x > 1 else x)

# check for null values
df.isnull().sum()

df = df.dropna()
df.isnull().sum()

df.describe()

df.to_csv('cleaned_heart.csv', index=False)
```

C Code for Kernel Density Estimation

```c
void gkr(double *x, double *y, int *n, double *b, double *g, int *m)
{
    int i,j;
    double sum = 0.0;
    double diff;
    for(i = 0; i < *m; i++)
    {

        sum = 0.0;
        for(j=0; j < *n; j++)
        {
            diff = pow(x[j] - g[i], 2);
            sum += exp((-1 * diff) / (2 * pow(*b, 2)));
        }

        y[i] = sum / (*n * sqrt(2 * M_PI * pow(*b, 2)));
    }
}
```

R Code for Final Project

```
# read in csv file checked by python script
heart_data = read.csv("cleaned_heart.csv")

num_table = table(heart_data$num)

# Plot a histogram of the frequencies
barplot(num_table,
        main = "Histogram of Response Variable 'num'",
        xlab = "'num' Levels",
        ylab = "Frequency",
        col = "skyblue",
        border = "black")



########################## Kernel Density Estimation
# lets do kernel density estimation on some of the continuous data predictors
system("R CMD SHLIB hw3.c")
dyn.unload("hw3.so")
dyn.load("hw3.so")

# lets examine the trestbps kernel density estimation
trestbps = as.numeric(heart_data[,4])
trestbps_bandwidth = bw.nrd(trestbps)

trestbps_grid = seq(min(trestbps),max(trestbps),length=100)
result = .C("gkr",x = trestbps, y = double(length(trestbps_grid)),
            n = as.integer(length(trestbps)), b = as.double(trestbps_bandwidth),
            g = trestbps_grid, m = as.integer(length(trestbps_grid)))


# lets compare men and women kernel density estimation
males_data = heart_data[heart_data$sex == 0, ]
females_data = heart_data[heart_data$sex == 1, ]

trestbps_m = as.numeric(males_data[,4])
trestbps_bandwidth_m = bw.nrd(trestbps_m)

trestbps_grid_m = seq(min(trestbps_m),max(trestbps_m),length=100)
result_trestbps_m = .C("gkr",x = trestbps_m, y = double(length(trestbps_grid_m)),
                       n = as.integer(length(trestbps_m)), b = as.double(trestbps_bandwidth_m),
                       g = trestbps_grid_m, m = as.integer(length(trestbps_grid_m)))

trestbps_f = as.numeric(females_data[,4])
trestbps_bandwidth_f = bw.nrd(trestbps_f)

trestbps_grid_f = seq(min(trestbps_f),max(trestbps_f),length=100)
result_trestbps_f = .C("gkr",x = trestbps_f, y = double(length(trestbps_grid_f)),
                       n = as.integer(length(trestbps_f)), b = as.double(trestbps_bandwidth_f),
                       g = trestbps_grid_f, m = as.integer(length(trestbps_grid_f)))
```

```r
plot(type = "l", ylim = c(0, 0.025), trestbps_grid, result$y,
     xlab="Resting Blood Pressure Magnitudes (in mm Hg)", ylab="Kernel Estimates",
     main="Kernel Density for Resting Blood Pressure Magnitudes", col="red")
lines(trestbps_grid_m, result_trestbps_m$y, col = "blue")
lines(trestbps_grid_f, result_trestbps_f$y, col = "green")
legend("topright", legend = c("Male and Female", "Male Only", "Female Only"),
       col = c("red", "blue", "green"), lwd = 2)


# now lets take a look at the Thalach kernel density estimation
thalch = as.numeric(heart_data[,8])
thalch_bandwidth = bw.nrd(thalch)

thalch_grid = seq(min(thalch),max(thalch),length=100)
result_thalch = .C("gkr",x = thalch, y = double(length(thalch_grid)),
                   n = as.integer(length(thalch)), b = as.double(thalch_bandwidth),
                   g = thalch_grid, m = as.integer(length(thalch_grid)))


# lets compare men and women kernel density estimation
thalch_m = as.numeric(males_data[,8])
thalch_bandwidth_m = bw.nrd(thalch_m)

thalch_grid_m = seq(min(thalch_m),max(thalch_m),length=100)
result_thalch_m = .C("gkr",x = thalch_m, y = double(length(thalch_grid_m)),
                     n = as.integer(length(thalch_m)), b = as.double(thalch_bandwidth_m),
                     g = thalch_grid_m, m = as.integer(length(thalch_grid_m)))

thalch_f = as.numeric(females_data[,8])
thalch_bandwidth_f = bw.nrd(thalch_f)

thalch_grid_f = seq(min(thalch_f),max(thalch_f),length=100)
result_thalch_f = .C("gkr",x = thalch_f, y = double(length(thalch_grid_f)),
                     n = as.integer(length(thalch_f)), b = as.double(thalch_bandwidth_f),
                     g = thalch_grid_f, m = as.integer(length(thalch_grid_f)))

plot(type = "l", ylim = c(0, 0.025), thalch_grid, result_thalch$y,
     xlab="thalch Magnitudes", ylab="Kernel Estimates",
     main="Kernel Density for thalch Magnitudes", col="red")
lines(thalch_grid_m, result_thalch_m$y, col = "blue")
lines(thalch_grid_f, result_thalch_f$y, col = "green")
legend("topleft", legend = c("Male and Female", "Male Only", "Female Only"),
       col = c("red", "blue", "green"), lwd = 1)


# Cholesterol Kernel Density Estimation
cholesterol = as.numeric(heart_data[,5])
cholesterol_bandwidth = bw.nrd(cholesterol)

cholesterol_grid = seq(min(cholesterol),max(cholesterol),length=100)
result_cholesterol = .C("gkr",x = cholesterol, y = double(length(cholesterol_grid)),
                        n = as.integer(length(cholesterol)), b = as.double(cholesterol_bandwidth),
                        g = cholesterol_grid, m = as.integer(length(cholesterol_grid)))
```

```r
# lets compare men and women kernel density estimation
cholesterol_m = as.numeric(males_data[,5])
cholesterol_bandwidth_m = bw.nrd(cholesterol_m)

cholesterol_grid_m = seq(min(cholesterol_m),max(cholesterol_m),length=100)
result_cholesterol_m = .C("gkr",x = cholesterol_m, y = double(length(cholesterol_grid_m)),
                          n = as.integer(length(cholesterol_m)), b = as.double(cholesterol_bandwidth_m)
                          g = cholesterol_grid_m, m = as.integer(length(cholesterol_grid_m)))

cholesterol_f = as.numeric(females_data[,5])
cholesterol_bandwidth_f = bw.nrd(cholesterol_f)

cholesterol_grid_f = seq(min(cholesterol_f),max(cholesterol_f),length=100)
result_cholesterol_f = .C("gkr",x = cholesterol_f, y = double(length(cholesterol_grid_f)),
                          n = as.integer(length(cholesterol_f)), b = as.double(cholesterol_bandwidth_f)
                          g = cholesterol_grid_f, m = as.integer(length(cholesterol_grid_f)))

plot(type = "l", ylim = c(0, 0.010), cholesterol_grid, result_cholesterol$y,
     xlab="Cholesterol Magnitudes (in mg/dl)", ylab="Kernel Estimates",
     main="Kernel Density for Cholesterol Magnitudes", col="red")
lines(cholesterol_grid_m, result_cholesterol_m$y, col = "blue")
lines(cholesterol_grid_f, result_cholesterol_f$y, col = "green")
legend("topright", legend = c("Male and Female", "Male Only", "Female Only"),
       col = c("red", "blue", "green"), lwd = 1)
```

```r
########################## Box Plots
par(mfrow = c(1, 1))
boxplot(trestbps ~ thal, data = heart_data,
        main = "Boxplot of Resting Blood Pressure vs Thalassemia",
        xlab = "thal", ylab = "trestbps (in mm Hg)",
        col = c("lightblue", "pink"))
```

```r
########################## Logistic Regression
# do an 90-10 split of data
num_rows=nrow(heart_data)
num_rows


train = heart_data[1:270,]
test = heart_data[271:299,]

# treat categorical variables as factors
```

```r
train$sex = factor(train$sex)
train$cp = factor(train$cp)
train$fbs = factor(train$fbs)
train$restecg = factor(train$restecg)
train$thal = factor(train$thal)

# fit logistic model on training data
hlm = glm(train$num ~ age + sex + cp + trestbps + chol + fbs + restecg + thalch + thal,
          data = train, family = "binomial")

summary(hlm)

test$sex = factor(test$sex)
test$cp = factor(test$cp)
test$fbs = factor(test$fbs)
test$restecg = factor(test$restecg)
test$thal = factor(test$thal)

# obtain initial accuracy results
fitted.results = predict(hlm,newdata=test,type='response')
fitted.results = ifelse(fitted.results > 0.5,1,0)
misClasificError = mean(fitted.results != test$num)
print(paste('Accuracy',1-misClasificError))

# lets refit the model with significant terms
hlm_1 = glm(train$num ~ sex + cp + trestbps + restecg + thalch + thal,
          data = train, family = "binomial")

summary(hlm_1)

# obtain final accuracy results
fitted_1.results = predict(hlm_1,newdata=test,type='response')
fitted_1.results = ifelse(fitted_1.results > 0.5,1,0)
misClasificError_1 = mean(fitted_1.results != test$num)
print(paste('Accuracy',1-misClasificError_1))




################################ KNN Model
# Lets try the KNN model
library(ggplot2)
library(caret)

train_knn = heart_data[1:270,]
test_knn = heart_data[271:299,]

# for caret package outcome
train_knn$num = as.factor(train_knn$num)
test_knn$num = as.factor(test_knn$num)

set.seed(123)
```

```r
knn_model = train(num ~ age + sex + cp + trestbps + chol + fbs + restecg + thalch + thal,
                  method = "knn",
                  data = train_knn,
                  tuneGrid = data.frame(k = seq(9, 71, 2)))
ggplot(knn_model, highlight = TRUE)

predictions = predict(knn_model, test_knn)
confusionMatrix(predictions, test_knn$num)

# The KNN performed slightly worse



################################ ROC Curves to Analyze the Models
library(pROC)
# Predicted probabilities from logistic regression
log_reg_probs = predict(hlm,newdata=test,type='response')

# Generate the ROC curve
log_reg_roc = roc(test$num, log_reg_probs)
plot(log_reg_roc, col = "blue", main = "ROC Curves")


# Predicted probabilities from k-NN
knn_probs = predict(knn_model, newdata = test_knn, type = "prob")[, 2]

# Generate the ROC curve
knn_roc = roc(test_knn$num, knn_probs)
plot(knn_roc, col = "red", add = TRUE)

legend("bottomright", legend = c("Logistic Regression", "kNN"), col = c("blue", "red"), lwd = 1)
# we see from the ROC curves that the logistic regression line
# is farther to the left showing better performance than the kNN model.

roc_test = roc.test(log_reg_roc, knn_roc)

# Print results
print(roc_test)

# the ROC.test is DeLong's test and we see that the logistic regression model
# performs better by the p-value is 0.43, this indicates however that the
# difference between the two models is not statistically significant.
```