MATH 4323, Project: Final Report
Using Supervised Learning to Predict Heart Disease

Members: Nathalie Martinez (1794003), David Silva (1776311), Keyz Chester (1828969), Sri Ganesh Ganta (2063306).

1. Introduction

This data was sourced from Kaggle, which describes itself as a machine learning and data science community. One of its many features is that it allows users access to workable datasets. We chose this particular one as heart failure is extremely prevalent in the United States and affects many of those around us. According to the CDC, 6.2 million adults will experience heart failure.1 It also costs the nation billions in health care costs.

The data contains 918 observations and 12 variables. There are 11 predictors that can be used to predict a possible heart failure episode.
The dataset contains the following features:

- Age: The age of the patient (years)
- Sex: The sex of the patient (M=Male, F=Female)
- ChestPainType: The type of chest pain (TA=Typical Angina, ATA= Atypical Angina, NAP=Non-Anginal Pain, ASY=Asymptomatic)
- Resting BP: The resting blood pressure (mm Hg)
- Cholesterol: The cholesterol (mm/di)
- FastingBS: The fasting blood sugar (1=if FastingBS>120 mg/dl, 0=otherwise)
- RestingECG: The resting electrocardiogram results (Normal, ST, LVH)
- MaxHR: The maximum heart rate achieved (Values are between 60 and 202)
- ExerciseAngina: The exercise-induced angina (Y=Yes, N=No)
- Oldpeak: ST depression induced by exercise relative to rest
- ST_Slope: The slope of the peak exercise ST segment (Up, Flat, Down)
- HeartDisease: The output (1=heart disease, 0=no heart disease

We will use this data to answer the research question: can we accurately predict heart disease based on this data?

2. Methodology

Our goal for this project is to accurately predict variable y: heart failure based on the 11 predictors. We will use supervised learning to perform the classification task.
Step-1) We will explore the data by observing the number of observations, seeing if there are any missing values, and checking the dimensions of the data set.
Step-2) The data will be charted in a variety of ways, including but not limited to scatter plots, bar graphs, and box-and-whisker plots in order to find obvious relations between variables. What we learn from this will direct the creation of our models.
Step-3) We will convert our categorical variables into dummy variables, scale the data, and split the data into a training and testing set.

Step-4) Using the KNN and SVM methods, we will proceed to build our models. What we learned during visualization and pre-processing will inform the selection of predictor variables, and we will use cross-validation techniques to tune our models.
Step 5)The last step is choosing which method will best predict whether or not a person is diagnosed with heart disease. This should be fairly straightforward; we just select the one with the lowest test error rate.. This is made simpler by the fact that the information we are trying to predict is binary; either the patient does have a heart condition, or they do not.

KNN Method:

The KNN (K-Nearest Neighbors) method assumes that observations close to one another are similar. We select a K number of neighbors around a random data point X. We calculate the Euclidean distance between X and each K number of closest neighbors around X. KNN then classifies X into a group based on the states of the nearest neighbors around it. If there are more nearest neighbors that are in group A then group B for example, then X will be in group A. Advantages of KNN are that it is fairly quick and simple, as it does not require us to tune various parameters. Some disadvantages are that it can be sensitive to noise, and it can become slow as the data set gets large.

We will begin by splitting the data into subgroups based on the categorical variables. Once this is complete, we will select predictors based on information gained in earlier steps. Then, we will split the data into testing and training sets and build our model, performing cross-validation in order to fine-tune the value of k. We will repeat this with different sets of predictors so as to find the best model, which would be the one with the lowest rate of false negatives, as while it would be inconvenient to a patient to misdiagnose them with a heart problem when they are healthy, it would be catastrophic to conclude that someone is healthy when they are not.

SVM Method:

The three kernels we will be generating are linear, polynomial and radial which will be less computationally demanding than working in the enlarged feature space. The hyperplane equations are as follows:

Linear Kernel: $\mathbf{K(xi\,,\,xj)} = \sum_{k=1}^{p} \mathbf{x_{ik}\ x_{jk}}$

Polynomial Kernel: $\mathbf{K(xi\,,\,xj)} = (1 + \sum_{k=1}^{p} \mathbf{x_{ik}\ x_{jk}})^{\mathbf{d}}$

Radial Kernel: $\mathbf{K(xi\,,\,xj)} = \mathbf{exp}(-\ \gamma \sum_{k=1}^{p} (\mathbf{x_{ik}\ x_{jk}})^2),\ \gamma > \mathbf{0}$

This model is known to be one of the best classifiers. Advantages include being less computationally demanding and less sensitive to noise. Disadvantages include difficulty in finding the right kernel, and finding optimal values can be quite slow. To perform SVM, we will split the data into a training set and a testing set. We will perform cross-validation to find optimal values for the hyperparameters involved in each kernel. We will then find test errors and select the kernel with the lowest error rate.

3. **Data Analysis**

a) Considerations

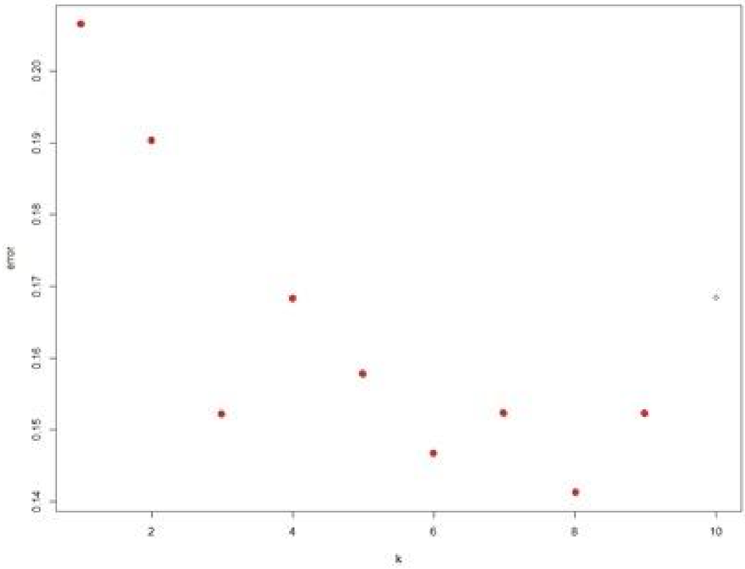
We did not exclude any predictors from the analysis because they all appeared to be relevant. The sex, age, and various pre-existing health conditions of a patient could all contribute to developing heart disease. We believed it would be useful to scale the data to prevent certain variables from being overrepresented. Before starting the KNN and SVM procedures, we are going to convert all categorical variables into qualitative variables by assigning them numbers such as 0, 1, 2, or 3. The dataset with all quantitative variables will be named "heart2."

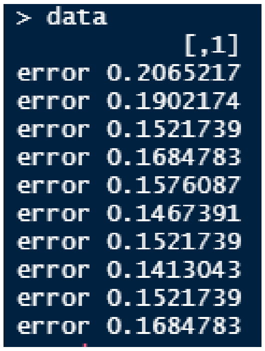
b) Optimal Parameters

K - Nearest Method

Since many of the variables are measured on different scales, all of the data will be scaled during the KNN procedure except the last column which is the Y variable. K - Nearest Method has a different approach for cross validation. We are going to split both of our Predictor and Target variables into train and test data. From the plot and data object which recorded the errors for each K value, the most optimal K value is 8 with a miscalculation error of 14.13%.

```
data = NULL
for (K in c(1:10)) {
  set.seed(1)
  knn.pred = knn(train = X.train, test = X.test, cl = y.train, k=K)
  error = mean(knn.pred != y.test)
  data = rbind(data,error)
}
plot(data, xlab="k", ylab="error")
```

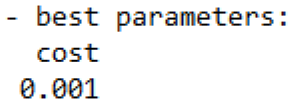




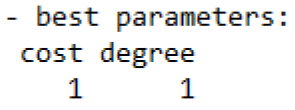
```
> data
           [,1]
error 0.2065217
error 0.1902174
error 0.1521739
error 0.1684783
error 0.1576087
error 0.1467391
error 0.1521739
error 0.1413043
error 0.1521739
error 0.1684783
```

SVM Method

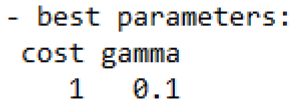
Like in the KNN procedure, we also converted all qualitative variables into numerical variables and used them to create the dataset “heart2”. We did not remove any of the variables. Using cross validation, we found the optimal values for linear, polynomial, and radial kernels.



```
- best parameters:
  cost
 0.001
```

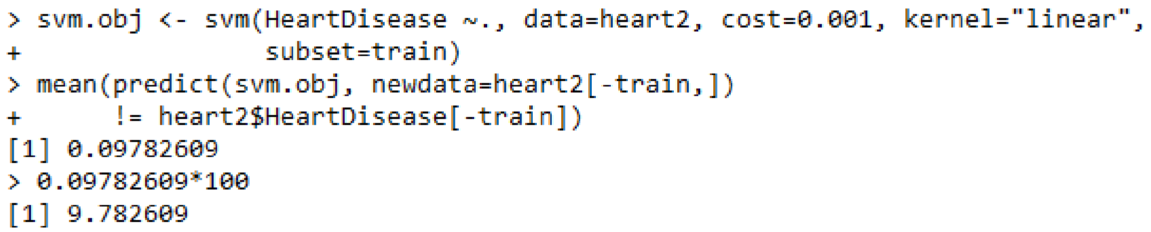


```
- best parameters:
 cost degree
    1      1
```



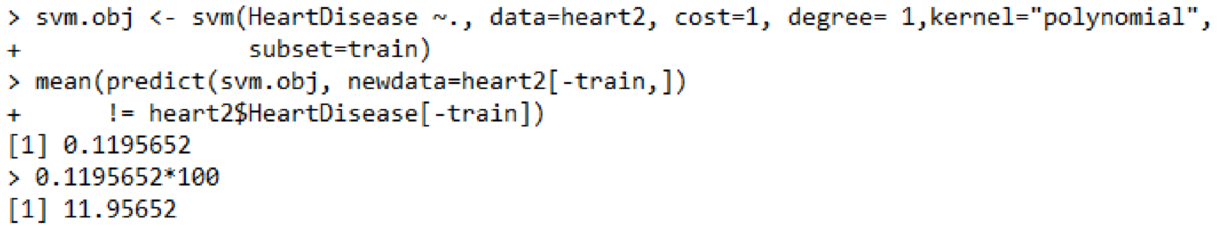
```
- best parameters:
 cost gamma
    1   0.1
```

Linear test error:



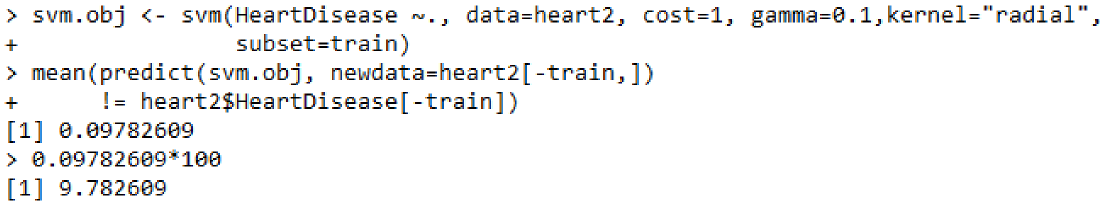
```
> svm.obj <- svm(HeartDisease ~., data=heart2, cost=0.001, kernel="linear",
+                 subset=train)
> mean(predict(svm.obj, newdata=heart2[-train,])
+      != heart2$HeartDisease[-train])
[1] 0.09782609
> 0.09782609*100
[1] 9.782609
```

Polynomial test error:



```
> svm.obj <- svm(HeartDisease ~., data=heart2, cost=1, degree= 1,kernel="polynomial",
+                 subset=train)
> mean(predict(svm.obj, newdata=heart2[-train,])
+      != heart2$HeartDisease[-train])
[1] 0.1195652
> 0.1195652*100
[1] 11.95652
```

Radial test error:



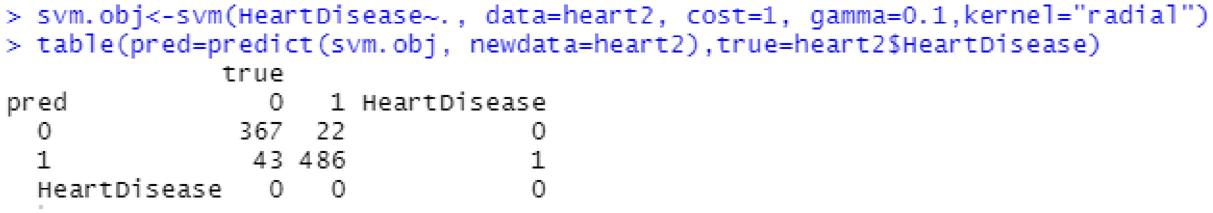
```
> svm.obj <- svm(HeartDisease ~., data=heart2, cost=1, gamma=0.1,kernel="radial",
+                 subset=train)
> mean(predict(svm.obj, newdata=heart2[-train,])
+      != heart2$HeartDisease[-train])
[1] 0.09782609
> 0.09782609*100
[1] 9.782609
```

c) Test Errors

To find test errors we split the data into 80% and 20% training and test subsets respectively. We trained the SVM model and calculated errors for all three kernels. Both linear and radial kernels had the best error rate of ~9.78%. However, we decided to go with the radial model with values of cost=1 and gamma=0.1 as it provided the most flexible boundaries. The error rate of 9.78% means that this was a clear choice over KNN with an error of 14.13%.

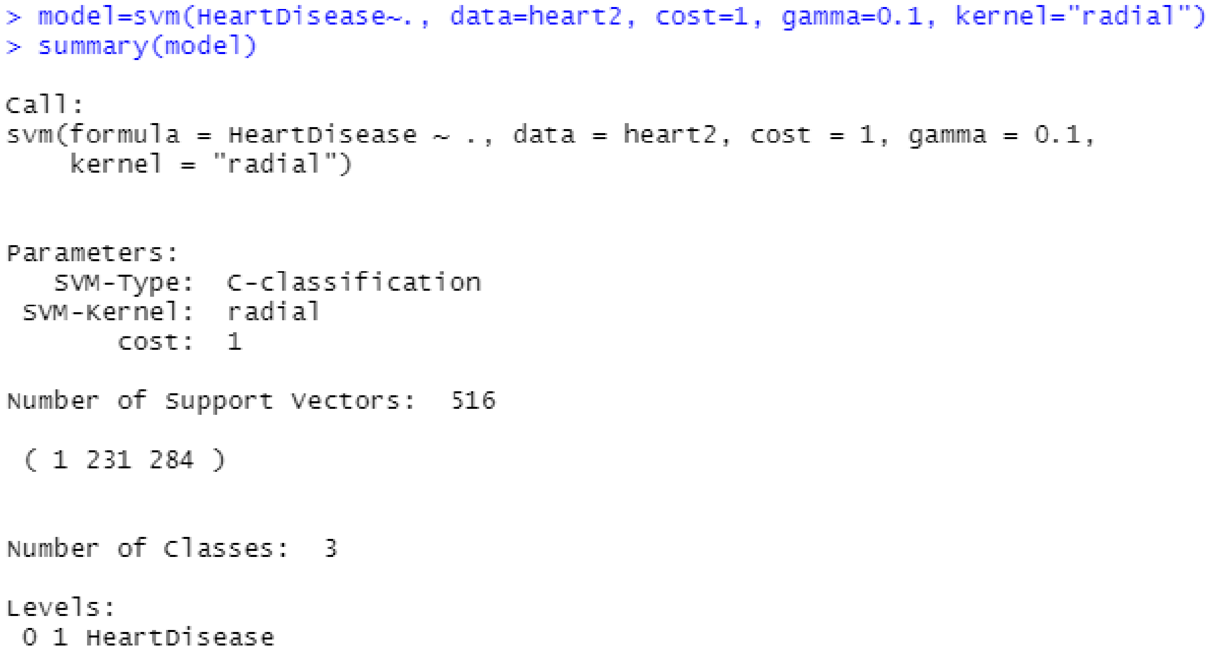
d) Best Model

Using the radial SVM, we fit it to the complete data set and obtained the following results:

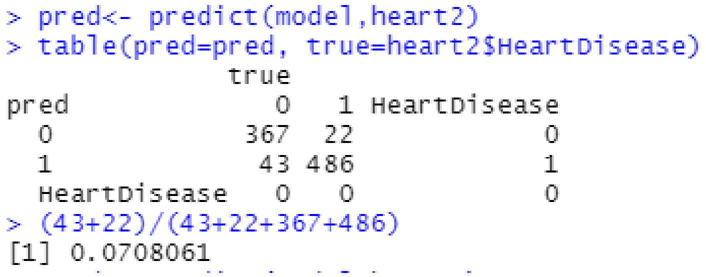


```
> svm.obj<-svm(HeartDisease~., data=heart2, cost=1, gamma=0.1,kernel="radial")
> table(pred=predict(svm.obj, newdata=heart2),true=heart2$HeartDisease)
              true
pred             0   1 HeartDisease
  0            367  22            0
  1             43 486            1
  HeartDisease   0   0            0
```

Here, we have provided the summary of the svm object:

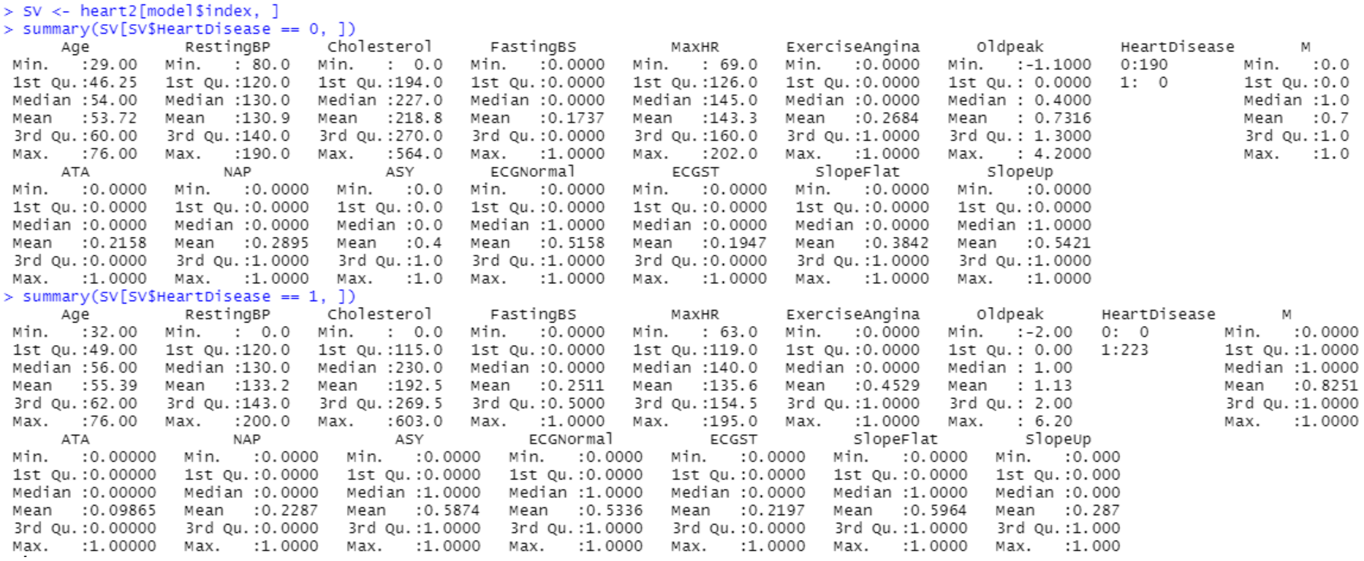


```
> model=svm(HeartDisease~., data=heart2, cost=1, gamma=0.1, kernel="radial")
> summary(model)

Call:
svm(formula = HeartDisease ~ ., data = heart2, cost = 1, gamma = 0.1,
    kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  516

 ( 1 231 284 )


Number of Classes:  3

Levels:
 0 1 HeartDisease
```



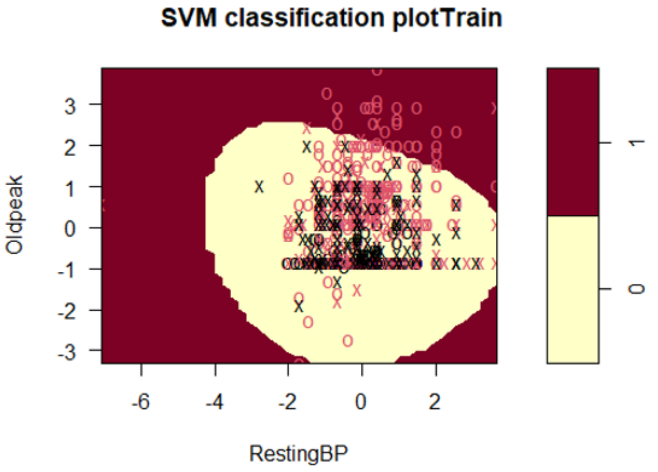
```
> pred<- predict(model,heart2)
> table(pred=pred, true=heart2$HeartDisease)
              true
pred             0   1 HeartDisease
  0            367  22            0
  1             43 486            1
  HeartDisease   0   0            0
> (43+22)/(43+22+367+486)
[1] 0.0708061
```

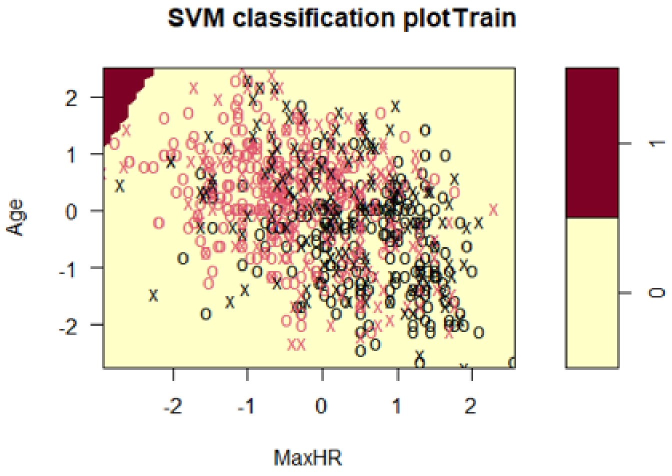
Support vectors:



```
> SV <- heart2[model$index, ]
> summary(SV[SV$HeartDisease == 0, ])
      Age          RestingBP      Cholesterol      FastingBS          MaxHR       ExerciseAngina      Oldpeak         HeartDisease       M      
 Min.   :29.00   Min.   : 80.0   Min.   :  0.0   Min.   :0.0000   Min.   : 69.0   Min.   :0.0000   Min.   :-1.1000   0:190         Min.   :0.0  
 1st Qu.:46.25   1st Qu.:120.0   1st Qu.:194.0   1st Qu.:0.0000   1st Qu.:126.0   1st Qu.:0.0000   1st Qu.: 0.0000   1:  0         1st Qu.:0.0  
 Median :54.00   Median :130.0   Median :227.0   Median :0.0000   Median :145.0   Median :0.0000   Median : 0.4000                 Median :1.0  
 Mean   :53.72   Mean   :130.9   Mean   :218.8   Mean   :0.1737   Mean   :143.3   Mean   :0.2684   Mean   : 0.7316                 Mean   :0.7  
 3rd Qu.:60.00   3rd Qu.:140.0   3rd Qu.:270.0   3rd Qu.:0.0000   3rd Qu.:160.0   3rd Qu.:1.0000   3rd Qu.: 1.3000                 3rd Qu.:1.0  
 Max.   :76.00   Max.   :190.0   Max.   :564.0   Max.   :1.0000   Max.   :202.0   Max.   :1.0000   Max.   : 4.2000                 Max.   :1.0  
      ATA              NAP              ASY         ECGNormal          ECGST          SlopeFlat         SlopeUp      
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000  
 Median :0.0000   Median :0.0000   Median :0.0   Median :1.0000   Median :0.0000   Median :0.0000   Median :1.0000  
 Mean   :0.2158   Mean   :0.2895   Mean   :0.4   Mean   :0.5158   Mean   :0.1947   Mean   :0.3842   Mean   :0.5421  
 3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.0   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.0000  
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000  
> summary(SV[SV$HeartDisease == 1, ])
      Age          RestingBP      Cholesterol      FastingBS          MaxHR       ExerciseAngina      Oldpeak      HeartDisease       M         
 Min.   :32.00   Min.   :  0.0   Min.   :  0.0   Min.   :0.0000   Min.   : 63.0   Min.   :0.0000   Min.   :-2.00   0:  0        Min.   :0.0000  
 1st Qu.:49.00   1st Qu.:120.0   1st Qu.:115.0   1st Qu.:0.0000   1st Qu.:119.0   1st Qu.:0.0000   1st Qu.: 0.00   1:223        1st Qu.:1.0000  
 Median :56.00   Median :130.0   Median :230.0   Median :0.0000   Median :140.0   Median :0.0000   Median : 1.00                Median :1.0000  
 Mean   :55.39   Mean   :133.2   Mean   :192.5   Mean   :0.2511   Mean   :135.6   Mean   :0.4529   Mean   : 1.13                Mean   :0.8251  
 3rd Qu.:62.00   3rd Qu.:143.0   3rd Qu.:269.5   3rd Qu.:0.5000   3rd Qu.:154.5   3rd Qu.:1.0000   3rd Qu.: 2.00                3rd Qu.:1.0000  
 Max.   :76.00   Max.   :200.0   Max.   :603.0   Max.   :1.0000   Max.   :195.0   Max.   :1.0000   Max.   : 6.20                Max.   :1.0000  
      ATA               NAP              ASY           ECGNormal          ECGST          SlopeFlat         SlopeUp     
 Min.   :0.00000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000  
 1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000  
 Median :0.00000   Median :0.0000   Median :1.0000   Median :1.0000   Median :0.0000   Median :1.0000   Median :0.000  
 Mean   :0.09865   Mean   :0.2287   Mean   :0.5874   Mean   :0.5336   Mean   :0.2197   Mean   :0.5964   Mean   :0.287  
 3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.000  
 Max.   :1.00000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000  
```

Plot of the fitted boundary:



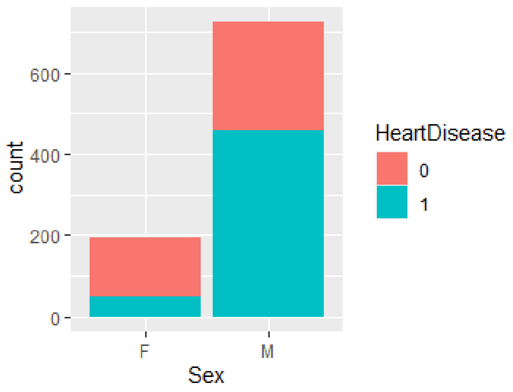


e) Results

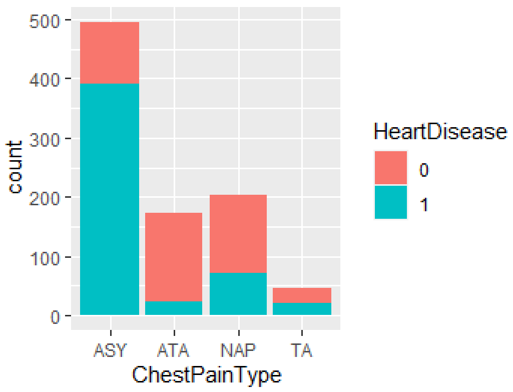
The radial SVM model shows that out of the 529 people that were predicted to have heart disease, 486 or 91.87% actually had it. Out of the 389 people who were predicted to not have the disease, 5.66% actually had it. The error rate was very low at 7.08%, and we had more false positives than false negatives, which is important when predicting disease. This ensures that almost everyone who can suffer serious health effects or death are identified and can receive treatment.

4. Conclusion

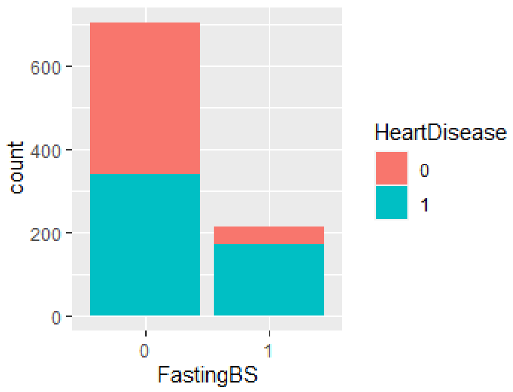
Based on the data analysis, we came to the conclusion that the data was successful in predicting heart disease. The low false negative rate of ~7% means that we are confident using this model to predict disease in a patient. Additionally we found that certain predictors were more highly correlated with a positive diagnosis. Old age, being male, high fasting blood sugar, angina, and a flat ST segment on an electrocardiogram reading were all major risk factors. Strangely, the data set had an unusual number of people reporting a value of 0 cholesterol which would lead to dangerous effects such as brain bleeding. Excluding this variable might improve our prediction results in a future analysis.

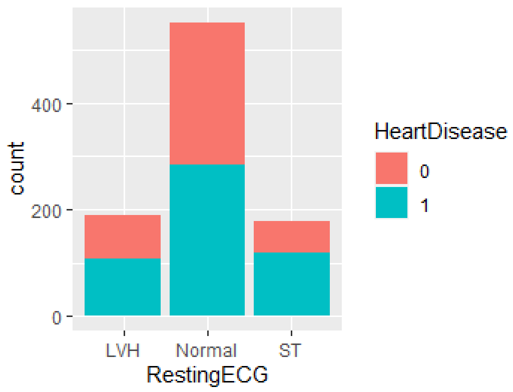


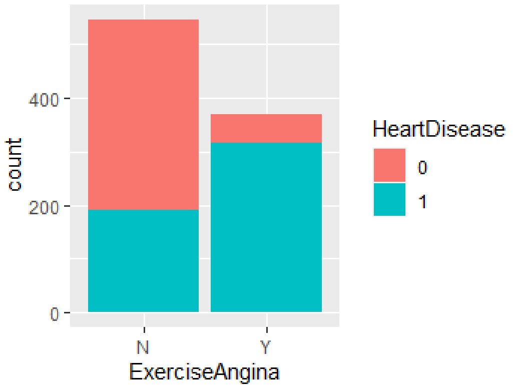
Males have a higher chance of getting heart disease.

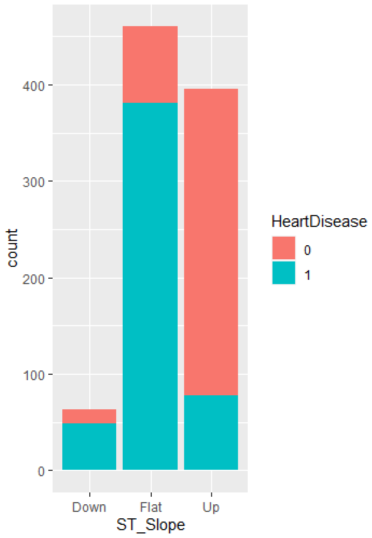


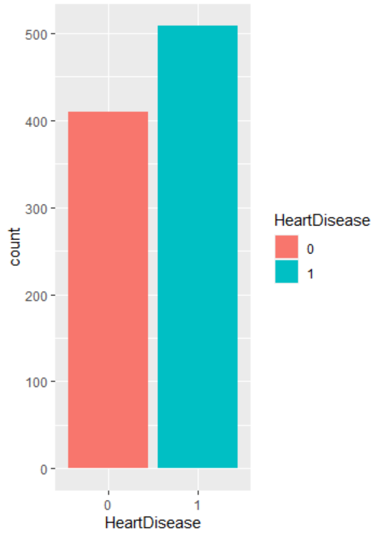
Asymptomatic (ASY) chest pain has a higher chance of heart disease.

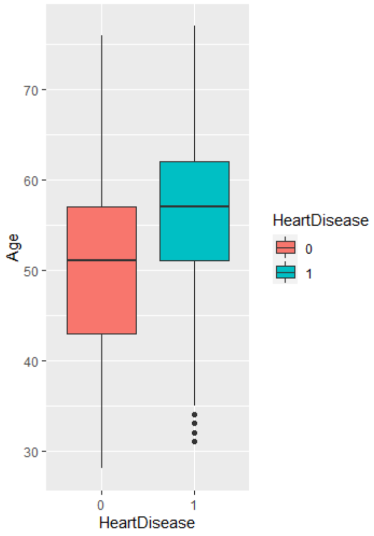


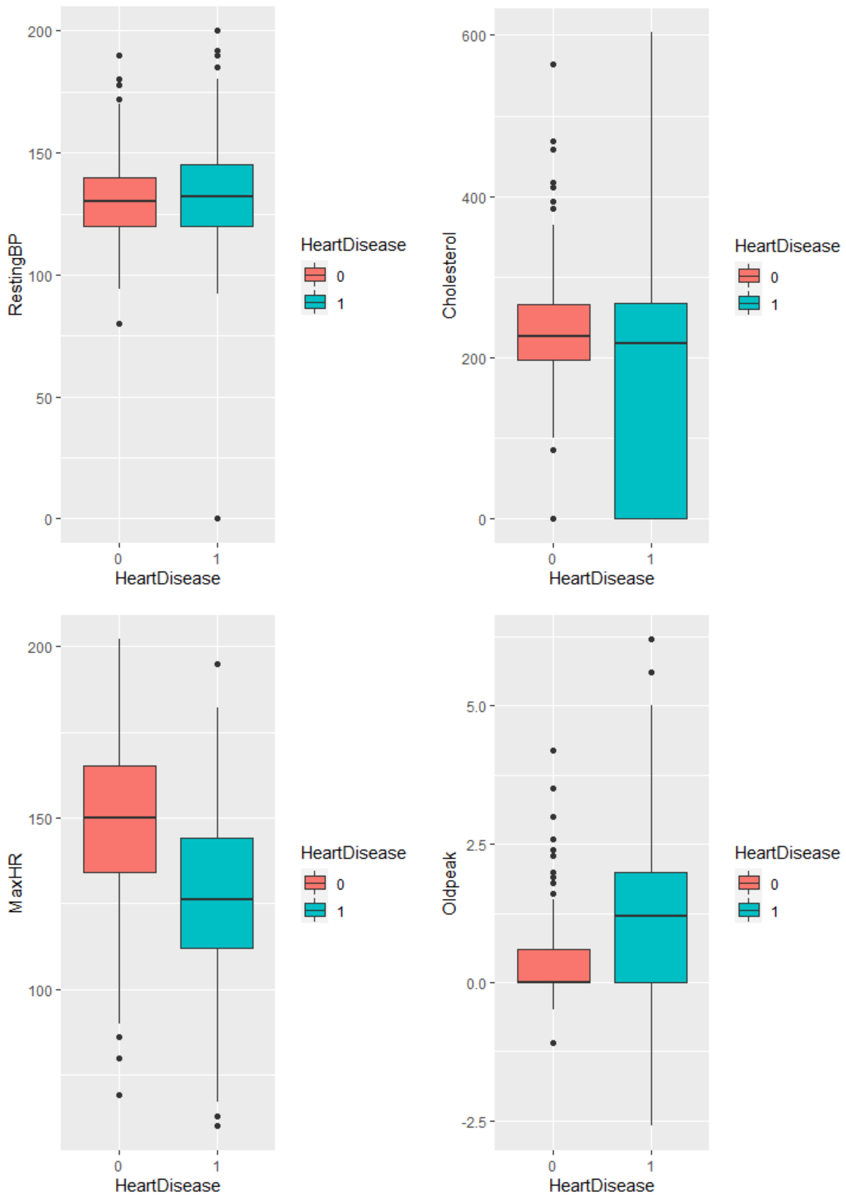



count
400
200
0
N
Y
ExerciseAngina
HeartDisease
0
1


count
400
300
200
100
0
Down
Flat
Up
ST_Slope
HeartDisease
0
1


count
500
400
300
200
100
0
0
1
HeartDisease
HeartDisease
0
1


Age
70
60
50
40
30
0
1
HeartDisease
HeartDisease
0
1


RestingBP
200
150
100
50
0
0
1
HeartDisease
HeartDisease
0
1
Cholesterol
600
400
200
0
0
1
HeartDisease
HeartDisease
0
1
MaxHR
200
150
100
0
1
HeartDisease
HeartDisease
0
1
Oldpeak
5.0
2.5
0.0
-2.5
0
1
HeartDisease
HeartDisease
0
1

5. References

1) https://www.kaggle.com/fedesoriano/heart-failure-prediction

2) Heart Failure | cdc.gov. Centers for Disease Control and Prevention. https://www.cdc.gov/heartdisease/heart_failure.htm. Published 2021. Accessed October 23, 2021

6. Appendix

Code for converting qualitative variables into numerical:

```
# F=0, M=1
n_Sex <- ifelse(heart$Sex == "M", 1, 0)
# TA=0, ATA=1, NAP=2, ASY=3
n_ChestPainType <- ifelse(heart$ChestPainType == "TA", 0,
                   ifelse(heart$ChestPainType == "ATA", 1,
                      ifelse(heart$ChestPainType == "NAP", 2, 3)))
# Normal=0, ST=1, LVH=2
n_RestingECG <- ifelse(heart$RestingECG == "Normal", 0,
                ifelse(heart$RestingECG == "ST", 1, 2))
# No=0, Y=1
n_ExerciseAngina <- ifelse(heart$ExerciseAngina == "Y", 1, 0)
# Up=0, Flat=1, Down=2
n_ST_Slope <- ifelse(heart$ST_Slope == "Up", 0,
              ifelse(heart$ST_Slope == "Flat", 1, 2))
```

Code for creating new dataset "heart2":

```
heart2<-heart
heart2[,2]=n_Sex
heart2[,3]=n_ChestPainType
heart2[,7]=n_RestingECG
heart2[,9]=n_ExerciseAngina
heart2[,11]=n_ST_Slope
summary(heart2)
```

Code for splitting into training/testing and scaling data:

```
n <- nrow(heart2)
RNGkind(sample.kind = "Rounding")
set.seed(1)
train <- sample(1:n, n*.8)

X.train <- scale(heart2[train, -12])
X.test <- scale(heart2[-train, -12], center = attr(X.train, "scaled:center"),
           scale = attr(X.train, "scaled:scale"))
y.train <- heart2$HeartDisease[train]
y.test <- heart2$HeartDisease[-train]
```

Code for finding optimal parameters for all 3 kernels:

```
> set.seed(1)
> n <- nrow(heart2); train <- sample(1:n, 0.8*n)

> set.seed(1)
> tune.out=tune(method=svm,
+         HeartDisease~.,data=heart2,
+         kernel="linear",
+         ranges=list(cost=c(0.001, 0.01, 0.1, 1,5,10,100)))
```

```
> summary(tune.out)
Parameter tuning of ‘svm’:
- sampling method: 10-fold cross validation
- best parameters:
  cost
 0.001

- best performance: 0.1567129

- Detailed performance results:
   cost     error dispersion
1 1e-03 0.1567129 0.03747688
2 1e-02 0.1588629 0.03403420
3 1e-01 0.1620999 0.03478088
4 1e+00 0.1838748 0.04382382
5 5e+00 0.2307095 0.03878122
6 1e+01 0.2448877 0.05290966
7 1e+02 0.2666388 0.05644958

> svm.obj <- svm(HeartDisease ~., data=heart2, cost=0.001, kernel="linear",
+                subset=train)
> mean(predict(svm.obj, newdata=heart2[-train,])
+      != heart2$HeartDisease[-train])
[1] 0.09782609
> 0.09782609*100
[1] 9.782609
*error for polynomial kernel with cost 1 degree1 = 11.96%*

> set.seed(1)
> tune.out=tune(method=svm,
+               HeartDisease~.,data=heart2,
+               kernel="polynomial",
+               ranges=list(cost=c(0.001, 0.01, 0.1, 1,5,10,100), degree=c(0.1,1,10,100)))
> summary(tune.out)
Parameter tuning of ‘svm’:
- sampling method: 10-fold cross validation
- best parameters:
 cost degree
    1      1
- best performance: 0.156689
- Detailed performance results:
    cost degree     error dispersion
1  1e-03    0.1 0.4472527 0.04652611
2  1e-02    0.1 0.4472527 0.04652611
3  1e-01    0.1 0.4472527 0.04652611
```

```
4  1e+00    0.1 0.4472527 0.04652611
5  5e+00    0.1 0.4472527 0.04652611
6  1e+01    0.1 0.4472527 0.04652611
7  1e+02    0.1 0.4472527 0.04652611
8  1e-03    1.0 0.4472527 0.04652611
9  1e-02    1.0 0.4472527 0.04652611
10 1e-01    1.0 0.4472527 0.04652611
11 1e+00    1.0 0.1566890 0.03404606
12 5e+00    1.0 0.1588629 0.03403420
13 1e+01    1.0 0.1599498 0.03029339
14 1e+02    1.0 0.1686216 0.03758219
15 1e-03   10.0 0.4472527 0.04652611
16 1e-02   10.0 0.4472527 0.04652611
17 1e-01   10.0 0.4472527 0.04652611
18 1e+00   10.0 0.4472527 0.04652611
19 5e+00   10.0 0.4472527 0.04652611
20 1e+01   10.0 0.4472527 0.04652611
21 1e+02   10.0 0.4472527 0.04652611
22 1e-03  100.0 0.4472527 0.04652611
23 1e-02  100.0 0.4472527 0.04652611
24 1e-01  100.0 0.4472527 0.04652611
25 1e+00  100.0 0.4472527 0.04652611
26 5e+00  100.0 0.4472527 0.04652611
27 1e+01  100.0 0.4472527 0.04652611
28 1e+02  100.0 0.4472527 0.04652611
```

```
> svm.obj <- svm(HeartDisease ~., data=heart2, cost=1, degree= 1,kernel="polynomial",
+                subset=train)
> mean(predict(svm.obj, newdata=heart2[-train,])
+      != heart2$HeartDisease[-train])
[1] 0.1195652
> 0.1195652*100
[1] 11.95652
```

error for radial kernel with cost 1 gamma 0.1= 9.782609%

```
> set.seed(1)
> tune.out=tune(svm,
+              HeartDisease~., data=heart2[train,],
+              kernel="radial",
+              ranges=list(cost=c(0.001, 0.01, 0.1, 1,5,10,100),
+                          gamma=c(0.001,0.01,0.1,0.5)))
> summary(tune.out)
Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
 cost gamma
    1   0.1

- best performance: 0.152425
- Detailed performance results:
    cost gamma     error dispersion
1  1e-03 0.001 0.4529619 0.05474582
2  1e-02 0.001 0.4529619 0.05474582
3  1e-01 0.001 0.4529619 0.05474582
4  1e+00 0.001 0.1701222 0.03824230
5  5e+00 0.001 0.1687708 0.04330666
6  1e+01 0.001 0.1688078 0.04004071
7  1e+02 0.001 0.1701777 0.03854469
8  1e-03 0.010 0.4529619 0.05474582
9  1e-02 0.010 0.4529619 0.05474582
10 1e-01 0.010 0.1728249 0.03960902
11 1e+00 0.010 0.1715106 0.03891847
12 5e+00 0.010 0.1701407 0.03604591
13 1e+01 0.010 0.1660866 0.03537615
14 1e+02 0.010 0.1974269 0.04201792
15 1e-03 0.100 0.4529619 0.05474582
16 1e-02 0.100 0.4529619 0.05474582
17 1e-01 0.100 0.1687338 0.03856396
18 1e+00 0.100 0.1524250 0.04486780
19 5e+00 0.100 0.1578675 0.03752578
20 1e+01 0.100 0.1578675 0.03752578
21 1e+02 0.100 0.1578675 0.03752578
22 1e-03 0.500 0.4529619 0.05474582
23 1e-02 0.500 0.4529619 0.05474582
24 1e-01 0.500 0.4529619 0.05474582
25 1e+00 0.500 0.2027767 0.05445431
26 5e+00 0.500 0.1974269 0.05154775
27 1e+01 0.500 0.1974269 0.05154775
28 1e+02 0.500 0.1974269 0.05154775

> svm.obj <- svm(HeartDisease ~., data=heart2, cost=1, gamma=0.1,kernel="radial",
+               subset=train)
> mean(predict(svm.obj, newdata=heart2[-train,])
+      != heart2$HeartDisease[-train])
[1] 0.09782609
> 0.09782609*100
[1] 9.782609
```

Code for SVM plots:

```
plot(tune.out, heart2, Age~MaxHR)
plot(tune.out, heart2, Age~RestingBP)
plot(tune.out, heart2, Age~OldPeak)
plot(tune.out, heart2, RestingBP~MaxHR)
plot(tune.out, heart2, OldPeak~MaxHR)
```