Classification using Logistic Regression

Identifying Heart Disease in Emergency Room Patients

Introduction

The goal of this project is to identify which emergency room patients presenting with heart disease symptoms truly have heart disease using a dataset of 297 Cleveland Clinic patients.

A logistic regression model to estimate the log-odds of having heart disease is developed. Model diagnostics are run to check the adequacy and appropriateness of the model. Finally, predictive performance is analyzed using a training and test split of the data set.

Exploratory Data Analysis

Load and Check Data

The Heart Disease data set was downloaded from the kmed package. The first few observations are displayed to check that the loading process was correctly applied. Initial data quality checks were completed including a check for any missing data points and a check for any duplicate observations.

```
heartDisease = data.frame(heart)
# Check data import
kbl(head(heartDisease, 4)) %>% kable_styling(position="center", font_size = 12)
```

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	class
63	TRUE	1	145	233	TRUE	2	150	FALSE	2.3	3	0	6	0
67	TRUE	4	160	286	FALSE	2	108	TRUE	1.5	2	3	3	2
67	TRUE	4	120	229	FALSE	2	129	TRUE	2.6	2	2	7	1
37	TRUE	3	130	250	FALSE	0	187	FALSE	3.5	3	0	3	0

```
# Check for any missing datapoints and any duplicate observations
cat("Missing Values Check:", paste(anyNA(heartDisease)),"\n")
```

```
## Missing Values Check: FALSE
```

```
cat("Number of Duplicate Observations:", paste(sum(duplicated(heartDisease))))
```

```
## Number of Duplicate Observations: 0
```

Evaluating the Data Structure

The data set structure is modified to assign the correct data type and appropriate labels for each binary/categorical feature.

```
heartDiseaseFinal = heartDisease %>% mutate(age = as.integer(age),
                                    sex = as.factor(ifelse(sex==TRUE,"M","F")),
                                    trestbps = as.integer(trestbps),
                                    chol = as.integer(chol),
```

```
                                    thalach = as.integer(thalach),
                                    ca = as.integer(ca),
                                    class = as.logical(ifelse(class == 0,0,1)))

heartDiseaseFinal$cp = recode_factor(heartDiseaseFinal$cp, "1" = "Typical/Atypical",
                    "2" = "Typical/Atypical","3" = "Non-Anginal", "4" = "Asymptomatic")

heartDiseaseFinal$restecg = recode_factor(heartDiseaseFinal$restecg, "0" = "(Ab)Normal",
                    "1" = "(Ab)Normal","2" = "Hypertrophy")

heartDiseaseFinal$slope = recode_factor(heartDiseaseFinal$slope, "1" = "Upsloping",
                    "2" = "Flat/Down", "3" = "Flat/Down")

heartDiseaseFinal$thal = recode_factor(heartDiseaseFinal$thal, "3" = "Normal/Fixed",
                    "6" = "Normal/Fixed", "7" = "Reversable")

rownames(heartDiseaseFinal) = seq.int(nrow(heartDiseaseFinal))   # fix row index numbering
names(heartDiseaseFinal)[14] = "hd"                              # rename supervisor
```

Overview of Features

The dataset consists of 1 supervisor and 13 features. 6 features are quantitative and are coded as either numeric or integer. The remaining 7 features are qualitative (either binary or categorical).

Summary statistics displaying the range of values for the quantitative features and the counts by level for the qualitative features are displayed below. This is a good way to check for any problems in the data such as any data points beyond the realistic range of possible values. Also, for any model to have value in a practical application, the sample data needs to accurately represent the overall population, the features need to have some plausible relationship to the supervisor and there needs to be sufficient data in each qualitative feature category.

```
descNumer = c("Patient Age","Resting Blood Pressure", "Serum Cholesterol mg/dl",
              "Max Heart Rate", "ST Depression", "Vessels Colored in Flouroscopy")

numerTable = data.frame(rbind(apply(heartDiseaseFinal[,c(1,4:5,8,10,12)],2,min),
                    apply(heartDiseaseFinal[,c(1,4:5,8,10,12)],2,median),
                    round(apply(heartDiseaseFinal[,c(1,4:5,8,10,12)],2,mean),1),
                    apply(heartDiseaseFinal[,c(1,4:5,8,10,12)],2,max)))

numerTable = rbind(descNumer, numerTable )

rownames(numerTable) = c("Desc.","Min","Median","Mean","Max")

kbl(numerTable, caption = "Summary of Continuous/Numerical Features", align = "c") %>%
  kable_styling(position="center", font_size = 14)
```

Summary of Continuous/Numerical Features

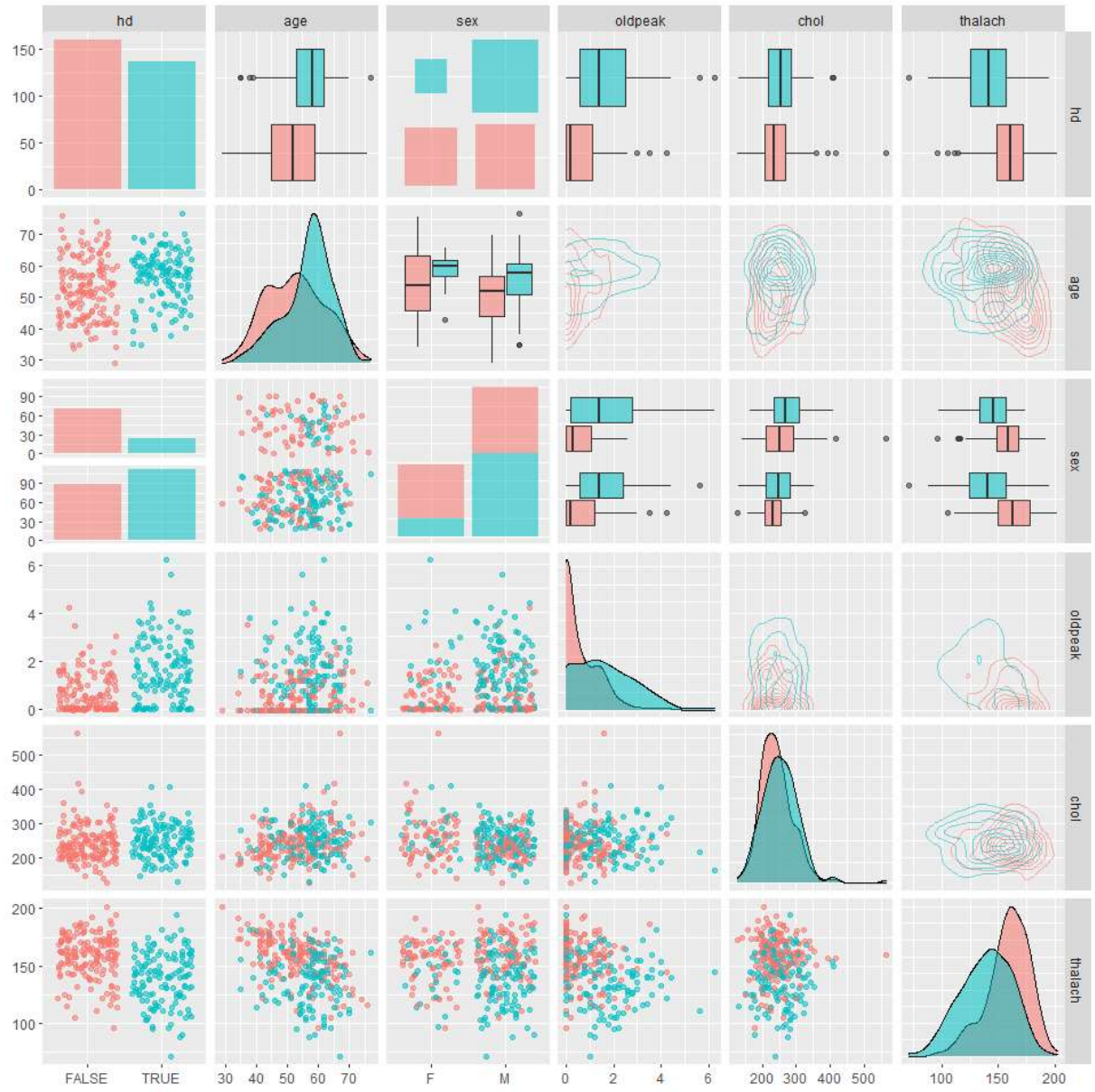
	age	trestbps	chol	thalach	oldpeak	ca
Desc.	Patient Age	Resting Blood Pressure	Serum Cholesterol mg/dl	Max Heart Rate	ST Depression	Vessels Colored in Flouroscopy
Min	29	94	126	71	0	0
Median	56	130	243	153	0.8	0
Mean	54.5	131.7	247.4	149.6	1.1	0.7
Max	77	200	564	202	6.2	3

Summary of Binary/Categorical Features			
Sex		*RestingECG*	
Female	96	Normal/Abnormal	151
Male	201	Hypertrophy	146
Exceed Fasting Blood Sugar		*Slope of Peak Exercise ST*	
FALSE	254	Upsloping	139
TRUE	43	Flat/Down	158
Exericse Induced Angina		*Chest Pain Type*	
FALSE	200	Typical/Atypical	72
TRUE	97	Non-Anginal	83
Thalassemia		Asymptomatic	142
Normal/Fixed	182		
Reversible	115		

Pairs Plots

Pairs plots are a visual way to evaluate each feature's relation to the supervisor, other features and it's own distribution. They also enable visual identification of outliers, imbalances in qualitative categories and whether some features might need a transformation or interaction term. A subset of the pairs plot is provided below:

```
ggpairs(
 heartDiseaseFinal[,c(14,1,2,10,5,8)],
 mapping = ggplot2::aes(color = hd, alpha = 0.85),
 upper = list(continuous = wrap("density"), combo = "box_no_facet"),
 lower = list(continuous = wrap("points"), combo = wrap("dot_no_facet")),
)
```



From the pairs plots it appears that there may be some outliers/extreme observations in both oldpeak (row 1, column 4) and chol (row 1, column 5). These observations are likely to have a larger influence (high leverage) on the regression model and will be evaluated in model diagnostics. An interaction term between two features may be needed if the covariance between the two varies for each class of the supervisor. Age and thalach might require an interaction term (row 6, column 2).

Modeling

Logistic regression requires that the following assumptions be met:
- the supervisor is binary
- the relationship between the log-odds of the supervisor and the features is linear and the features are correctly specified
- observations are independent
- stong multicollinearity is absent
- outliers don't exhibit high leverage
- sample size is sufficiently large

Check for Skewness

If the density of a feature is strongly skewed, then the log-odds can depend on both the feature and the natural log transformation of that same feature. A commonly used heuristic is to try a natural log transformation if the absolute value of the skewness exceeds 2. None of the quantitative features meet this criteria.

```
# check for high skewness
skewInd = c(1,4,5,8,10,12)
skewnessVec = round(heartDiseaseFinal[,skewInd] %>%
                      sapply(., e1071::skewness, na.rm = TRUE),3)
cat("Number of features with high skew:", paste(sum(abs(skewnessVec)>=2)))
```

```
## Number of features with high skew: 0
```

Check for Multicollinearity

Multicollinearity occurs when two or more features are strongly correlated with one another, meaning that the information provided by each feature with respect to the supervisor isn't separable or distinct. This can result in model fitting problems but can be alleviated by removing one of the highly correlated features or applying regularization techniques such as ridge regression. A partial check for multicollinearity is performed below.

```
multiCol = model.matrix(~., data=heartDiseaseFinal[,-14], fullRank = TRUE)[,-1] %>%
  cor(use="pairwise.complete.obs")
cat("Number of feature pairs with high correlation:",
    paste(sum(multiCol[upper.tri(multiCol, diag = FALSE)] >= .85)))
```

```
## Number of feature pairs with high correlation: 0
```

MODEL 1

A preliminary model to estimate the log odds of having heart disease is estimated using all 13 features.

```
model1Out = glm(hd~., data = trainData, family = "binomial")
summary(model1Out)
```

```
##
## Call:
## glm(formula = hd ~ ., family = "binomial", data = trainData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9465  -0.4379  -0.1262   0.4014   2.6126
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -4.675433   3.581504  -1.305  0.19174
## age                -0.018530   0.029429  -0.630  0.52893
## sexM                1.630470   0.600826   2.714  0.00665 **
## cpNon-Anginal      -0.081637   0.632628  -0.129  0.89732
## cpAsymptomatic      1.311305   0.585528   2.240  0.02512 *
## trestbps            0.036586   0.013913   2.630  0.00855 **
## chol                0.003248   0.004759   0.683  0.49482
## fbsTRUE            -1.546799   0.772560  -2.002  0.04527 *
## restecgHypertrophy  0.314249   0.478855   0.656  0.51166
## thalach            -0.027123   0.013369  -2.029  0.04248 *
## exangTRUE           1.181442   0.549400   2.150  0.03152 *
## oldpeak             0.316996   0.261305   1.213  0.22508
## slopeFlat/Down      0.455674   0.604592   0.754  0.45104
## ca                  1.270983   0.317522   4.003 6.26e-05 ***
## thalReversable      1.586373   0.496849   3.193  0.00141 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.12  on 207  degrees of freedom
## Residual deviance: 133.22  on 193  degrees of freedom
## AIC: 163.22
##
## Number of Fisher Scoring iterations: 6
```

Model 1 Interpretation

In logistic regression a "base case" for the qualitative features is assigned. Here, the base case is a female patient with (a)typical chest pain, an upsloping slope of peak exercise CT, normal Thalassemia tests, and no excess fasting blood sugar, hypertrophy or exercise induced angina. The log-odds of having heart disease for someone that meets this criteria is contained in the intercept coefficient (excluding the values for the quantitative features). As a result, The coefficients of the other qualitative predictors measure the change in log-odds of having heart disease versus this base case.

For example, for a female with asymptomatic chest pain, the estimated log-odds of having heart disease increases by 1.31, on average, controlling or holding all other features constant. This means that the odds of having heart disease when asymptomatic chest pain is present are $e^{1.31}$ = 3.7 times higher, on average, as compared to the base case, controlling for all other features.

The interpretation of continuous feature coefficients is more straightforward. For example, a one year increase in age is associated with an expected average .018 decrease in the log-odds of having heart disease,

(or $e^{-.018}$ = a 0.982 *multiplicative* effect, which is about a 2% decrease in odds), again controlling for the other features.

The estimated coefficient for age provides an interesting result. While the coefficient is negative, the practical effect is very small, especially compared to the standard error which indicates that there is **not** strong evidence that age is associated with heart disease. The test statistic is: $z = \frac{-.018-0}{.029} = -0.63$ which translates to a p-value of 0.528, much higher than the typical 0.05 (5%) level for statistical significance. A 95% confidence interval for the estimated age coefficient is:

$$(est.\ coef.)\pm(1.96) * (std.\ error)$$

$$(-.018)\pm(1.96) * (.029) = (-0.075, 0.039)$$

The confidence interval for the estimated effect of age on the log-odds of heart disease contains 0, indicating that the model has not found significant evidence that age is associated with heart disease.

Model Diagnostics

Model 1 Validation

The next step is to review model diagnostics to check for misspecifications, violations of modeling assumptions or any other problems with the model.

Misspecification Check

One method to check for model misspecification is to use linktest which regresses the supervisor on the predicted value ("fit") and squared predicted value ("fit2") from model 1. A correctly specified model will have a statistically significant fit value and a non-significant fit2 value, which is the case here.

```
print(blr_linktest(model1Out)$coef, digits=2)
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.122      0.262    0.47  6.4e-01
## fit            1.012      0.137    7.36  1.8e-13
## fit2          -0.045      0.049   -0.91  3.6e-01
```

Goodness of Fit

There are a number of goodness of fit (GOF) metrics that can be used to check a logistic regression model. However, each has some drawbacks. Here we look at three GOF metrics.

Hosmer-Lemeshow's GOF separates observations into groups and compares the model's predicted number of patients with heart disease to the actual observed number in each group. Large differences between these two in each group indicates a poor fit and would be indicated by a low p-value. A high p-value, which we have here, indicates an adequate fit.

```
logitgof(trainData$hd, fitted(model1Out), g=10)
```

```
##
##  Hosmer and Lemeshow test (binary model)
##
## data:  trainData$hd, fitted(model1Out)
## X-squared = 5.5845, df = 8, p-value = 0.6937
```

Psuedo R-squared

R-squared in linear regression measures the proportion of variance explained by the model. In logistic regression, this isn't available as the variance is fixed. But a similar measure using the proportion of log likelihoods between the model with features and an intercept only model can be used. There are several versions of this GOF metric which include various adjustments. The more commonly used metrics are included here, and all indicate that the model with features has sufficient GOF.

```
print(PseudoR2(model1Out, which = c("McFadden","CoxSnell","AldrichNelson",
                                    "VeallZimmermann")), digits=3)
```

```
##        McFadden        CoxSnell   AldrichNelson VeallZimmermann
##           0.536           0.523           0.425           0.733
```

Full vs Intercept Only Model

A third GOF check tests whether the proposed model is better than an intercept only model with no features. Deviance tests can be used to compare these two models with a lower deviance being better. Deviance

compares the maximized log-likelihoods of the proposed model with a saturated model where there are as many parameters as there are observations. To test whether Model 1 is better than an intercept only model, the differences in their respective deviances can be used.

Null Hypothesis - H_0 : There is no significant reduction in deviance between the Null Model and Model 1
Alternative - H_A : The reduction in deviance is significant, Model 1 is appropriate

```
m1NullDev = summary(model1Out)$null.deviance
m1NullDF = summary(model1Out)$df.null
m1Dev = summary(model1Out)$deviance
m1df =  summary(model1Out)$df.residual
(pValue = 1 - pchisq(m1NullDev-m1Dev, df = (m1NullDF-m1df)))
```

```
## [1] 0
```

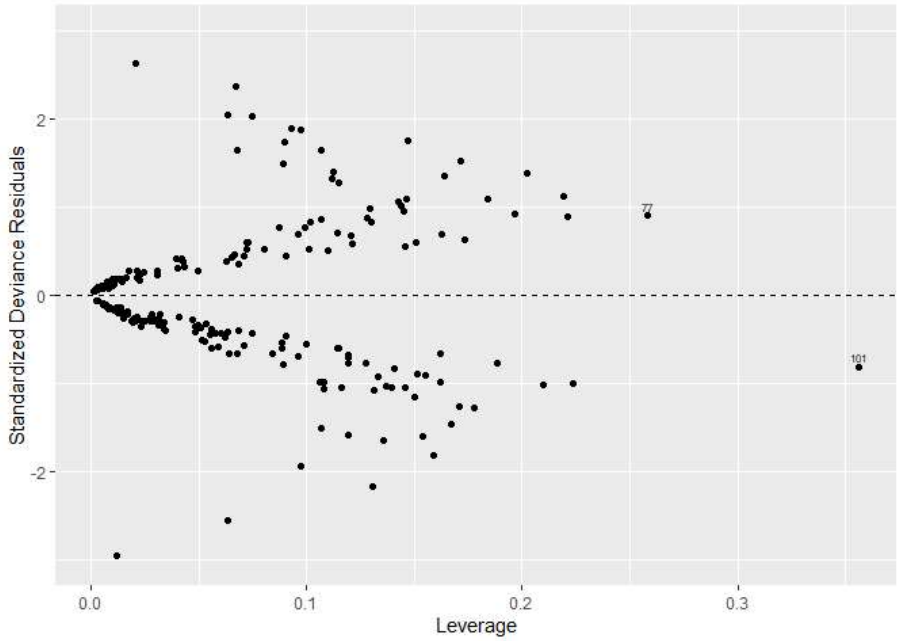
The p-value is very close to 0 indicating a rejection of H_0. There is strong evidence that the reduction in deviance is significant and that Model 1 is a more appropriate model than the intercept only model.

Leverage and Deviance

A plot of the standardized deviance residuals by leverage can help identify any remaining modeling or data problems. Points 101 and 77 in the test dataset correspond to data points that have unusually high leverage, indicating that they have an outsized effect on the model's estimated coefficients as compared to other data points. These two points correspond to people with very high levels of cholesterol (and for point 77, high levels of oldpeak). As cholesterol isn't a statistically significant feature, it will be removed from the model. After doing so, this plot can be rerun which would show that no points with unusually high leverage remain.

```
#Figure 8.13 on page 291
par(mfrow=c(1,1))
hvalues <- influence(model1Out)$hat
stanresDeviance <- residuals(model1Out)/sqrt(1-hvalues)
leverageDF = bind_cols(hvalues, stanresDeviance)
leverageDF$row = seq.int(nrow(leverageDF))
leverageDF$rowch = as.character(leverageDF$row)
leverageDF$rowch[hvalues<=.25] = ""

ggplot(leverageDF, aes(x=hvalues, y=stanresDeviance, label = rowch)) +
  geom_point() + geom_hline(yintercept = 0, lty=2) +
  geom_text(aes(label = rowch), vjust = - 0.5, size = 2) +
  scale_y_continuous(limits = c(-3,3)) +
  labs(x = "Leverage", y = "Standardized Deviance Residuals")
```



The two data outliers with high levels of chol:

```
kbl((trainData[c(77,101),])) %>%
  kable_styling(position="center", font_size = 8) %>%
  column_spec(6, color = "red")
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	hd
113	43	F	Asymptomatic	132	341	TRUE	Hypertrophy	136	TRUE	3.0	Flat/Down	0	Reversable	TR
152	67	F	Non-Anginal	115	564	FALSE	Hypertrophy	160	FALSE	1.6	Flat/Down	0	Reversable	FA

MODEL 2 - Remove Chol

Chol is removed as a feature. The only notable change in the estimated model coefficients is a decrease in the magnitude of the intercept term, although it remains statistically insignificant.

```
hdReduced = trainData[,-5]
model2Out = glm(hd~., data = hdReduced, family = "binomial")
summary(model2Out)$coef
```

```
##                       Estimate Std. Error    z value     Pr(>|z|)
## (Intercept)        -3.89641239 3.38920059 -1.1496553 2.502859e-01
## age                -0.01848635 0.02933652 -0.6301481 5.285977e-01
## sexM                1.49599855 0.55958617  2.6734016 7.508631e-03
## cpNon-Anginal      -0.06399098 0.63329976 -0.1010437 9.195157e-01
## cpAsymptomatic      1.32254657 0.58694270  2.2532806 2.424146e-02
## trestbps            0.03675951 0.01387337  2.6496450 8.057638e-03
## fbsTRUE            -1.53365947 0.76763711 -1.9978965 4.572789e-02
## restecgHypertrophy  0.39360190 0.46441470  0.8475225 3.967040e-01
## thalach            -0.02687625 0.01335516 -2.0124238 4.417528e-02
## exangTRUE           1.15638277 0.54345704  2.1278274 3.335140e-02
## oldpeak             0.32704824 0.26079865  1.2540258 2.098326e-01
## slopeFlat/Down      0.44313184 0.60605329  0.7311764 4.646714e-01
## ca                  1.26304878 0.31684177  3.9863708 6.709162e-05
## thalReversable      1.62386094 0.49586375  3.2748128 1.057320e-03
```

The model diagnostics were repeated for Model 2, but had no notable changes and are thus excluded here. Further model validation steps could be taken, including marginal model plots, and feature selection methods could also be employed, such as stepwise selection or LASSO.

Model Performance

Once the model is trained and the feature coefficients are estimated, they can be applied to the test data to check for how accurate the model is in correctly classifying patients. An alternative and/or complement to a training/test split would be to conduct cross validation. Cross validation is skipped for this report.

Accuracy

Accuracy measures the portion of observations in the test data set that the model correctly identifies (as having or not having heart disease). One technical note: the logistic regression model estimates the average log-odds of having heart disease using data provided by the features. This estimate must be converted into a binary classifier, as either having or not having heart disease. Thus a cutoff, or threshold value dividing the two classes must be set to assign the predicted classifications. Typically, a threshold probability of 50% is used. Log-odds can be converted to a probability by the following formula:

$$\frac{e^{log-odds}}{1+e^{log-odds}}$$

```
hdReducedTest = testData[,-5]
yHatProbM2 =  predict(model2Out,newdata=hdReducedTest,type='response')
yHatM2 = ifelse(yHatProbM2 >= 0.5,TRUE,FALSE)   # applies a .5 threshold
accM2 = mean(yHatM2 == hdReducedTest$hd)
cat("Accuracy:", paste(round(accM2,3)*100), "%")
```

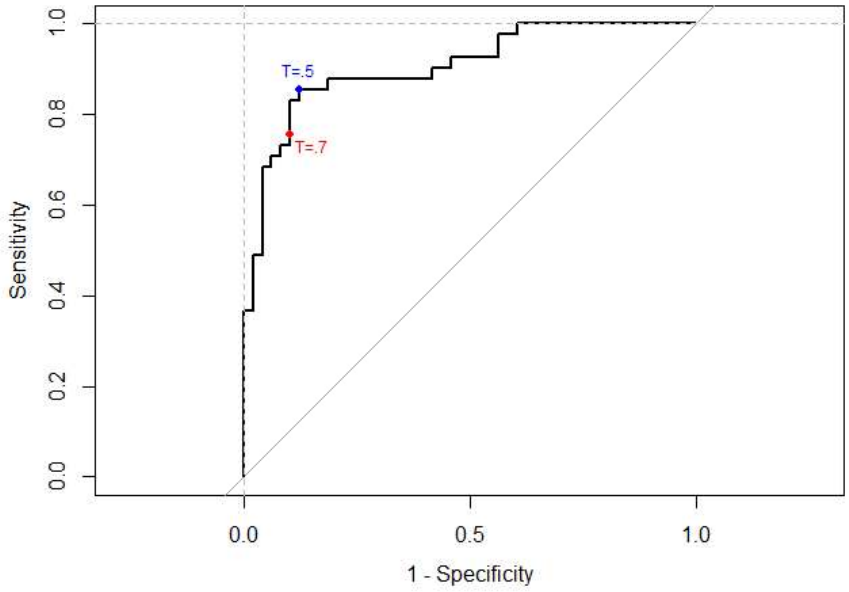
```
## Accuracy: 85.4 %
```

85.4% of the test data observations were correctly predicted. This is fairly good performance for a simple model but probably not sufficient for use in an emergency room setting. Applying feature selection methods or gathering more data might result in better performance.

ROC Curve

The ROC curve plots the tradeoff between sensitivity and specificity (or 1-specificity) for every possible threshold probability value. Sensitivity is the percentage of correctly identified patients with heart disease. Specificity is the percentage of correctly identified patients without heart disease. It may be the case that a researcher is more interested in one of these measures over the other and can adjust the probability threshold to increase the model's performance in one of these metrics. The ROC curve below shows points for thresholds of 50% (in blue) and 70% (in red). Increasing the threshold probability increases the specificity but decreases the sensitivity. Decreasing the threshold will have the opposite effect.

```
yTestROC = as.numeric(hdReducedTest$hd)
rocCurveM2 = roc(yTestROC, yHatProbM2)
plot(rocCurveM2, legacy.axe = TRUE)
abline(h = 1, lty = 2, col = "gray")
abline(v = 1, lty = 2, col = "gray")
# add threshold points
points(x= rocCurveM2$specificities[min(which(rocCurveM2$thresholds[2:84]>=.5))],
       y=rocCurveM2$sensitivities[min(which(rocCurveM2$thresholds[2:84]>=.5))],
       col = "blue", pch = 16, cex = .9)
text(x= 0.88, y=0.90, "T=.5", col = "blue", cex = .75)
points(x= rocCurveM2$specificities[min(which(rocCurveM2$thresholds[2:84]>=.7))],
       y=rocCurveM2$sensitivities[min(which(rocCurveM2$thresholds[2:84]>=.7))],
       col = "red", pch = 16, cex = .9)
text(x= rocCurveM2$specificities[min(which(rocCurveM2$thresholds[2:84]>=.5))]-.025,
     y=rocCurveM2$sensitivities[min(which(rocCurveM2$thresholds[2:84]>=.7))]-.025,
     "T=.7", col = "red", cex = .75)
```




```
cat("Area Under Curve:", paste(round(rocCurveM2$auc,3)))
```

```
## Area Under Curve: 0.904
```

A perfect model would have an ROC curve in the shape of an upside down L, following the plotted dotted lines. The area under the ROC curve, or "AUC" is a frequently used metric that describes the performance of a classification model averaged across all possible values of the threshold probability. The maximum possible AUC is 1. Here, the AUC value of .9 is quite good.

Confusion Matrix

A confusion matrix is a table showing the observed and predicted classes in the test dataset (using the selected 50% probability threshold). The model correctly predicts 42 patients as not having heart disease (False) and 34 as having heart disease (True). The model incorrectly identifies 6 patients with heart disease as not having it (false negative) and 7 patients without heart disease as having it (false positive).

```
confusionOut = confusionMatrix(reference = as.factor(yHatM2),
                               data = as.factor(hdReducedTest$hd))
confusionOut[2]
```

```
## $table
##           Reference
## Prediction FALSE TRUE
##      FALSE    42    6
##      TRUE      7   34
```

Conclusion

The goal of this project was to demonstrate the steps in building and evaluating a classification model using logistic regression, to determine which features are associated with heart disease and to evaluate model performance. Given a small amount of features and data, the model performs fairly well but not quite well enough to use in an emergency room setting.