Logistic Regression

This situation required a model to be fit on the *GermanCredit* data set. Since the response variable was a binary variable, whether applicants were a good credit risk or not, logistic regression was an appropriate approach.

Model Fitting

After categorical variables had been converted to binary variables, a model was ready to be fit. A cross-validated logistic regression model was fit to compare the accuracies of various models.

```
logit.1 = train(V21 ~.,
                data = german,
                trControl = train.control,
                method = "glm",
                family=binomial(link='logit'))

summary(logit.1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3410  -0.6994  -0.3752   0.7095   2.6116
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.005e-01  1.084e+00   0.369 0.711869
## V1A12       -3.749e-01  2.179e-01  -1.720 0.085400 .
## V1A13       -9.657e-01  3.692e-01  -2.616 0.008905 **
## V1A14       -1.712e+00  2.322e-01  -7.373 1.66e-13 ***
## V2           2.786e-02  9.296e-03   2.997 0.002724 **
## V3A31        1.434e-01  5.489e-01   0.261 0.793921
## V3A32       -5.861e-01  4.305e-01  -1.362 0.173348
## V3A33       -8.532e-01  4.717e-01  -1.809 0.070470 .
## V3A34       -1.436e+00  4.399e-01  -3.264 0.001099 **
## V4A41       -1.666e+00  3.743e-01  -4.452 8.51e-06 ***
## V4A410      -1.489e+00  7.764e-01  -1.918 0.055163 .
## V4A42       -7.916e-01  2.610e-01  -3.033 0.002421 **
## V4A43       -8.916e-01  2.471e-01  -3.609 0.000308 ***
## V4A44       -5.228e-01  7.623e-01  -0.686 0.492831
## V4A45       -2.164e-01  5.500e-01  -0.393 0.694000
## V4A46        3.628e-02  3.965e-01   0.092 0.927082
## V4A48       -2.059e+00  1.212e+00  -1.699 0.089297 .
## V4A49       -7.401e-01  3.339e-01  -2.216 0.026668 *
```

```
## V5           1.283e-04  4.444e-05   2.887 0.003894 **
## V6A62       -3.577e-01  2.861e-01  -1.250 0.211130
## V6A63       -3.761e-01  4.011e-01  -0.938 0.348476
## V6A64       -1.339e+00  5.249e-01  -2.551 0.010729 *
## V6A65       -9.467e-01  2.625e-01  -3.607 0.000310 ***
## V7A72       -6.691e-02  4.270e-01  -0.157 0.875475
## V7A73       -1.828e-01  4.105e-01  -0.445 0.656049
## V7A74       -8.310e-01  4.455e-01  -1.866 0.062110 .
## V7A75       -2.766e-01  4.134e-01  -0.669 0.503410
## V8           3.301e-01  8.828e-02   3.739 0.000185 ***
## V9A92       -2.755e-01  3.865e-01  -0.713 0.476040
## V9A93       -8.161e-01  3.799e-01  -2.148 0.031718 *
## V9A94       -3.671e-01  4.537e-01  -0.809 0.418448
## V10A102      4.360e-01  4.101e-01   1.063 0.287700
## V10A103     -9.786e-01  4.243e-01  -2.307 0.021072 *
## V11          4.776e-03  8.641e-02   0.055 0.955920
## V12A122      2.814e-01  2.534e-01   1.111 0.266630
## V12A123      1.945e-01  2.360e-01   0.824 0.409743
## V12A124      7.304e-01  4.245e-01   1.721 0.085308 .
## V13         -1.454e-02  9.222e-03  -1.576 0.114982
## V14A142     -1.232e-01  4.119e-01  -0.299 0.764878
## V14A143     -6.463e-01  2.391e-01  -2.703 0.006871 **
## V15A152     -4.436e-01  2.347e-01  -1.890 0.058715 .
## V15A153     -6.839e-01  4.770e-01  -1.434 0.151657
## V16          2.721e-01  1.895e-01   1.436 0.151109
## V17A172      5.361e-01  6.796e-01   0.789 0.430160
## V17A173      5.547e-01  6.549e-01   0.847 0.397015
## V17A174      4.795e-01  6.623e-01   0.724 0.469086
## V18          2.647e-01  2.492e-01   1.062 0.288249
## V19A192     -3.000e-01  2.013e-01  -1.491 0.136060
## V20A202     -1.392e+00  6.258e-01  -2.225 0.026095 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1221.73  on 999  degrees of freedom
## Residual deviance:  895.82  on 951  degrees of freedom
## AIC: 993.82
##
## Number of Fisher Scoring iterations: 5
```

As it can be seen above, there are a number of insignificant predictors in this model. Next, these were removed and the model was refit on only the significant predictors.

```
logit.1.sum = summary(logit.1)
coeff = names(which(logit.1.sum$coefficients[,4] < 0.05))
formula = paste('V21', paste(coeff, collapse=' + '), sep = ' ~ ')
for(c in coeff){
  splits = str_split(c,'A',2,simplify = T)
```

```
  col = splits[1,1]
  val = paste0('A',splits[1,2])
  if(val != 'A'){
    german[paste0(c)] = as.data.frame(ifelse(german[col] == val,1,0))
  }
}
logit.2 = train(as.formula(formula),
                data = german,
                trControl = train.control,
                method = "glm",
                family=binomial(link='logit'))
```

Afterwards, the accuracy of the full and reduced models were compared.

```
logit.1$results$RMSE
```

```
## [1] 0.4115816
```

```
logit.2$results$RMSE
```

```
## [1] 0.4041013
```

```
logit.1$results$Rsquared
```

```
## [1] 0.2087795
```

```
logit.2$results$Rsquared
```

```
## [1] 0.2334721
```

From this, it can be seen that the model containing only significant predictors had a lower cross-validated RMSE and accounted for more variance in the data (23.35%) than the full model. Because of this, the reduced model would be retained for further analysis. The predictors used and their coefficients for this model can be seen below:

```
summary(logit.2)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1782  -0.7569  -0.3950   0.8553   2.7117
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.053e-01  3.584e-01  -1.131 0.258042
## V1A13       -8.309e-01  3.400e-01  -2.444 0.014527 *
## V1A14       -1.644e+00  1.944e-01  -8.454  < 2e-16 ***
## V2           2.860e-02  8.703e-03   3.286 0.001015 **
## V3A34       -7.562e-01  1.988e-01  -3.804 0.000142 ***
```

```
## V4A41       -1.423e+00  3.406e-01  -4.179 2.93e-05 ***
## V4A42       -4.717e-01  2.231e-01  -2.114 0.034492 *
## V4A43       -8.714e-01  2.146e-01  -4.062 4.88e-05 ***
## V4A49       -6.703e-01  2.875e-01  -2.332 0.019714 *
## V5           1.073e-04  3.966e-05   2.704 0.006844 **
## V6A64       -1.308e+00  4.909e-01  -2.665 0.007709 **
## V6A65       -9.788e-01  2.433e-01  -4.023 5.76e-05 ***
## V8           3.071e-01  8.213e-02   3.739 0.000184 ***
## V9A93       -5.909e-01  1.684e-01  -3.509 0.000449 ***
## V10A103     -1.022e+00  3.984e-01  -2.566 0.010292 *
## V14A143     -6.650e-01  1.960e-01  -3.394 0.000690 ***
## V20A202     -1.083e+00  6.261e-01  -1.729 0.083812 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1221.73  on 999  degrees of freedom
## Residual deviance:  948.52  on 983  degrees of freedom
## AIC: 982.52
##
## Number of Fisher Scoring iterations: 5
```

Optimal Threshold

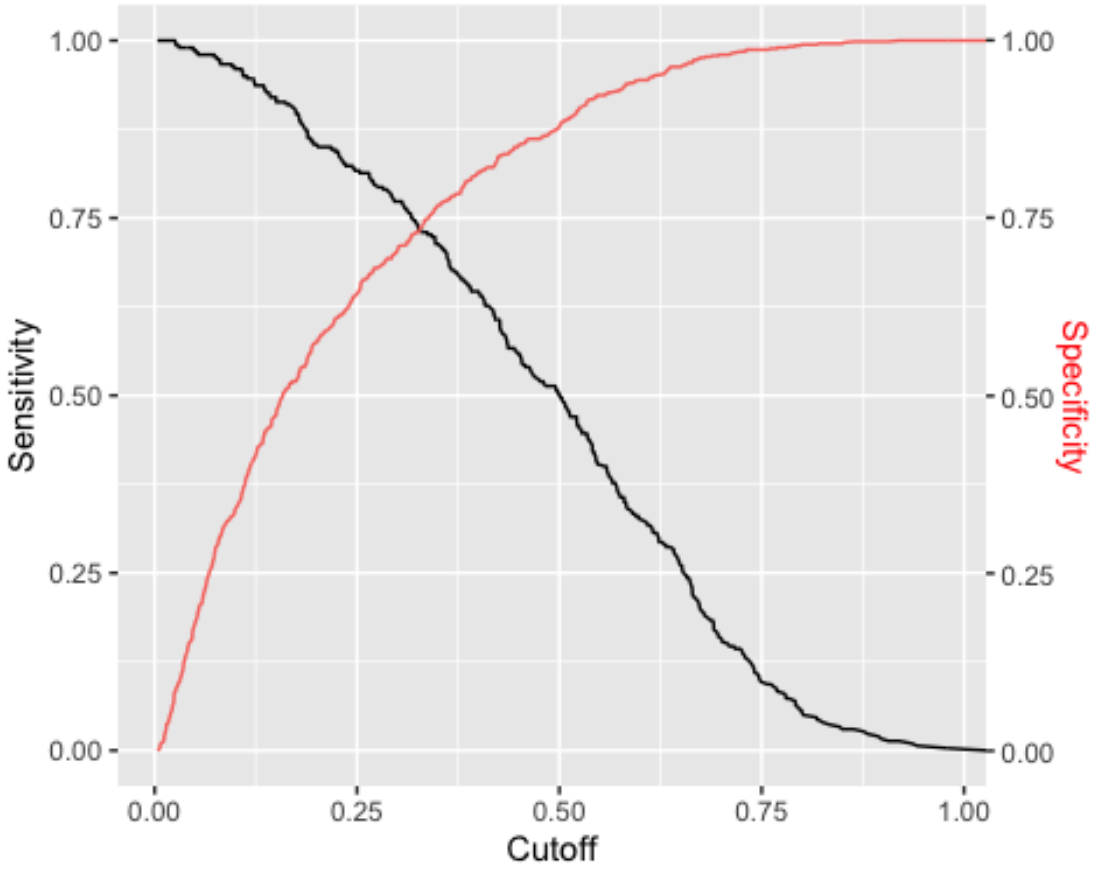
The estimate for the *GermanCredit* data set is that incorrectly identifying a bad customer as good is 5 times worse than incorrectly classifying a good customer as bad. To account for this, the optimal threshold probability needed to be calculated for the selected logistic regression model.

To do so, the intersection between sensitivity and specificity metric for this model needed to be determined.

```
logit.pred = predict(logit.2.final,german,type='response')
predictions = prediction(logit.pred, german$V21)
sens = data.frame(x=unlist(performance(predictions, "sens")@x.values), y=unli
st(performance(predictions, "sens")@y.values))
spec = data.frame(x=unlist(performance(predictions, "spec")@x.values), y=unli
st(performance(predictions, "spec")@y.values))
opt.cut = sens[which.min(apply(sens, 1, function(x) min(colSums(abs(t(spec) -
x))))), 1]
opt.cut

## [1] 0.3270131
```

As it can be seen above, the optimal threshold probability for the model chosen is 0.3270131. The intersection of these two metrics can be seen below:



Model Accuracy

Using this optimal threshold, a confusion matrix was created for this model.

```
pred.round = as.integer(logit.pred > opt.cut )
conf.mat = confusionMatrix(pred.round, german$V21, threshold = opt.cut)
conf.mat
```

```
##     0   1
## 0 514 186
## 1  80 220
```

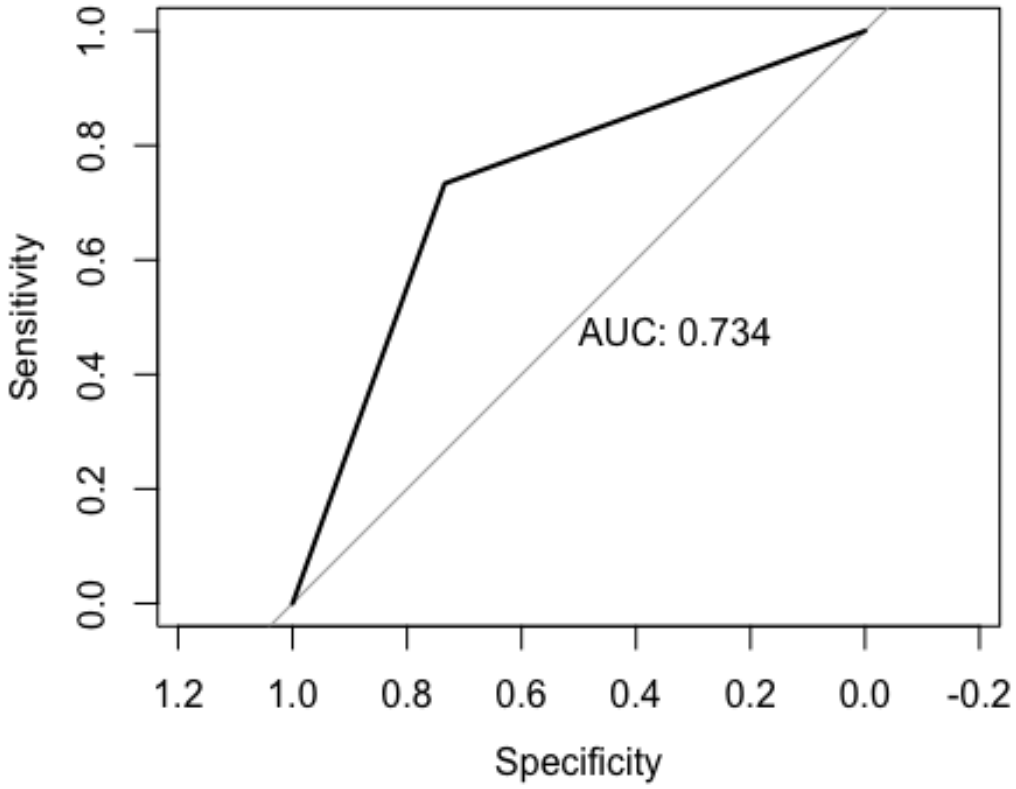
```
# Accuracy
(conf.mat[1,1]+ conf.mat[2,2]) / sum(conf.mat)
```

```
## [1] 0.734
```

```
# Missclassificaton Rate
(conf.mat[2,1]+ conf.mat[1,2]) / sum(conf.mat)
```

```
## [1] 0.266
```

From this, it can be seen that the model is correct 73.4% of the time, and is incorrect 26.6% of the time. A plot of the ROC curve can be seen below:


Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
AUC: 0.734
1.2
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
Specificity