BHao_HW3

DATA EXPLORATION

- Summary data indicate that ~49% of neighborhoods have crime rates above the median, which makes sense
- Also, there are no NA data points that we need to address via preprocessing for logistic regression

```
crime = read.csv('crime-training-data.csv')
str(crime)
```

```
## 'data.frame':    466 obs. of  14 variables:
##  $ zn     : num  0 0 0 30 0 0 0 0 0 80 ...
##  $ indus  : num  19.58 19.58 18.1 4.93 2.46 ...
##  $ chas   : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.605 0.871 0.74 0.428 0.488 0.52 0.693 0.693 0.515 0.392 ...
##  $ rm     : num  7.93 5.4 6.49 6.39 7.16 ...
##  $ age    : num  96.2 100 100 7.8 92.2 71.3 100 100 38.1 19.1 ...
##  $ dis    : num  2.05 1.32 1.98 7.04 2.7 ...
##  $ rad    : int  5 5 24 6 3 5 24 24 5 1 ...
##  $ tax    : int  403 403 666 300 193 384 666 666 224 315 ...
##  $ ptratio: num  14.7 14.7 20.2 16.6 17.8 20.9 20.2 20.2 20.2 16.4 ...
##  $ black  : num  369 397 387 375 394 ...
##  $ lstat  : num  3.7 26.82 18.85 5.19 4.82 ...
##  $ medv   : num  50 13.4 15.4 23.7 37.9 26.5 5 7 22.2 20.9 ...
##  $ target : int  1 1 1 0 0 0 1 1 0 0 ...
```

```
summary(crime)
```

```
##        zn             indus             chas              nox
##  Min.   :  0.00   Min.   : 0.460   Min.   :0.00000   Min.   :0.3890
##  1st Qu.:  0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
##  Median :  0.00   Median : 9.690   Median :0.00000   Median :0.5380
##  Mean   : 11.58   Mean   :11.105   Mean   :0.07082   Mean   :0.5543
##  3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
##  Max.   :100.00   Max.   :27.740   Max.   :1.00000   Max.   :0.8710
##        rm             age              dis               rad
##  Min.   :3.863   Min.   :  2.90   Min.   : 1.130   Min.   : 1.00
##  1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
##  Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
##  Mean   :6.291   Mean   : 68.37   Mean   : 3.796   Mean   : 9.53
##  3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.00
##       tax           ptratio         black            lstat
##  Min.   :187.0   Min.   :12.6   Min.   :  0.32   Min.   : 1.730
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.:375.61   1st Qu.: 7.043
##  Median :334.5   Median :18.9   Median :391.34   Median :11.350
##  Mean   :409.5   Mean   :18.4   Mean   :357.12   Mean   :12.631
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:396.24   3rd Qu.:16.930
##  Max.   :711.0   Max.   :22.0   Max.   :396.90   Max.   :37.970
##       medv           target
##  Min.   : 5.00   Min.   :0.0000
##  1st Qu.:17.02   1st Qu.:0.0000
```

```
##  Median :21.20   Median :0.0000
##  Mean   :22.59   Mean   :0.4914
##  3rd Qu.:25.00   3rd Qu.:1.0000
##  Max.   :50.00   Max.   :1.0000
```

DATA PREPARATION

- Again, there were no NAs within the data, so no need to address those
- Since I'll be using the caret package for modeling, I need to convert the categorical variables to factors
- Lastly, we'll use caret's center and scale preprocessing features to automatically transform the data during modeling

```
crime$chas = factor(crime$chas, labels = c('not_bordered', 'bordered'))
crime$target = factor(crime$target, labels = c('below_median', 'above_median'))
str(crime)
```

```
## 'data.frame':    466 obs. of  14 variables:
##  $ zn     : num  0 0 0 30 0 0 0 0 0 80 ...
##  $ indus  : num  19.58 19.58 18.1 4.93 2.46 ...
##  $ chas   : Factor w/ 2 levels "not_bordered",..: 1 2 1 1 1 1 1 1 1 1 ...
##  $ nox    : num  0.605 0.871 0.74 0.428 0.488 0.52 0.693 0.693 0.515 0.392 ...
##  $ rm     : num  7.93 5.4 6.49 6.39 7.16 ...
##  $ age    : num  96.2 100 100 7.8 92.2 71.3 100 100 38.1 19.1 ...
##  $ dis    : num  2.05 1.32 1.98 7.04 2.7 ...
##  $ rad    : int  5 5 24 6 3 5 24 24 5 1 ...
##  $ tax    : int  403 403 666 300 193 384 666 666 224 315 ...
##  $ ptratio: num  14.7 14.7 20.2 16.6 17.8 20.9 20.2 20.2 20.2 16.4 ...
##  $ black  : num  369 397 387 375 394 ...
##  $ lstat  : num  3.7 26.82 18.85 5.19 4.82 ...
##  $ medv   : num  50 13.4 15.4 23.7 37.9 26.5 5 7 22.2 20.9 ...
##  $ target : Factor w/ 2 levels "below_median",..: 2 2 2 1 1 1 2 2 1 1 ...
```

BUILD MODELS

- In terms of setup, we are using 10-fold cross validation to measure out-of-sample performance and are using the same folds for each model to ensure comparable results
- We then start by including all variables and then remove statistically insignificant ones at the 5% level until all remaining are significant
- We then tried a glmnet model which combines lasso and ridge regression; given that it penalizes large magnitude and the number of non-zero coefficients, it can be used for variable selection
- As such, we fit a simple logistic regression model based on the variables selected by the glmnet model: nox, rad and age (which was then dropped as it was insignificant)
- Lastly, we fit a random forest model just for fun
- Based on the ROC dot plot (the x-axis is actually AUC), surprisingly the simple logistic regression model based on nox and rad performed the best and is our final selected model

- You can also see the improvement as variables were removed; also note how well the glmnet and rf models performed without manual tuning

```r
library(caret)
library(caretEnsemble)

set.seed(123)
# use cross validation to compare out-of-sample ROC for all models
# use the same folds for each model to ensure comparable results
myFolds = createFolds(crime$target, k = 10)

# confirm that folks preserve class distribution
table(crime$target) / nrow(crime)
```

```
##
## below_median above_median
##    0.5085837    0.4914163
```

```r
table(crime$target[myFolds$Fold1]) / length(myFolds$Fold1)
```

```
##
## below_median above_median
##    0.5106383    0.4893617
```

```r
myControl = trainControl(summaryFunction = twoClassSummary, classProbs = TRUE, verboseIter = FALSE,
                         savePredictions = TRUE, index = myFolds)  # used instead of method = 'cv', num

# model using glm model
model_glm_full = train(target ~ ., data = crime, metric = 'ROC', method = 'glm',
                  preProcess = c('center', 'scale'), trControl = myControl)
summary(model_glm_full)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2854  -0.1372  -0.0017   0.0020   3.4721
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.6213     0.7387   3.549 0.000387 ***
## zn            -1.4421     0.8040  -1.794 0.072868 .
## indus         -0.4969     0.3323  -1.495 0.134894
## chasbordered   0.2651     0.1951   1.359 0.174139
## nox            5.8519     0.9391   6.231 4.62e-10 ***
## rm            -0.4879     0.5226  -0.934 0.350548
## age            0.9777     0.3932   2.487 0.012895 *
## dis            1.6135     0.4939   3.267 0.001087 **
## rad            5.7589     1.4343   4.015 5.94e-05 ***
## tax           -1.1070     0.5145  -2.152 0.031422 *
## ptratio        0.9715     0.2905   3.344 0.000825 ***
## black         -1.1957     0.6100  -1.960 0.049974 *
## lstat          0.3378     0.3871   0.873 0.382802
## medv           1.8455     0.6562   2.812 0.004919 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 186.15  on 452  degrees of freedom
## AIC: 214.15
##
## Number of Fisher Scoring iterations: 9
```

```
# let's drop any statistically insignificant variables at 5%
model_glm_sig1 = train(target ~ nox + age + dis + rad + tax + ptratio + black + medv, data = crime,
                       metric = 'ROC', method = 'glm', preProcess = c('center', 'scale'), trControl = my
summary(model_glm_sig1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.42422  -0.19292  -0.01400   0.00279   3.06740
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.0772     0.6630   4.642 3.46e-06 ***
## nox           4.9187     0.7787   6.317 2.67e-10 ***
## age           0.8784     0.3025   2.904 0.003684 **
## dis           0.9224     0.3635   2.538 0.011165 *
## rad           6.1101     1.2186   5.014 5.33e-07 ***
## tax          -1.4680     0.4385  -3.348 0.000813 ***
## ptratio       0.8690     0.2471   3.517 0.000437 ***
## black        -1.1406     0.6173  -1.848 0.064662 .
## medv          0.9348     0.3152   2.966 0.003020 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 198.28  on 457  degrees of freedom
## AIC: 216.28
##
## Number of Fisher Scoring iterations: 9
```

```
# let's drop any additional statistically insigificant variables at 5%
model_glm_sig2 = train(target ~ nox + age + dis + rad + tax + ptratio + medv, data = crime,
                       metric = 'ROC', method = 'glm', preProcess = c('center', 'scale'), trControl = my
summary(model_glm_sig2)
```

```
##
## Call:
## NULL
##
```

```
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.01059  -0.19744  -0.01371   0.00402   3.06424
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.8134     0.6425   4.379 1.19e-05 ***
## nox           4.9395     0.7746   6.377 1.81e-10 ***
## age           0.9030     0.3028   2.982 0.002867 **
## dis           0.9051     0.3621   2.500 0.012433 *
## rad           6.0955     1.2110   5.033 4.82e-07 ***
## tax          -1.3830     0.4255  -3.250 0.001153 **
## ptratio       0.8273     0.2393   3.458 0.000545 ***
## medv          0.8653     0.3100   2.791 0.005255 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 203.45  on 458  degrees of freedom
## AIC: 219.45
##
## Number of Fisher Scoring iterations: 9
```

```r
# it looks like there are still some variables to drop; let's drop any additional statistically insigif
model_glm_sig3 = train(target ~ nox + age + rad + tax + ptratio + medv, data = crime,
                       metric = 'ROC', method = 'glm', preProcess = c('center', 'scale'), trControl = my
summary(model_glm_sig3)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.98644  -0.22493  -0.01416   0.00378   2.84776
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.6685     0.6361   4.195 2.73e-05 ***
## nox           3.9422     0.6099   6.464 1.02e-10 ***
## age           0.6439     0.2750   2.341 0.019227 *
## rad           6.2574     1.2121   5.162 2.44e-07 ***
## tax          -1.5056     0.4332  -3.476 0.000509 ***
## ptratio       0.7242     0.2321   3.120 0.001807 **
## medv          0.5684     0.2666   2.132 0.033014 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 209.55  on 459  degrees of freedom
## AIC: 223.55
```

```
##
## Number of Fisher Scoring iterations: 8
```

```r
# let's try a glmnet model that combines ridge vs. lasso regression
# since it penalizes either or both magnitude and number of non-zero coefficients, it can be used for v
model_glmnet = train(target ~ ., data = crime, metric = 'ROC', method = 'glmnet',
                     preProcess = c('center', 'scale'), trControl = myControl)
coef(model_glmnet$finalModel, s = model_glmnet$finalModel$tuneValue$lambda)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.09432227
## zn           .
## indus        .
## chasbordered .
## nox          1.19548163
## rm           .
## age          0.21242418
## dis          .
## rad          0.52642418
## tax          .
## ptratio      .
## black        .
## lstat        .
## medv         .
```

```r
# let's build a simple linear model based on the variables selected by the glmnet model
# here no penalty terms or regularization is introduced
model_glm_sig4 = train(target ~ nox + rad, data = crime,
                       metric = 'ROC', method = 'glm', preProcess = c('center', 'scale'), trControl = my
summary(model_glm_sig4)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8769  -0.3447  -0.0692   0.0068   2.5803
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.5191     0.5482   4.596 4.32e-06 ***
## nox           3.1729     0.3770   8.415  < 2e-16 ***
## rad           4.4633     0.9397   4.750 2.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 239.51  on 463  degrees of freedom
## AIC: 245.51
##
## Number of Fisher Scoring iterations: 8
```

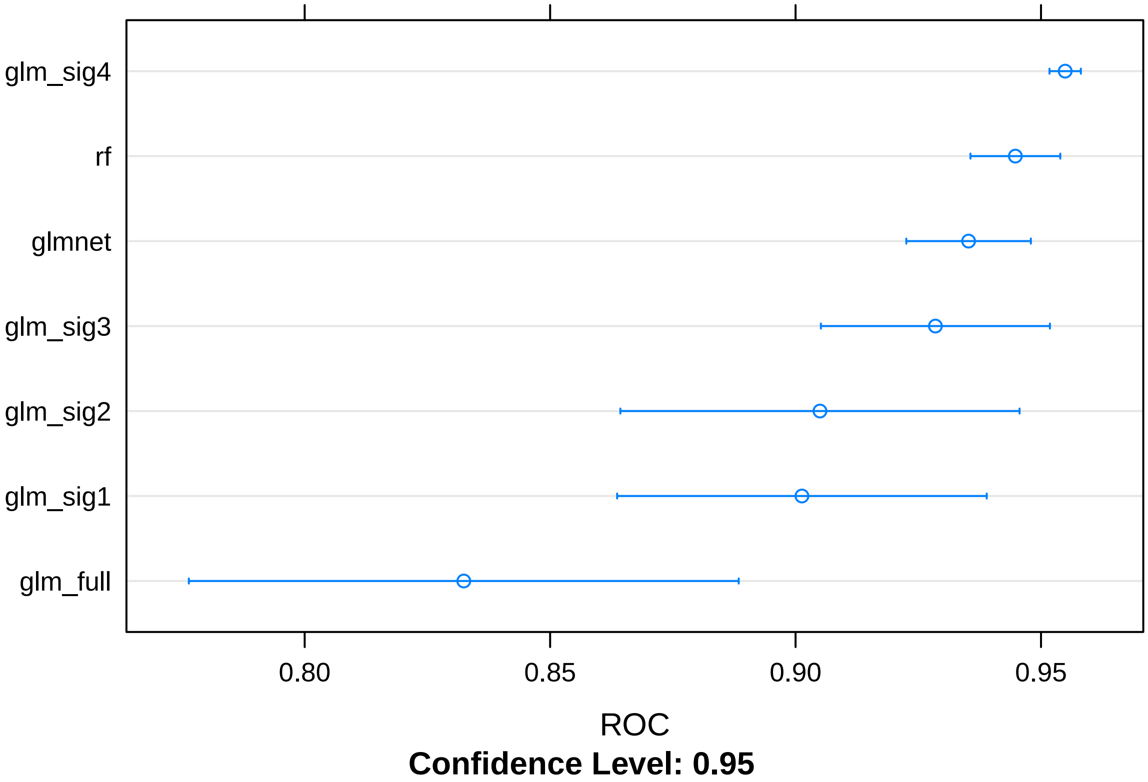
```r
# let's also model using random forest just for fun
model_rf = train(target ~ ., data = crime, metric = 'ROC', method = 'ranger',
                 trControl = myControl)

# compare models
model_list = list(glm_full = model_glm_full, glm_sig1 = model_glm_sig1, glm_sig2 = model_glm_sig2, glm_s
                  glm_sig4 = model_glm_sig4, glmnet = model_glmnet, rf = model_rf)

# collect resamples from the CV folds
resamps = resamples(model_list)
summary(resamps)
```

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: glm_full, glm_sig1, glm_sig2, glm_sig3, glm_sig4, glmnet, rf
## Number of resamples: 10
##
## ROC
##             Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## glm_full 0.6854  0.8001 0.8388 0.8324  0.8678 0.9561    0
## glm_sig1 0.8289  0.8479 0.9167 0.9013  0.9456 0.9666    0
## glm_sig2 0.8086  0.8730 0.9353 0.9050  0.9443 0.9670    0
## glm_sig3 0.8719  0.9080 0.9400 0.9285  0.9468 0.9665    0
## glm_sig4 0.9482  0.9525 0.9547 0.9549  0.9562 0.9647    0
## glmnet   0.8911  0.9338 0.9367 0.9352  0.9444 0.9531    0
## rf       0.9274  0.9361 0.9423 0.9448  0.9546 0.9665    0
##
## Sens
##             Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## glm_full 0.6244  0.7333 0.8341 0.8040  0.8732 0.9249    0
## glm_sig1 0.7371  0.8300 0.8876 0.8762  0.9484 0.9673    0
## glm_sig2 0.6714  0.8475 0.9108 0.8780  0.9485 0.9673    0
## glm_sig3 0.7230  0.8462 0.9206 0.8875  0.9343 0.9765    0
## glm_sig4 0.6808  0.8760 0.9134 0.8823  0.9249 0.9579    0
## glmnet   0.7793  0.8746 0.9040 0.8954  0.9379 0.9765    0
## rf       0.7746  0.8405 0.8806 0.8781  0.9190 0.9484    0
##
## Spec
##             Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## glm_full 0.6311  0.7184 0.8180 0.7772  0.8386 0.8841    0
## glm_sig1 0.7136  0.7767 0.8495 0.8355  0.8936 0.9466    0
## glm_sig2 0.7087  0.7767 0.8277 0.8297  0.8994 0.9272    0
## glm_sig3 0.7573  0.7816 0.8325 0.8486  0.9116 0.9757    0
## glm_sig4 0.7476  0.7685 0.8301 0.8244  0.8386 0.9709    0
## glmnet   0.7039  0.7427 0.7670 0.7763  0.8131 0.8641    0
## rf       0.7573  0.7828 0.8329 0.8389  0.8871 0.9515    0
```

```r
dotplot(resamps, metric = 'ROC')
```



SELECT MODEL

- The final model was selected because it performed the best, is very simple and is highly intuitive
- Since the coefficients for both nox and rad are positive, it suggests that higher nox levels and being closer to highways are associated with higher rates of crime (which makes sense intuitively)
- After the final model was selected, we then re-fit the model to the entire data set (i.e. no cross validation) to ensure that we maximize use of all the available data
- The final model is then used to predict the classes and probabilities on the test data

```
final_model = train(target ~ nox + rad, data = crime, metric = 'ROC', method = 'glm', preProcess = c('c
                    trControl = trainControl(summaryFunction = twoClassSummary, classProbs = TRUE, verb

summary(model_glm_sig4)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8769  -0.3447  -0.0692   0.0068   2.5803
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.5191     0.5482   4.596 4.32e-06 ***
```

```
## nox           3.1729     0.3770   8.415  < 2e-16 ***
## rad           4.4633     0.9397   4.750 2.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 239.51  on 463  degrees of freedom
## AIC: 245.51
##
## Number of Fisher Scoring iterations: 8
```

```r
crime_test = read.csv('crime-evaluation-data.csv')
pred_class = predict(final_model, newdata = crime_test)
pred_prob = predict(final_model, newdata = crime_test, type = 'prob')
write.csv(cbind(pred_class, pred_prob), 'crime_evaluation-prediction.csv')

# assuming the evaluation data is similarly class balanced, the predictions do not seem unreasonable
table(pred_class) / length(pred_class)
```

```
## pred_class
## below_median above_median
##        0.575        0.425
```