# Homework #4: Classification

**Hyunsuk Ko**

Due: Wed Sept 28 | 11:45am

**DS 6030 | Fall 2022 | University of Virginia**

This is an **independent assignment**. Do not discuss or work with classmates.

## Required R packages and Directories

```
data.dir = 'https://mdporter.github.io/DS6030/data/' # data directory
library(R6030)     # functions for SYS-6030
library(tidyverse) # functions for data manipulation
library(glmnet)
library(yardstick)
library(dplyr)
```

## Crime Linkage

Crime linkage attempts to determine if two or more unsolved crimes share a common offender. *Pairwise* crime linkage is the more simple task of deciding if two crimes share a common offender; it can be considered a binary classification problem. The linkage training data has 8 evidence variables that measure the similarity between a pair of crimes:

- `spatial` is the spatial distance between the crimes
- `temporal` is the fractional time (in days) between the crimes
- `tod` and `dow` are the differences in time of day and day of week between the crimes
- `LOC`, `POA`, and `MOA` are binary with a 1 corresponding to a match (type of property, point of entry, method of entry)
- `TIMERANGE` is the time between the earliest and latest possible times the crime could have occurred (because the victim was away from the house during the crime).
- The response variable indicates if the crimes are linked ($y = 1$) or unlinked ($y = 0$).

These problems use the linkage-train (https://mdporter.github.io/DS6030/data/linkage_train.csv) and linkage-test (https://mdporter.github.io/DS6030/data/linkage_test.csv) datasets (click on links for data).

```
train = read.csv('linkage_train.csv')
test = read.csv('linkage_test.csv')

X = glmnet::makeX(
  train = train %>% select(-y),
  test = test
)

X.train = X$x
Y.train = train$y
X.test = X$xtest
```

## Problem 1: Penalized Regression for Crime Linkage

a. Fit a penalized *linear regression* model to predict linkage. Use a lasso, ridge, or elasticnet penalty (your choice).

- Report the value of $\alpha$ used (if elasticnet)
- Report the value of $\lambda$ used
- Report the estimated coefficients

```
set.seed(2022)
ridge_cv <- cv.glmnet(X.train, Y.train, alpha = 0, nfolds = 10)
ridge_cv$lambda.min
```

```
#> [1] 0.002327
```

```
coef(ridge_cv, s = "lambda.min")
```

```
#> 9 x 1 sparse Matrix of class "dgCMatrix"
#>                     s1
#> (Intercept)  9.263e-02
#> spatial     -2.319e-03
#> temporal    -1.548e-04
#> tod         -2.213e-03
#> dow         -5.795e-03
#> LOC          4.263e-02
#> POA          9.321e-03
#> MOA          7.297e-03
#> TIMERANGE    2.499e-05
```

```
yhat_ridge = predict(ridge_cv, X.test, s = "lambda.min")
```

b. Fit a penalized *logistic regression* model to predict linkage. Use a lasso, ridge, or elasticnet penalty (your choice).

- Report the value of $\alpha$ used (if elasticnet)
- Report the value of $\lambda$ used
- Report the estimated coefficients

```r
set.seed(2022)
# alpha = 0.5
fit.enet = cv.glmnet(X.train, Y.train, alpha = 0.5, family = "binomial")
fit.enet$lambda.min
```

```
#> [1] 6.912e-05
```

```r
coef(fit.enet, s = "lambda.min")
```

```
#> 9 x 1 sparse Matrix of class "dgCMatrix"
#>                    s1
#> (Intercept) -0.038162
#> spatial     -0.284236
#> temporal    -0.012414
#> tod         -0.113157
#> dow         -0.240099
#> LOC          1.388922
#> POA          0.441772
#> MOA          0.212107
#> TIMERANGE    0.000615
```

```r
yhat_enet = predict(fit.enet, X.test, s = "lambda.min")
```

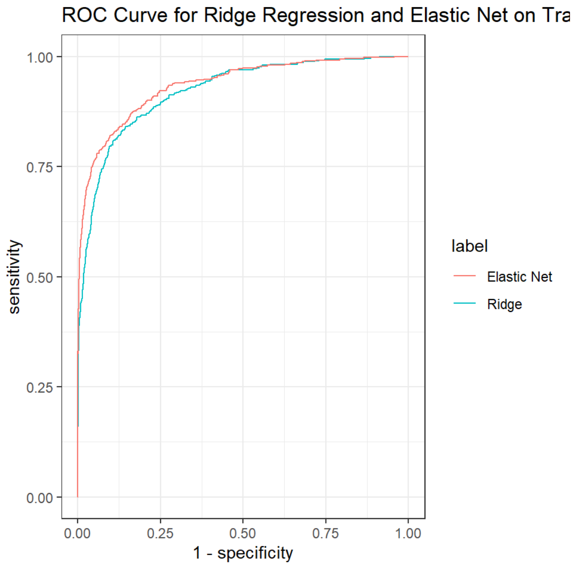
c. Produce one plot that has the ROC curves, using the *training data*, for both models (from part a and b). Use color and/or linetype to distinguish between models and include a legend.

```r
gamma_ridge = predict(ridge_cv, X.train, s = "lambda.min", type = 'link')
gamma_enet = predict(fit.enet, X.train, s = "lambda.min", type = 'link')

ROC_ridge = tibble(truth = factor(Y.train, levels=c(1,0)), gamma = gamma_ridge[,
1]) %>%
  yardstick::roc_curve(truth, gamma) %>%
  mutate(label = "Ridge")

ROC_enet = tibble(truth = factor(Y.train, levels=c(1,0)), gamma = gamma_enet[,1
]) %>%
  yardstick::roc_curve(truth, gamma) %>%
  mutate(label = "Elastic Net")

ggplot() +
  geom_line(data = ROC_ridge, aes(x = 1-specificity, y = sensitivity, color = la
bel)) +
  geom_line(data = ROC_enet, aes(x = 1-specificity, y = sensitivity, color = lab
el)) +
  labs(title = "ROC Curve for Ridge Regression and Elastic Net on Train Data")
```




d. Recreate the ROC curve from the penalized logistic regression model using repeated hold-out data. The following steps will guide you:

- Fix $\alpha = .75$
- Run the following steps 25 times:
    i. Hold out 500 observations
    ii. Use the remaining observations to estimate $\lambda$ using 10-fold CV
    iii. Predict the probability of linkage for the 500 hold-out observations
    iv. Store the predictions and hold-out labels
- Combine the results and produce the hold-out based ROC curve
- Note: by estimating $\lambda$ each iteration, we are incorporating the uncertainty present in estimating that tuning parameter.

```
set.seed(2022)

alpha = .75
M = 1#25
K = 10
x = nrow(X.train)
n = 500

holdout_labels = c()
lambdas = c()
linkages = c()
truths = c()
gammas = c()

for (i in 1:M) {
  ind = sample(x, size = n, replace = FALSE)
  holdout = X.train[ind,]
  x_train = X.train[-ind,]
  y_train = Y.train[-ind]

  fit.enet = cv.glmnet(x_train, y_train, alpha = alpha, family = "binomial")
  lambda = fit.enet$lambda.min
  gamma_enet = predict(fit.enet, X.train, s = "lambda.min", type = 'link')

  holdout_labels = append(holdout_labels, i)
  lambdas = append(lambdas, lambda)
  linkages = append(linkages, gamma_enet[,1])
  truths = append(truths, factor(Y.train, levels=c(1,0)))
  gammas = append(gammas, gamma_enet[,1])
}

ROC_tibble = tibble(truth = truths, gamma = gammas)

ROC_enet = ROC_tibble %>%
  yardstick::roc_curve(truth, gamma)
```

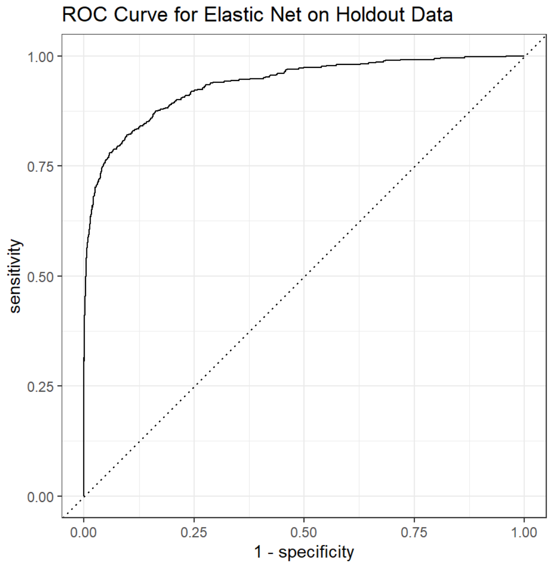
```
ROC_enet
```

| .threshold <dbl> | specificity <dbl> | sensitivity <dbl> |
|---:|---:|---:|
| -Inf | 0.0000000 | 1.000 |
| -21.265 | 0.0000000 | 1.000 |
| -19.876 | 0.0000339 | 1.000 |
| -19.791 | 0.0000678 | 1.000 |
| -19.643 | 0.0001017 | 1.000 |
| -19.444 | 0.0001356 | 1.000 |
| -19.321 | 0.0001695 | 1.000 |
| -19.295 | 0.0002034 | 1.000 |

| .threshold <dbl> | specificity <dbl> | sensitivity <dbl> |
|---|---|---|
| -18.974 | 0.0002373 | 1.000 |
| -18.879 | 0.0002712 | 1.000 |

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 ... 1000 Next

```
ROC_enet %>%
  ggplot(aes(1-specificity, sensitivity)) + geom_line() +
  geom_abline(lty = 3) +
  coord_equal() +
  labs(title = "ROC Curve for Elastic Net on Holdout Data")
```




```
#table(predicted = ROC_tibble$gamma, truth = ROC_tibble$truth) %>% addmargins()
```

e. Contest Part 1: Predict the estimated *probability* of linkage for the test data (using any model).

- Submit a .csv file (ensure comma separated format) named `lastname_firstname_1.csv` that includes the column named **p** that is your estimated posterior probability. We will use automated evaluation, so the format must be exact.
- You are free to use any tuning parameters
- You are free to use any data transformation or feature engineering
- You will receive credit for a proper submission; the top five scores will receive 2 bonus points.
- Your probabilities will be evaluated with respect to the mean negative Bernoulli log-likelihood (known as the average *log-loss* metric)

$$L = -\frac{1}{M}\sum_{i=1}^{m}[y_i \log \hat{p}_i + (1 - y_i) \log (1 - \hat{p}_i)]$$

where $M$ is the number of test observations, $\hat{p}_i$ is the prediction for the $i$th test observation, and $y_i \in \{0, 1\}$ are the true test set labels.

```
fit.lm = glm(y ~ spatial + temporal + tod + dow + LOC + POA + TIMERANGE, family
= 'binomial', data = train)

p.hat = predict(fit.lm, test, type = 'response')

my_phat <- data.frame(p = p.hat)
my_phat
```

| | p <dbl> |
|---|---|
| 1 | 6.287e-02 |
| 2 | 8.713e-03 |
| 3 | 7.362e-02 |
| 4 | 1.257e-05 |
| 5 | 5.127e-04 |
| 6 | 1.043e-02 |
| 7 | 3.609e-02 |
| 8 | 1.852e-02 |
| 9 | 1.216e-02 |
| 10 | 2.843e-05 |

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 ... 1000 Next

```
write_csv(my_phat, 'ko_hyunsuk_1.csv')
```

f. Contest Part 2: Predict the linkages for the test data (using any model).

- Submit a .csv file (ensure comma separated format) named `lastname_firstname_2.csv` that includes the column named **linkage** that takes the value of 1 for linkages and 0 for unlinked pairs. We will use automated evaluation, so the format must be exact.
- You are free to use any tuning parameters.
- You are free to use any data transformation or feature engineering.
- Your labels will be evaluated based on total cost, where cost is equal to `1*FP + 8*FN`. This implies that False Negatives (FN) are 8 times as costly as False Positives (FP)
- You will receive credit for a proper submission; the top five scores will receive 2 bonus points. Note: you only will get bonus credit for one of the two contests.

```
G.hat = ifelse(p.hat >= .1, 1, 0)

my_ghat <- data.frame(linkage = G.hat)
my_ghat
```

| | linkage <dbl> |
|---|---|

| | linkage <dbl> |
|---|---|
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |
| 8 | 0 |
| 9 | 0 |
| 10 | 0 |

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 ... 1000 Next

```
write_csv(my_ghat, 'ko_hyunsuk_2.csv')
```