homework5_isye6501

Zach Olivier

6/9/2018

Question 11.1

Question:

Using the crime data set uscrime.txt from Questions 8.2, 9.1, and 10.1, build a regression model using: 1. Stepwise regression 2. Lasso 3. Elastic net For Parts 2 and 3, remember to scale the data first - otherwise, the regression coefficients will be on different scales and the constraint won't have the desired effect. For Parts 2 and 3, use the glmnet function in R

Answer:

Below are my steps to fit a stepwise regression, lasso regression, and elastic net regression to the crime data set.

We start by reading in the data and then transform the data using caret's PreProcess function. Then I develop training and test data sets to prepare for fitting each model. We can use the same training data to fit each model and judge their relative performance via cross validation (k = 10).

Output from each model and the best tuned model's coefficients are shown below.

Here are the results:

```
- Stepwise: R^2 = 82%
- LASSO: R^2 = 81% (alpha = 1, lambda = .8)
- Elastic Net: R^2 = 80% (aplha = .5, lambda = 5.5)
```

Overall it seems the stepwise regression performs the best on our training dataset. It will be interesting to see if these results generalize to new data, the regularization of the LASSO and Elastic Net may boost test set accuracy. It is also interesting that in both the LASSO and Elastic Net model regularization is fairly small. Time is the only variable that is taken completely to zero by LASSO and Elastic Net with such a small regularization penalty. Plots of the regularized factor coefficients are shown below.

```
set.seed(110)

# read in the crime data
crime_df = read_delim('11.1uscrimeSummer2018.txt', delim = '\t') %>%
        as.data.frame()


# using cool caret functions to transform predictor data
transform_df = caret::preProcess(
        crime_df %>% dplyr::select(., -Crime),
        method = c('center', 'scale', 'nzv')
        )

crime_mod_df <- predict(transform_df, crime_df)


# set up train and testing split
train <- createDataPartition(crime_mod_df$Crime, p = .75, list = F)

# set up test and train datasets
crime_train <- crime_mod_df[train,]
crime_test <- crime_mod_df[-train,]

# check splits
dim(crime_train); dim(crime_test)
```

```
## [1] 36 16
```

```
## [1] 11 16
```

```
set.seed(110)


# fit model - stepwise
crime_step <- train(
        Crime ~ .,
        data = crime_train,
        method = 'glmStepAIC',
        trControl = trainControl(method = 'cv'),
        trace = 0
        )

coef(crime_step$finalModel)
```

```
## (Intercept)           M          So          Ed         Po1         Po2
##   877.36279   157.94437   105.32763   323.38881   883.66350  -526.54502
##         Pop          U2      Wealth        Ineq        Prob
##   -59.17099   106.40983   -98.18501   184.62981   -88.00276
```

```
crime_step$results
```

```
##   parameter     RMSE  Rsquared      MAE   RMSESD RsquaredSD    MAESD
## 1      none 194.9725 0.8204522 163.2867 100.6364  0.2089959 89.10375
```

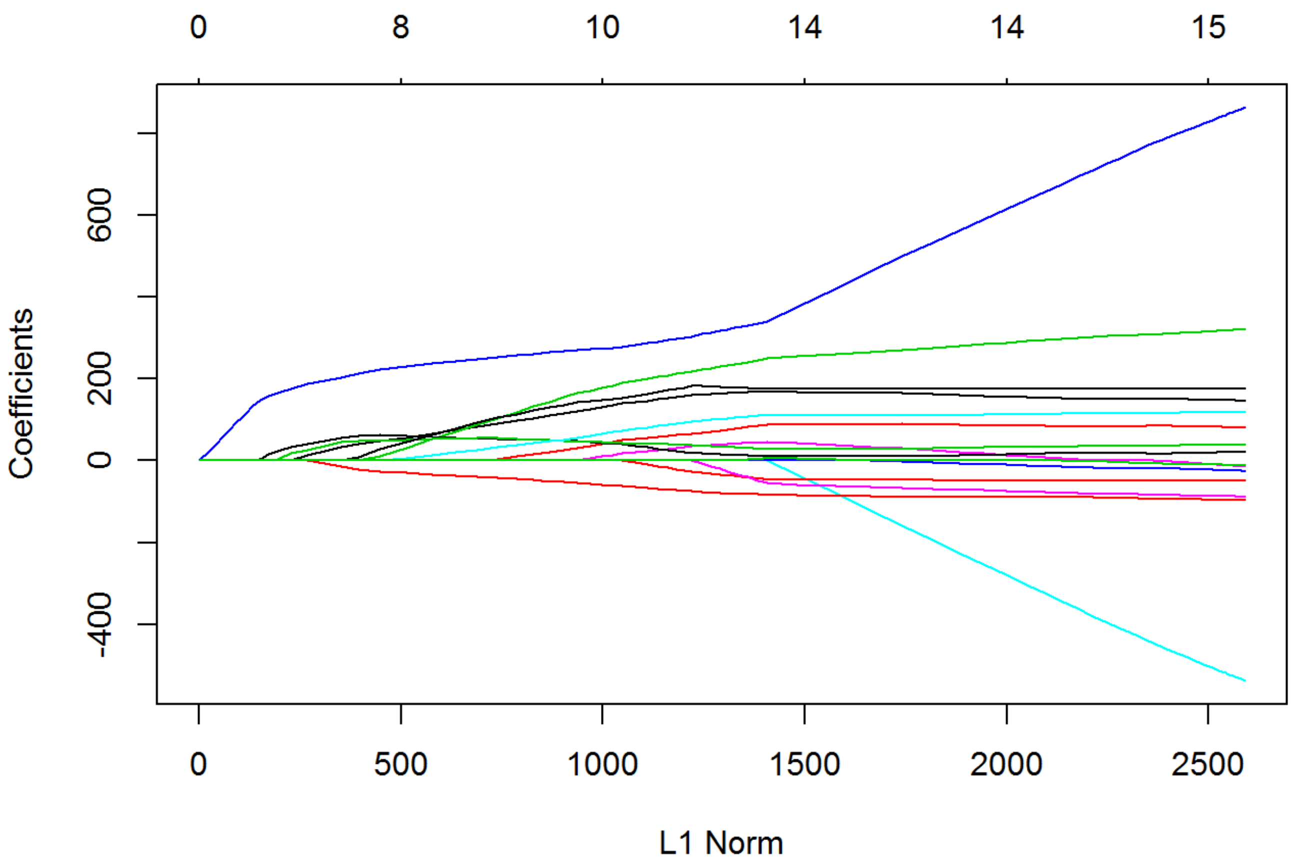
```
set.seed(110)


# fit model - lasso
crime_lasso <- train(
        Crime ~ .,
        data = crime_train,
        method = 'glmnet',
        trControl = trainControl(method = 'cv'),
        tuneGrid=expand.grid(
              .alpha=1,
              .lambda=seq(0, 100, by = .1))
        )

# lasso coefficients with best tuned regularization parameter lambda (alpha hard co
ded at 1 for lasso)
coef(crime_lasso$finalModel, crime_lasso$bestTune$lambda==crime_lasso$bestTune$lamb
da)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  883.730647
## M            154.547717
## So            84.916199
## Ed           293.854327
## Po1          649.230992
## Po2         -315.338636
## LF             9.954312
## M.F           17.324512
## Pop          -48.224798
## NW            33.870001
## U1           -12.756266
## U2           114.287282
## Wealth       -76.332428
## Ineq         176.200935
## Prob         -88.512846
## Time           .
```

```
# best tune performance
crime_lasso$results %>% as_tibble() %>%
  filter(
    lambda == crime_lasso$bestTune$lambda
    )
```

```
## # A tibble: 1 x 8
##   alpha lambda  RMSE Rsquared   MAE RMSESD RsquaredSD MAESD
##   <dbl>  <dbl> <dbl>    <dbl> <dbl>  <dbl>      <dbl> <dbl>
## 1     1    4.1  200.    0.812  167.   86.3      0.239  77.4
```

```
plot(crime_lasso$finalModel)
```



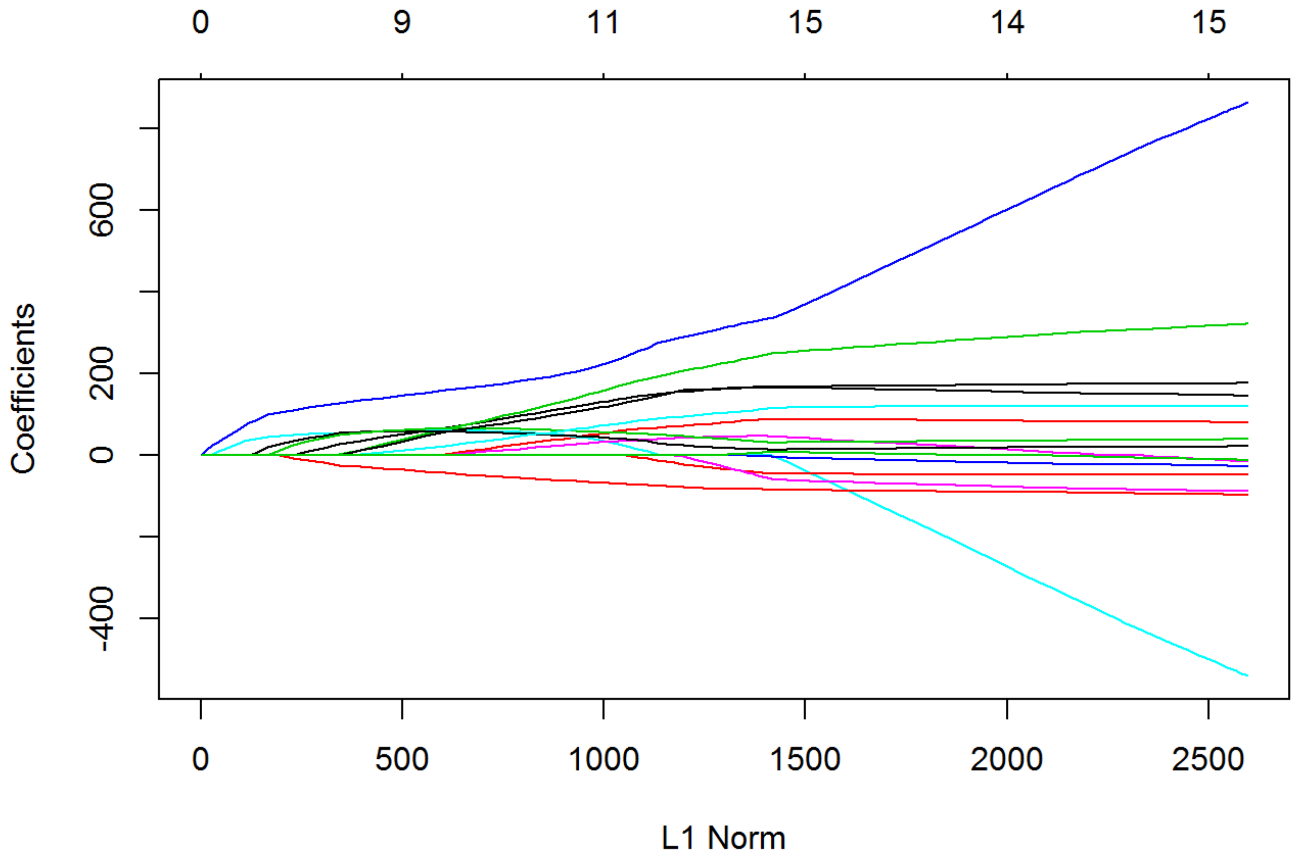
```
set.seed(110)


# fit model - elastic net
crime_elastic <- train(
        Crime ~ .,
        data = crime_train,
        method = 'glmnet',
        trControl = trainControl(method = 'cv'),
        tuneGrid=expand.grid(
              .alpha= .5,
              .lambda=seq(0, 100, by = .1))
        )

# elastic net coefficients with best tuned regularization parameter lambda and alph
a
coef(
  crime_elastic$finalModel,
  crime_elastic$bestTune$lambda==crime_elastic$bestTune$lambda &
  crime_elastic$bestTune$alpha == crime_elastic$bestTune$alpha
     )
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept)  884.31429
## M            154.85813
## So            84.43070
## Ed           293.09784
## Po1          621.65015
## Po2         -291.48846
## LF            12.21340
## M.F           19.82396
## Pop          -47.24458
## NW            36.05988
## U1           -18.88317
## U2           120.94781
## Wealth       -77.46653
## Ineq         173.30558
## Prob         -89.69596
## Time           .
```

```
# best tune performance
crime_elastic$results %>% as_tibble() %>%
  filter(
    alpha == crime_elastic$bestTune$alpha,
    lambda == crime_elastic$bestTune$lambda
    )
```

```
## # A tibble: 1 x 8
##   alpha lambda  RMSE Rsquared   MAE RMSESD RsquaredSD MAESD
##   <dbl>  <dbl> <dbl>    <dbl> <dbl>  <dbl>      <dbl> <dbl>
## 1   0.5    5.5  201.    0.804  169.   85.1      0.238  74.4
```

```
plot(crime_elastic$finalModel)
```



Question 12.1

Question:

Describe a situation or problem from your job, everyday life, current events, etc., for which a design of experiments approach would be appropriate.

Answer:

One situation I can imagine is combining a model's output with a carefully designed study to learn more about our target population. We can develop a model that scores customers with probabilities of purchase. Then we can market to these customers with the highest probability. This seems great, but what if those customers were most likely to purchase anyway, and we wasted marketing on 'sure things'? Experiment design can help us with this problem. We can group our customers into probability bands and see which probability group is more responsive to marketing emails. It may turn out that we should market to customers in the 40% range - they are on the edge of purchase and need marketing to persuade them. This combination of model output and a cleverly designed experiment and be really powerful.

Question 12.2

Question:

To determine the value of 10 different yes/no features to the market value of a house (large yard, solar roof, etc.), a real estate agent plans to survey 50 potential buyers, showing a fictitious house with different combinations of features. To reduce the survey size, the agent wants to show just 16 fictitious houses. Use R's FrF2 function (in the FrF2 package) to find a fractional factorial design for this experiment: what set of features should each of the 16 fictitious houses have? Note: the output of FrF2 is "1" (include) or "-1" (don't include) for each feature

Answer:

Below is the implementation of the fractional factorial design for 16 open houses based on 10 features. Output below shows which house should have which features to show to prospective house buyers.

```
set.seed(40)

(houses <- FrF2(
  16, 10,
  factor.names = c(
    'Large Yard', 'Pool', 'Solar Roof', 'Long Driveway', 'Multi Car Garage',
    'Walk-In Closet', 'Man Cave', 'Full Bar', 'Gazebo', 'Elevator'
    ),
  default.levels = c('Yes', 'No')
  ) %>%
  as_tibble() %>%
  rownames_to_column('House')
 )
```

```
## # A tibble: 16 x 11
##    House Large.Yard Pool  Solar.Roof Long.Driveway Multi.Car.Garage
##    <chr> <fct>      <fct> <fct>      <fct>         <fct>
##  1 1     Yes        No    Yes        No            Yes
##  2 2     No         Yes   No         No            Yes
##  3 3     No         Yes   Yes        No            Yes
##  4 4     No         Yes   Yes        Yes           Yes
##  5 5     Yes        No    Yes        Yes           Yes
##  6 6     No         Yes   No         Yes           Yes
##  7 7     No         No    Yes        No            No
##  8 8     No         No    No         No            No
##  9 9     Yes        No    No         No            Yes
## 10 10    Yes        Yes   Yes        Yes           No
## 11 11    Yes        No    No         Yes           Yes
## 12 12    No         No    Yes        Yes           No
## 13 13    Yes        Yes   Yes        No            No
## 14 14    Yes        Yes   No         Yes           No
## 15 15    Yes        Yes   No         No            No
## 16 16    No         No    No         Yes           No
## # ... with 5 more variables: Walk.In.Closet <fct>, Man.Cave <fct>,
## #   Full.Bar <fct>, Gazebo <fct>, Elevator <fct>
```

Question 13.1

Question:

For each of the following distributions, give an example of data that you would expect to follow this distribution (besides the examples already discussed in class). a. Binomial b. Geometric c. Poisson d. Exponential e. Weibull

Answer:

Examples of the distributions above include:

```
- Binomial: probability of acceptance to a prestigious college (success p = .15)
- Geometric: probability of passing an exam on a certain attempt (i.e we keep taking it the exam until we pass)
- Poisson: number of visits to a certain webpage for given time period
- Exponential: times between when cars arrive to the work parking garage
- Weibull: how long will it take for my car battery to fail
```