HW Lasso Reg, Elastic Reg

Me

March 5, 2025

loading data

```
mydata <- read.table("C:/ISYE6040/data/data 8.2/uscrime.txt", header = TRUE)
set.seed(123)
#install.packages("glmnet")
library(glmnet)

## Warning: package 'glmnet' was built under R version 4.4.3

## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

creating lm model

```
lm.model <- lm(Crime~., data = mydata)
r2_lm <- summary(lm.model)$r.squared
r2_lm_adj <- summary(lm.model)$adj.r.squared
summary(lm.model)

##
## Call:
## lm(formula = Crime ~ ., data = mydata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -395.74  -98.09   -6.69  112.99  512.67
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
## M            8.783e+01  4.171e+01   2.106 0.043443 *
## So          -3.803e+00  1.488e+02  -0.026 0.979765
## Ed           1.883e+02  6.209e+01   3.033 0.004861 **
## Po1          1.928e+02  1.061e+02   1.817 0.078892 .
## Po2         -1.094e+02  1.175e+02  -0.931 0.358830
## LF          -6.638e+02  1.470e+03  -0.452 0.654654
## M.F          1.741e+01  2.035e+01   0.855 0.398995
## Pop         -7.330e-01  1.290e+00  -0.568 0.573845
## NW           4.204e+00  6.481e+00   0.649 0.521279
## U1          -5.827e+03  4.210e+03  -1.384 0.176238
## U2           1.678e+02  8.234e+01   2.038 0.050161 .
## Wealth       9.617e-02  1.037e-01   0.928 0.360754
```

```
## Ineq         7.067e+01  2.272e+01   3.111 0.003983 **
## Prob        -4.855e+03  2.272e+03  -2.137 0.040627 *
## Time        -3.479e+00  7.165e+00  -0.486 0.630708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.1 on 31 degrees of freedom
## Multiple R-squared:  0.8031, Adjusted R-squared:  0.7078
## F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

performing Stepwise Regression on the lm model to add and remove predictors

```
stepwise.model <- step(lm.model, direction = 'both')
r2_stepwise <- summary(stepwise.model)$r.squared
r2_stepwise_adj <- summary(stepwise.model)$adj.r.squared

summary(stepwise.model)
```

```
##
## Call:
## lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,
##     data = mydata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -444.70 -111.07    3.03  122.15  483.30
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6426.10    1194.61  -5.379 4.04e-06 ***
## M              93.32      33.50   2.786  0.00828 **
## Ed            180.12      52.75   3.414  0.00153 **
## Po1           102.65      15.52   6.613 8.26e-08 ***
## M.F            22.34      13.60   1.642  0.10874
## U1          -6086.63    3339.27  -1.823  0.07622 .
## U2            187.35      72.48   2.585  0.01371 *
## Ineq           61.33      13.96   4.394 8.63e-05 ***
## Prob        -3796.03    1490.65  -2.547  0.01505 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 195.5 on 38 degrees of freedom
## Multiple R-squared:  0.7888, Adjusted R-squared:  0.7444
## F-statistic: 17.74 on 8 and 38 DF,  p-value: 1.159e-10
```

notice how the stepwise model reduced the number of predictors and increased the adjusted R-squared value. We have a simpler model that performs better and that is less prone to overfitting random patterns.

Seperating data and perfoming Lasso Regression and cross-validation

```
x <- as.matrix(mydata[,1:15])
y <- mydata[,16]
x_scaled = scale(x)
lasso_model <- glmnet(x_scaled,y, alpha = 1)
summary(lasso_model)
cv_lasso <- cv.glmnet(x_scaled,y, alpha = 1)
summary(cv_lasso)
```

#finding the best lambda and re-running model

```
best_lambda_lasso <- cv_lasso$lambda.min
best_lasso <- glmnet(x_scaled,y, alpha = 1, lambda = best_lambda_lasso)
summary(best_lasso)
```

#calculating the r-squared of the best lasso_model

```
best_lasso_predictions <- predict(best_lasso, newx = x_scaled)
rss <- sum((y - best_lasso_predictions)^2)
tss <- sum((y-mean(y))^2)
r2_lasso <- 1 - (rss/tss)
r2_lasso_adj <- 1 - (((1-r2_lasso)*(47-1))/(47-15-1))
```

#repeating steps for elastic model

```
elastic_model <- glmnet(x_scaled, y, alpha = 0.5)
cv.elastic <-cv.glmnet(x_scaled, y, alpha = 0.5)
best_lam <- cv.elastic$lambda.min
best_elastic <- glmnet(x_scaled, y, alpha = 0.5, lambda = best_lam)
elastic_pred <- predict(best_elastic, newx = x_scaled)
rss_e <- sum((y-elastic_pred)^2)
r2_elastic <- 1 - (rss_e/tss)
r2_elastic_adj <- 1 - (((1-r2_elastic)*(47-1))/(47-15-1))
```

#comparing r-squared and r-squared adjusted values

```
results <- data.frame(
  Model = c("Linear Regression", "Stepwise Regression", "Lasso Regression",
"Elastic Net Regression"),
  R2 = c(r2_lm, r2_stepwise, r2_lasso, r2_elastic),
  R2_Adjusted = c(r2_lm_adj, r2_stepwise_adj, r2_lasso_adj, r2_elastic_adj)
)
results

##                    Model        R2 R2_Adjusted
## 1      Linear Regression 0.8030868   0.7078062
## 2    Stepwise Regression 0.7888268   0.7443692
## 3       Lasso Regression 0.7937693   0.6939803
## 4 Elastic Net Regression 0.7492885   0.6279765
```

While Stepwise, Lasso and Elastic regression generally underperformed compared to the linear regression this is not necessarily a bad thing. This could be the result of the non linear regression models reducing overfitting by reducing dimensionality and colinearity. By simplfying the models using these alternate forms of regression it trades off capturing all of the variance for a more simplistic model which doesn't fit the training data as well. However, these simplier models should perfom better on unseen data than linear regression.