Conducting and Presenting Multiple Linear Regression Analysis Using R

Jishnu Matra

About the Analysis

Packages Used:

- `glmnet` - For Lasso regression
- `lm` - Ordinary least squares regression model
- `MASS` - Contains Boston dataset

Boston Housing Dataset:

- 506 observations
- 14 variables
- Target: `medv` (median home value)
- Predictors: crime rate, room number, age, etc.

Key Concepts

Coefficients: Numerical values that represent the relationship strength between each predictor variable and the target variable. A larger absolute value indicates stronger influence.

R-squared (R^2): A statistical measure (ranging from 0 to 1) that indicates the proportion of variance in the dependent variable that is predictable from the independent variables. Higher values indicate better model fit.

Penalty (L1): In Lasso regression, the penalty is the sum of absolute values of coefficients. This can force some coefficients to become exactly zero, effectively performing feature selection.

Regularization: A technique to prevent overfitting by adding a penalty term to the loss function, discouraging overly complex models with large coefficients.

Lambda ( ): The regularization strength parameter. Higher values create stronger penalties, leading to more coefficients being shrunk toward zero.

Step 1: Install & Load Packages

```
# Install and load packages
if (!require(glmnet)) {
  install.packages("glmnet")
}
library(glmnet)
library(MASS)
```

Purpose: Set up the required libraries for our analysis

Step 2: Load & Prepare Data

```
# Load and prepare data
data(Boston)
head(Boston, 3)
```

```
     crim zn indus chas   nox    rm  age    dis rad tax ptratio  black lstat
1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83  4.03
  medv
1 24.0
2 21.6
3 34.7
```

```
X <- as.matrix(Boston[, -14])
y <- Boston$medv
```

What we did: Loaded the Boston dataset and separated predictors (X) from the target variable (y)

Step 3: Fit Linear Model

```
# Fit a normal linear model
lm_model <- lm(medv ~ ., data = Boston)
lm_coef <- coef(lm_model)
lm_r_squared <- summary(lm_model)$r.squared

cat("--- Base lm() Coefficients ---\n")
```

```
--- Base lm() Coefficients ---
```

```
print(lm_coef)
```

```
  (Intercept)          crim            zn         indus          chas
 3.645949e+01 -1.080114e-01  4.642046e-02  2.055863e-02  2.686734e+00
          nox            rm           age           dis           rad
-1.776661e+01  3.809865e+00  6.922246e-04 -1.475567e+00  3.060495e-01
          tax       ptratio         black         lstat
-1.233459e-02 -9.527472e-01  9.311683e-03 -5.247584e-01
```

```
cat("\nR-squared:", lm_r_squared, "\n")
```

```

R-squared: 0.7406427
```

Step 4: Fit Lasso Model

```
# Fit a lasso model (with cross-validation)
cv_model <- cv.glmnet(X, y, alpha = 1)
best_lambda <- cv_model$lambda.min

cat("Best lambda from cross-validation:", best_lambda, "\n")
```

```
Best lambda from cross-validation: 0.007613501
```

```r
lasso_model <- glmnet(X, y, alpha = 1, lambda = best_lambda)
```

Key Point: Cross-validation automatically selects the optimal lambda value

---

Step 5: Extract Lasso Results

```r
# Extract Lasso coefficients and predictions
lasso_coef <- coef(lasso_model)
lasso_predictions <- predict(lasso_model, newx = X)
lasso_r_squared <- 1 - (sum((y - lasso_predictions)^2) /
                        sum((y - mean(y))^2))

cat("--- Lasso Coefficients ---\n")
```

```
--- Lasso Coefficients ---
```

```r
print(lasso_coef)
```

```
14 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept)  35.841545884
crim         -0.105275607
zn            0.044824451
indus         0.008841511
chas          2.699117520
nox         -17.262113725
rm            3.828134920
age           .
dis          -1.460765357
rad           0.288031707
tax          -0.011418008
ptratio      -0.944235679
black         0.009225021
lstat        -0.522848702
```

Step 6: Compare Results

```
# R-squared Comparison
cat("--- R-squared Comparison ---\n")
```

```
--- R-squared Comparison ---
```

```
cat("Base lm() R-squared:", lm_r_squared, "\n")
```

```
Base lm() R-squared: 0.7406427
```

```
cat("Lasso glmnet R-squared:", lasso_r_squared, "\n")
```

```
Lasso glmnet R-squared: 0.7405855
```

```
cat("\nDifference:", lm_r_squared - lasso_r_squared, "\n")
```

```
Difference: 5.720643e-05
```

Observation: Lasso achieves similar performance while potentially simplifying the model by shrinking some coefficients

---

Business Applications

1. **Sales Forecasting**
   Use regression to predict future sales using variables like marketing spend, pricing, or seasonality.
2. **Customer Churn Prediction**
   Identify which customer behaviors or attributes most strongly indicate churn risk.
3. **Marketing Channel Optimization**
   Use Lasso to find which marketing channels contribute most to revenue and remove low-impact ones.

4. **Employee Attrition & Performance Modeling**
   Predict employee outcomes based on workload, satisfaction, and manager ratings.
5. **Demand Forecasting for Inventory**
   Estimate product demand using promotions, weather, or store traffic to improve stock planning.

Exercise: Try With Another Dataset

Task:
Run the same workflow (load → prepare → fit → compare) using a *different dataset* such as `mtcars`, `iris`, or your own CSV.

Steps:

1. Load a dataset: `df <- mtcars`
2. Pick a target variable: `y <- df$mpg`
3. Create predictors: `X <- as.matrix(df[, -1])`
4. Fit linear & Lasso models (same code as earlier)
5. Compare R^2 and identify important predictors.

Goal:
See how model behavior changes with a new dataset.

Conclusion

- **Linear Regression**: Uses all predictors, no regularization
- **Lasso Regression**: Applies L1 penalty, can eliminate irrelevant features
- **Performance**: Both models show similar R^2 values on this dataset
- **Advantage of Lasso**: Provides feature selection and helps prevent overfitting
- **Use Case**: Lasso is particularly useful when you have many predictors and want a simpler, more interpretable model

---

Thank You!

Questions?