Regression Analysis of Regularization and Dimension Reduction Techniques

Ambachow Kahsay

2025-10-16

Contents

###Introduction

This report explores the California Housing dataset, performs Exploratory Data Analysis (EDA), and builds a Multiple Linear Regression model to predict median house values.

```r
library(recipes)
```

0.0.0.0.1 Libraries

```
## Loading required package: dplyr

## Warning: package 'dplyr' was built under R version 4.3.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

##
## Attaching package: 'recipes'

## The following object is masked from 'package:stats':
##
##     step
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3

## Loading required package: ggplot2

## Loading required package: lattice
```

```r
library(magrittr)

library(magrittr)
```

```r
library(caret)
library(magrittr)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-10
```

```r
# Load the dataset using the absolute path
housing <- read.csv("C:/Users/Admin/OneDrive - United States International University (USIU)/Documents/U

# View the first few rows of the data
head(housing)
```

0.0.0.1 Load the Dataset

```
##   longitude latitude housing_median_age total_rooms total_bedrooms population
## 1   -122.23    37.88                 41         880            129        322
## 2   -122.22    37.86                 21        7099           1106       2401
## 3   -122.24    37.85                 52        1467            190        496
## 4   -122.25    37.85                 52        1274            235        558
## 5   -122.25    37.85                 52        1627            280        565
## 6   -122.25    37.85                 52         919            213        413
##   households median_income median_house_value ocean_proximity
## 1        126        8.3252             452600        NEAR BAY
## 2       1138        8.3014             358500        NEAR BAY
## 3        177        7.2574             352100        NEAR BAY
## 4        219        5.6431             341300        NEAR BAY
## 5        259        3.8462             342200        NEAR BAY
## 6        193        4.0368             269700        NEAR BAY
```

```r
# Check the structure of the dataset
str(housing)
```

0.0.0.2 Inspect Data

```
## 'data.frame':    20640 obs. of  10 variables:
##  $ longitude         : num  -122 -122 -122 -122 -122 ...
##  $ latitude          : num  37.9 37.9 37.9 37.9 37.9 ...
##  $ housing_median_age: num  41 21 52 52 52 52 52 52 42 52 ...
##  $ total_rooms       : num  880 7099 1467 1274 1627 ...
##  $ total_bedrooms    : num  129 1106 190 235 280 ...
##  $ population        : num  322 2401 496 558 565 ...
##  $ households        : num  126 1138 177 219 259 ...
##  $ median_income     : num  8.33 8.3 7.26 5.64 3.85 ...
##  $ median_house_value: num  452600 358500 352100 341300 342200 ...
##  $ ocean_proximity   : chr  "NEAR BAY" "NEAR BAY" "NEAR BAY" "NEAR BAY" ...
```

```r
housing <- na.omit(housing)
head(housing)
```

0.0.0.3 Handle Missing Values

```
##   longitude latitude housing_median_age total_rooms total_bedrooms population
## 1   -122.23    37.88                 41         880            129        322
## 2   -122.22    37.86                 21        7099           1106       2401
## 3   -122.24    37.85                 52        1467            190        496
## 4   -122.25    37.85                 52        1274            235        558
## 5   -122.25    37.85                 52        1627            280        565
## 6   -122.25    37.85                 52         919            213        413
##   households median_income median_house_value ocean_proximity
## 1        126        8.3252             452600        NEAR BAY
## 2       1138        8.3014             358500        NEAR BAY
## 3        177        7.2574             352100        NEAR BAY
## 4        219        5.6431             341300        NEAR BAY
## 5        259        3.8462             342200        NEAR BAY
## 6        193        4.0368             269700        NEAR BAY
```

```
housing <- dummyVars(" ~ .", data = housing) %>%
  predict(newdata = housing) %>%
  as.data.frame()
```

```
dummies <- dummyVars(" ~ .", data = housing)
housing <- predict(dummies, newdata = housing)
housing <- as.data.frame(housing)
```

```
str(housing)
```

0.0.0.4 Encode Categorical Variable

```
## 'data.frame':    20433 obs. of  14 variables:
##  $ longitude                  : num  -122 -122 -122 -122 -122 ...
##  $ latitude                   : num  37.9 37.9 37.9 37.9 37.9 ...
##  $ housing_median_age         : num  41 21 52 52 52 52 52 52 42 52 ...
##  $ total_rooms                : num  880 7099 1467 1274 1627 ...
##  $ total_bedrooms             : num  129 1106 190 235 280 ...
##  $ population                 : num  322 2401 496 558 565 ...
##  $ households                 : num  126 1138 177 219 259 ...
##  $ median_income              : num  8.33 8.3 7.26 5.64 3.85 ...
##  $ median_house_value         : num  452600 358500 352100 341300 342200 ...
##  $ `ocean_proximity<1H OCEAN` : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ocean_proximityINLAND      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ocean_proximityISLAND      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ `ocean_proximityNEAR BAY`  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ `ocean_proximityNEAR OCEAN`: num  0 0 0 0 0 0 0 0 0 0 ...
```

```
summary(housing)
```

```
##    longitude         latitude     housing_median_age  total_rooms
##  Min.   :-124.3   Min.   :32.54   Min.   : 1.00      Min.   :    2
##  1st Qu.:-121.8   1st Qu.:33.93   1st Qu.:18.00      1st Qu.: 1450
##  Median :-118.5   Median :34.26   Median :29.00      Median : 2127
##  Mean   :-119.6   Mean   :35.63   Mean   :28.63      Mean   : 2636
##  3rd Qu.:-118.0   3rd Qu.:37.72   3rd Qu.:37.00      3rd Qu.: 3143
##  Max.   :-114.3   Max.   :41.95   Max.   :52.00      Max.   :39320
##  total_bedrooms     population      households     median_income
```

```
##  Min.   :   1.0   Min.   :    3   Min.   :   1.0   Min.   : 0.4999
##  1st Qu.: 296.0   1st Qu.:  787   1st Qu.: 280.0   1st Qu.: 2.5637
##  Median : 435.0   Median : 1166   Median : 409.0   Median : 3.5365
##  Mean   : 537.9   Mean   : 1425   Mean   : 499.4   Mean   : 3.8712
##  3rd Qu.: 647.0   3rd Qu.: 1722   3rd Qu.: 604.0   3rd Qu.: 4.7440
##  Max.   :6445.0   Max.   :35682   Max.   :6082.0   Max.   :15.0001
##  median_house_value `ocean_proximity<1H OCEAN` ocean_proximityINLAND
##  Min.   : 14999     Min.   :0.0000            Min.   :0.0000
##  1st Qu.:119500     1st Qu.:0.0000            1st Qu.:0.0000
##  Median :179700     Median :0.0000            Median :0.0000
##  Mean   :206864     Mean   :0.4421            Mean   :0.3179
##  3rd Qu.:264700     3rd Qu.:1.0000            3rd Qu.:1.0000
##  Max.   :500001     Max.   :1.0000            Max.   :1.0000
##  ocean_proximityISLAND `ocean_proximityNEAR BAY` `ocean_proximityNEAR OCEAN`
##  Min.   :0.0000000     Min.   :0.0000            Min.   :0.0000
##  1st Qu.:0.0000000     1st Qu.:0.0000            1st Qu.:0.0000
##  Median :0.0000000     Median :0.0000            Median :0.0000
##  Mean   :0.0002447     Mean   :0.1111            Mean   :0.1286
##  3rd Qu.:0.0000000     3rd Qu.:0.0000            3rd Qu.:0.0000
##  Max.   :1.0000000     Max.   :1.0000            Max.   :1.0000
```

```r
colSums(is.na(housing))
```

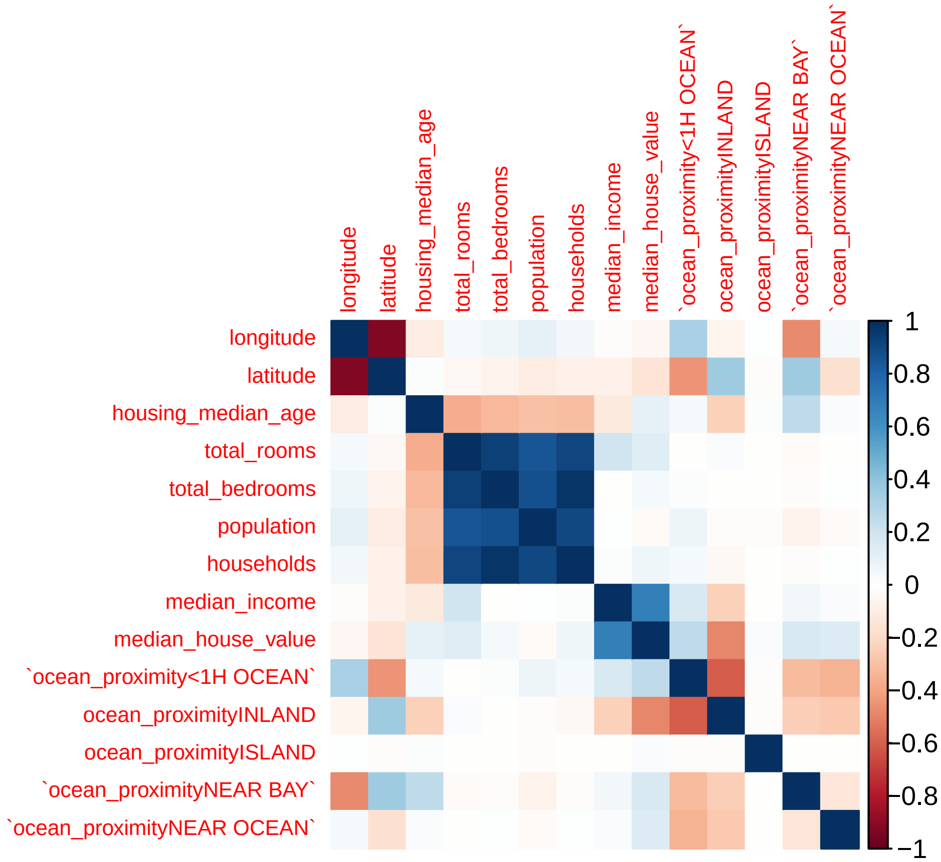
```
##                  longitude                   latitude
##                          0                          0
##         housing_median_age                total_rooms
##                          0                          0
##             total_bedrooms                 population
##                          0                          0
##                 households              median_income
##                          0                          0
##         median_house_value  `ocean_proximity<1H OCEAN`
##                          0                          0
##      ocean_proximityINLAND      ocean_proximityISLAND
##                          0                          0
##   `ocean_proximityNEAR BAY` `ocean_proximityNEAR OCEAN`
##                          0                          0
```

```r
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3
```

```
## corrplot 0.95 loaded
```

```r
corr_matrix <- cor(housing)
corrplot(corr_matrix, method = "color", tl.cex = 0.7)
```




```r
set.seed(123)  # ensures reproducibility

# Use caret's createDataPartition function
library(caret)

split <- createDataPartition(housing$median_house_value, p = 0.8, list = FALSE)

train_data <- housing[split, ]
test_data  <- housing[-split, ]
```

0.0.0.5 Split the Data

```r
nrow(train_data)
```

0.0.0.5.1 Check the Split

```
## [1] 16348
```

```r
nrow(test_data)
```

```
## [1] 4085
```

```r
colnames(train_data)
```

```
##  [1] "longitude"                    "latitude"
##  [3] "housing_median_age"           "total_rooms"
```

```
##  [5] "total_bedrooms"              "population"
##  [7] "households"                  "median_income"
##  [9] "median_house_value"          "`ocean_proximity<1H OCEAN`"
## [11] "ocean_proximityINLAND"       "ocean_proximityISLAND"
## [13] "`ocean_proximityNEAR BAY`"   "`ocean_proximityNEAR OCEAN`"
```

```r
colnames(test_data)
```

```
##  [1] "longitude"                   "latitude"
##  [3] "housing_median_age"          "total_rooms"
##  [5] "total_bedrooms"              "population"
##  [7] "households"                  "median_income"
##  [9] "median_house_value"          "`ocean_proximity<1H OCEAN`"
## [11] "ocean_proximityINLAND"       "ocean_proximityISLAND"
## [13] "`ocean_proximityNEAR BAY`"   "`ocean_proximityNEAR OCEAN`"
```

```r
model_train <- lm(median_house_value ~ ., data = train_data)
summary(model_train)
```

0.0.0.6 Fit the Model on Training Data

```
##
## Call:
## lm(formula = median_house_value ~ ., data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -560505  -42590  -10327   28586  734628
##
## Coefficients: (1 not defined because of singularities)
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -2.255e+06  9.861e+04 -22.870  < 2e-16 ***
## longitude                         -2.672e+04  1.138e+03 -23.475  < 2e-16 ***
## latitude                          -2.549e+04  1.122e+03 -22.713  < 2e-16 ***
## housing_median_age                 1.074e+03  4.904e+01  21.911  < 2e-16 ***
## total_rooms                       -6.607e+00  8.820e-01  -7.491 7.16e-14 ***
## total_bedrooms                     1.088e+02  7.824e+00  13.913  < 2e-16 ***
## population                        -3.597e+01  1.172e+00 -30.702  < 2e-16 ***
## households                         3.785e+01  8.426e+00   4.492 7.09e-06 ***
## median_income                      3.961e+04  3.828e+02 103.474  < 2e-16 ***
## `\\`ocean_proximity<1H OCEAN\\``  -4.668e+03  1.748e+03  -2.671  0.00758 **
## ocean_proximityINLAND             -4.443e+04  2.510e+03 -17.700  < 2e-16 ***
## ocean_proximityISLAND              1.477e+05  3.086e+04   4.786 1.71e-06 ***
## `\\`ocean_proximityNEAR BAY\\``   -7.389e+03  2.444e+03  -3.024  0.00250 **
## `\\`ocean_proximityNEAR OCEAN\\``         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68870 on 16335 degrees of freedom
## Multiple R-squared:  0.6441, Adjusted R-squared:  0.6438
## F-statistic:  2463 on 12 and 16335 DF,  p-value: < 2.2e-16
```

0.0.0.7 California Housing Price Prediction (Linear Regression)

```r
predictions <- predict(model_train, newdata = test_data)
```

0.0.0.7.1 Predict on Test Data

```r
# Calculate RMSE and R-squared
rmse <- sqrt(mean((test_data$median_house_value - predictions)^2))
r2 <- cor(test_data$median_house_value, predictions)^2

cat("RMSE:", rmse, "\nR-squared:", r2)
```

0.0.0.7.2 Evaluate Model Performance

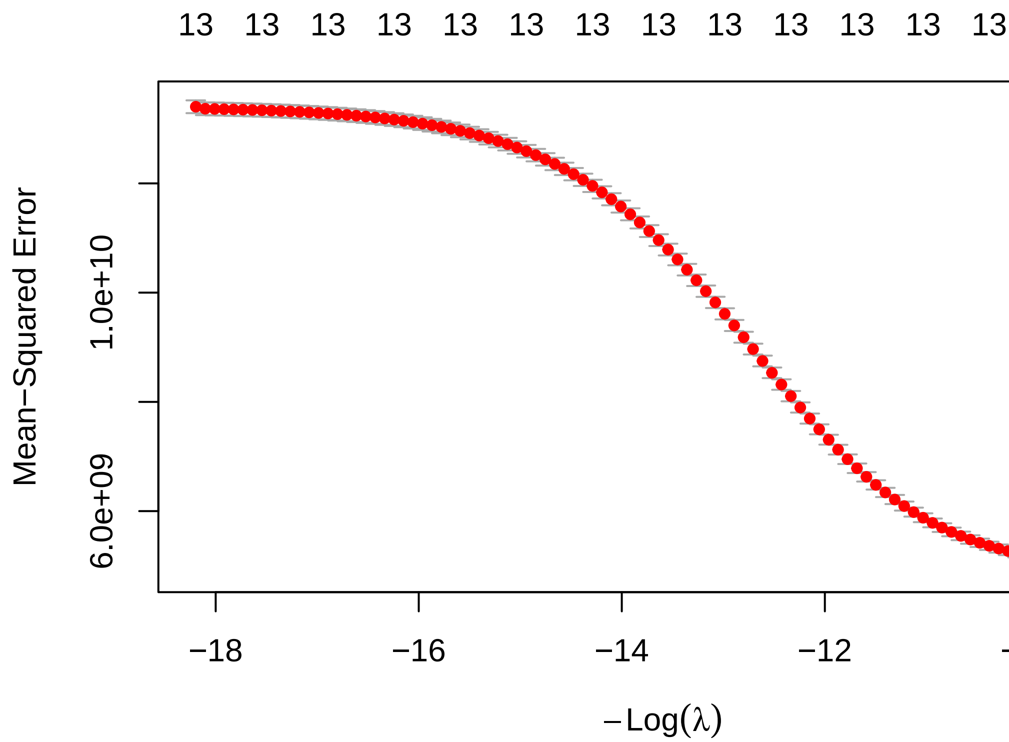
```
## RMSE: 67873.12
## R-squared: 0.6558641
```

0.0.1 Regularization Techniques

```r
x <- model.matrix(median_house_value ~ ., data = housing)[, -1]  # remove intercept
y <- housing$median_house_value

set.seed(123)
train_idx <- sample(1:nrow(x), 0.8 * nrow(x))
x_train <- x[train_idx, ]
y_train <- y[train_idx]
x_test <- x[-train_idx, ]
y_test <- y[-train_idx]
```

0.0.1.1 Prepare Data for Regularization

```r
ridge_model <- cv.glmnet(x_train, y_train, alpha = 0)
plot(ridge_model)
```



0.0.1.2 Ridge Regression (L2)

```
#  Find best lambda (regularization parameter)
best_lambda_ridge <- ridge_model$lambda.min
best_lambda_ridge
```

0.0.1.2.1 regularization parameter

```
## [1] 7987.83
```

```
ridge_pred <- predict(ridge_model, s = ridge_model$lambda.min, newx = x_test)
```

0.0.1.3 Prediction_Ridge

```
ridge_rmse <- sqrt(mean((ridge_pred - y_test)^2))
ridge_r2 <- 1 - sum((ridge_pred - y_test)^2) / sum((y_test - mean(y_test))^2)
cat("Ridge Regression:\n")
```

0.0.1.4 Evaluation Ridgr Regression

```
## Ridge Regression:
```

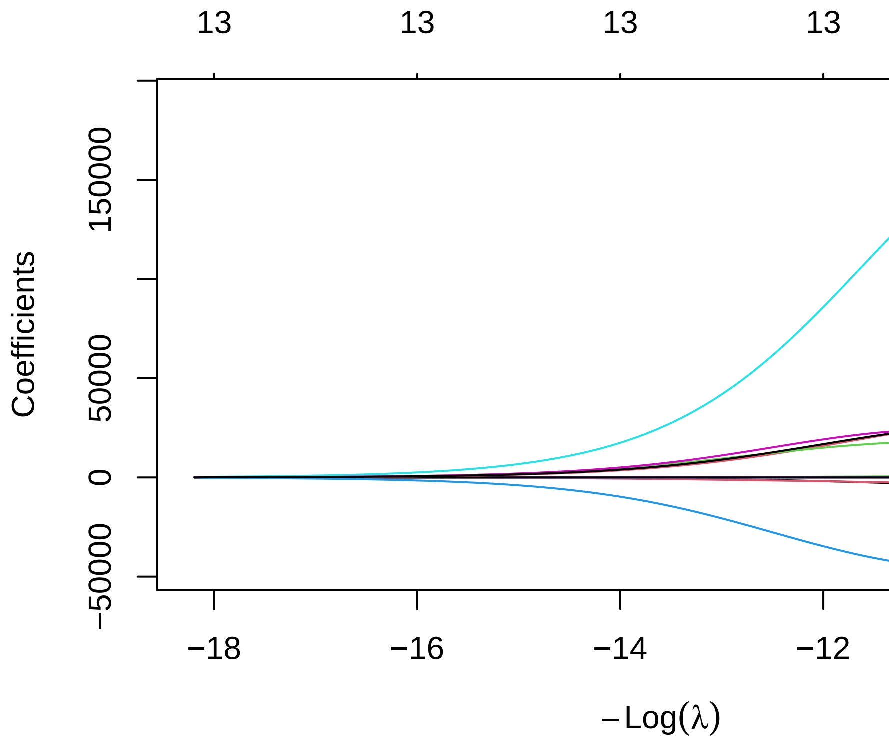
```
cat("  RMSE:", ridge_rmse, "\n")
```

```
##   RMSE: 68931.86
```

```r
cat("  R-squared:", ridge_r2, "\n\n")
```

```
##   R-squared: 0.63499
```

```r
library(glmnet)
plot(ridge_model$glmnet.fit, xvar = "lambda")
```

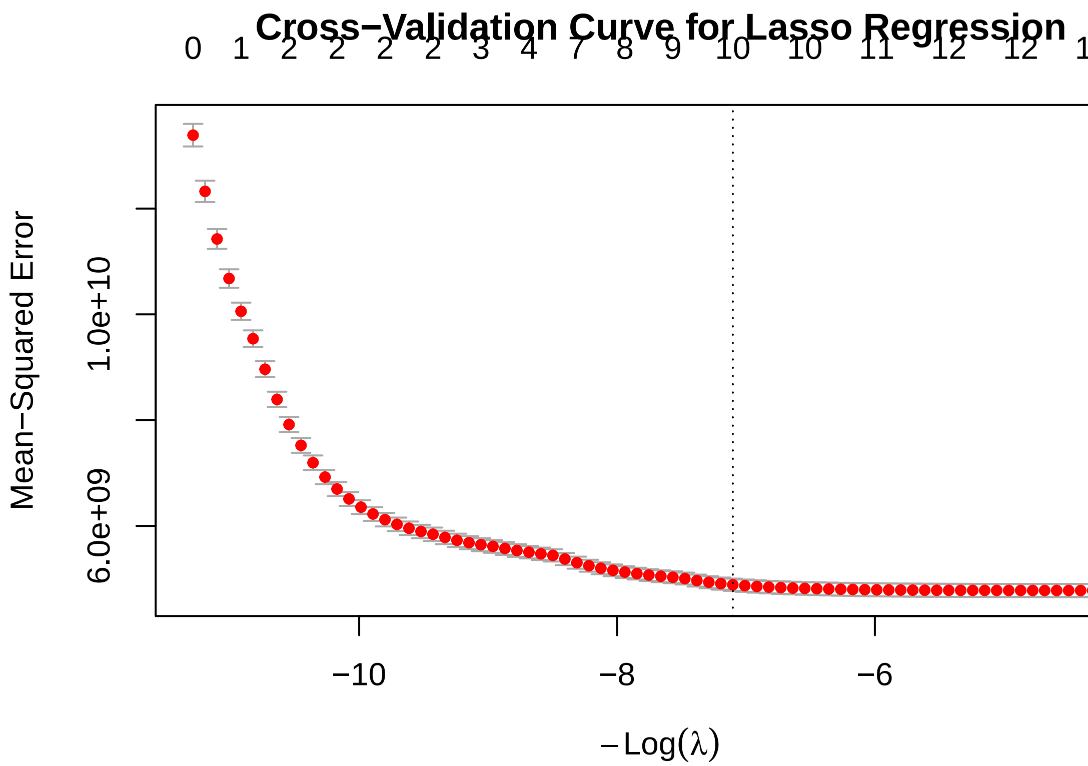


0.0.1.4.1 Ridge Regression Diagnostics

```r
coef(ridge_model, s = ridge_model$lambda.min)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                                        s=7987.83
## (Intercept)                        -7.958822e+05
## longitude                          -9.432991e+03
## latitude                           -8.406588e+03
## housing_median_age                  1.003642e+03
## total_rooms                         7.246974e-01
## total_bedrooms                      4.134851e+01
## population                         -2.514532e+01
## households                          4.091730e+01
## median_income                       3.544553e+04
## `\\`ocean_proximity<1H OCEAN\\``    1.770221e+04
## ocean_proximityINLAND              -4.397253e+04
## ocean_proximityISLAND               1.912247e+05
## `\\`ocean_proximityNEAR BAY\\``     2.023143e+04
## `\\`ocean_proximityNEAR OCEAN\\``   2.758138e+04
```

```r
# Perform Lasso Regression (L1 Regularization)
set.seed(123)
lasso_model <- cv.glmnet(x_train, y_train, alpha = 1)

# Cross-validated Lasso plot with title
plot(lasso_model, main = "Cross-Validation Curve for Lasso Regression")
```



0.0.1.5 Laso Regression

```r
# Best lambda
best_lambda_lasso <- lasso_model$lambda.min
best_lambda_lasso
```

```
## [1] 56.35256
```

```r
# Lasso Regression
lasso_pred <- predict(lasso_model, s = lasso_model$lambda.min, newx = x_test)
```

0.0.1.6 Lasso Prediction

```r
lasso_rmse <- sqrt(mean((lasso_pred - y_test)^2))
lasso_r2 <- 1 - sum((lasso_pred - y_test)^2) / sum((y_test - mean(y_test))^2)

cat("Lasso Regression:\n")
```

0.0.1.6.1 Evaluation

```
## Lasso Regression:
```

```r
cat("  RMSE:", lasso_rmse, "\n")
```

```
##   RMSE: 67681.33
```

```r
cat("  R-squared:", lasso_r2, "\n")
```

```
##   R-squared: 0.6481135
```

```r
coef(lasso_model, s = best_lambda_lasso)
```

0.0.1.6.2 Lasso Regression Diagnostics

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                                        s=56.35256
## (Intercept)                         -2.196103e+06
## longitude                           -2.596847e+04
## latitude                            -2.470979e+04
## housing_median_age                   1.055381e+03
## total_rooms                         -5.562909e+00
## total_bedrooms                       8.775694e+01
## population                          -3.700647e+01
## households                           5.858007e+01
## median_income                        3.906495e+04
## `\\`ocean_proximity<1H OCEAN\\``      .
## ocean_proximityINLAND               -3.921343e+04
## ocean_proximityISLAND                1.682914e+05
## `\\`ocean_proximityNEAR BAY\\``    -2.839300e+03
## `\\`ocean_proximityNEAR OCEAN\\``   5.092122e+03
```

```r
# Prepare data
x <- model.matrix(median_house_value ~ ., train_data)[, -1]
y <- train_data$median_house_value

# Create training control
train_control <- trainControl(method = "cv", number = 10)
```

```r
# Fit PLS Regression model
set.seed(123)
pls_model <- train(
  x = x,
  y = y,
  method = "pls",
  trControl = train_control,
  tuneLength = 10,
  preProcess = c("center", "scale")
)
# Show best model
print(pls_model)
```

```
## Partial Least Squares
##
## 16348 samples
##    13 predictor
##
```

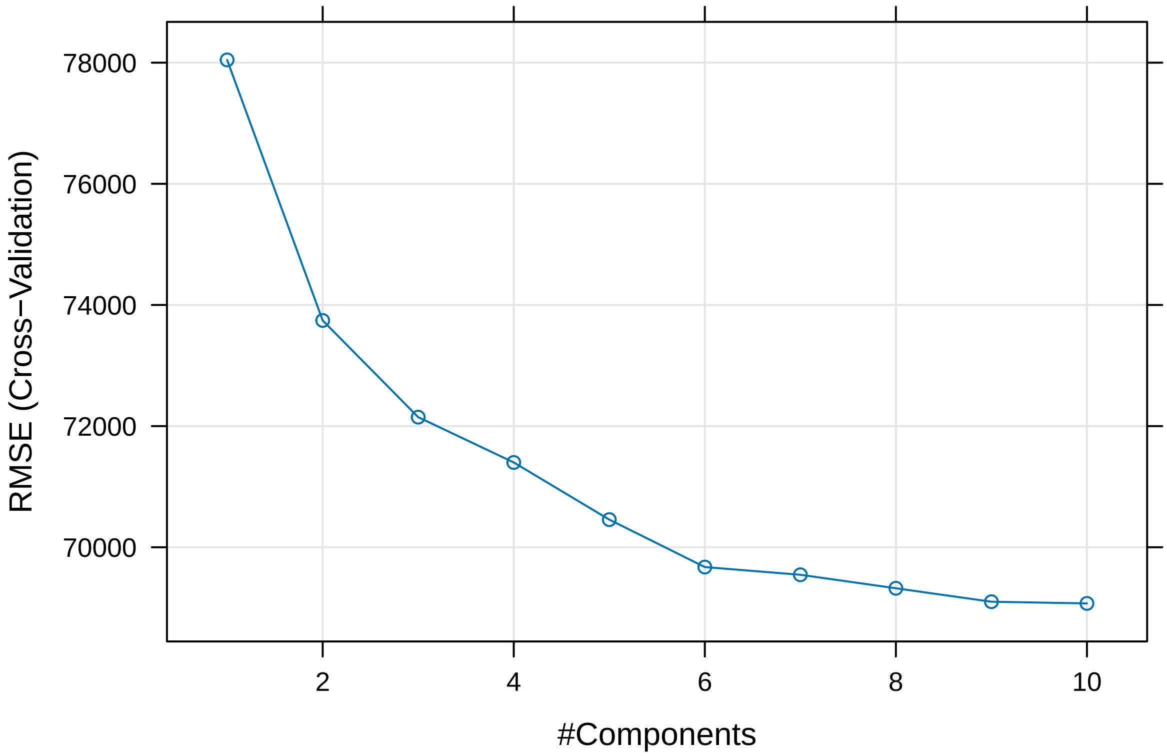
```
## Pre-processing: centered (13), scaled (13)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 14713, 14712, 14714, 14713, 14713, 14715, ...
## Resampling results across tuning parameters:
##
##   ncomp  RMSE      Rsquared   MAE
##    1     78045.73  0.5425952  58709.15
##    2     73743.99  0.5915625  54199.26
##    3     72147.29  0.6091188  52064.38
##    4     71400.04  0.6171795  51586.66
##    5     70456.33  0.6272779  51248.63
##    6     69672.94  0.6357131  50232.80
##    7     69545.56  0.6370352  50083.46
##    8     69322.88  0.6393446  49949.82
##    9     69100.78  0.6416603  49772.12
##   10     69072.99  0.6419112  49766.69
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was ncomp = 10.
```

```r
# Evaluate on test data
x_test <- model.matrix(median_house_value ~ ., test_data)[, -1]
y_test <- test_data$median_house_value

pls_predictions <- predict(pls_model, newdata = x_test)
```

```r
# Compute metrics
pls_rmse <- sqrt(mean((y_test - pls_predictions)^2))
pls_r2 <- cor(y_test, pls_predictions)^2

cat("PLS Regression:\n",
    "  RMSE:", pls_rmse, "\n",
    "  R-squared:", pls_r2, "\n")
```

```
## PLS Regression:
##    RMSE: 67902.41
##    R-squared: 0.6555516
```

```r
library(caret)
plot(pls_model)
```




```r
# If you used caret::train with method = "pls"
pls_coef <- coef(pls_model$finalModel, ncomp = 10)  # ncomp = optimal number of components
pls_coef
```

```
## , , 10 comps
##
##                                                          .outcome
## longitude                                              -41823.080
## latitude                                               -42022.359
## housing_median_age                                      13738.501
## total_rooms                                            -20290.447
## total_bedrooms                                          37074.890
## population                                             -42468.432
## households                                              30392.935
## median_income                                           75905.282
## `\\`\\\\\\`ocean_proximity<1H OCEAN\\\\\\`\\``          6732.612
## ocean_proximityINLAND                                  -14691.680
## ocean_proximityISLAND                                    3375.103
## `\\`\\\\\\`ocean_proximityNEAR BAY\\\\\\`\\``           3530.472
## `\\`\\\\\\`ocean_proximityNEAR OCEAN\\\\\\`\\``         6948.294
```

```r
set.seed(123)
ridge_cv <- cv.glmnet(x, y, alpha = 0, nfolds = 10)
best_lambda_ridge <- ridge_cv$lambda.min
rmse_ridge <- sqrt(min(ridge_cv$cvm))  # CV RMSE

# Lasso CV
```

```r
lasso_cv <- cv.glmnet(x, y, alpha = 1, nfolds = 10)
best_lambda_lasso <- lasso_cv$lambda.min
rmse_lasso <- sqrt(min(lasso_cv$cvm))  # CV RMSE

# PLS CV
library(caret)
set.seed(123)
pls_model <- train(
  median_house_value ~ ., data = train_data,
  method = "pls",
  preProcess = c("center", "scale"),
  tuneLength = 10,
  trControl = trainControl(method = "cv", number = 10)
)
rmse_pls <- pls_model$results$RMSE[pls_model$results$ncomp == pls_model$bestTune$ncomp]

# Linear regression RMSE (CV)
set.seed(123)
lm_model <- train(
  median_house_value ~ ., data = train_data,
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)
rmse_lm <- lm_model$results$RMSE

# Combine results
cv_results <- data.frame(
  Model = c("Linear Regression", "Ridge Regression", "Lasso Regression", "PLS Regression"),
  CV_RMSE = c(rmse_lm, rmse_ridge, rmse_lasso, rmse_pls)
)

cv_results
```

```
##                Model  CV_RMSE
## 1 Linear Regression 68971.66
## 2  Ridge Regression 70126.12
## 3  Lasso Regression 69013.66
## 4    PLS Regression 69072.99
```

```r
# Predictions on test set
x_test <- model.matrix(median_house_value ~ ., test_data)[, -1]
y_test <- test_data$median_house_value

# Linear Regression
pred_lm <- predict(lm_model, newdata = test_data)

# Ridge
pred_ridge <- predict(ridge_model, s = best_lambda_ridge, newx = x_test)

# Lasso
pred_lasso <- predict(lasso_model, s = best_lambda_lasso, newx = x_test)

# PLS
pred_pls <- predict(pls_model, newdata = test_data)
```

```r
library(ggplot2)
plot_df <- data.frame(
  Actual = y_test,
  Linear = as.vector(pred_lm),
  Ridge = as.vector(pred_ridge),
  Lasso = as.vector(pred_lasso),
  PLS = as.vector(pred_pls)
)
```

```r
library(tidyr)
```

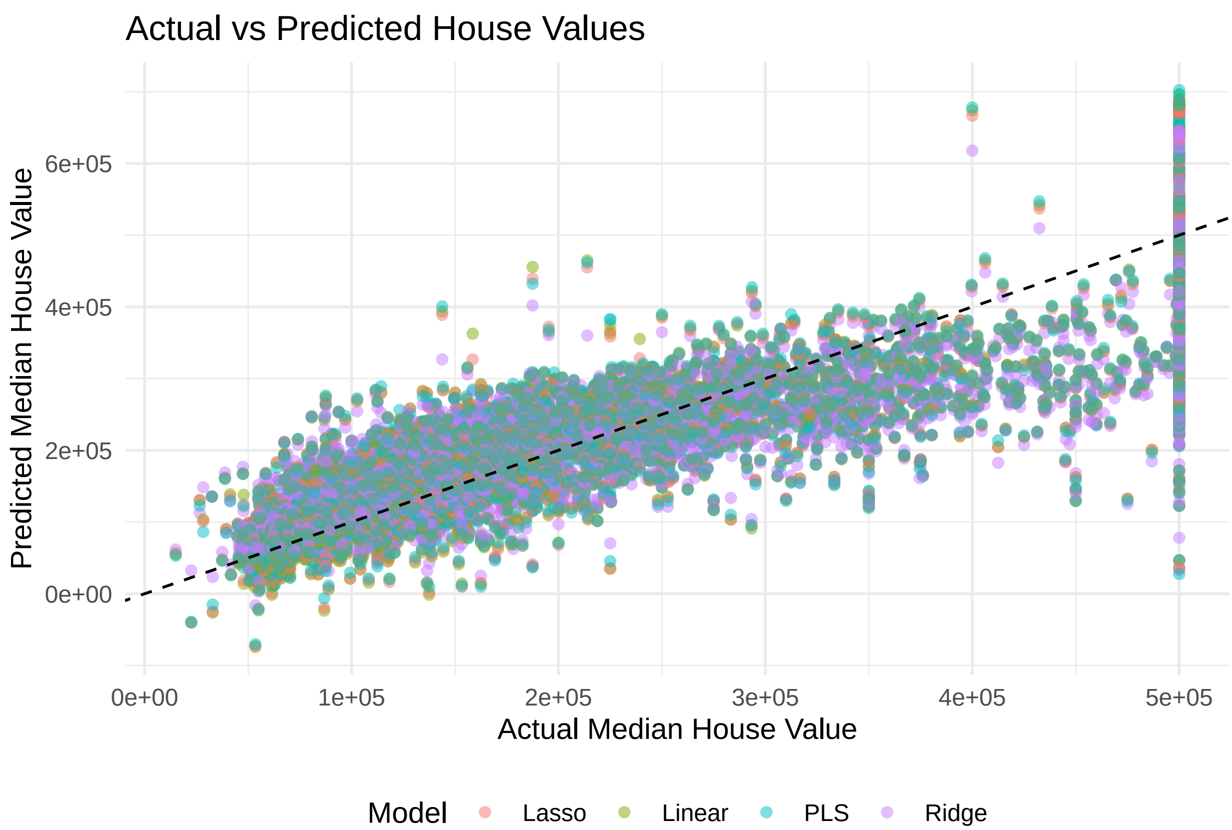
```
## Warning: package 'tidyr' was built under R version 4.3.3
```

```
##
## Attaching package: 'tidyr'
```

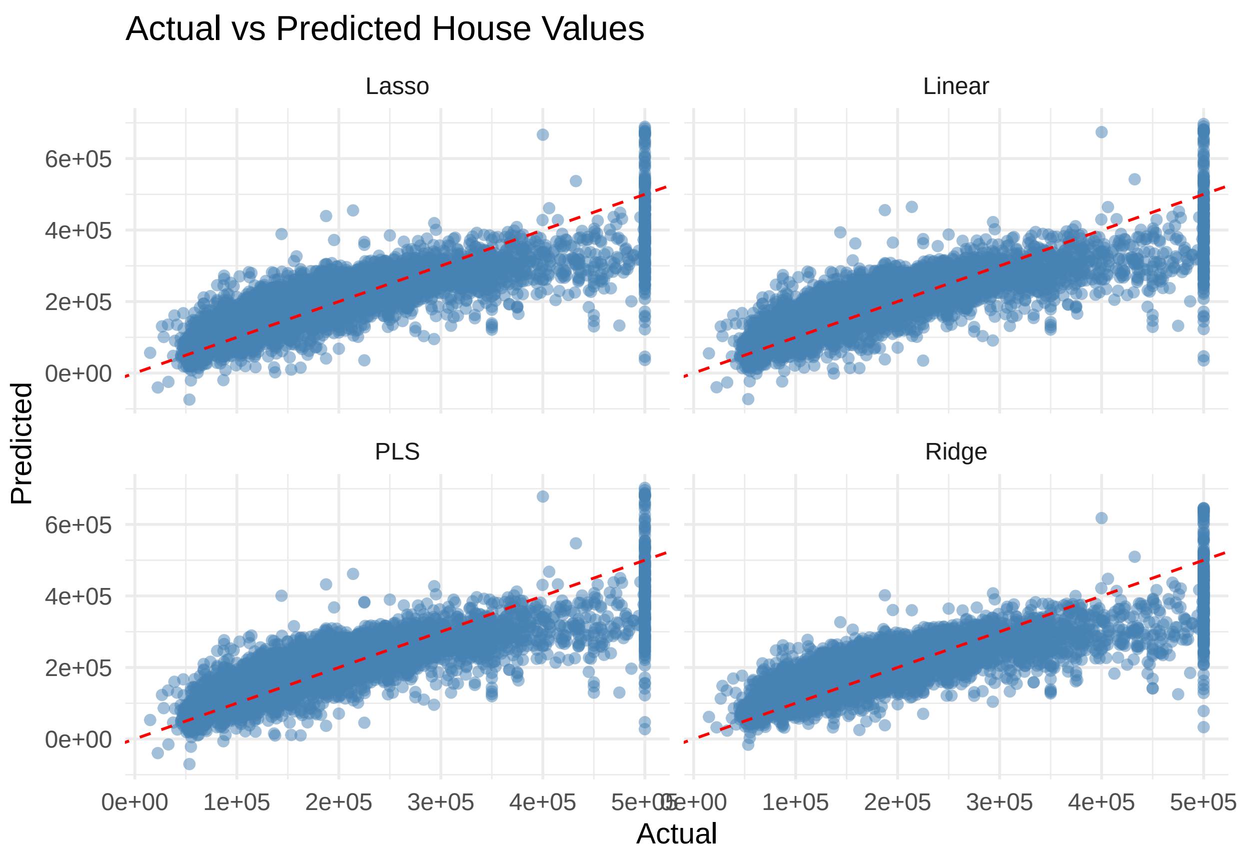
```
## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
```

```
## The following object is masked from 'package:magrittr':
##
##     extract
```

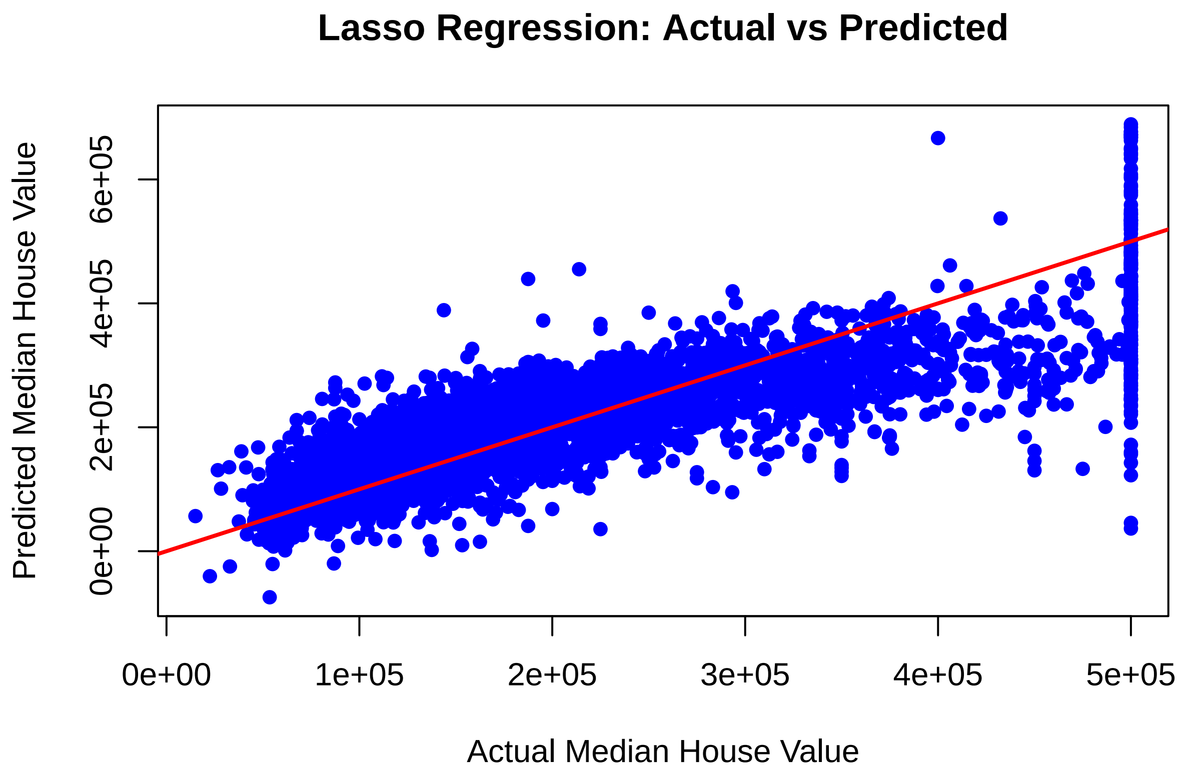
```r
# Convert to long format for ggplot
plot_long <- pivot_longer(plot_df, cols = -Actual, names_to = "Model", values_to = "Predicted")

ggplot(plot_long, aes(x = Actual, y = Predicted, color = Model)) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed", color = "black") +
  labs(title = "Actual vs Predicted House Values",
       x = "Actual Median House Value",
       y = "Predicted Median House Value") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



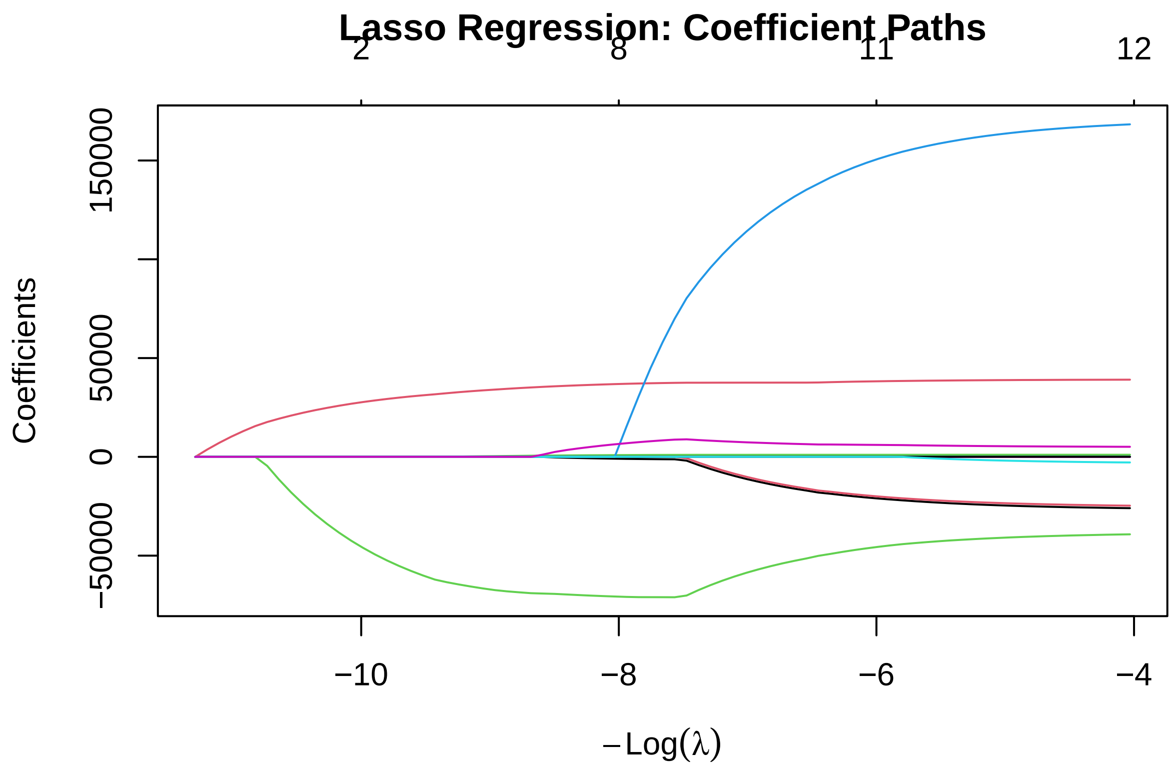

```r
ggplot(plot_long, aes(x = Actual, y = Predicted)) +
  geom_point(alpha = 0.5, color = "steelblue") +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed", color = "red") +
  labs(title = "Actual vs Predicted House Values", x = "Actual", y = "Predicted") +
  facet_wrap(~Model) +
  theme_minimal()
```



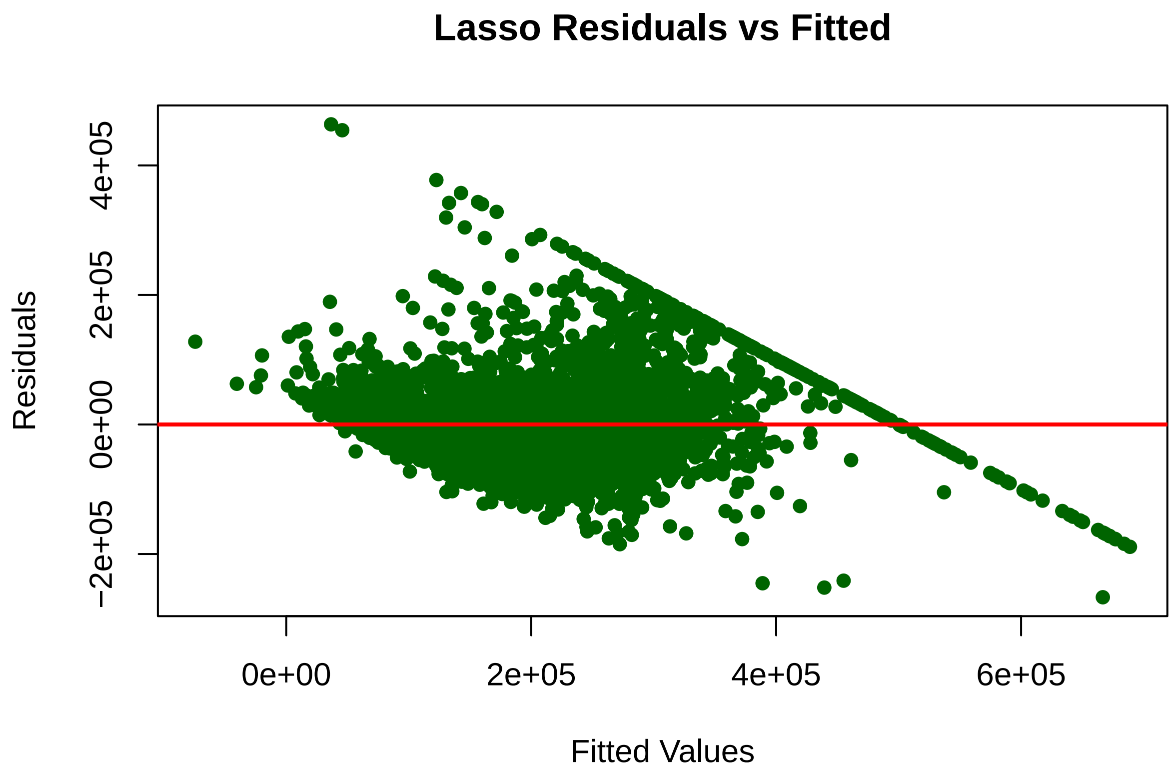

```r
# Predict on test data
y_pred <- predict(lasso_model, newx = x_test, s = "lambda.min")

# Scatter plot: actual vs predicted
plot(y_test, y_pred,
     xlab = "Actual Median House Value",
     ylab = "Predicted Median House Value",
     main = "Lasso Regression: Actual vs Predicted",
     col = "blue", pch = 16)
abline(a = 0, b = 1, col = "red", lwd = 2) # perfect fit line
```



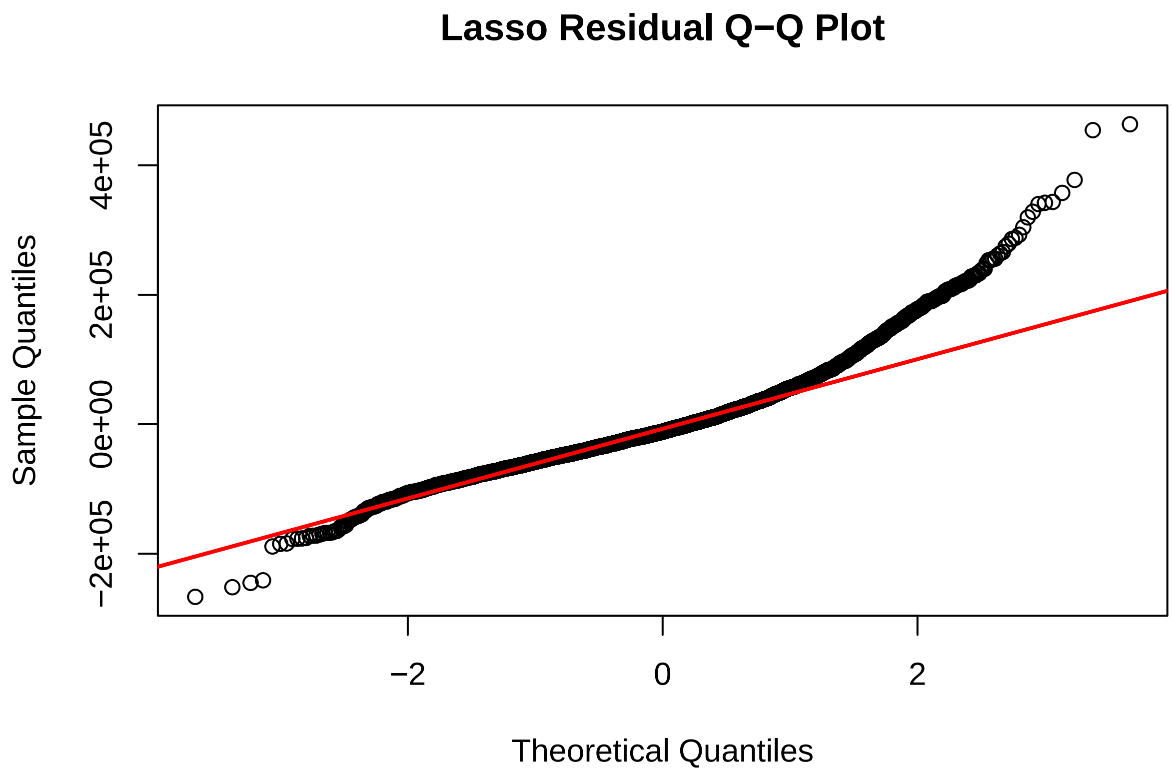

```
plot(lasso_model$glmnet.fit, xvar = "lambda", main = "Lasso Regression: Coefficient Paths")
```




```r
# Residuals
residuals_lasso <- y_test - y_pred

# Residuals vs Fitted
plot(y_pred, residuals_lasso,
     xlab = "Fitted Values", ylab = "Residuals",
     main = "Lasso Residuals vs Fitted", pch = 16, col = "darkgreen")
abline(h = 0, col = "red", lwd = 2)
```




```r
# Normal Q-Q plot
qqnorm(residuals_lasso, main = "Lasso Residual Q-Q Plot")
qqline(residuals_lasso, col = "red", lwd = 2)
```


Lasso Residual Q−Q Plot
Sample Quantiles
4e+05
2e+05
0e+00
−2e+05
−2
0
2
Theoretical Quantiles