Lab block 2

johmo870, nisra674

2024-12-07

Statement of Contribution

- Nisal Amashan(nisra674) - Assignment 1, Assignment 3, Report
- John Möller (johmo870) - Assignment 2, Assignment 4, Report

Assignment 1: Explicit Regularization

The goal is to investigate whether a near-infrared absorbance spectrum can predict fat content in meat samples.

The dataset contains:

- **Features**: A 100-channel spectrum of absorbance measurements.
- **Target**: Fat content in meat samples.

We perform:

1. Linear regression to model fat content.
2. LASSO regression to explore feature selection and sparsity.
3. Ridge regression for comparison.
4. Cross-validation to find the optimal penalty parameter.

Results

1. Linear Regression

Training and Test Errors

- **Training MSE**: `0.0089`
  This indicates a very good fit to the training data, suggesting that the model captures the relationships in the training set effectively.
- **Testing MSE**: `337.2898`
  This high value for the test data suggests poor generalization to unseen data, indicating overfitting. The model performs well on the training set but fails to predict accurately on the test set, compromising its predictive capability.

Cost Function for LASSO Regression

The LASSO regression cost function is defined as:

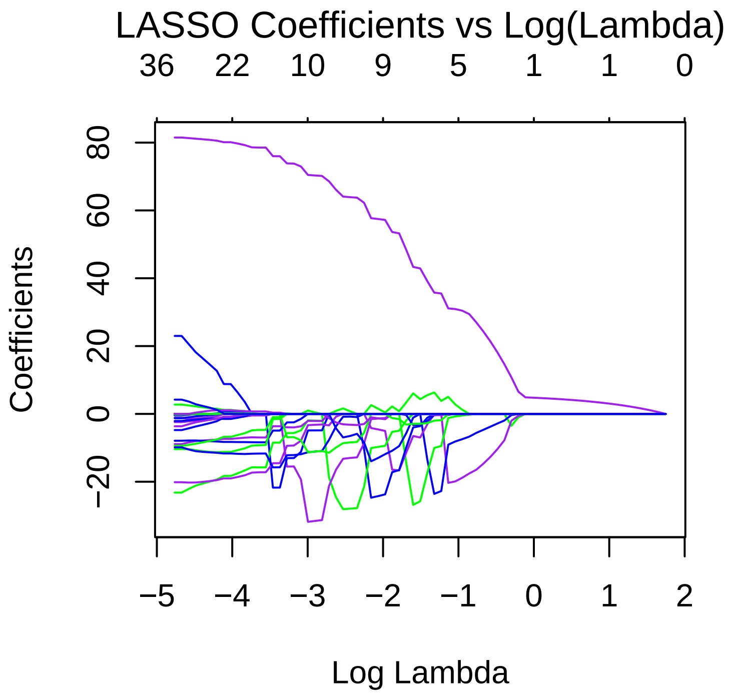
$$J(\beta) = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

Where:

- N: Number of samples.
- p: Number of features (100 absorbance channels).
- β_0: Intercept.
- β_j: Coefficients of the features.
- x_{ij}: Value of the j-th feature for the i-th sample.
- λ: Regularization parameter (controls the strength of penalty on coefficients).

2. **The parameter λ determines the balance:**

   - **Larger λ:** Stronger regularization, more coefficients shrink to zero.
   - **Smaller λ:** Weaker regularization, βs closer to ordinary least squares.



```
## Lambda for 3 features: 0.6764156
```

```
##        Feature Coefficient
## 1 (Intercept)   18.362617
## 2    Channel7   -7.785416
## 3    Channel8   -1.940834
## 4   Channel41   14.654522
```

Selecting a Model with 3 Features

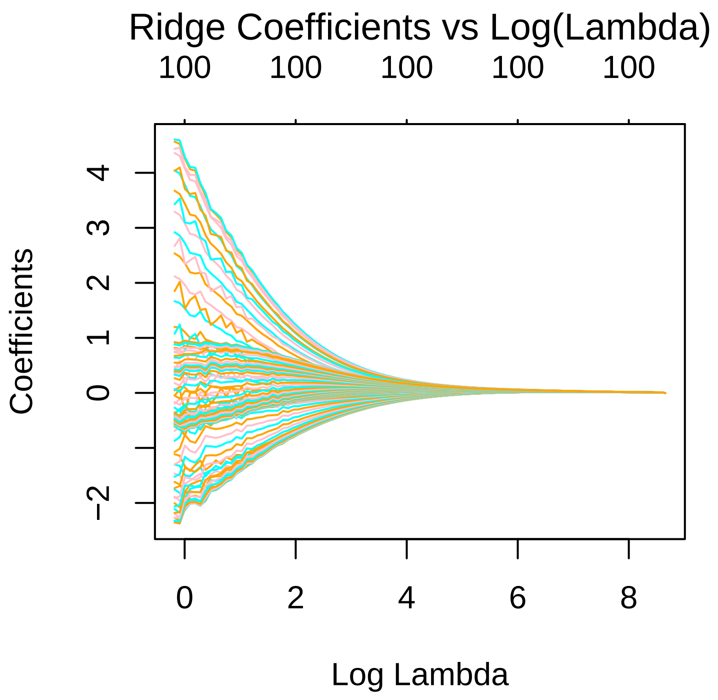
To identify a model with exactly 3 non-zero features, the coefficients were examined across different lambda values. The penalty factor Lambda for which only 3 features remain non-zero is:

Lambda = 0.6764

At this value of Lambda, the retained features and their coefficients are:

Feature	Coefficient
(Intercept)	18.36
Channel7	-7.79
Channel8	-1.94
Channel41	14.65

- The plot clearly illustrates the shrinking effect of Lambda on the coefficients and highlights the ability of LASSO to select a sparse set of features.
- At Lambda = 0.6764, the model selects **Channel7**, **Channel8**, and **Channel41** as the most significant predictors for Fat, along with the intercept term.
- This demonstrates LASSO's effectiveness in both feature selection and reducing model complexity.



Key Observations from the Plots

LASSO Plot

- Coefficients gradually shrink, with many becoming exactly zero as λ increases, indicating feature selection.
- Demonstrates sparsity, where only a small subset of features remains significant at higher λ values.

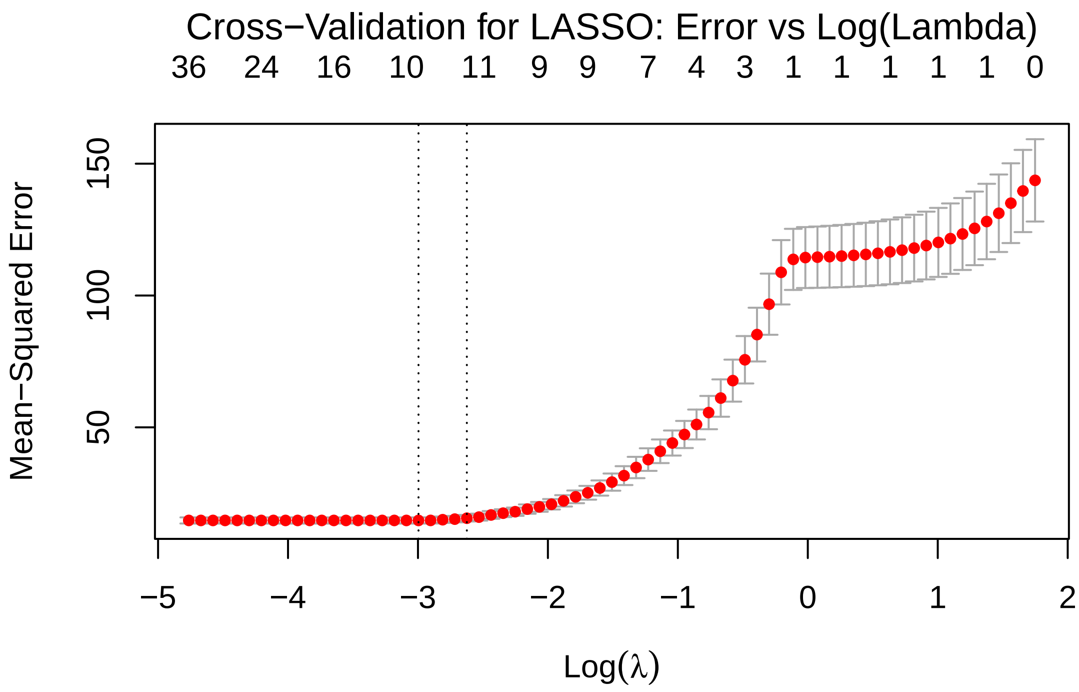
Ridge Plot

- Coefficients decrease smoothly and uniformly, without any feature being completely eliminated.
- Highlights Ridge's ability to regularize the model while retaining all features, which is useful when all predictors are assumed to be relevant.

Conclusions

- **LASSO** is more effective for feature selection in datasets with many irrelevant predictors. It provides a sparse model by retaining only a subset of features, which can improve interpretability.
- **Ridge** is more suitable for scenarios where all predictors are expected to contribute to the outcome. It minimizes overfitting by shrinking coefficients without eliminating any predictors.
- In this case, **LASSO** might be more appropriate if only a few absorbance channels are truly predictive of Fat. **Ridge**, on the other hand, would be beneficial if all features have some predictive value.

4. Cross-Validation



The LASSO regression model was fitted using cross-validation to determine the optimal value. The plot below shows the mean-squared error (MSE) as a function of log( ) with error bars indicating variability across the folds.

- The CV score decreases as decreases (log( ) increases), reaching a minimum at the optimal .
- After this point, the CV score increases as the model becomes less regularized and overfitting occurs.
- The optimal is identified as:

$$\lambda_{optimal} = 0.0549$$

At this value, the model achieves the lowest cross-validation error.

Selected Features and Test Performance

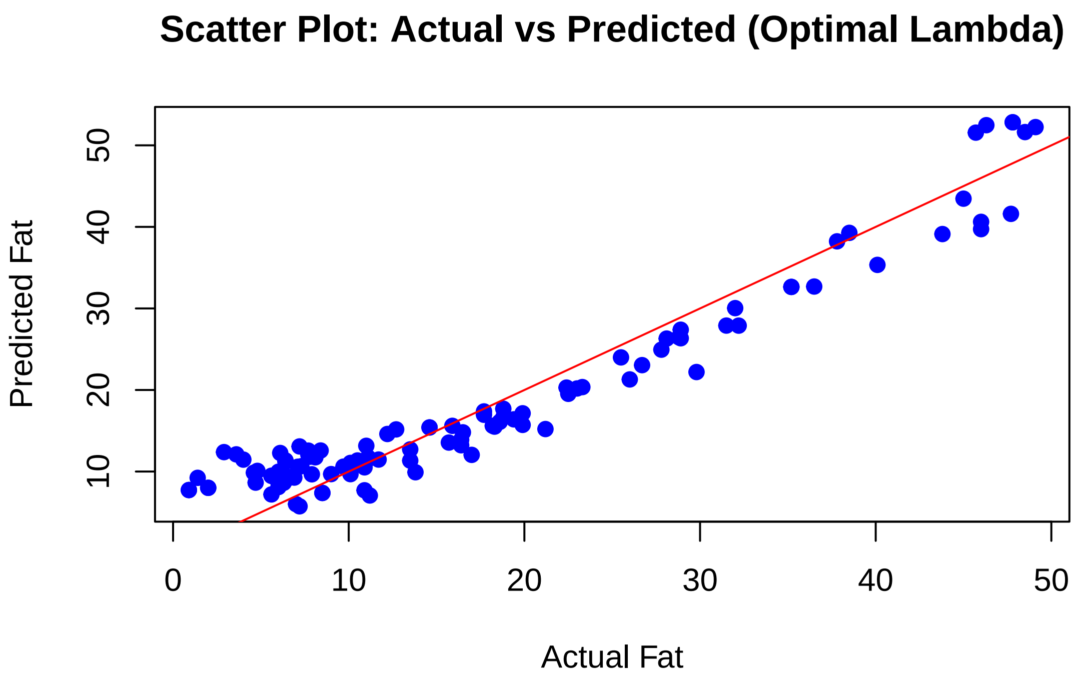
- **Number of Non-Zero Features:** At $\lambda_{optimal}$, the model retains 9 non-zero features, highlighting its ability to select important predictors while enforcing sparsity.
- **Test Set Performance:** The model's mean-squared error (MSE) on the test set at $\lambda_{optimal}$ is 14.57.

Comparison with $\log(\lambda) = -4$

- At $\log(\lambda) = -4$, the test MSE is 13.09, which is slightly lower than the test MSE for the optimal λ.
- While the test MSE for $\lambda_{optimal}$ is slightly worse, the difference is minor, suggesting that both models perform comparably.
- However, the model at $\lambda_{optimal}$ may be preferable due to its fewer retained features and reduced complexity.

Scatter Plot: Actual vs Predicted (Optimal Lambda)

The scatter plot of actual versus predicted values for $\lambda_{optimal}$ is shown below:



- The points lie close to the diagonal line, indicating good agreement between actual and predicted values.
- This suggests that the model provides reasonable predictions with the selected features.

Final Comments

- The cross-validation process effectively balances model complexity and generalization, selecting $\lambda_{optimal} = 0.0549$.

- While the test MSE for $\lambda_{optimal}$ is slightly higher than for $\log(\lambda) = -4$, the optimal λ model is simpler and less prone to overfitting.
- The scatter plot further confirms the predictive power of the model, as it closely aligns actual and predicted values.

Appendix

Code for Assignment 1

```r
# Load the data
meat_data <- read.csv("data/tecator.csv")

# Set seed for reproducibility
set.seed(123)

# Extract features and target variable
absorbance_features <- meat_data[, 2:101]
fat_content <- meat_data$Fat

# Split data into training (50%) and testing (50%)
train_indices <- sample(1:nrow(meat_data), size = nrow(meat_data) * 0.5)
train_features <- absorbance_features[train_indices, ]
train_target <- fat_content[train_indices]
test_features <- absorbance_features[-train_indices, ]
test_target <- fat_content[-train_indices]

# Standardize features
train_features_scaled <- scale(train_features)
test_features_scaled <- scale(test_features)

# Linear Regression
linear_model <- lm(train_target ~ ., data = as.data.frame(cbind(train_target, train_features_scaled)))
train_predictions <- predict(linear_model, newdata = as.data.frame(train_features_scaled))
test_predictions <- predict(linear_model, newdata = as.data.frame(test_features_scaled))
mse_train <- mean((train_target - train_predictions)^2)
mse_test <- mean((test_target - test_predictions)^2)
cat("Training MSE:", mse_train, "\n")
cat("Testing MSE:", mse_test, "\n")

# Load glmnet library
library(glmnet)
train_features_matrix <- as.matrix(train_features_scaled)
train_target_vector <- as.vector(train_target)

# LASSO Regression
lasso_model <- glmnet(train_features_matrix, train_target_vector, alpha = 1)
plot(lasso_model, xvar = "lambda", label = TRUE, col = c("blue", "green", "purple"))
title("LASSO Coefficients vs Log(Lambda)")
```

```
# Lambda with 3 non-zero features
lambda_values <- lasso_model$lambda
for (lambda in lambda_values) {
  coef_nonzero <- coef(lasso_model, s = lambda)[-1]
  num_features <- sum(coef_nonzero != 0)
  if (num_features == 3) {
    cat("Lambda for 3 features:", lambda, "\n")
    break
  }
}

# Extract and display coefficients for 3-feature lambda
selected_coefficients <- coef(lasso_model, s = lambda)
selected_coefficients_matrix <- as.matrix(selected_coefficients)
nonzero_indices <- which(selected_coefficients_matrix != 0)
selected_features <- data.frame(
  Feature = rownames(selected_coefficients_matrix)[nonzero_indices],
  Coefficient = selected_coefficients_matrix[nonzero_indices]
)
print(selected_features)

# Ridge Regression
ridge_model <- glmnet(train_features_matrix, train_target_vector, alpha = 0)
plot(ridge_model, xvar = "lambda", label = TRUE, col = c("orange", "cyan", "pink"))
title("Ridge Coefficients vs Log(Lambda)")

# Cross-Validation for LASSO
cv_lasso <- cv.glmnet(train_features_matrix, train_target_vector, alpha = 1)
plot(cv_lasso, col = "darkgreen")
title("Cross-Validation for LASSO: Error vs Log(Lambda)")
optimal_lambda <- cv_lasso$lambda.min
cat("Optimal Lambda:", optimal_lambda, "\n")

# Non-zero features for optimal lambda
optimal_coefficients <- coef(cv_lasso, s = "lambda.min")
nonzero_features_count <- sum(optimal_coefficients != 0) - 1
cat("Number of non-zero features at optimal lambda:", nonzero_features_count, "\n")

# Test performance for optimal lambda
lasso_test_predictions <- predict(cv_lasso, s = optimal_lambda, newx = as.matrix(test_features_scaled))
mse_test_lasso <- mean((test_target - lasso_test_predictions)^2)
cat("Test MSE for LASSO with Optimal Lambda:", mse_test_lasso, "\n")

# Comparison with log(lambda) = -4
lambda_at_log_minus4 <- exp(-4)
test_predictions_log_minus4 <- predict(lasso_model, s = lambda_at_log_minus4, newx = as.matrix(test_fea
mse_test_log_minus4 <- mean((test_target - test_predictions_log_minus4)^2)
cat("Test MSE for LASSO with log(lambda) = -4:", mse_test_log_minus4, "\n")

# Scatter plot for optimal lambda
plot(test_target, lasso_test_predictions,
     main = "Scatter Plot: Actual vs Predicted (Optimal Lambda)",
     xlab = "Actual Fat", ylab = "Predicted Fat", pch = 19, col = "blue")
```

```r
abline(0, 1, col = "red")
```

Code for Assignment 2

Code for Assignment 3