# Linear Regerssion

Yuqi Zhang

2024-05-17

```r
library(dplyr)
library(caret)
library(ggplot2)
library(rjags)
library(coda)
library(bookdown)
```

## Linear Regression Model

### Data Preparation

We first prepare the data for each cluster.

```r
aggregated_data <- read.csv("AggregatedData1.csv")
daily_data <- read.csv("Daily_AggregatedData1.csv")

cluster1 <- aggregated_data[, -c(2,3,4,5,6)]
cluster2 <- aggregated_data[, -c(1,3,4,5,6)]
cluster3 <- aggregated_data[, -c(1,2,4,5,6)]
cluster4 <- aggregated_data[, -c(1,2,3,5,6)]
cluster5 <- aggregated_data[, -c(1,2,3,4,6)]
cluster6 <- aggregated_data[, -c(1,2,3,4,5)]

cluster1_trans <- cluster1 %>%
  mutate(Cluster.1_lag1 = lag(Cluster.1, n = 1)) %>%
  mutate(
    tod_poly1 = tod,
    tod_poly2 = tod^2,
    tod_poly3 = tod^3,
    tod_poly4 = tod^4,
    weekend_dummy = ifelse(weekend == "TRUE", 1, 0),
    toy_sin = sin(toy),
    toy_cos = cos(toy)
  ) %>%
  na.omit()

cluster2_trans <- cluster2 %>%
  mutate(Cluster.2_lag1 = lag(Cluster.2, n = 1)) %>%
  mutate(
    tod_poly1 = tod,
    tod_poly2 = tod^2,
    tod_poly3 = tod^3,
    tod_poly4 = tod^4,
    weekend_dummy = ifelse(weekend == "TRUE", 1, 0),
    toy_sin = sin(toy),
    toy_cos = cos(toy)
  ) %>%
  na.omit()

cluster3_trans <- cluster3 %>%
  mutate(Cluster.3_lag1 = lag(Cluster.3, n = 1)) %>%
  mutate(
    tod_poly1 = tod,
    tod_poly2 = tod^2,
    tod_poly3 = tod^3,
    tod_poly4 = tod^4,
    weekend_dummy = ifelse(weekend == "TRUE", 1, 0),
    toy_sin = sin(toy),
    toy_cos = cos(toy)
  ) %>%
  na.omit()


cluster4_trans <- cluster4 %>%
  mutate(Cluster.4_lag1 = lag(Cluster.4, n = 1)) %>%
  mutate(
    tod_poly1 = tod,
    tod_poly2 = tod^2,
    tod_poly3 = tod^3,
    tod_poly4 = tod^4,
    weekend_dummy = ifelse(weekend == "TRUE", 1, 0),
    toy_sin = sin(toy),
    toy_cos = cos(toy)
  ) %>%
  na.omit()

cluster5_trans <- cluster5 %>%
  mutate(Cluster.5_lag1 = lag(Cluster.5, n = 1)) %>%
  mutate(
    tod_poly1 = tod,
    tod_poly2 = tod^2,
    tod_poly3 = tod^3,
    tod_poly4 = tod^4,
    weekend_dummy = ifelse(weekend == "TRUE", 1, 0),
    toy_sin = sin(toy),
    toy_cos = cos(toy)
  ) %>%
  na.omit()

cluster6_trans <- cluster6 %>%
  mutate(Cluster.6_lag1 = lag(Cluster.6, n = 1)) %>%
  mutate(
    tod_poly1 = tod,
    tod_poly2 = tod^2,
    tod_poly3 = tod^3,
    tod_poly4 = tod^4,
    weekend_dummy = ifelse(weekend == "TRUE", 1, 0),
    toy_sin = sin(toy),
    toy_cos = cos(toy)
  ) %>%
  na.omit()
```

### Clusrer 1

### Linear Regression Model for Cluster 1

On the basis of `cluster1_trans`, we fit a linear regression model and evaluate its performance:

```
# Split the data into training and testing sets
train_index_1 <- 1:floor(0.8 * nrow(cluster1_trans))
train_data_1 <- cluster1_trans[train_index_1, ]
test_data_1 <- cluster1_trans[-train_index_1, ]

# Fit the linear regression model
linear_model_1 <- lm(Cluster.1 ~ Cluster.1_lag1 + temp + tod_poly1 + tod_poly2 + tod_poly3 + tod_poly4
```

### Performance of Predictions

```
# Make predictions on the testing set
test_data_1$predictions <- predict(linear_model_1, newdata = test_data_1)

# Calculate prediction error metrics
mae_1 <- mean(abs(test_data_1$Cluster.1 - test_data_1$predictions))
mse_1 <- mean((test_data_1$Cluster.1 - test_data_1$predictions)^2)
rmse_1 <- sqrt(mse_1)

# Print the results
cat("Mean Absolute Error (MAE):", mae_1, "\n")
```

```
## Mean Absolute Error (MAE): 0.04186565
```

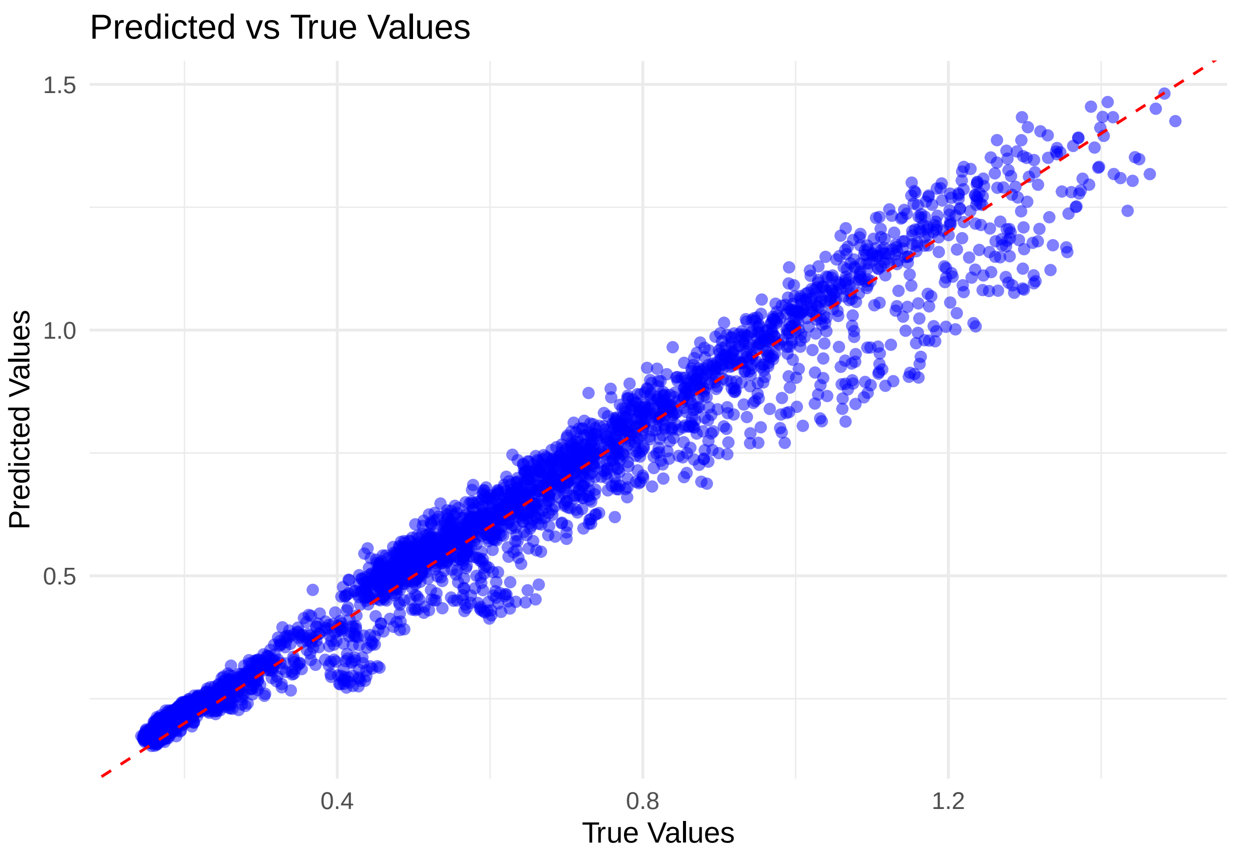
```
cat("Mean Squared Error (MSE):", mse_1, "\n")
```

```
## Mean Squared Error (MSE): 0.003343571
```

```
cat("Root Mean Squared Error (RMSE):", rmse_1, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.05782362
```

```
# Plot predicted vs true values
ggplot(test_data_1, aes(x = Cluster.1, y = predictions)) +
  geom_point(color = 'blue', alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, color = 'red', linetype = "dashed") +
  labs(title = "Predicted vs True Values",
       x = "True Values",
       y = "Predicted Values") +
  theme_minimal()
```



**Linear Regression Model Summary**

```r
summary(linear_model_1)
```

```
##
## Call:
## lm(formula = Cluster.1 ~ Cluster.1_lag1 + temp + tod_poly1 +
##     tod_poly2 + tod_poly3 + tod_poly4 + weekend_dummy + toy_sin +
##     toy_cos, data = train_data_1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.165153 -0.026932 -0.008028  0.017576  0.300232
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -3.660e-02  1.999e-02  -1.831   0.0671 .
## Cluster.1_lag1  9.265e-01  3.346e-03 276.867  < 2e-16 ***
## temp           -9.932e-04  1.286e-04  -7.725  1.2e-14 ***
## tod_poly1       2.293e-02  6.069e-04  37.780  < 2e-16 ***
## tod_poly2      -1.602e-03  5.276e-05 -30.367  < 2e-16 ***
## tod_poly3       5.057e-05  1.632e-06  30.989  < 2e-16 ***
## tod_poly4      -5.649e-07  1.682e-08 -33.597  < 2e-16 ***
## weekend_dummy   1.519e-03  9.309e-04   1.632   0.1027
## toy_sin        -9.073e-03  9.531e-03  -0.952   0.3411
## toy_cos        -3.571e-02  1.860e-02  -1.920   0.0549 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.048 on 13428 degrees of freedom
## Multiple R-squared:  0.9653, Adjusted R-squared:  0.9652
## F-statistic: 4.145e+04 on 9 and 13428 DF,  p-value: < 2.2e-16
```

### Cluster 2

### Linear Regression Model for Cluster 2

```r
# Split the data into training and testing sets
train_index_2 <- 1:floor(0.8 * nrow(cluster2_trans))
train_data_2 <- cluster2_trans[train_index_2, ]
test_data_2 <- cluster2_trans[-train_index_2, ]

# Fit the linear regression model
linear_model_2 <- lm(Cluster.2 ~ Cluster.2_lag1 + temp + tod_poly1 + tod_poly2 + tod_poly3 + tod_poly4 +
```

### Performance of Predictions

```r
# Make predictions on the testing set
test_data_2$predictions <- predict(linear_model_2, newdata = test_data_2)

# Calculate prediction error metrics
mae_2 <- mean(abs(test_data_2$Cluster.2 - test_data_2$predictions))
mse_2 <- mean((test_data_2$Cluster.2 - test_data_2$predictions)^2)
rmse_2 <- sqrt(mse_2)

# Print the results
cat("Mean Absolute Error (MAE):", mae_2, "\n")
```

```
## Mean Absolute Error (MAE): 0.03925669
```

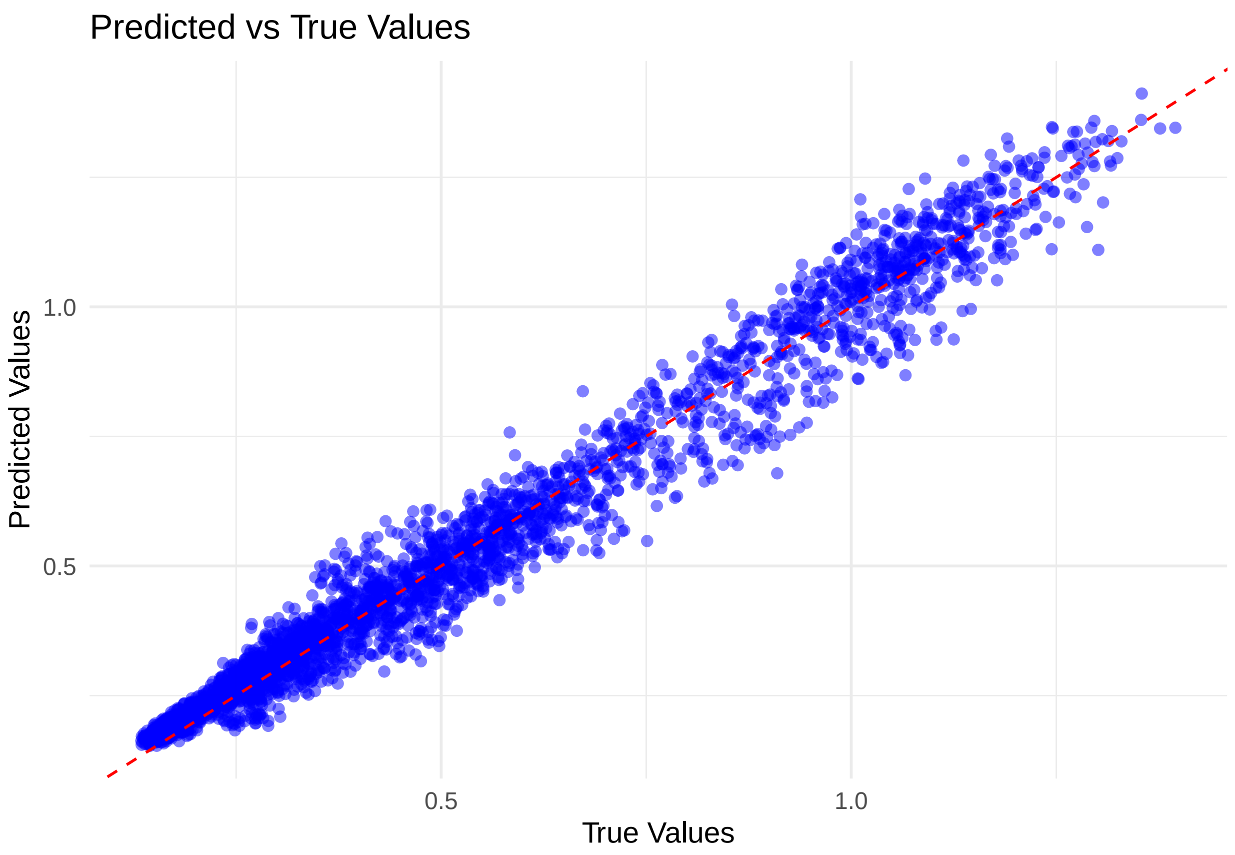
```r
cat("Mean Squared Error (MSE):", mse_2, "\n")
```

```
## Mean Squared Error (MSE): 0.002707433
```

```r
cat("Root Mean Squared Error (RMSE):", rmse_2, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.052033
```

```r
# Plot predicted vs true values
ggplot(test_data_2, aes(x = Cluster.2, y = predictions)) +
  geom_point(color = 'blue', alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, color = 'red', linetype = "dashed") +
  labs(title = "Predicted vs True Values",
       x = "True Values",
       y = "Predicted Values") +
  theme_minimal()
```



**Linear Regression Model Summary**

```r
summary(linear_model_2)
```

```
##
## Call:
## lm(formula = Cluster.2 ~ Cluster.2_lag1 + temp + tod_poly1 +
##     tod_poly2 + tod_poly3 + tod_poly4 + weekend_dummy + toy_sin +
##     toy_cos, data = train_data_2)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.196463 -0.024745 -0.005851  0.020245  0.263215
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.276e-01  1.877e-02  -6.799 1.10e-11 ***
## Cluster.2_lag1  9.571e-01  2.936e-03 326.014  < 2e-16 ***
## temp           -9.467e-04  1.217e-04  -7.778 7.87e-15 ***
## tod_poly1       3.788e-02  5.538e-04  68.407  < 2e-16 ***
## tod_poly2      -3.201e-03  4.786e-05 -66.870  < 2e-16 ***
## tod_poly3       1.053e-04  1.510e-06  69.782  < 2e-16 ***
## tod_poly4      -1.148e-06  1.579e-08 -72.691  < 2e-16 ***
## weekend_dummy   1.161e-03  8.849e-04   1.312    0.190
## toy_sin         1.321e-02  8.965e-03   1.474    0.141
## toy_cos         5.188e-03  1.752e-02   0.296    0.767
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.04535 on 13428 degrees of freedom
## Multiple R-squared:  0.9669, Adjusted R-squared:  0.9669
## F-statistic: 4.359e+04 on 9 and 13428 DF,  p-value: < 2.2e-16
```

### Cluster 3

#### Linear Regression Model for Cluster 3

```r
# Split the data into training and testing sets
train_index_3 <- 1:floor(0.8 * nrow(cluster3_trans))
train_data_3 <- cluster3_trans[train_index_3, ]
test_data_3 <- cluster3_trans[-train_index_3, ]

# Fit the linear regression model
linear_model_3 <- lm(Cluster.3 ~ Cluster.3_lag1 + temp + tod_poly1 + tod_poly2 + tod_poly3 + tod_poly4 +
```

#### Performance of Predictions

```r
# Make predictions on the testing set
test_data_3$predictions <- predict(linear_model_3, newdata = test_data_3)

# Calculate prediction error metrics
mae_3 <- mean(abs(test_data_3$Cluster.3 - test_data_3$predictions))
mse_3 <- mean((test_data_3$Cluster.3 - test_data_3$predictions)^2)
rmse_3 <- sqrt(mse_3)

# Print the results
cat("Mean Absolute Error (MAE):", mae_3, "\n")
```

```
## Mean Absolute Error (MAE): 0.0290271
```

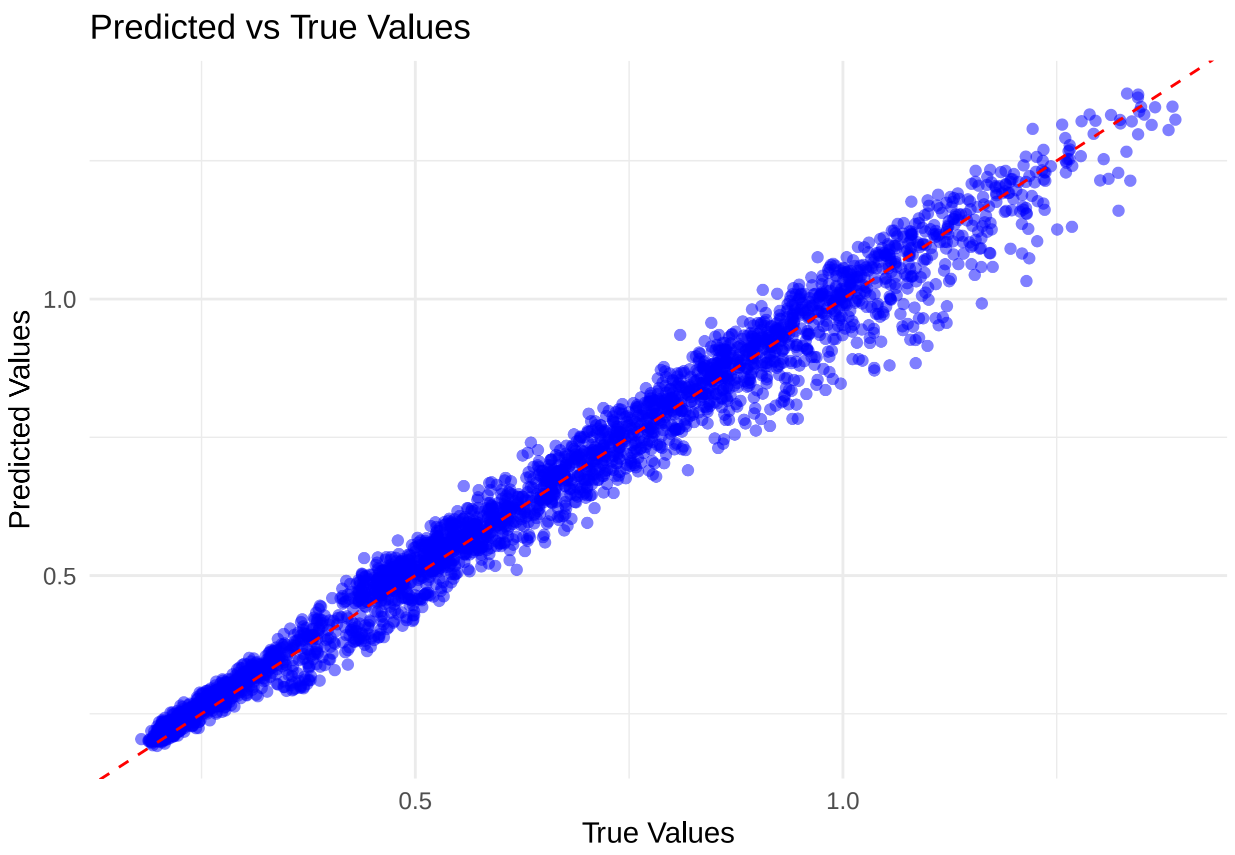
```r
cat("Mean Squared Error (MSE):", mse_3, "\n")
```

```
## Mean Squared Error (MSE): 0.001502561
```

```r
cat("Root Mean Squared Error (RMSE):", rmse_3, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.03876289
```

```r
# Plot predicted vs true values
ggplot(test_data_3, aes(x = Cluster.3, y = predictions)) +
  geom_point(color = 'blue', alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, color = 'red', linetype = "dashed") +
  labs(title = "Predicted vs True Values",
       x = "True Values",
       y = "Predicted Values") +
  theme_minimal()
```



**Linear Regression Model Summary**

```r
summary(linear_model_3)
```

```
##
## Call:
## lm(formula = Cluster.3 ~ Cluster.3_lag1 + temp + tod_poly1 +
##     tod_poly2 + tod_poly3 + tod_poly4 + weekend_dummy + toy_sin +
##     toy_cos, data = train_data_3)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.118353 -0.018677 -0.002823  0.016148  0.198274
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -2.665e-02  1.459e-02  -1.827  0.06779 .
## Cluster.3_lag1  9.349e-01  3.359e-03 278.292  < 2e-16 ***
## temp           -7.840e-04  8.744e-05  -8.966  < 2e-16 ***
## tod_poly1       1.486e-02  5.223e-04  28.443  < 2e-16 ***
## tod_poly2      -7.698e-04  4.278e-05 -17.994  < 2e-16 ***
## tod_poly3       2.094e-05  1.248e-06  16.775  < 2e-16 ***
## tod_poly4      -2.247e-07  1.237e-08 -18.173  < 2e-16 ***
## weekend_dummy   1.569e-03  6.391e-04   2.455  0.01409 *
## toy_sin        -1.191e-02  6.667e-03  -1.786  0.07406 .
## toy_cos        -3.624e-02  1.299e-02  -2.790  0.00529 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.03262 on 13428 degrees of freedom
## Multiple R-squared:  0.9797, Adjusted R-squared:  0.9797
## F-statistic: 7.193e+04 on 9 and 13428 DF,  p-value: < 2.2e-16
```

### Cluster 4

### Linear Regression Model for Cluster 4

```r
# Split the data into training and testing sets
train_index_4 <- 1:floor(0.8 * nrow(cluster4_trans))
train_data_4 <- cluster4_trans[train_index_4, ]
test_data_4 <- cluster4_trans[-train_index_4, ]

# Fit the linear regression model
linear_model_4 <- lm(Cluster.4 ~ Cluster.4_lag1 + temp + tod_poly1 + tod_poly2 + tod_poly3 + tod_poly4 

# Make predictions on the testing set
test_data_4$predictions <- predict(linear_model_4, newdata = test_data_4)
```

### Performance of Predictions

```r
# Calculate prediction error metrics
mae_4 <- mean(abs(test_data_4$Cluster.4 - test_data_4$predictions))
mse_4 <- mean((test_data_4$Cluster.4 - test_data_4$predictions)^2)
rmse_4 <- sqrt(mse_4)

# Print the results
cat("Mean Absolute Error (MAE):", mae_4, "\n")
```

```
## Mean Absolute Error (MAE): 0.04281978
```

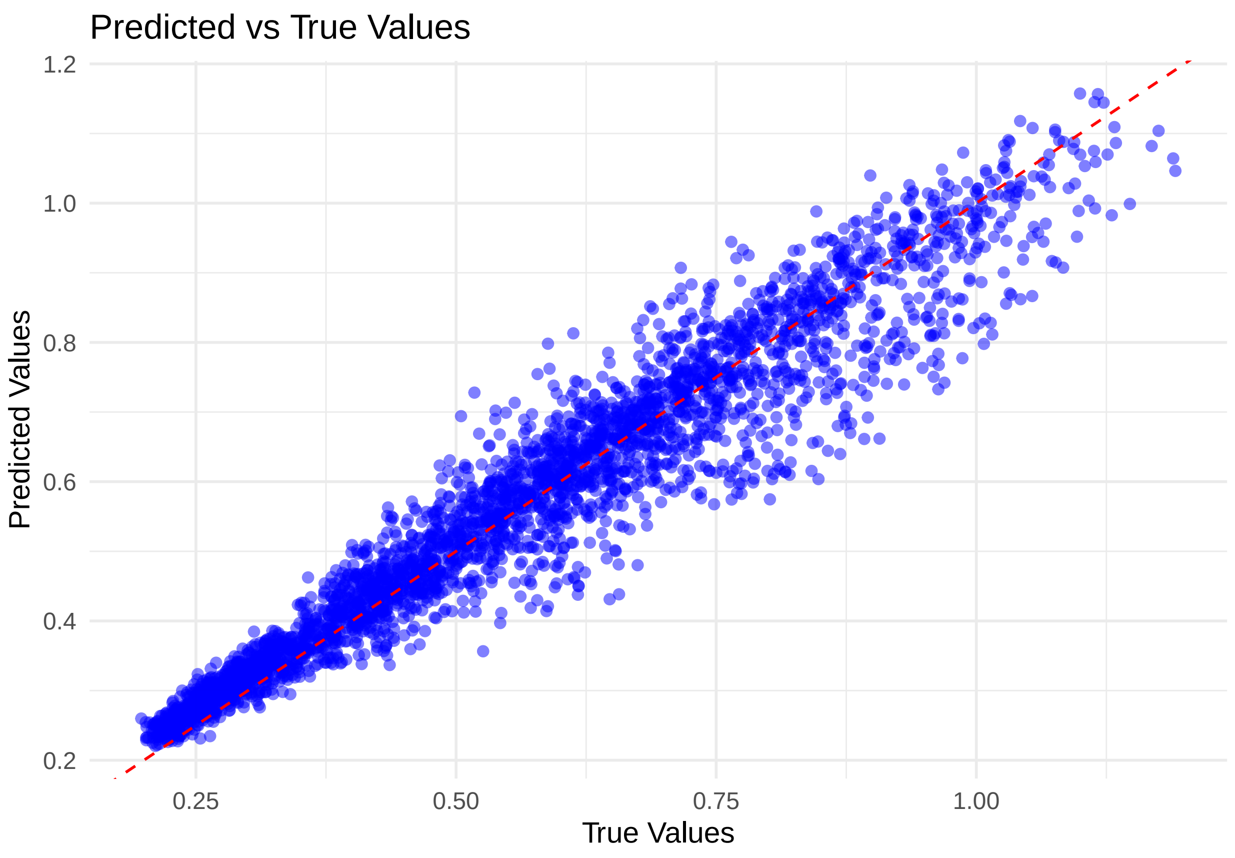
```r
cat("Mean Squared Error (MSE):", mse_4, "\n")
```

```
## Mean Squared Error (MSE): 0.003455547
```

```r
cat("Root Mean Squared Error (RMSE):", rmse_4, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.0587839
```

```r
# Plot predicted vs true values
ggplot(test_data_4, aes(x = Cluster.4, y = predictions)) +
  geom_point(color = 'blue', alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, color = 'red', linetype = "dashed") +
  labs(title = "Predicted vs True Values",
       x = "True Values",
       y = "Predicted Values") +
  theme_minimal()
```



**Linear Regression Model Summary**

```r
summary(linear_model_4)
```

```
##
## Call:
## lm(formula = Cluster.4 ~ Cluster.4_lag1 + temp + tod_poly1 +
##     tod_poly2 + tod_poly3 + tod_poly4 + weekend_dummy + toy_sin +
##     toy_cos, data = train_data_4)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.220407 -0.030273 -0.006641  0.024947  0.279164
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.487e-02  2.205e-02   1.581  0.11389
## Cluster.4_lag1  8.858e-01  3.853e-03 229.912  < 2e-16 ***
## temp           -1.079e-03  1.438e-04  -7.502  6.7e-14 ***
## tod_poly1       2.221e-02  6.335e-04  35.058  < 2e-16 ***
## tod_poly2      -1.660e-03  5.618e-05 -29.545  < 2e-16 ***
## tod_poly3       5.144e-05  1.786e-06  28.805  < 2e-16 ***
## tod_poly4      -5.492e-07  1.866e-08 -29.431  < 2e-16 ***
## weekend_dummy   1.669e-03  1.035e-03   1.613  0.10670
## toy_sin        -2.580e-02  1.059e-02  -2.436  0.01486 *
## toy_cos        -6.427e-02  2.065e-02  -3.112  0.00186 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.05364 on 13428 degrees of freedom
## Multiple R-squared:  0.9081, Adjusted R-squared:  0.908
## F-statistic: 1.474e+04 on 9 and 13428 DF,  p-value: < 2.2e-16
```

### Cluster 5

### Linear Regression Model for Cluster 5

```r
# Split the data into training and testing sets
train_index_5 <- 1:floor(0.8 * nrow(cluster5_trans))
train_data_5 <- cluster5_trans[train_index_5, ]
test_data_5 <- cluster5_trans[-train_index_5, ]

# Fit the linear regression model
linear_model_5 <- lm(Cluster.5 ~ Cluster.5_lag1 + temp + tod_poly1 + tod_poly2 + tod_poly3 + tod_poly4 
```

### Performance of Predictions

```r
# Make predictions on the testing set
test_data_5$predictions <- predict(linear_model_5, newdata = test_data_5)

# Calculate prediction error metrics
mae_5 <- mean(abs(test_data_5$Cluster.5 - test_data_5$predictions))
mse_5 <- mean((test_data_5$Cluster.5 - test_data_5$predictions)^2)
rmse_5 <- sqrt(mse_5)

# Print the results
cat("Mean Absolute Error (MAE):", mae_5, "\n")
```

```
## Mean Absolute Error (MAE): 0.03355503
```

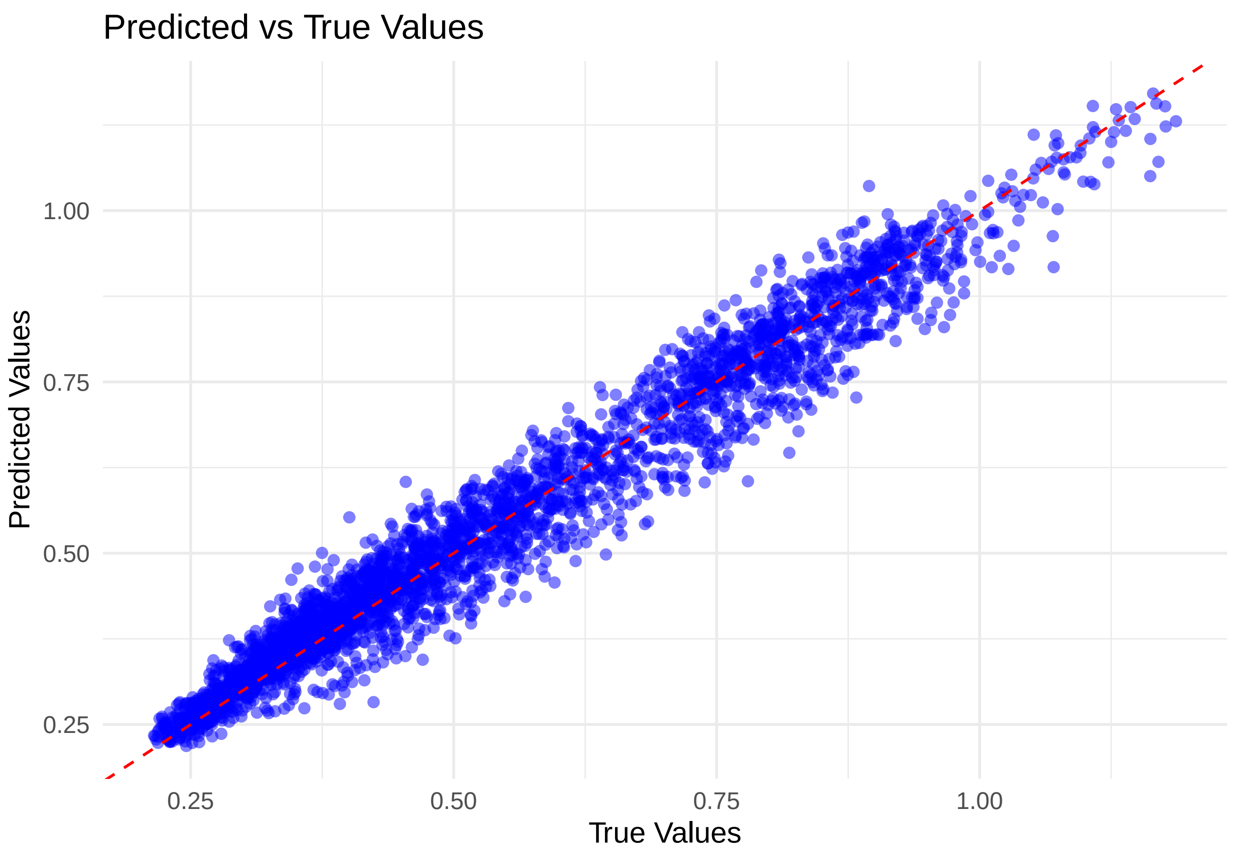
```r
cat("Mean Squared Error (MSE):", mse_5, "\n")
```

```
## Mean Squared Error (MSE): 0.001909648
```

```r
cat("Root Mean Squared Error (RMSE):", rmse_5, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.04369951
```

```r
# Plot predicted vs true values
ggplot(test_data_5, aes(x = Cluster.5, y = predictions)) +
  geom_point(color = 'blue', alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, color = 'red', linetype = "dashed") +
  labs(title = "Predicted vs True Values",
       x = "True Values",
       y = "Predicted Values") +
  theme_minimal()
```



**Linear Regression Model Summary**

```r
summary(linear_model_5)
```

```
##
## Call:
## lm(formula = Cluster.5 ~ Cluster.5_lag1 + temp + tod_poly1 +
##     tod_poly2 + tod_poly3 + tod_poly4 + weekend_dummy + toy_sin +
##     toy_cos, data = train_data_5)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.139892 -0.024421 -0.002343  0.020704  0.202596
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     8.670e-02  1.768e-02   4.905 9.45e-07 ***
## Cluster.5_lag1  8.866e-01  4.144e-03 213.968  < 2e-16 ***
## temp           -1.149e-03  1.040e-04 -11.049  < 2e-16 ***
## tod_poly1       4.041e-03  6.185e-04   6.535 6.60e-11 ***
## tod_poly2      -9.191e-05  4.936e-05  -1.862   0.0626 .
## tod_poly3       1.671e-06  1.483e-06   1.126   0.2601
## tod_poly4      -1.001e-08  1.514e-08  -0.661   0.5086
## weekend_dummy   3.885e-03  7.573e-04   5.131 2.93e-07 ***
## toy_sin        -3.745e-02  7.988e-03  -4.688 2.78e-06 ***
## toy_cos        -7.989e-02  1.546e-02  -5.167 2.41e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.0387 on 13428 degrees of freedom
## Multiple R-squared:  0.9521, Adjusted R-squared:  0.9521
## F-statistic: 2.966e+04 on 9 and 13428 DF,  p-value: < 2.2e-16
```

### Cluster 6

### Linear Regression Model for Cluster 6

```r
# Split the data into training and testing sets
train_index_6 <- 1:floor(0.8 * nrow(cluster6_trans))
train_data_6 <- cluster6_trans[train_index_6, ]
test_data_6 <- cluster6_trans[-train_index_6, ]

# Fit the linear regression model
linear_model_6 <- lm(Cluster.6 ~ Cluster.6_lag1 + temp + tod_poly1 + tod_poly2 + tod_poly3 + tod_poly4 
```

### Performance of Predictions

```r
# Make predictions on the testing set
test_data_6$predictions <- predict(linear_model_6, newdata = test_data_6)

# Calculate prediction error metrics
mae_6 <- mean(abs(test_data_6$Cluster.6 - test_data_6$predictions))
mse_6 <- mean((test_data_6$Cluster.6 - test_data_6$predictions)^2)
rmse_6 <- sqrt(mse_6)

# Print the results
cat("Mean Absolute Error (MAE):", mae_6, "\n")
```

```
## Mean Absolute Error (MAE): 0.03879818
```

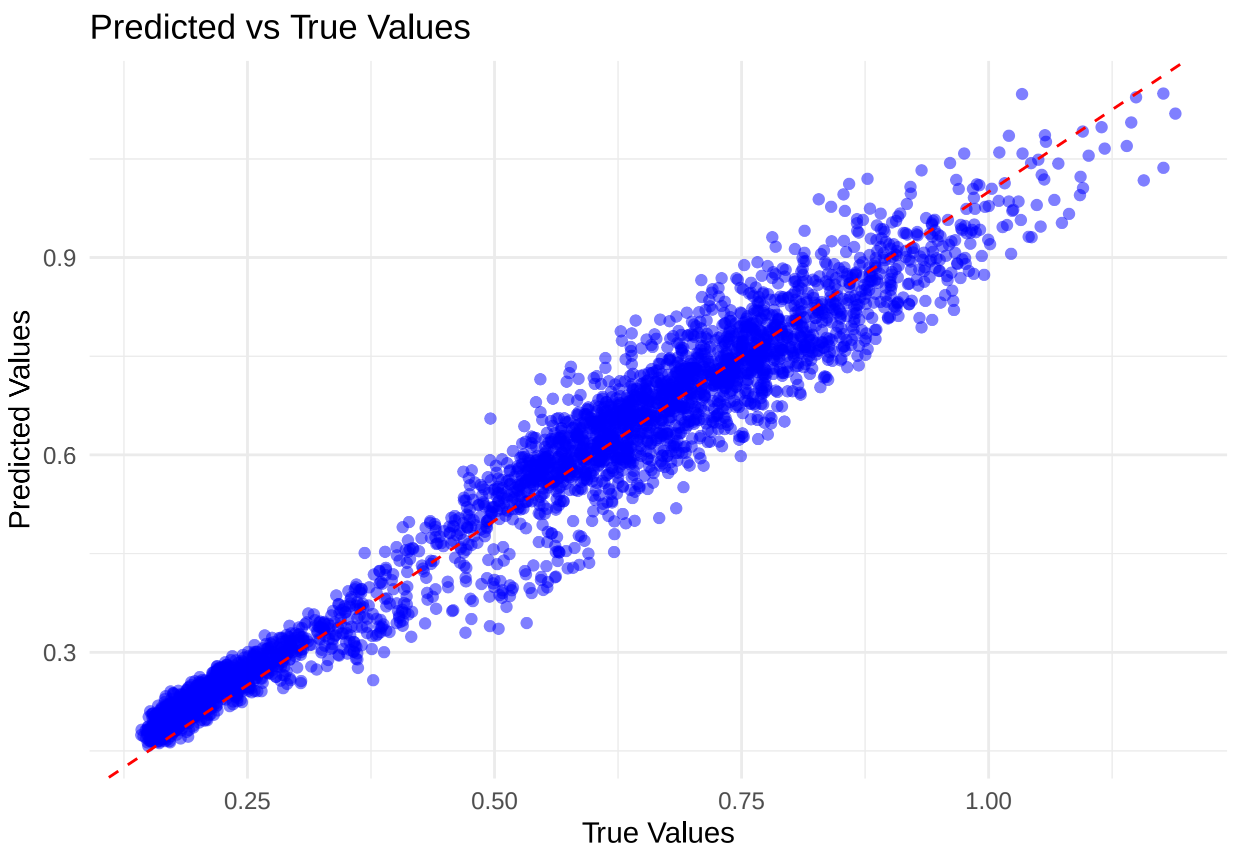
```r
cat("Mean Squared Error (MSE):", mse_6, "\n")
```

```
## Mean Squared Error (MSE): 0.002503867
```

```r
cat("Root Mean Squared Error (RMSE):", rmse_6, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.05003866
```

```r
# Plot predicted vs true values
ggplot(test_data_6, aes(x = Cluster.6, y = predictions)) +
  geom_point(color = 'blue', alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1, color = 'red', linetype = "dashed") +
  labs(title = "Predicted vs True Values",
       x = "True Values",
       y = "Predicted Values") +
  theme_minimal()
```



**Linear Regression Model Summary**

```r
summary(linear_model_6)
```

```
##
## Call:
## lm(formula = Cluster.6 ~ Cluster.6_lag1 + temp + tod_poly1 +
##     tod_poly2 + tod_poly3 + tod_poly4 + weekend_dummy + toy_sin +
##     toy_cos, data = train_data_6)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.213397 -0.027366 -0.004184  0.025137  0.239803
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.241e-02  2.120e-02   1.529 0.126388
## Cluster.6_lag1  8.936e-01  4.408e-03 202.700  < 2e-16 ***
## temp           -1.131e-03  1.308e-04  -8.649  < 2e-16 ***
## tod_poly1       1.288e-02  7.602e-04  16.943  < 2e-16 ***
## tod_poly2      -4.028e-04  7.078e-05  -5.692 1.28e-08 ***
## tod_poly3       4.116e-06  2.202e-06   1.870 0.061538 .
## tod_poly4      -1.317e-08  2.220e-08  -0.593 0.553046
## weekend_dummy   1.425e-03  9.487e-04   1.502 0.133196
## toy_sin        -2.796e-02  9.974e-03  -2.803 0.005074 **
## toy_cos        -6.734e-02  1.937e-02  -3.476 0.000511 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.04909 on 13428 degrees of freedom
## Multiple R-squared:  0.9454, Adjusted R-squared:  0.9453
## F-statistic: 2.582e+04 on 9 and 13428 DF,  p-value: < 2.2e-16
```

### Conclusion for Linear Regression Models

```
coefficients_1 <- summary(linear_model_1)$coefficients
coefficients_2 <- summary(linear_model_2)$coefficients
coefficients_3 <- summary(linear_model_3)$coefficients
coefficients_4 <- summary(linear_model_4)$coefficients
coefficients_5 <- summary(linear_model_5)$coefficients
coefficients_6 <- summary(linear_model_6)$coefficients

performance_1 <- data.frame(MAE = mae_1, MSE = mse_1, RMSE = rmse_1)
performance_2 <- data.frame(MAE = mae_2, MSE = mse_2, RMSE = rmse_2)
performance_3 <- data.frame(MAE = mae_3, MSE = mse_3, RMSE = rmse_3)
performance_4 <- data.frame(MAE = mae_4, MSE = mse_4, RMSE = rmse_4)
performance_5 <- data.frame(MAE = mae_5, MSE = mse_5, RMSE = rmse_5)
performance_6 <- data.frame(MAE = mae_6, MSE = mse_6, RMSE = rmse_6)

performance_summary <- rbind(
  data.frame(Cluster = "Cluster 1", performance_1),
  data.frame(Cluster = "Cluster 2", performance_2),
  data.frame(Cluster = "Cluster 3", performance_3),
  data.frame(Cluster = "Cluster 4", performance_4),
  data.frame(Cluster = "Cluster 5", performance_5),
  data.frame(Cluster = "Cluster 6", performance_6)
)

knitr::kable(performance_summary, caption = "Performance Metrics for Each Cluster", booktabs = TRUE)
```

Table 1: Performance Metrics for Each Cluster

| Cluster | MAE | MSE | RMSE |
|---|---|---|---|
| Cluster 1 | 0.0418657 | 0.0033436 | 0.0578236 |
| Cluster 2 | 0.0392567 | 0.0027074 | 0.0520330 |
| Cluster 3 | 0.0290271 | 0.0015026 | 0.0387629 |
| Cluster 4 | 0.0428198 | 0.0034555 | 0.0587839 |
| Cluster 5 | 0.0335550 | 0.0019096 | 0.0436995 |
| Cluster 6 | 0.0387982 | 0.0025039 | 0.0500387 |

```
coef_summary <- data.frame(
  #Variable = rownames(coefficients_1),
  `Cluster 1` = coefficients_1[, "Estimate"],
  `Cluster 2` = coefficients_2[, "Estimate"],
  `Cluster 3` = coefficients_3[, "Estimate"],
  `Cluster 4` = coefficients_4[, "Estimate"],
  `Cluster 5` = coefficients_5[, "Estimate"],
  `Cluster 6` = coefficients_6[, "Estimate"]
)

knitr::kable(coef_summary, caption = "Coefficients Summary for Each Cluster", booktabs = TRUE)
```

Table 2: Coefficients Summary for Each Cluster

| | Cluster.1 | Cluster.2 | Cluster.3 | Cluster.4 | Cluster.5 | Cluster.6 |
|---|---|---|---|---|---|---|
| (Intercept) | -0.0365963 | -0.1276174 | -0.0266471 | 0.0348692 | 0.0867022 | 0.0324100 |
| Cluster.1_lag1 | 0.9264920 | 0.9571412 | 0.9348833 | 0.8857962 | 0.8866198 | 0.8935789 |
| temp | -0.0009932 | -0.0009467 | -0.0007840 | -0.0010790 | -0.0011490 | -0.0011315 |
| tod_poly1 | 0.0229293 | 0.0378848 | 0.0148552 | 0.0222100 | 0.0040415 | 0.0128805 |
| tod_poly2 | -0.0016023 | -0.0032007 | -0.0007698 | -0.0016599 | -0.0000919 | -0.0004028 |
| tod_poly3 | 0.0000506 | 0.0001053 | 0.0000209 | 0.0000514 | 0.0000017 | 0.0000041 |
| tod_poly4 | -0.0000006 | -0.0000011 | -0.0000002 | -0.0000005 | 0.0000000 | 0.0000000 |
| weekend_dummy | 0.0015193 | 0.0011610 | 0.0015693 | 0.0016690 | 0.0038854 | 0.0014246 |
| toy_sin | -0.0090731 | 0.0132123 | -0.0119092 | -0.0257991 | -0.0374516 | -0.0279551 |
| toy_cos | -0.0357117 | 0.0051879 | -0.0362438 | -0.0642703 | -0.0798886 | -0.0673353 |

The tables referred to here as Table 1 and Table 2 provide a detailed overview of the initial performance metrics (using MAE (Mean Absolute Error), MSE (Mean Squared Error), and RMSE (Root Mean Squared Error)) and model coefficients for each cluster. These serve as a foundational baseline for assessing the effectiveness of the linear regression models. The primary purpose of these tables is to establish initial values and benchmarks for evaluating model performance. The insights gained from these metrics and coefficients are critical for understanding the model's predictive capabilities and identifying areas for potential improvement.

This foundational information will guide further analysis and model refinement, thereby enhancing our understanding of the underlying structure of the dataset and the effectiveness of the linear regression models.