assignment_2

2024-10-16

```
library(caTools)
```

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

```
library(ggplot2)
library(gridExtra)
```

Problem 1. Regression

```
data <- read.csv("qsar_aquatic_toxicity.csv", sep = ";", header = FALSE)
names(data) <- c(
    "TPSA",
    "SAacc",
    "H050",
    "MLOGP",
    "RDCHI",
    "GATS1p",
    "nN",
    "C040",
    "LC50"
)

head(data)
```

```
##     TPSA   SAacc H050 MLOGP RDCHI GATS1p nN C040  LC50
## 1   0.00   0.000    0 2.419 1.225  0.667  0    0 3.740
## 2   0.00   0.000    0 2.638 1.401  0.632  0    0 4.330
## 3   9.23  11.000    0 5.799 2.930  0.486  0    0 7.019
## 4   9.23  11.000    0 5.453 2.887  0.495  0    0 6.723
## 5   9.23  11.000    0 4.068 2.758  0.695  0    0 5.979
## 6 215.34 327.629    3 0.189 4.677  1.333  0    4 6.064
```

a. Split the data into a training and a test set, with approximately 2/3 and 1/3 of the observations, respectively.

```
# Use 70% of dataset as training set and remaining 30% as testing set
sample <- sample.split(data$LC50, SplitRatio = 0.7)
train  <- subset(data, sample == TRUE)
test   <- subset(data, sample == FALSE)
```

```r
cat("Dimension of Training Set:", paste(dim(train), collapse = "x"), "\nDimension of Test Set:", paste(
```

```
## Dimension of Training Set: 382x9
## Dimension of Test Set: 164x9
```

(i) Model each of them directly as a linear effect

```r
train_i = train
test_i = test
```

```r
# Fit linear regression model on training data
model <- lm(LC50 ~ ., data=train_i)

summary(model)
```

```
##
## Call:
## lm(formula = LC50 ~ ., data = train_i)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4817 -0.8215 -0.1059  0.6366  4.7001
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.551232   0.311360   8.194 4.09e-15 ***
## TPSA         0.026898   0.003428   7.847 4.54e-14 ***
## SAacc       -0.015250   0.002634  -5.789 1.50e-08 ***
## H050        -0.010285   0.075442  -0.136 0.891635
## MLOGP        0.451685   0.077045   5.863 1.00e-08 ***
## RDCHI        0.510956   0.166623   3.067 0.002323 **
## GATS1p      -0.332342   0.193356  -1.719 0.086479 .
## nN          -0.205046   0.056465  -3.631 0.000321 ***
## C040         0.042407   0.090135   0.470 0.638282
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.214 on 373 degrees of freedom
## Multiple R-squared:  0.4465, Adjusted R-squared:  0.4347
## F-statistic: 37.62 on 8 and 373 DF,  p-value: < 2.2e-16
```

```r
# Predict on training and test datasets
pred_train <- predict(model, newdata=train_i)
pred_test <- predict(model, newdata=test_i)
```

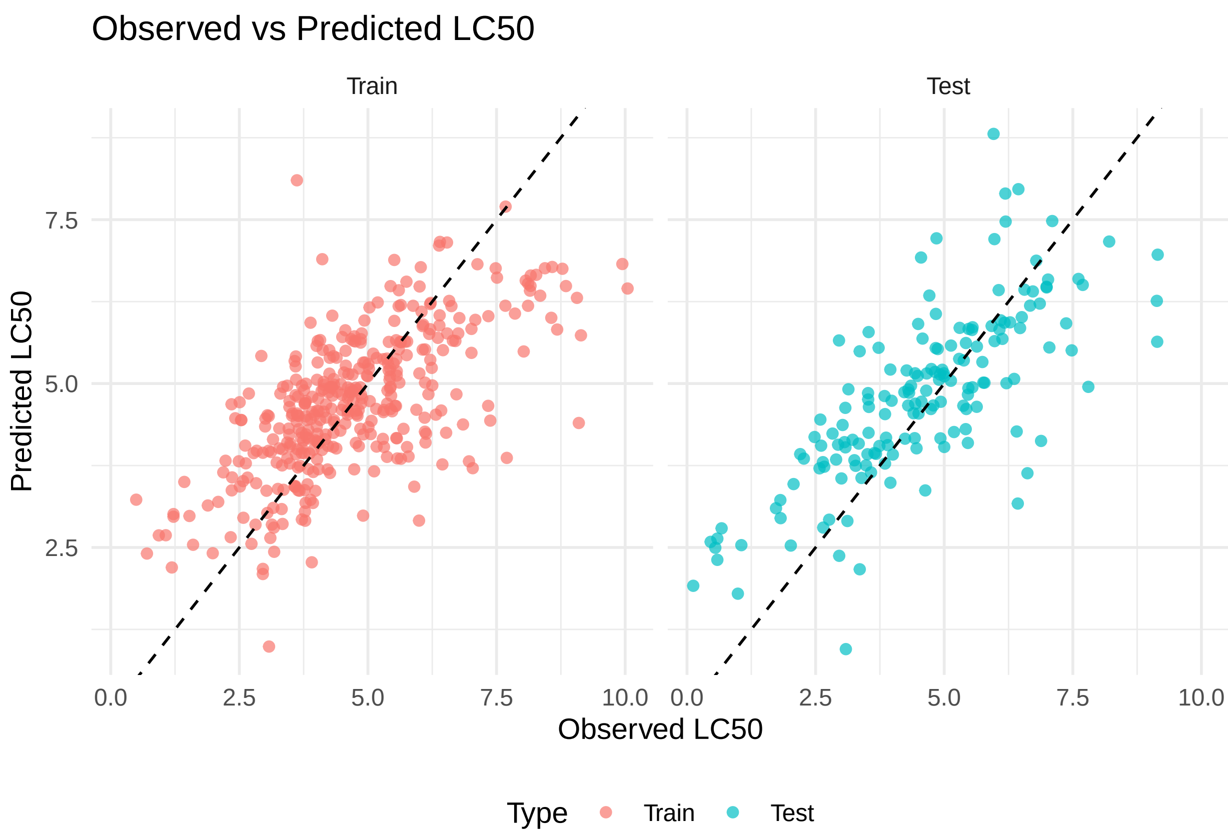
```r
# Adding predictions columns to the datasets
train_i$predicted_LC50 <- pred_train
test_i$predicted_LC50 <- pred_test
```

```r
# Evaluate model: calculate MSE, RMSE, and R-squared for training and test sets
mse_train <- mean((train_i$LC50 - train_i$predicted_LC50)^2)
rmse_train <- sqrt(mse_train)
r2_train <- 1 - (sum((train_i$LC50 - train_i$predicted_LC50)^2) / sum((train_i$LC50 - mean(train_i$LC50)

mse_test <- mean((test_i$LC50 - test_i$predicted_LC50)^2)
rmse_test <- sqrt(mse_test)
r2_test <- 1 - (sum((test_i$LC50 - test_i$predicted_LC50)^2) / sum((test_i$LC50 - mean(test_i$LC50))^2)
```

```r
cat(paste0(
  "Training Metrics:\n",
  "MSE (Train): ", mse_train, "\n",
  "RMSE (Train): ", rmse_train, "\n",
  "R-squared (Train): ", r2_train, "\n\n",

  "Test Metrics:\n",
  "MSE (Test): ", mse_test, "\n",
  "RMSE (Test): ", rmse_test, "\n",
  "R-squared (Test): ", r2_test, "\n"
))
```

```
## Training Metrics:
## MSE (Train): 1.43816162556801
## RMSE (Train): 1.19923376602229
## R-squared (Train): 0.446537490133659
##
## Test Metrics:
## MSE (Test): 1.45799438483515
## RMSE (Test): 1.20747438268278
## R-squared (Test): 0.537535791029839
```

```r
# Combine data for plotting
train_i$Type <- 'Train'
test_i$Type <- 'Test'
combined_data <- rbind(train_i, test_i)

combined_data$Type <- factor(combined_data$Type, levels = c('Train', 'Test'))

# Plotting observed vs predicted LC50 values
ggplot(combined_data, aes(x = LC50, y = predicted_LC50, color = Type)) +
  geom_point(alpha = 0.7) +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed") +
  labs(title = "Observed vs Predicted LC50", x = "Observed LC50", y = "Predicted LC50") +
  theme_minimal() +
  facet_wrap(~Type) +
  theme(legend.position = "bottom")
```



(ii). Transform each of them using a 0/1 dummy encoding where 0 represents absence of the specific atom and 1 represents presence of the specific atoms.

```
# To make sure we use the same split in (i)
train_ii = train
test_ii = test
```

```
# Transform 3 count variable (H050, nN, C040) into 0/1 in train and test datasets

train_ii$H050 <- ifelse(train_ii$H050 > 0, 1, 0)
train_ii$nN <- ifelse(train_ii$nN > 0, 1, 0)
train_ii$C040 <- ifelse(train_ii$C040 > 0, 1, 0)

test_ii$H050 <- ifelse(test_ii$H050 > 0, 1, 0)
test_ii$nN <- ifelse(test_ii$nN > 0, 1, 0)
test_ii$C040 <- ifelse(test_ii$C040 > 0, 1, 0)
```

```
head(train_ii)
```

```
##       TPSA   SAacc H050 MLOGP RDCHI GATS1p nN C040  LC50
## 2     0.00   0.000    0 2.638 1.401  0.632  0    0 4.330
## 6   215.34 327.629    1 0.189 4.677  1.333  0    1 6.064
## 7     9.23  11.000    0 2.723 2.321  1.165  0    0 7.337
```

```
## 8    0.00   0.000    0 3.267 2.318  0.963  0    0 4.100
## 9    0.00   0.000    0 2.067 1.800  1.250  0    0 3.941
## 10   0.00   0.000    0 2.746 1.667  1.400  0    0 3.809
```

```r
# Fit linear regression model on transformed training data
model_transform_dummy <- lm(LC50 ~ ., data = train_ii)

summary(model_transform_dummy)
```

```
##
## Call:
## lm(formula = LC50 ~ ., data = train_ii)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2602 -0.7723 -0.1402  0.6336  4.8362
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.631299   0.317988   8.275 2.31e-15 ***
## TPSA         0.021993   0.003434   6.405 4.53e-10 ***
## SAacc       -0.012924   0.002374  -5.443 9.52e-08 ***
## H050        -0.194007   0.155401  -1.248  0.21266
## MLOGP        0.465837   0.076862   6.061 3.32e-09 ***
## RDCHI        0.459961   0.169024   2.721  0.00681 **
## GATS1p      -0.308447   0.191170  -1.613  0.10749
## nN          -0.014816   0.148970  -0.099  0.92083
## C040        -0.053874   0.163704  -0.329  0.74227
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.234 on 373 degrees of freedom
## Multiple R-squared:  0.428,  Adjusted R-squared:  0.4157
## F-statistic: 34.89 on 8 and 373 DF,  p-value: < 2.2e-16
```

```r
# Predict on training and test datasets
pred_train_transform_dummy <- predict(model, newdata=train_ii)
pred_test_transform_dummy <- predict(model, newdata=test_ii)
```

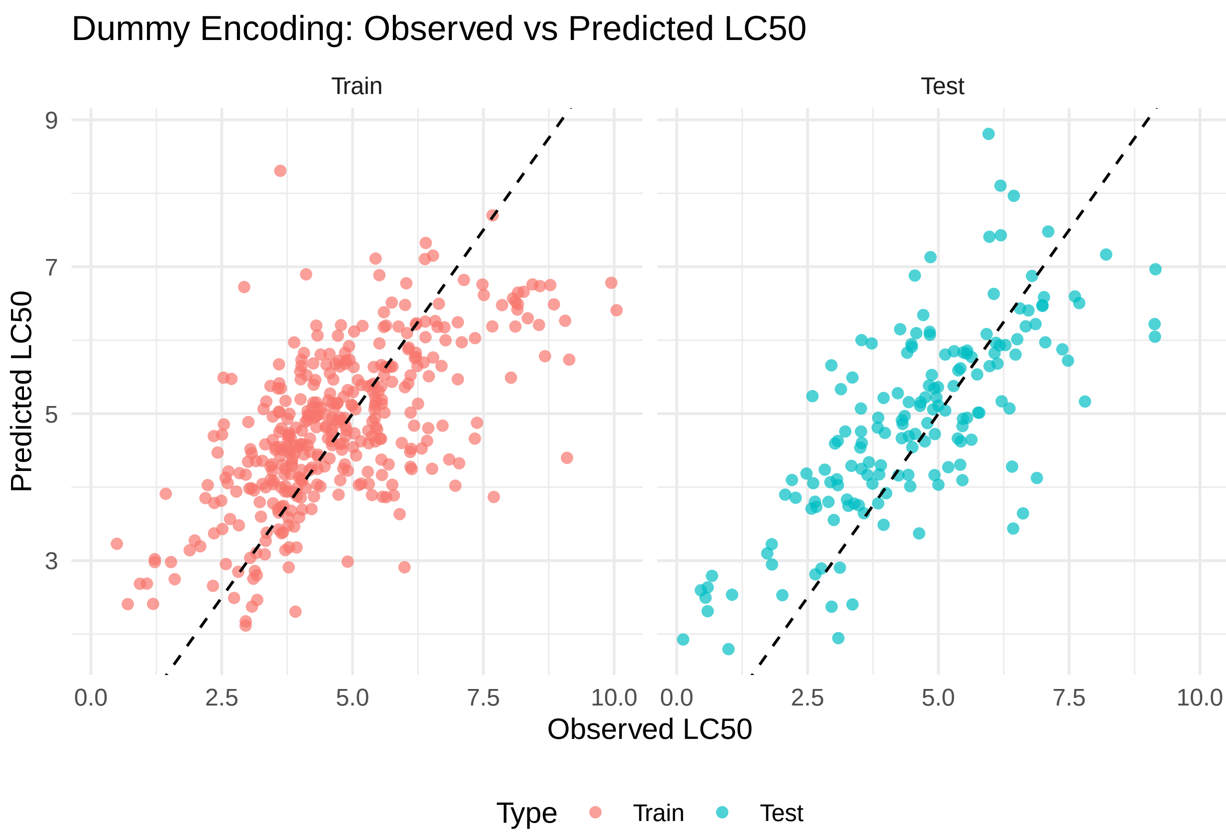
```r
# Adding predictions columns to the datasets
train_ii$predicted_LC50 <- pred_train_transform_dummy
test_ii$predicted_LC50 <- pred_test_transform_dummy
```

```r
# Evaluate model: calculate MSE, RMSE, and R-squared for training and test sets
mse_train_transform_dummy <- mean((train_ii$LC50 - train_ii$predicted_LC50)^2)
rmse_train_transform_dummy <- sqrt(mse_train_transform_dummy)
r2_train_transform_dummy <- 1 - (sum((train_ii$LC50 - train_ii$predicted_LC50)^2) / sum((train_ii$LC50 -

mse_test_transform_dummy <- mean((test_ii$LC50 - test_ii$predicted_LC50)^2)
rmse_test_transform_dummy <- sqrt(mse_test_transform_dummy)
r2_test_transform_dummy <- 1 - (sum((test_ii$LC50 - test_ii$predicted_LC50)^2) / sum((test_ii$LC50 - mea
```

```r
cat(paste0(
  "Training Metrics:\n",
  "MSE (Train): ", mse_train_transform_dummy, "\n",
  "RMSE (Train): ", rmse_train_transform_dummy, "\n",
  "R-squared (Train): ", r2_train_transform_dummy, "\n\n",

  "Test Metrics:\n",
  "MSE (Test): ", mse_test_transform_dummy, "\n",
  "RMSE (Test): ", rmse_test_transform_dummy, "\n",
  "R-squared (Test): ", r2_test_transform_dummy, "\n"
))
```

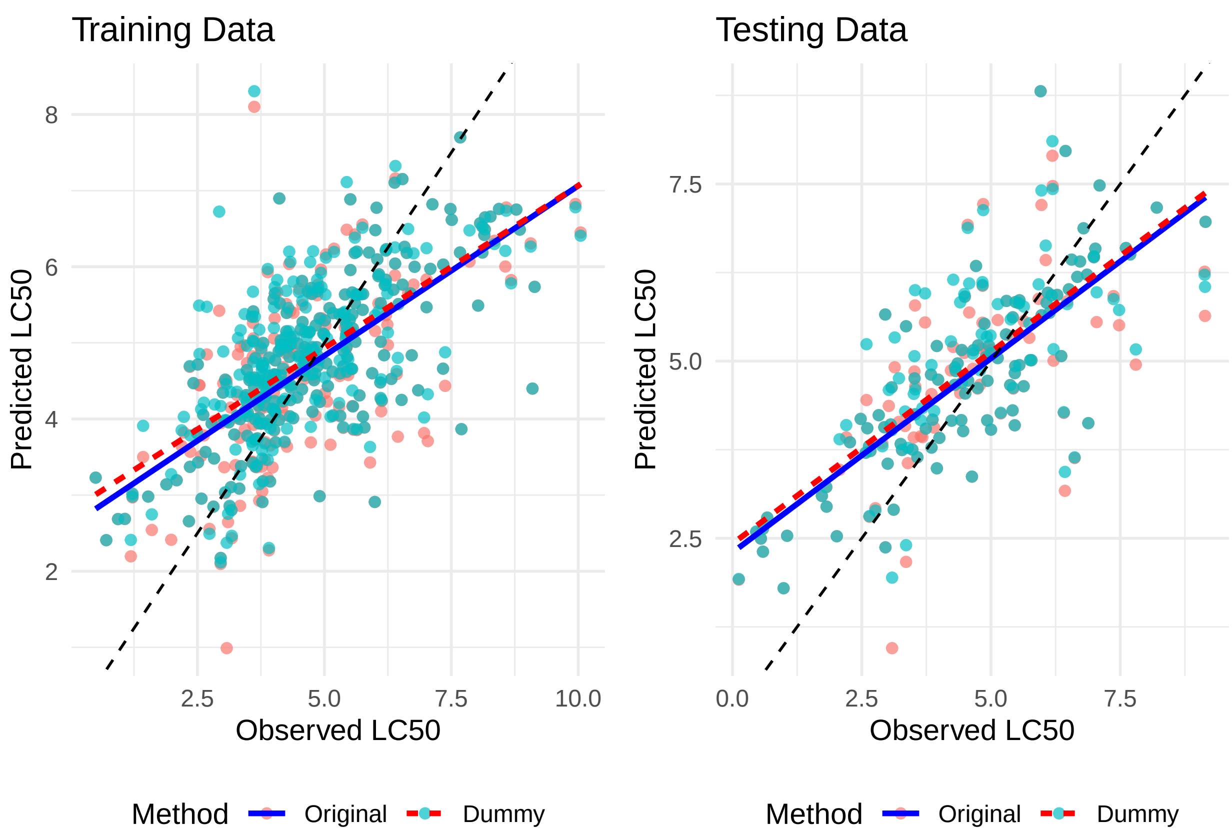
```
## Training Metrics:
## MSE (Train): 1.52494290800825
## RMSE (Train): 1.23488578743471
## R-squared (Train): 0.413140557873127
##
## Test Metrics:
## MSE (Test): 1.54950599294663
## RMSE (Test): 1.24479154598135
## R-squared (Test): 0.508509037636925
```

```r
# Combine data for plotting
train_ii$Type <- 'Train'
test_ii$Type <- 'Test'
combined_data <- rbind(train_ii, test_ii)

combined_data$Type <- factor(combined_data$Type, levels = c('Train', 'Test'))

# Plotting observed vs predicted LC50 values
ggplot(combined_data, aes(x = LC50, y = predicted_LC50, color = Type)) +
  geom_point(alpha = 0.7) +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed") +
  labs(title = "Dummy Encoding: Observed vs Predicted LC50", x = "Observed LC50", y = "Predicted LC50")
  theme_minimal() +
  facet_wrap(~Type) +
  theme(legend.position = "bottom")
```




```
# Prepare combined data
train_combined <- train_i[, c("LC50", "predicted_LC50")]
train_combined$Method <- 'Original'
train_combined$Type <- 'Train'
train_ii_combined <- train_ii[, c("LC50", "predicted_LC50")]
train_ii_combined$Method <- 'Dummy'
train_ii_combined$Type <- 'Train'
train_combined_all <- rbind(train_combined, train_ii_combined)

test_combined <- test_i[, c("LC50", "predicted_LC50")]
test_combined$Method <- 'Original'
test_combined$Type <- 'Test'
test_ii_combined <- test_ii[, c("LC50", "predicted_LC50")]
test_ii_combined$Method <- 'Dummy'
test_ii_combined$Type <- 'Test'
test_combined_all <- rbind(test_combined, test_ii_combined)

# Convert 'Method' and 'Type' to factors
train_combined_all$Method <- factor(train_combined_all$Method, levels = c('Original', 'Dummy'))
test_combined_all$Method <- factor(test_combined_all$Method, levels = c('Original', 'Dummy'))

# Function to draw regression lines
add_regression_lines <- function(df, original_model, dummy_model) {
  ggplot(df, aes(x = LC50, y = predicted_LC50, color = Method)) +
    geom_point(alpha = 0.7) +
    geom_smooth(method = "lm", formula = y ~ x, se = FALSE,
```

```r
                 aes(linetype = Method),
                 data = df[df$Method == 'Original', ],
                 color = 'blue') +
    geom_smooth(method = "lm", formula = y ~ x, se = FALSE,
                 aes(linetype = Method),
                 data = df[df$Method == 'Dummy', ],
                 color = 'red') +
    geom_abline(intercept = 0, slope = 1, linetype = "dashed") +
    labs(x = "Observed LC50", y = "Predicted LC50", title = df$Type[1]) +
    theme_minimal() +
    theme(legend.position = "bottom")
}

# Plot training data with both regression lines
train_plot <- add_regression_lines(train_combined_all, model, model_transform_dummy)
train_plot <- train_plot + labs(title = "Training Data")

# Plot testing data with both regression lines
test_plot <- add_regression_lines(test_combined_all, model, model_transform_dummy)
test_plot <- test_plot + labs(title = "Testing Data")

# Display plots side by side
grid.arrange(train_plot, test_plot, ncol = 2)
```



b.

```r
# Initialize vectors to store test errors
mse_test_errors_i <- numeric(200)
rmse_test_errors_i <- numeric(200)
r2_test_errors_i <- numeric(200)
mse_test_errors_ii <- numeric(200)
rmse_test_errors_ii <- numeric(200)
r2_test_errors_ii <- numeric(200)

# Repeat the procedure 200 times
set.seed(2)
for (i in 1:200) {
  # Split the data
  sample <- sample.split(data$LC50, SplitRatio = 0.7)
  train <- subset(data, sample == TRUE)
  test <- subset(data, sample == FALSE)

  # Option (i): Original model
  model <- lm(LC50 ~ ., data=train)
  pred_test_i <- predict(model, newdata=test)
  mse_test_i <- mean((test$LC50 - pred_test_i)^2)
  rmse_test_i <- sqrt(mse_test_i)
  r2_test_i <- 1 - (sum((test$LC50 - pred_test_i)^2) / sum((test$LC50 - mean(test$LC50))^2))

  # Option (ii): Dummy encoding
  train$H050 <- ifelse(train$H050 > 0, 1, 0)
  train$nN <- ifelse(train$nN > 0, 1, 0)
  train$C040 <- ifelse(train$C040 > 0, 1, 0)

  test$H050 <- ifelse(test$H050 > 0, 1, 0)
  test$nN <- ifelse(test$nN > 0, 1, 0)
  test$C040 <- ifelse(test$C040 > 0, 1, 0)

  model_ii <- lm(LC50 ~ ., data = train)
  pred_test_ii <- predict(model_ii, newdata = test)
  mse_test_ii <- mean((test$LC50 - pred_test_ii)^2)
  rmse_test_ii <- sqrt(mse_test_ii)
  r2_test_ii <- 1 - (sum((test$LC50 - pred_test_ii)^2) / sum((test$LC50 - mean(test$LC50))^2))

  # Record the test errors
  mse_test_errors_i[i] <- mse_test_i
  rmse_test_errors_i[i] <- rmse_test_i
  r2_test_errors_i[i] <- r2_test_i

  mse_test_errors_ii[i] <- mse_test_ii
  rmse_test_errors_ii[i] <- rmse_test_ii
  r2_test_errors_ii[i] <- r2_test_ii
}
```

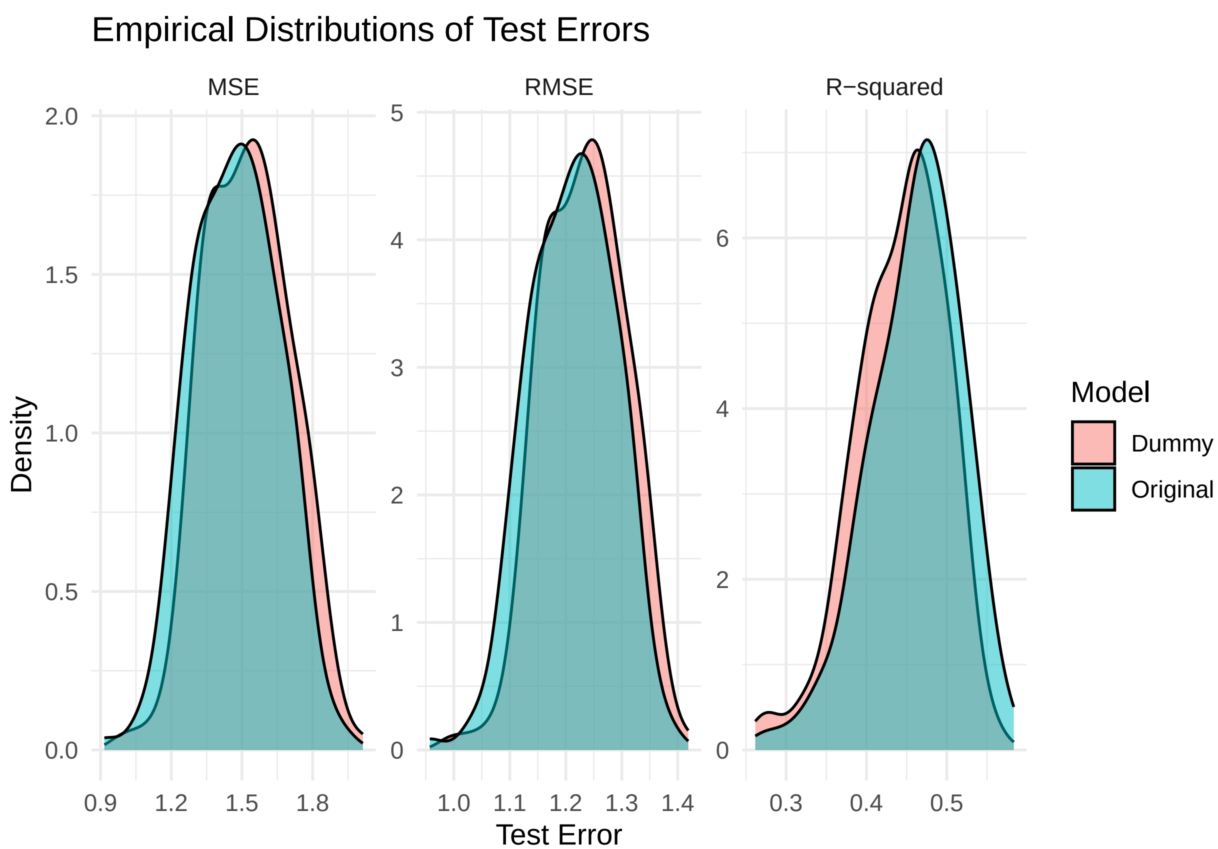
- Method 1: performs better in term of MSE
- Method 2: better in reduce overfitting

```r
# Calculate and print average test errors
average_test_error_i <- mean(mse_test_errors_i)
average_rmse_error_i <- mean(rmse_test_errors_i)
average_r2_error_i <- mean(r2_test_errors_i)

average_test_error_ii <- mean(mse_test_errors_ii)
average_rmse_error_ii <- mean(rmse_test_errors_ii)
average_r2_error_ii <- mean(r2_test_errors_ii)

cat(paste0(
  "Average Test Errors (Original Model):\n",
  "MSE: ", average_test_error_i, "\n",
  "RMSE: ", average_rmse_error_i, "\n",
  "R-squared: ", average_r2_error_i, "\n\n",

  "Average Test Errors (Dummy Model):\n",
  "MSE: ", average_test_error_ii, "\n",
  "RMSE: ", average_rmse_error_ii, "\n",
  "R-squared: ", average_r2_error_ii, "\n"
))
```

```
## Average Test Errors (Original Model):
## MSE: 1.47416671253053
## RMSE: 1.2118365144871
## R-squared: 0.461029936280147
##
## Average Test Errors (Dummy Model):
## MSE: 1.52473049238122
## RMSE: 1.23264425343633
## R-squared: 0.442463420670575
```

```r
# Create data frames for plotting
errors_df_mse <- data.frame(
  Error = c(mse_test_errors_i, mse_test_errors_ii),
  Metric = 'MSE',
  Model = factor(rep(c("Original", "Dummy"), each = 200))
)
errors_df_rmse <- data.frame(
  Error = c(rmse_test_errors_i, rmse_test_errors_ii),
  Metric = 'RMSE',
  Model = factor(rep(c("Original", "Dummy"), each = 200))
)
errors_df_r2 <- data.frame(
  Error = c(r2_test_errors_i, r2_test_errors_ii),
  Metric = 'R-squared',
  Model = factor(rep(c("Original", "Dummy"), each = 200))
)
errors_df <- rbind(errors_df_mse, errors_df_rmse, errors_df_r2)

# Ensure the 'Metric' factor has the correct level order
errors_df$Metric <- factor(errors_df$Metric, levels = c('MSE', 'RMSE', 'R-squared'))

# Plot the empirical distributions of the test errors
```

```r
ggplot(errors_df, aes(x = Error, fill = Model)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~ Metric, scales = "free") +
  labs(title = "Empirical Distributions of Test Errors", x = "Test Error", y = "Density") +
  theme_minimal()
```




```r
# Plot the empirical distributions of the test errors using boxplots
ggplot(errors_df, aes(x = Metric, y = Error, fill = Model)) +
  geom_boxplot(alpha = 0.7) +
  labs(title = "Boxplots of Test Errors", x = "Error Metric", y = "Error Value") +
  theme_minimal() +
  theme(legend.position = "top")
```


Boxplots of Test Errors
Model
Dummy
Original
2.0
1.5
1.0
0.5
Error Value
MSE
RMSE
R−squared
Error Metric