

Exercise 2 - Advanced Methods for Regression and Classification

12433732 - Stefan Merdian

2024-10-26

```
load("building.RData")

set.seed(1)

sample <- sample(c(TRUE, FALSE), nrow(df), replace=TRUE, prob=c(0.7,0.3))
train_x <- df[sample, !(names(df) %in% "y")]
test_x <- df[!sample, !(names(df) %in% "y")]
train_y <- df$y[sample ]
test_y <- df$y[!sample ]
train_data <- cbind(train_x, y = train_y)
```

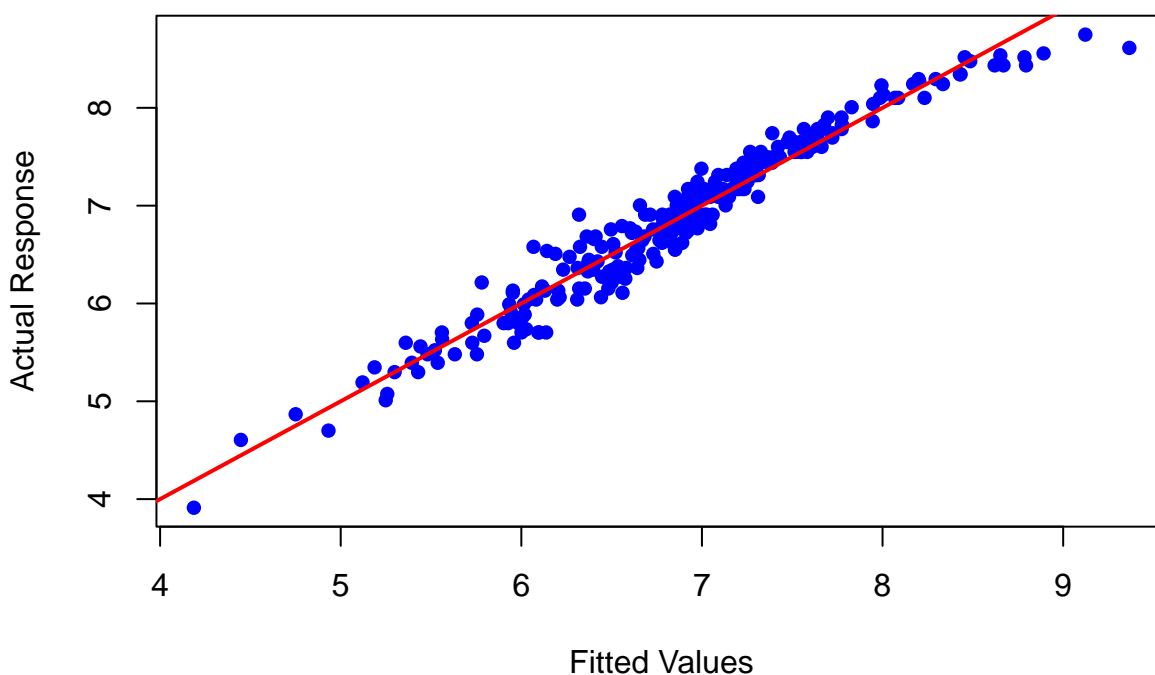
1) Compute the full model with lm()

```
full_model <- lm(y ~ ., data = train_data)

##a)

fitted_values <- predict(full_model, newdata = train_x)
plot(fitted_values, train_y,
     xlab = "Fitted Values",
     ylab = "Actual Response",
     main = "Fitted Values vs Response",
     pch = 16,
     col = "blue")
abline(0, 1, col = "red", lwd = 2)
```

Fitted Values vs Response



[illegible]

[illegible]

[illegible]


```
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
```

```
print(cv_results)
```

```
## 5-fold CV results:
##      CV
## 5.354344
```

```
head(cv_results)
```

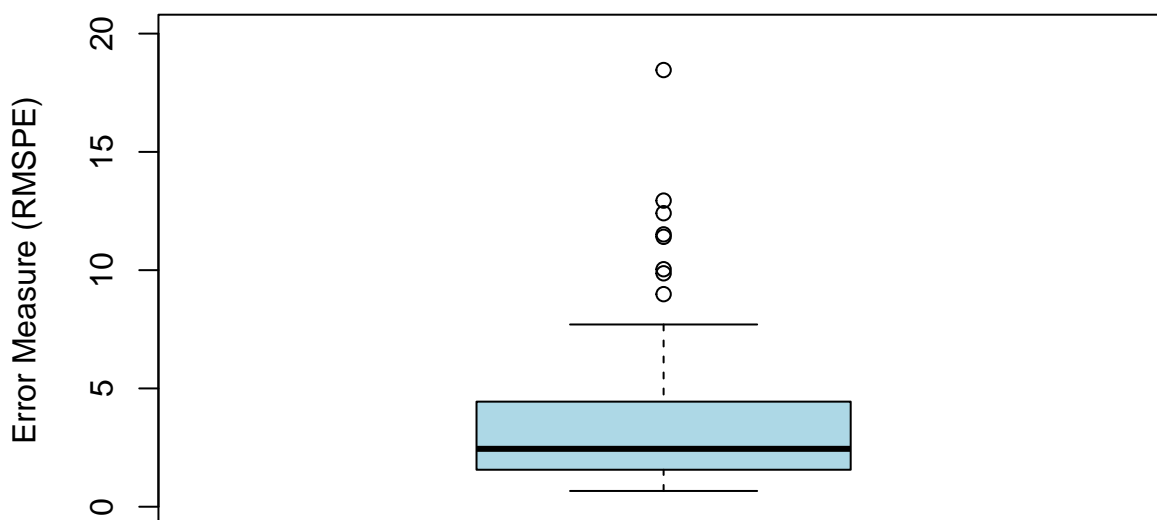
```
## $n
## [1] 266
##
## $K
## [1] 5
##
## $R
## [1] 100
##
## $cv
##      CV
## 5.354344
##
## $se
##      CV
## 16.82486
##
## $reps
##      CV
## [1,] 7.5704603
## [2,] 1.6045326
## [3,] 1.3417887
## [4,] 10.0390765
## [5,] 1.2457804
## [6,] 2.5088754
## [7,] 1.2142269
## [8,] 2.6510171
## [9,] 2.5757940
## [10,] 3.1213937
## [11,] 1.1602581
## [12,] 1.7430730
## [13,] 1.2090027
```

##	[14,]	3.9911132
##	[15,]	3.2634101
##	[16,]	1.2195155
##	[17,]	2.2505057
##	[18,]	12.9399172
##	[19,]	1.2550598
##	[20,]	2.7546252
##	[21,]	24.8649234
##	[22,]	3.9715800
##	[23,]	1.5339370
##	[24,]	1.9713203
##	[25,]	4.5606503
##	[26,]	11.4094603
##	[27,]	167.7031263
##	[28,]	1.2880653
##	[29,]	5.4672522
##	[30,]	2.3721369
##	[31,]	1.3352450
##	[32,]	4.3230004
##	[33,]	2.5712648
##	[34,]	1.7872145
##	[35,]	7.2677972
##	[36,]	6.1040144
##	[37,]	2.6485721
##	[38,]	2.0674127
##	[39,]	4.1489601
##	[40,]	1.4372441
##	[41,]	3.2607569
##	[42,]	1.3107157
##	[43,]	3.7606655
##	[44,]	1.7336455
##	[45,]	1.2865366
##	[46,]	1.6838006
##	[47,]	0.9435679
##	[48,]	4.6797280
##	[49,]	6.9101069
##	[50,]	9.8671988
##	[51,]	2.9824880
##	[52,]	8.9859484
##	[53,]	6.1326896
##	[54,]	2.2887412
##	[55,]	7.7032258
##	[56,]	1.3111808
##	[57,]	1.5469368
##	[58,]	2.5310655
##	[59,]	1.3974958
##	[60,]	1.7680144
##	[61,]	18.4600920
##	[62,]	5.9766021
##	[63,]	0.6961156
##	[64,]	0.6661405
##	[65,]	2.4386434
##	[66,]	2.0162137
##	[67,]	2.1805661
##	[68,]	0.8009422
##	[69,]	6.0036789
##	[70,]	1.0626394
##	[71,]	2.6876745
##	[72,]	2.6263966
##	[73,]	1.4818683
##	[74,]	1.4741641
##	[75,]	2.8549525
##	[76,]	1.6364202
##	[77,]	2.2014598

```
## [78,] 2.4559043
## [79,] 2.0441174
## [80,] 5.7975318
## [81,] 3.0172548
## [82,] 2.6953252
## [83,] 5.7132009
## [84,] 2.1329193
## [85,] 2.2989836
## [86,] 6.0700405
## [87,] 12.4080715
## [88,] 2.6368115
## [89,] 11.5136970
## [90,] 1.5775984
## [91,] 2.3059699
## [92,] 1.1765431
## [93,] 4.7642514
## [94,] 2.1687152
## [95,] 1.6176437
## [96,] 1.6448532
## [97,] 2.8362420
## [98,] 4.1067177
## [99,] 1.6417314
## [100,] 0.9685696
```

```
cv_errors <- cv_results$reps[, 1]
boxplot(cv_errors,
        ylim= c(0, 20),
        main = "Distribution of Cross-Validation Errors",
        ylab = "Error Measure (RMSPE)",
        col = "lightblue")
```

Distribution of Cross-Validation Errors



- The interquartile range is around 1 and 6.
- The median is around 1-2.
- The whisker extend from approximately 1 to 12.
- Outliers going from around 13- 18.

What do we conclude?:

- The model generally performs well, but there are some instances of significant error that may affect its reliability.
- To improve the model, it would be beneficial to investigate the outliers further. -> Whats the cause for that?

c)

What are we doing? - We will perform Cross-Validation again with the same data, but this time we will change the cost function to RTMSPE. RTMSPE provides an error value in percentage terms rather than in absolute values.

```
cv_resultsT <- cvFit(full_model,data = train_data, y = train_data$y, cost = rtmspe, K = 5, R = 100)
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

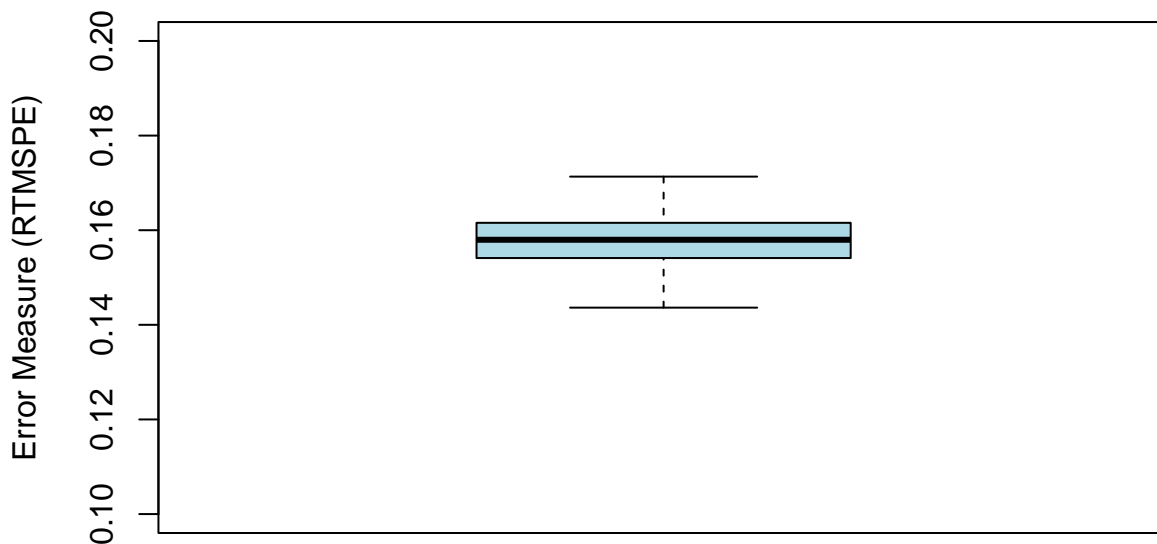
[illegible]

[illegible]


```
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
## Warning in predict.lm(...): prediction from rank-deficient fit; attr(*,
## "non-estim") has doubtful cases
```

```
cv_errorsT <- cv_resultsT$reps[, 1]
boxplot(cv_errorsT,
        ylim= c(0.1, 0.2),
        main = "Distribution of Cross-Validation Errors",
        ylab = "Error Measure (RTMSPE)",
        col = "lightblue")
```

Distribution of Cross-Validation Errors



- the RTMSPE values are significantly lower ranging between 14-17% - the space is more compact and has lower variability - The errors are tightly clustered around a median

→ the RTMSPE boxplot is tighter compared to the RMSPE, because of the root transformation → which reduces the effect of outliers and compresses the higher error values

d)

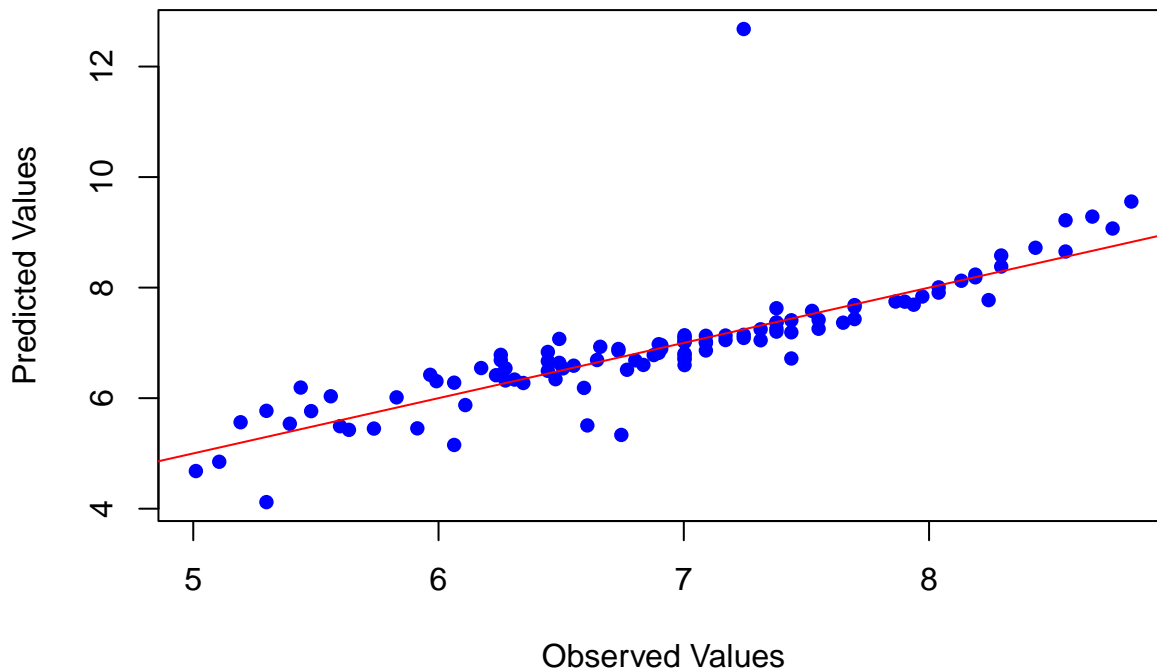
```
test_pred <- predict(full_model, newdata = test_x)
```

```
## Warning in predict.lm(full_model, newdata = test_x): prediction from
## rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
plot(test_y, test_pred,
     main = "Observed vs. Predicted (Test Data)",
     xlab = "Observed Values",
     ylab = "Predicted Values",
     col = "blue", pch = 16)

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

Observed vs. Predicted (Test Data)



```
rmse <- sqrt(mean((test_y - test_pred)^2))
cat('<\n> RMSE for the test data: ', rmse)
```

```
## <
## > RMSE for the test data: 0.6334959
```

- The model shows a good linear relationship.
- Predicts very well already.
- Has some outliers which probably lead also to the RMSE value around **0.63**

2) Best subset regression:

a)

First we remove all coef which don't get value.

(Possible Reason for na: - Some predictors had missing values from the beginning. - Some predictors might not be used much by the model, especially if they are less important.)

```
current_model <- full_model
model_coefs <- coef(full_model)
na_predictors <- names(model_coefs[is.na(model_coefs)])

if (length(na_predictors) > 0) {
  na_predictors_str <- paste(na_predictors, collapse = " - ")
  updated_formula <- as.formula(paste(". ~ . -", na_predictors_str))
  current_model <- update(full_model, updated_formula, data = train_data)

  message("Removed predictors with NA coefficients: ", na_predictors_str)
} else {
  current_model <- full_model
}
```

```
## Removed predictors with NA coefficients: PhysFin7 - Econ4.lag3 - Econ5.lag3 - Econ6.lag3 - Econ7.lag3 - Econ8.lag3
```

As the next step, we use the `drop1()` function repeatedly until the model is reduced to just 50 predictors. In each step, we drop the least significant predictor based on its contribution to reducing the model error, continuing this process until only 50 predictors remain

```
while (length(attr(terms(current_model), "term.labels")) > 50) {
  drop_result <- drop1(current_model, test = "F")
  term_to_drop <- rownames(drop_result)[which.max(drop_result$`Pr(>F)`)]
  updated_formula <- as.formula(paste(".", term_to_drop))
  current_model <- update(current_model, updated_formula)
}

final_model_50 <- current_model

summary(final_model_50)
```

```
##
## Call:
## lm(formula = y ~ START.YEAR + COMPLETION.YEAR + COMPLETION.QUARTER +
##     PhysFin1 + PhysFin2 + PhysFin3 + PhysFin4 + PhysFin5 + PhysFin6 +
##     PhysFin8 + Econ2 + Econ3 + Econ5 + Econ6 + Econ7 + Econ8 +
##     Econ9 + Econ13 + Econ15 + Econ17 + Econ18 + Econ19 + Econ1.lag1 +
##     Econ2.lag1 + Econ4.lag1 + Econ5.lag1 + Econ6.lag1 + Econ7.lag1 +
##     Econ9.lag1 + Econ10.lag1 + Econ13.lag1 + Econ15.lag1 + Econ16.lag1 +
##     Econ17.lag1 + Econ19.lag1 + Econ1.lag2 + Econ2.lag2 + Econ3.lag2 +
##     Econ5.lag2 + Econ6.lag2 + Econ7.lag2 + Econ8.lag2 + Econ11.lag2 +
##     Econ12.lag2 + Econ14.lag2 + Econ15.lag2 + Econ18.lag2 + Econ19.lag2 +
##     Econ2.lag3 + Econ3.lag3, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.7526 -0.1135  0.0269  0.1169  0.5316
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.199e+01  1.014e+01   3.156 0.001830 **
## START.YEAR      -4.351e-01  1.334e-01  -3.260 0.001293 **
## COMPLETION.YEAR  9.379e-02  2.668e-02   3.516 0.000535 ***
## COMPLETION.QUARTER 4.247e-02  1.409e-02   3.014 0.002887 **
## PhysFin1       -2.799e-02  3.672e-03  -7.621 7.93e-13 ***
## PhysFin2        9.604e-05  4.097e-05   2.344 0.020000 *
## PhysFin3       -2.906e-04  1.173e-04  -2.478 0.013996 *
## PhysFin4       -1.502e-04  9.477e-05  -1.585 0.114408
## PhysFin5       -2.507e-03  5.652e-04  -4.436 1.46e-05 ***
## PhysFin6        5.508e-04  8.995e-05   6.124 4.29e-09 ***
## PhysFin8        5.257e-04  3.103e-05  16.943 < 2e-16 ***
## Econ2          1.725e-01  5.736e-02   3.008 0.002944 **
## Econ3          2.084e-01  3.457e-02   6.028 7.15e-09 ***
## Econ5         -1.255e-05  4.566e-06  -2.749 0.006479 **
## Econ6         -3.125e-04  6.910e-05  -4.522 1.01e-05 ***
## Econ7         -8.626e-02  1.750e-02  -4.929 1.65e-06 ***
## Econ8          6.716e-03  1.337e-03   5.023 1.07e-06 ***
## Econ9         -4.851e-05  1.159e-05  -4.187 4.13e-05 ***
## Econ13        -1.030e-04  2.861e-05  -3.601 0.000394 ***
## Econ15         1.106e-01  3.085e-02   3.584 0.000419 ***
## Econ17         2.921e-04  6.444e-05   4.533 9.67e-06 ***
## Econ18         2.155e-05  5.538e-06   3.892 0.000133 ***
## Econ19        -2.414e-06  9.669e-07  -2.497 0.013283 *
## Econ1.lag1     -2.016e-04  6.110e-05  -3.299 0.001135 **
## Econ2.lag1     -3.944e-01  8.987e-02  -4.389 1.79e-05 ***
## Econ4.lag1      2.389e-01  7.796e-02   3.065 0.002457 **
## Econ5.lag1      2.276e-05  5.056e-06   4.501 1.11e-05 ***
## Econ6.lag1      3.386e-04  7.746e-05   4.370 1.93e-05 ***
## Econ7.lag1     -7.387e-02  1.309e-02  -5.643 5.25e-08 ***
```

```
## Econ9.lag1      -3.757e-05  9.929e-06  -3.784  0.000200 ***
## Econ10.lag1     -2.167e-01  7.056e-02  -3.072  0.002402 **
## Econ13.lag1     -1.554e-04  3.940e-05  -3.945  0.000108 ***
## Econ15.lag1      3.132e-01  5.909e-02   5.301  2.84e-07 ***
## Econ16.lag1     -2.584e-02  2.494e-02  -1.036  0.301505
## Econ17.lag1     -3.784e-04  6.955e-05  -5.441  1.44e-07 ***
## Econ19.lag1     -5.391e-06  9.905e-07  -5.443  1.43e-07 ***
## Econ1.lag2      -1.762e-04  4.707e-05  -3.744  0.000233 ***
## Econ2.lag2       1.845e-01  4.773e-02   3.866  0.000147 ***
## Econ3.lag2      -1.096e-01  2.188e-02  -5.007  1.15e-06 ***
## Econ5.lag2       7.863e-06  4.965e-06   1.584  0.114733
## Econ6.lag2      2.898e-04  8.300e-05   3.491  0.000584 ***
## Econ7.lag2       4.043e-02  1.072e-02   3.772  0.000209 ***
## Econ8.lag2      -1.095e-02  1.981e-03  -5.528  9.33e-08 ***
## Econ11.lag2     -4.356e-04  2.716e-04  -1.604  0.110223
## Econ12.lag2      7.963e-04  3.982e-04   2.000  0.046770 *
## Econ14.lag2     -1.461e-04  6.132e-05  -2.382  0.018093 *
## Econ15.lag2     -3.239e-01  7.379e-02  -4.389  1.79e-05 ***
## Econ18.lag2      5.150e-05  9.113e-06   5.652  5.02e-08 ***
## Econ19.lag2      1.834e-06  8.422e-07   2.177  0.030549 *
## Econ2.lag3      -1.540e-01  3.238e-02  -4.756  3.62e-06 ***
## Econ3.lag3       1.805e-01  3.573e-02   5.052  9.32e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2079 on 215 degrees of freedom
## Multiple R-squared:  0.9495, Adjusted R-squared:  0.9378
## F-statistic: 80.92 on 50 and 215 DF,  p-value: < 2.2e-16
```

We save the remaining predictors and build a new `reduced_model` with them.

```
selected_predictors <- attr(terms(final_model_50), "term.labels")

train_data_reduced <- train_data[, c(selected_predictors, "y"), drop = FALSE]
```

Now we identify the best subset of predictors from the `reduced_dataset`. We limiting the number of predictors to **10** (`nvmax = 10`).

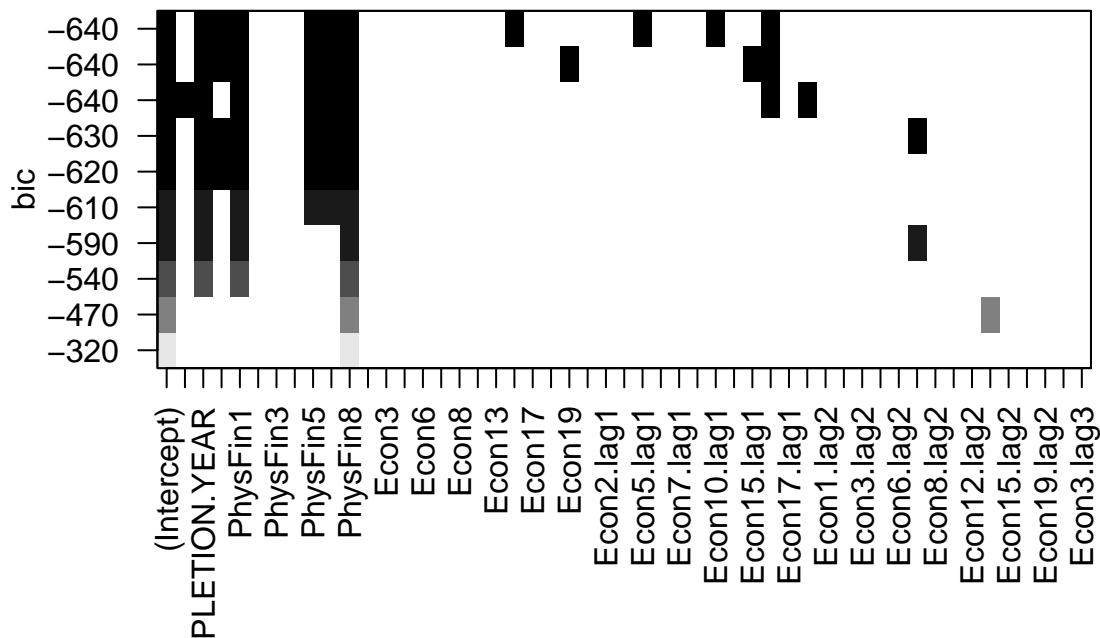
```
library(leaps)

best_subset <- regsubsets(y ~ ., data = train_data_reduced, nvmax = 10, really.big = TRUE)

best_subset_summary <- summary(best_subset)
```

b)

```
plot(best_subset)
```

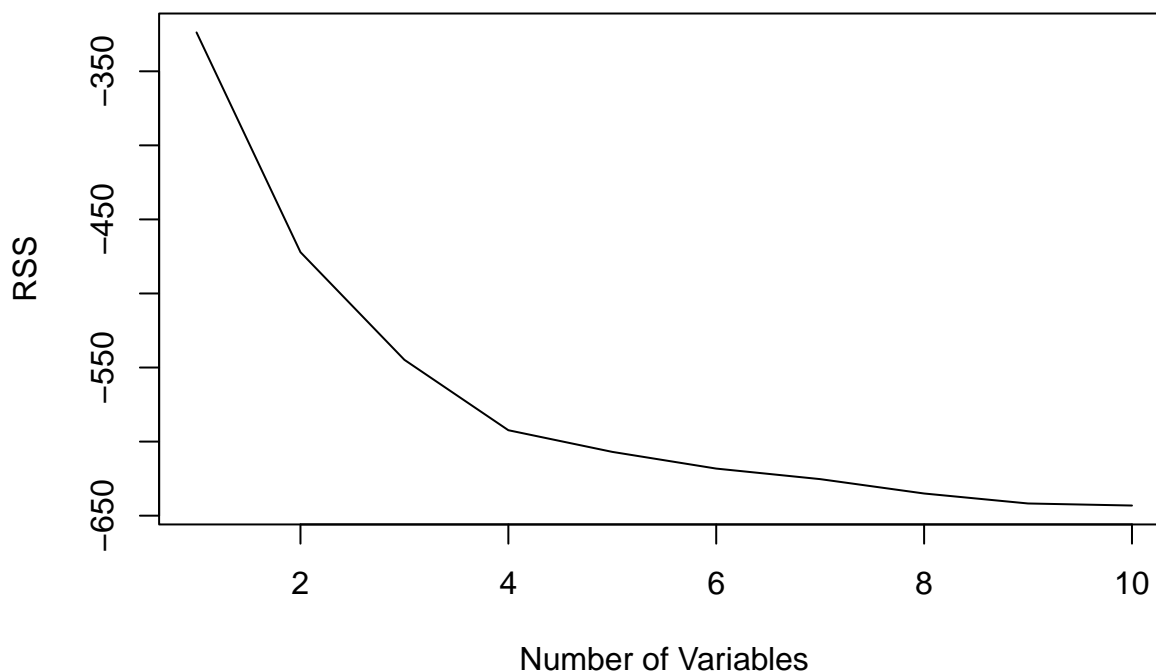


We can see that the best model, which has the lowest BIC value, is represented in the first row of the plot. All black-colored squares indicate the predictors that are included in this model.

```
names(best_subset_summary)
```

```
## [1] "which" "rsq" "rss" "adjr2" "cp" "bic" "outmat" "obj"
```

```
plot(best_subset_summary$bic, xlab = "Number of Variables", ylab = "RSS", type = "l")
```



```
coef(best_subset, 10)
```

```
##      (Intercept)      COMPLETION.YEAR COMPLETION.QUARTER      PhysFin1
##      -8.477957e+00      1.674334e-01      5.383538e-02      -3.183562e-02
##      PhysFin5      PhysFin6      PhysFin8      Econ15
##      -2.711553e-03      4.802016e-04      5.028358e-04      2.583284e-02
##      Econ5.lag1      Econ10.lag1      Econ16.lag1
##      7.606901e-07      8.484362e-02      -3.167488e-02
```

c)

- Extracts the coefficients of the best model with 10 predictors.

- Identifies the names of the selected predictors, excluding the intercept.
- Creates a formula for the model using these selected predictors.
- Reduces both the training and testing datasets

```
best_coef <- coef(best_subset, 10)
selected_predictors <- names(best_coef)[-1]
final_formula <- as.formula(paste("y ~", paste(selected_predictors, collapse = " + ")))
train_data_reduced <- train_data[, c(selected_predictors, "y"), drop = FALSE]
test_data_reduced <- test_x[, c(selected_predictors), drop = FALSE]
```

Train model and perform CrossValidation again.

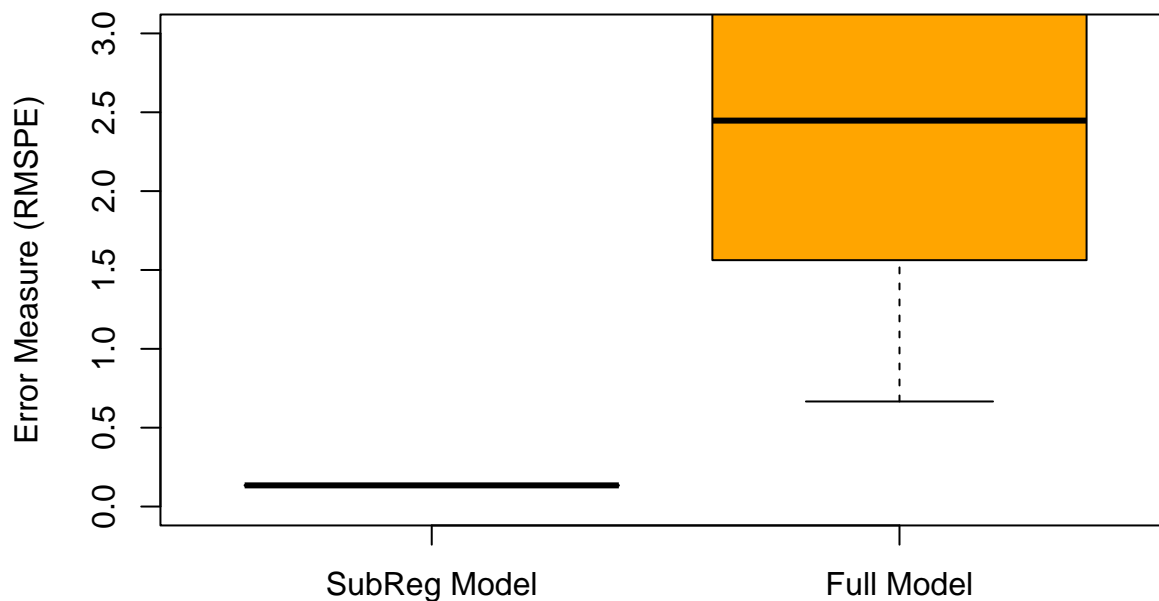
```
reg_model <- lm(y ~ ., data = train_data_reduced)
cv_resultsRegModelRmspe <- cvFit(reg_model, data = train_data_reduced, y = train_data$y, cost = rtmspe, K = 5,
cv_resultsRegModelRtmspe <- cvFit(reg_model, data = train_data_reduced, y = train_data$y, cost = rtmspe, K = 5,
```

Plotting results.

```
cv_errors_subReg <- cv_resultsRegModelRmspe$reps[, "CV"]
boxplot(cv_errors_subReg, cv_errors, ylim= c(0, 3),
        main = "Distribution of Cross-Validation Errors",
        ylab = "Error Measure (RMSPE)", names = c("SubReg Model", "Full Model"), col = c("skyblue", "orange"))
```

```
## Warning in (function (z, notch = FALSE, width = NULL, varwidth = FALSE, :
## Duplicated argument main = "RMSPE" is disregarded
```

Distribution of Cross-Validation Errors

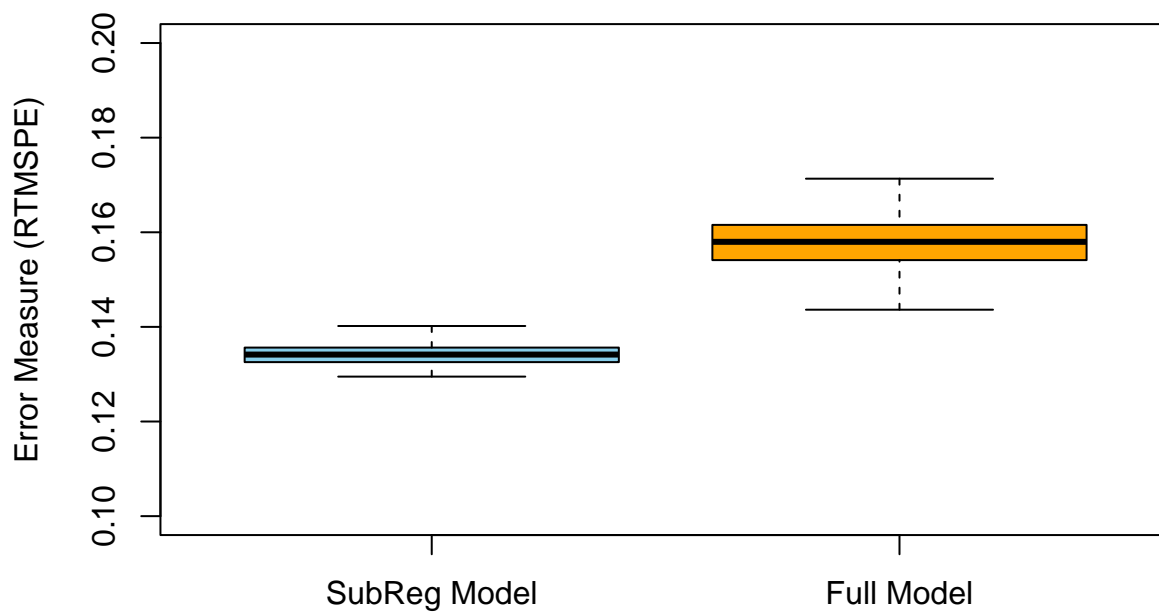


- The SubReg Model has significantly lower cross-validation errors compared to the Full Model and much more compact error spread.

```
cv_errors_subRegT <- cv_resultsRegModelRmspe$reps[, "CV"]
boxplot(cv_errors_subRegT, cv_errorsT, ylim= c(0.1, 0.2),
        main = "Distribution of Cross-Validation Errors",
        ylab = "Error Measure (RTMSPE)", names = c("SubReg Model", "Full Model"), col = c("skyblue", "orange"))
```

```
## Warning in (function (z, notch = FALSE, width = NULL, varwidth = FALSE, :
## Duplicated argument main = "RMSPE" is disregarded
```


Distribution of Cross-Validation Errors



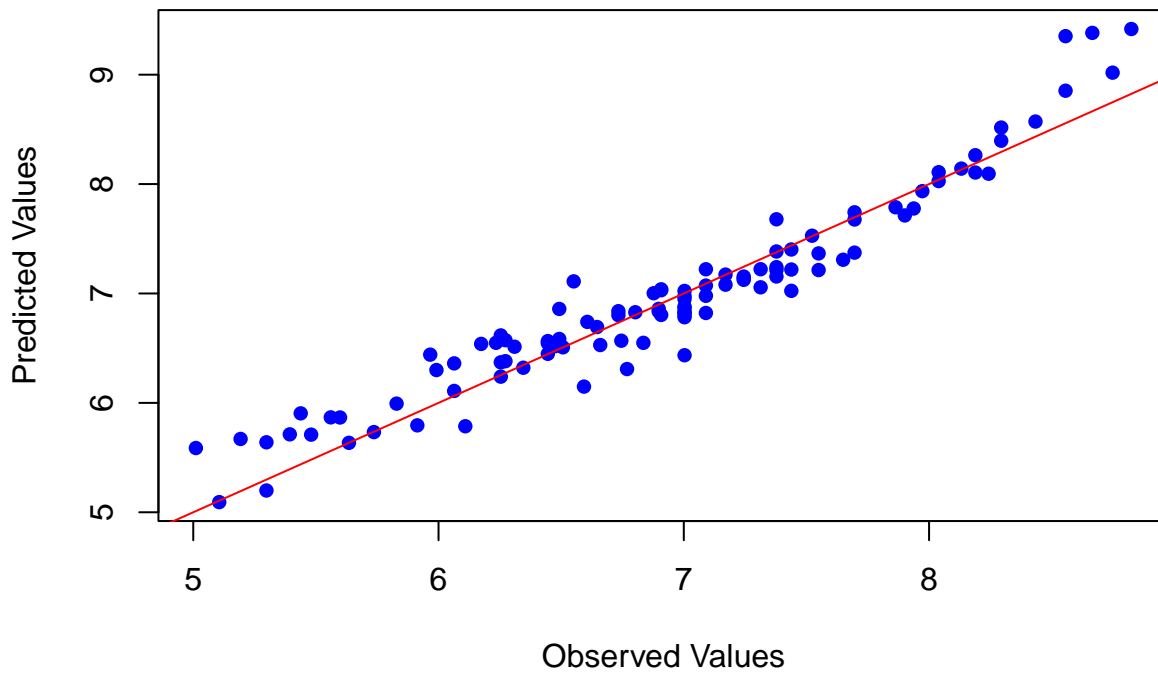
d)

```
test_predSubreg <- predict(reg_model, newdata = test_data_reduced)

plot(
  test_y,
  test_predSubreg,
  main = "Observed vs. Predicted (Test Data)",
  xlab = "Observed Values",
  ylab = "Predicted Values",
  col = "blue",
  pch = 16
)

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

Observed vs. Predicted (Test Data)



```
rmseSunReg <- sqrt(mean((test_y - test_predSubreg)^2))  
cat("\nThe RMSE value for the subReg model is:", rmseSunReg)
```

```
##  
## The RMSE value for the subReg model is: 0.2527903
```

```
cat("\nThe RMSE value for the model with all predictors was:", rmse)
```

```
##  
## The RMSE value for the model with all predictors was: 0.6334959
```