STT 380

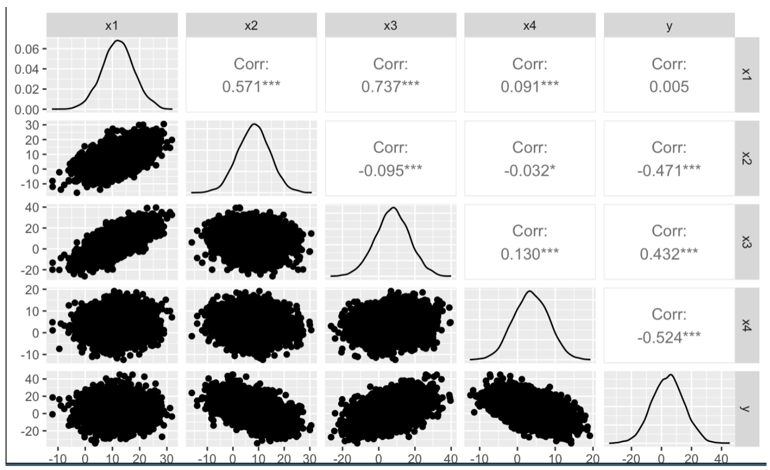
In-Class Assignment 23

1. For the sampregdata dataset,

   a. Create a plot with ggpairs (no color needed). Evaluate potential multicollinearity issues with multiple x's.

      i. library(GGally)

      ii. ggpairs(sampregdata)



   b. Split the data into training and test datasets, using a 60/40 split.

      i. split_pct <- 0.6

      ii. n <- length(sampregdata$y)*split_pct # train size

      iii. row_samp <- sample(1:length(sampregdata$y), n, replace = FALSE)

      iv. train <- sampregdata[row_samp,]

      v. test <- sampregdata[-row_samp,]

      vi. length(train$y) = 3236

      vii. length(test$y) = 2158

   c. Build a linear regression model for price on the training dataset, based on the variable with the strongest correlation.

      i. sampregdata_mod <- lm(data = train, y ~ x4)

      ii. sampregdata_mod

   d. Build the prediction for the test dataset. How does the test RMSE compare to that for the training model?

      i. test_pred <- predict(sampregdata_mod,test)

ii. rmse_train <- sqrt(mean(sampregdata_mod$residuals^2))

iii. summary(test_pred)

iv. Min. 1st Qu. Median Mean 3rd Qu. Max.

v. -25.315 -1.433 4.610 4.562 10.885 40.197

vi. summary(rmse_train)

vii. Min. 1st Qu. Median Mean 3rd Qu. Max.

viii. 5.353 5.353 5.353 5.353 5.353 5.353

ix. So the train is a constant and the test is a range of 60 ish values.

2. Next, create a model with all four x's, again using the same 60/40 split. Evaluate the model in terms of the fit of parameters and the errors. Is the x with the strongest fit the same as the one with the highest correlation?

   a. split_pct <- 0.6
   b. n <- length(sampregdata$y)*split_pct # train size
   c. row_samp <- sample(1:length(sampregdata$y), n, replace = FALSE)
   d. train <- sampregdata[row_samp,]
   e. test <- sampregdata[-row_samp,]
   f. sampregdata_mod <- lm(data = train, y ~ x1 + x2 + x3 + x4)
   g. test_pred <- predict(sampregdata_mod,test)
   h. test_error <- test$y - test_pred
   i. rmse_train <- sqrt(mean(sampregdata_mod$residuals^2))
   j. rmse_test <- sqrt(mean(test_error^2))
   k. rmse_train
   l. rmse_test
   m. [1] 5.352711
   n. [1] 5.376122
   o.