Midterm

AUTHOR
Collin Real (yhi267)

Section I - True (T) or False (F) (20 Points)

1. **False:** Principal Component Regression is unsupervised learning.
2. **False:** Collinearity increases the errors associated with the coefficients, causing them to worsen.
3. **False:** R-squared of 0 means the model explains none of the variation in response Y, not -1.
4. **False:** If the number of predictors is > the number of observations, unique least squares coefficient estimate is eliminated.
5. **True**
6. **True**
7. **True**
8. **True**
9. **False:** OLS cannot be used.
10. **False:** Last name is Wang, first name is Min.

Section II - Free Response Questions (25 Points)

Problem 1 (Total: 15 Points)

Explain whether each scenario is a classification or regression problem, and indicate whether we are most interested in inference or prediction. Finally, provide n and p.

1a) We collect a set of data on the top 500 firms in the US. For each firm we record profit, number of employees, industry, and the CEO salary. We are interested in understanding which factors affect CEO salary. (5 Points)

Since CEO salary is a continuous variable, this is a **regression** problem. The focus is on **inference** because we want to understand the relationship between CEO salary and the other variables. The number of observations (**n**) is 500 (i.e., the top 500 firms) and the number of features (**p**) is 3 (i.e., profit, number of employees, and industry).

1b) We are considering launching a new product and wish to know whether it will be a success or a failure. We collect data on 20 similar products that were previously launched. For each product we have recorded whether it was a success or failure, price charged for the product, marketing budget, competition price, and ten other variables. (5 Points)

Since the outcome is a categorical variable, this is a **classification** problem. The focus is on **prediction** because we want to predict the outcome of a new product launch. The number of observations (**n**) is 20 (i.e., 20 products) and the number of features (**p**) is 13.

1c) We are interested in predicting the % change in the USD/Euro exchange rate in relation to the weekly changes in the world stock markets. Hence we collect weekly data for all of

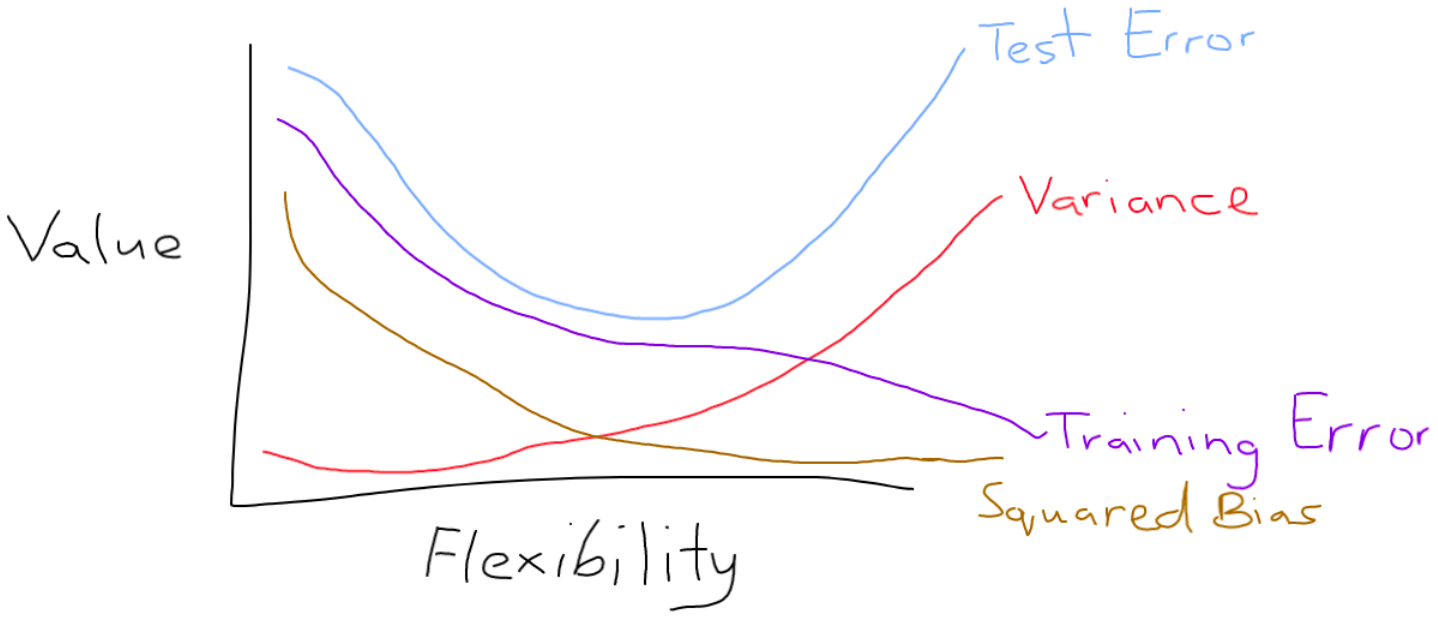
2012. For each week we record the % change in the USD/Euro, the % change in the US market, the % change in the British market, and the % change in the German market. (5 Points)

Since the percentage change in the USD/Euro exchange rate is a continuous variable, this is a **regression** problem. The focus is on **prediction** because we want to predict future changes in the rate. The number of observations (**n**) is 52 (i.e., 52 weeks of data) and the number of features (**p**) is 3.

Problem 2 (Total: 10 Points)

In this class, we discussed the bias-variance trade-off. Answer the following questions.

2a) Provide a sketch of typical (squared) bias, variance, training error, test error curves, on a single plot, as we go from less flexible statistical learning methods towards more flexible approaches. The x-axis should represent the amount of flexibility in the method, and the y-axis should represent the values for each curve. There should be four curves. Make sure to label each one.



Statistical learning methods flexibility sketch

2b) Briefly explain why each of the four curves has the shape displayed in part (a).

Squared bias: Since more complex models can fit the training data better, this curve begins high and decreases as complexity increases. High squared bias means the model is too simple and underfitting the data, not identifying the underlying patterns.
Variance: Measures the difference between the model's predictions and actual values. Variance increases as the model becomes more complex due to random noise captured from the data, causing underfitting.
Training Error: Complex models fit the training data extremely well leading to lower training errors. As the model becomes simpler, the training error increases because simpler models cannot capture all of the data's complexity.
Test Error: The curve is U-shaped: starting high for simple models resulting from high bias, decreasing to a minimum, then increasing for models with high complexity due to higher variance.

Section III - Coding Questions (55 Points)

Problem 3 (Total: 18 Points - 3 points each)

This exercise involves the Auto data set studied in the lab. Make sure that the missing values have been removed from the data. We may start R and use these commands to load the data:

```
library(ISLR)
library(dplyr)
library(rlist)
library(ggplot2)
library(corrplot)
library(tidyr)
library(caret)
data('Auto')
str(Auto)
```

```
'data.frame':   392 obs. of  9 variables:
 $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
 $ cylinders   : num  8 8 8 8 8 8 8 8 8 8 ...
 $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
 $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
 $ weight      : num  3504 3693 3436 3433 3449 ...
 $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
 $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
 $ origin      : num  1 1 1 1 1 1 1 1 1 1 ...
 $ name        : Factor w/ 304 levels "amc ambassador brougham",..: 49 36 231 14 161 141 54
223 241 2 ...
```

```
# Convert numeric to factor: cylinders & origin
Auto$cylinders <- as.factor(Auto$cylinders)
Auto$origin <- as.factor(Auto$origin)

# Verify updated data types
str(Auto)
```

```
'data.frame':   392 obs. of  9 variables:
 $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
 $ cylinders   : Factor w/ 5 levels "3","4","5","6",..: 5 5 5 5 5 5 5 5 5 5 ...
 $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
 $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
 $ weight      : num  3504 3693 3436 3433 3449 ...
 $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
 $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
 $ origin      : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
 $ name        : Factor w/ 304 levels "amc ambassador brougham",..: 49 36 231 14 161 141 54
223 241 2 ...
```

```
### Check for missing values
Auto_missing_values <- sum(
    is.na(
        Auto
    )
```

```
)

print(
    paste0(
        "Missing values: ",
        Auto_missing_values
    )
)
```

```
[1] "Missing values: 0"
```

3a) Which of the predictors are quantitative, and which are qualitative?

```
# Quantitative predictors
quant_preds <- colnames(
    select_if(
        Auto,
        is.numeric
    )
)

# Qualitative predictors
qual_preds <- colnames(
    select_if(
        Auto,
        is.factor
    )
)
```

```
cat(
    "Quantitative predictor(s): ",
    paste0(
        quant_preds
    ),
    sep = "\n"
)
```

```
Quantitative predictor(s):
mpg
displacement
horsepower
weight
acceleration
year
```

```
cat(
    'Qualititative predictor(s): ',
    paste0(
        qual_preds
    ),
    sep = "\n"
)
```

```
Qualititative predictor(s):
cylinders
origin
name
```

```
# Create key values for dictionary for pretty printing
quant_preds_list <- list(
    colnames(
        select_if(
            Auto,
            is.numeric
        )
    )
)
quant_preds_keys <- c()
colon <- ": "
for (pred in quant_preds_list) {
    new_pred = paste(
        pred,
        colon,
        sep = ""
    )
    quant_preds_keys = append(
        quant_preds_keys,
        new_pred
    )
}
names(quant_preds) <- cat(quant_preds_keys)
```

```
mpg:  displacement:  horsepower:  weight:  acceleration:  year:
```

3b) What is the range of each quantitative predictor? You can answer this using the range() function.

```
quant_preds_range <- sapply(
    Auto[, quant_preds],
    range
)
cat(
    'Quantitative predictor ranges:',
    paste0(
        quant_preds_keys,
        quant_preds_range[1,],
        " - ",
        quant_preds_range[2,]
    ),
    sep = "\n"
)
```

```
Quantitative predictor ranges:
mpg: 9 - 46.6
displacement: 68 - 455
horsepower: 46 - 230
```

```
weight: 1613 - 5140
acceleration: 8 - 24.8
year: 70 - 82
```

3c) What is the mean and standard deviation of each quantitative predictor?

```
quant_preds_mean <- round(
    sapply(
        Auto[, quant_preds],
        mean
    ),
    2
)
quant_preds_sd <- round(
    sapply(
        Auto[, quant_preds],
        sd
    ),
    2
)
cat(
    'Quantitative predictor means:',
    paste0(
        quant_preds_keys,
        quant_preds_mean
    ),
    sep = "\n"
)
```

```
Quantitative predictor means:
mpg: 23.45
displacement: 194.41
horsepower: 104.47
weight: 2977.58
acceleration: 15.54
year: 75.98
```

```
cat(
    'Quantitative predictor standard deviations:',
    paste0(
        quant_preds_keys,
        quant_preds_sd
    ),
    sep = "\n"
)
```

```
Quantitative predictor standard deviations:
mpg: 7.81
displacement: 104.64
horsepower: 38.49
weight: 849.4
acceleration: 2.76
year: 3.68
```

3d) Now remove the 20th through 80th observations. What is the range, mean, and standard deviation of each predictor in the subset of the data that remains?

```
# Create subset dataset, removing 20th through 80th observations
subset_Auto <- Auto[-c(20:80), ]

# Calucate new range
range_Auto_subset <- sapply(
    subset_Auto[, quant_preds],
    range
)

# Calucate new mean
mean_Auto_subset <- round(
    sapply(
    subset_Auto[, quant_preds],
    mean
    ),
    2
)

# Calucate new standard deviation
sd_Auto_subset <- round(
    sapply(
    subset_Auto[, quant_preds],
    sd
    ),
    2
)
```

```
cat(
    'Quantitative predictor subset ranges:',
    paste0(
        quant_preds_keys,
        range_Auto_subset[1,],
        " - ",
        range_Auto_subset[2,]
    ),
    sep = "\n"
)
```

```
Quantitative predictor subset ranges:
mpg: 11 - 46.6
displacement: 68 - 455
horsepower: 46 - 230
weight: 1649 - 4997
acceleration: 8 - 24.8
year: 70 - 82
```

```
cat(
    'Quantitative predictor subset means:',
    paste0(
        quant_preds_keys,
```

```
        mean_Auto_subset
    ),
    sep = "\n"
)
```

```
Quantitative predictor subset means:
mpg: 24.22
displacement: 189.44
horsepower: 101.86
weight: 2935.02
acceleration: 15.61
year: 76.85
```

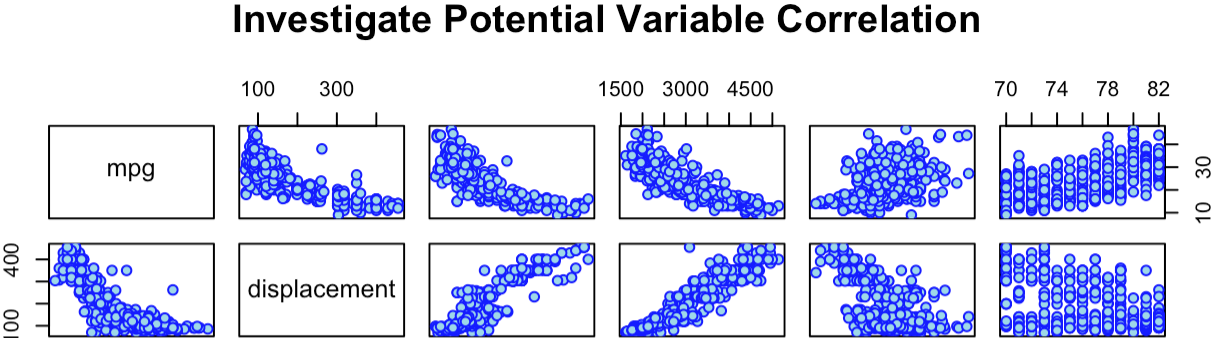
```
cat(
    'Quantitative predictor subset standard deviations:',
    paste0(
        quant_preds_keys,
        sd_Auto_subset
    ),
    sep = "\n"
)
```

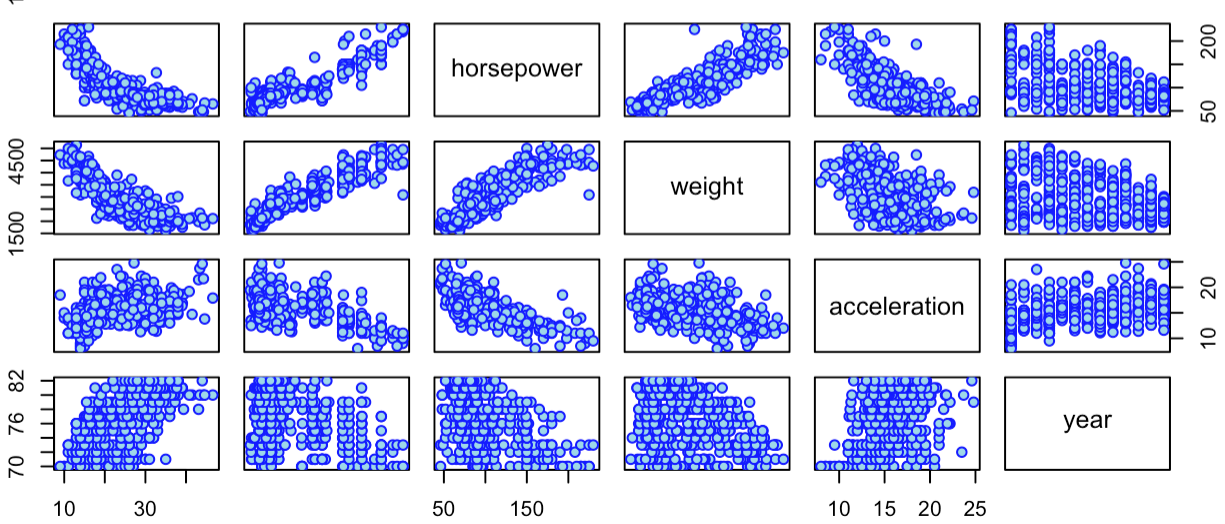
```
Quantitative predictor subset standard deviations:
mpg: 7.82
displacement: 101.76
horsepower: 36.6
weight: 805.48
acceleration: 2.76
year: 3.32
```

3e) Using the full data set, investigate the predictors graphically, using scatterplots or other tools of your choice. Create some plots highlighting the relationships among the predictors. Comment on your findings.

Scatterplot Correlation

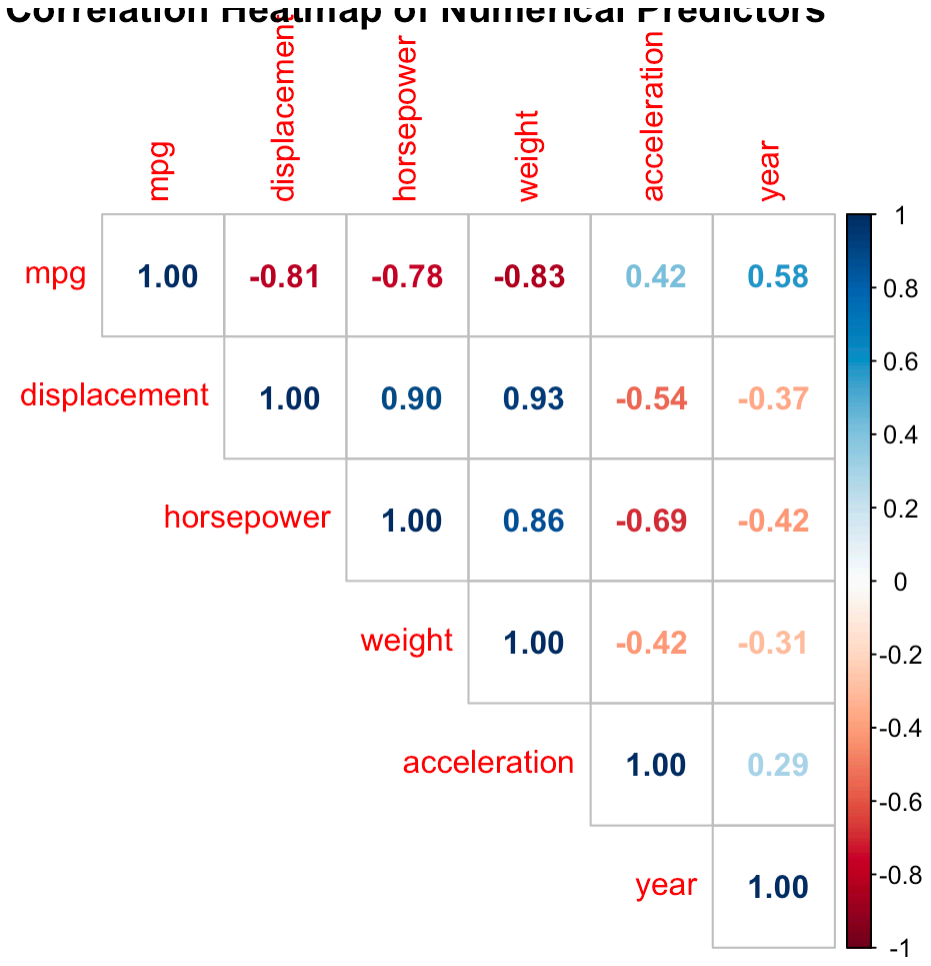
```
plot(~mpg+displacement+horsepower+weight+acceleration+year,
    data=Auto,
    main="Investigate Potential Variable Correlation",
    col='blue',
    bg='lightblue',
    pch = 21)
```





Heatmap Correlation

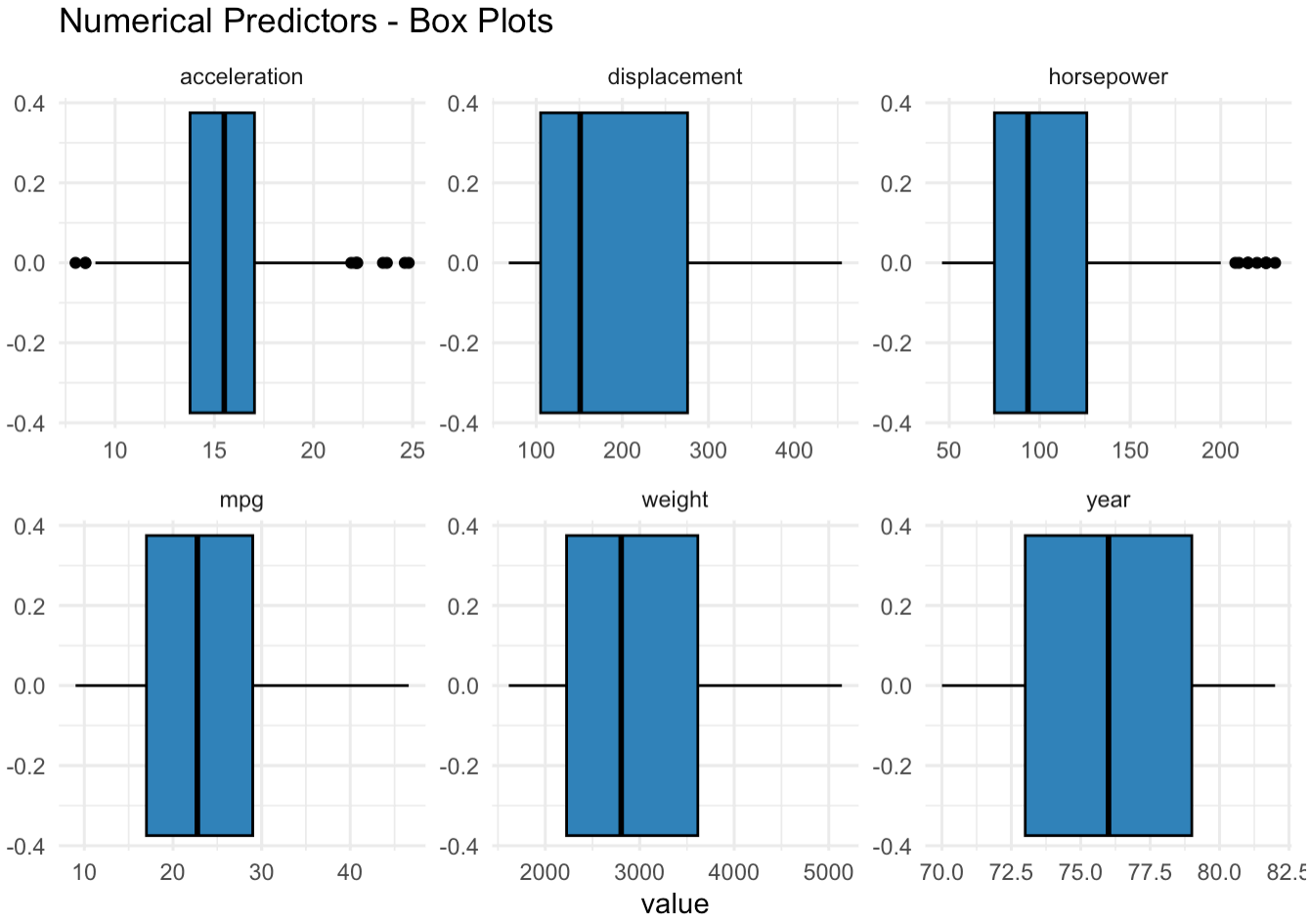
```
Auto %>%
  select_if(is.numeric) %>%
  cor() %>%
  corrplot(
    method = "number",
    title = 'Correlation Heatmap of Numerical Predictors',
    type = 'upper')
```



Box Plots

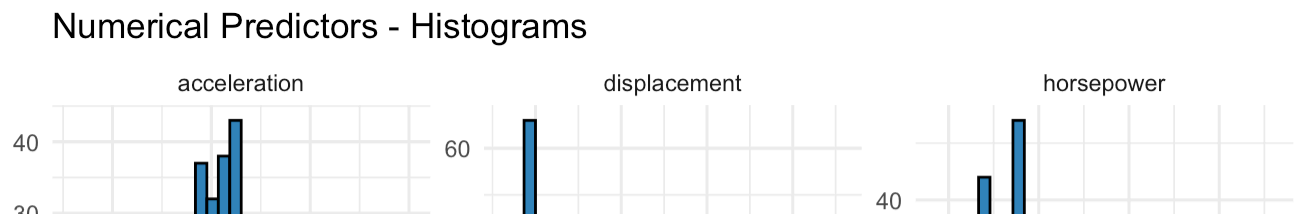
Box Plots

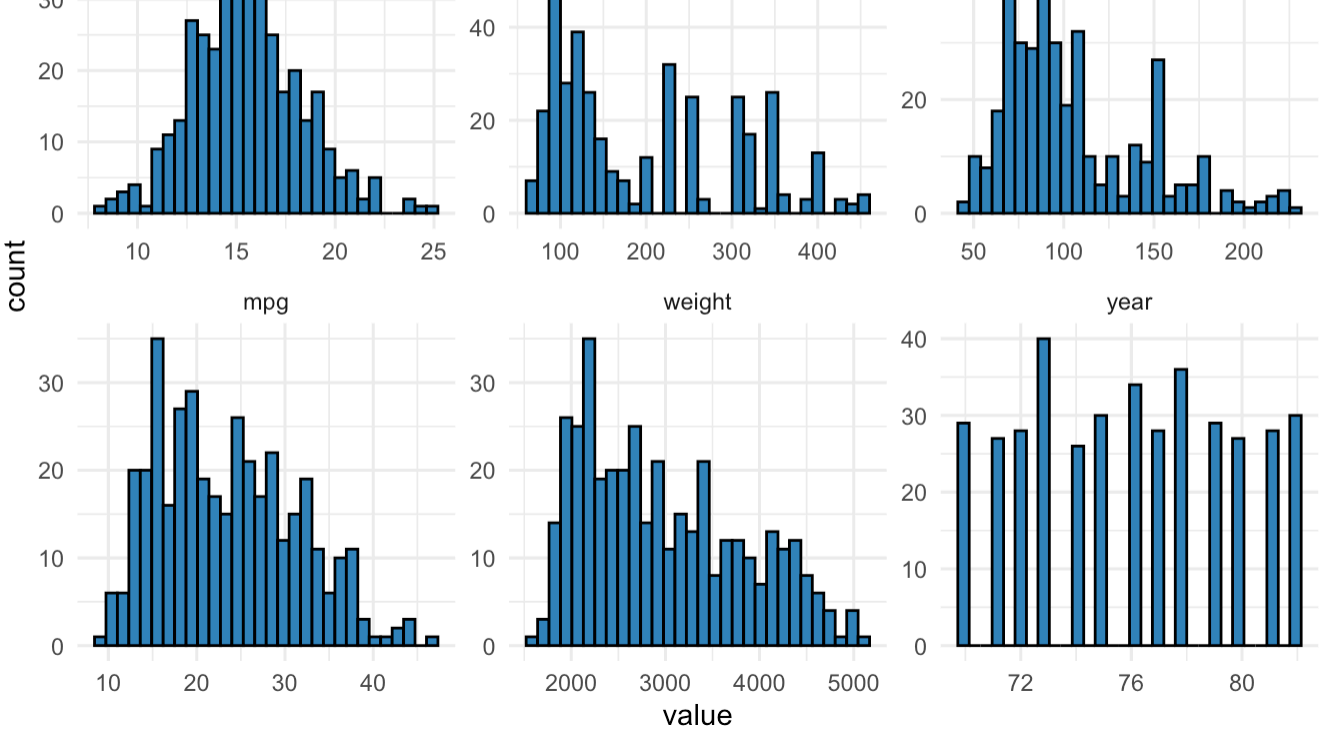
```
Auto %>%
  select_if(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_boxplot(color='black', fill='steelblue') +
  facet_wrap(~key, scales = 'free') +
  ggtitle(("Numerical Predictors - Box Plots")) +
  theme_minimal()
```



Histogram Plots

```
Auto %>%
  select_if(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_histogram(color='black', fill='steelblue') +
  facet_wrap(~key, scales = 'free') +
  ggtitle(("Numerical Predictors - Histograms")) +
  theme_minimal()
```





Comments on predictor relationships: Based on our scatterplots and heatmap, mpg has a strong negative relationship with displacement, horsepower, and weight as well as a moderate positive relationship with year. Horsepower has a moderately strong negative relationship with acceleration, and a strong positive relationship with weight. Displacement has a strong positive relationship with horsepower and weight, and a moderate negative relationship with acceleration.

3f) Suppose that we wish to predict gas mileage (mpg) on the basis of the other variables. Do your plots suggest that any of the other variables might be useful in predicting mpg? Justify your answer.

The correlation coefficients for variables displacement, horsepower, and weight are -0.81, -0.78, and -0.83, respectively. Since the magnitudes of these coefficients fall between the range -0.9 and -0.7, these variables have a strong negative correlation with mpg: - As engine displacement (inches) increases, the car gets less mpg. - As horsepower increases, mpg decreases. - As vehicle weight increases (lbs), mpg decreases. Addtionally, mpg has a moderate positive correlation (coeff = 0.58) with the model year of the vehicle, indicating that newer cars get better gas mileage. Considering our observations, these four variables might serve as useful predictors for building a model to predict mpg.

Problem 4 (Total: 15 Points)

We will predict the number of applications received using the other variables in the College data set available in the R package ISLR, which can be accessed as follows.

```
library(ISLR)
library(e1071)
library(nnet)
library(earth)
data(College)
#data basic information
head(College)
```

```
             Private Apps Accept Enroll Top10perc Top25perc
```

```
Abilene Christian University     Yes 1660   1232    721        23        52
Adelphi University               Yes 2186   1924    512        16        29
Adrian College                   Yes 1428   1097    336        22        50
Agnes Scott College              Yes  417    349    137        60        89
Alaska Pacific University        Yes  193    146     55        16        44
Albertson College                Yes  587    479    158        38        62
                           F.Undergrad P.Undergrad Outstate Room.Board Books
Abilene Christian University        2885         537     7440       3300   450
Adelphi University                  2683        1227    12280       6450   750
Adrian College                      1036          99    11250       3750   400
Agnes Scott College                  510          63    12960       5450   450
Alaska Pacific University            249         869     7560       4120   800
Albertson College                    678          41    13500       3335   500
                           Personal PhD Terminal S.F.Ratio perc.alumni Expend
Abilene Christian University     2200  70       78      18.1          12   7041
Adelphi University               1500  29       30      12.2          16  10527
Adrian College                   1165  53       66      12.9          30   8735
Agnes Scott College               875  92       97       7.7          37  19016
Alaska Pacific University        1500  76       72      11.9           2  10922
Albertson College                 675  67       73       9.4          11   9727
                           Grad.Rate
Abilene Christian University      60
Adelphi University                56
Adrian College                    54
Agnes Scott College               59
Alaska Pacific University         15
Albertson College                 55
```

```r
dim(College)
```

```
[1] 777  18
```

```r
# The column Apps is the response variable, and others may be treated as predictors.
#For instance, for linear regression model in R, you may use
lm(Apps~.,data=College)
```

```
Call:
lm(formula = Apps ~ ., data = College)

Coefficients:
(Intercept)   PrivateYes       Accept       Enroll    Top10perc    Top25perc
 -445.08413   -494.14897      1.58581     -0.88069     49.92628    -14.23448
F.Undergrad  P.Undergrad     Outstate   Room.Board        Books     Personal
    0.05739      0.04445     -0.08587      0.15103      0.02090      0.03110
         PhD     Terminal    S.F.Ratio  perc.alumni       Expend    Grad.Rate
   -8.67850     -3.33066     15.38961      0.17867      0.07790      8.66763
```

4a) Appropriately split the data set into a training set (80%) and a test set (20%). [3 points]

```r
# Split the data
set.seed(123)
```

```r
train_index <- createDataPartition(College$Apps, p = 0.8, list = FALSE)
train_data <- College[train_index, ]
test_data <- College[-train_index, ]
```

4b) Fit a support vector machine (SVM) model with the radial basis function function model using least squares on the training set. Clearly report the test error obtained. [3 points]

```r
# Fit SVM-RBF model
svm_mod <- svm(Apps ~ ., train_data, kernel = "radial")

# Predict test data
svm_preds <- predict(svm_mod, test_data)

# Calcuate MSE (Mean Squared Error)
svm_mse <- round(mean((svm_preds - test_data$Apps)^2),2)
cat("SVM MSE:", paste0(svm_mse))
```

```
SVM MSE: 622415.53
```

4c) Fit a neural network model on the training set by creating an appropriate grid for tuning parameters. Clearly report the test error obtained. [3 points]

```r
# Neural network tuning grid
grid_neural_network <- expand.grid(size = c(5, 10), decay = c(0.1, 0.5))

# Fit neural network model
set.seed(123)
model_neural_network <- train(
  Apps ~ .,
  data = train_data,
  method = "nnet",
  trControl = trainControl(method = "cv", number = 5),
  tuneGrid = grid_neural_network,
  trace = FALSE
)

# Predict test data
preds_neural_network <- predict(model_neural_network, test_data)

# Calculate the MSE
neural_network_mse <- round(mean((preds_neural_network - test_data$Apps)^2),2)
cat("Neural Network MSE:", paste0(neural_network_mse))
```

```
Neural Network MSE: 14683008.9
```

4d) Fit a Multivariate Adaptive Regression Splines (MARS) model on the training set with tuning parameters chosen by cross-validation. Clearly report the test error obtained, along with the importance of each predictor. estimates. [3 points]

```r
# Fit MARS model with cross-validation
set.seed(123)
```

```r
mars_mod <- train(
  Apps ~ .,
  data = train_data,
  method = "earth",
  trControl = trainControl(method = "cv", number = 5),
  tuneLength = 5
)

# Predict test data
mars_preds <- predict(mars_mod, test_data)

# Calculate the MSE
mars_mse <- round(mean((mars_preds - test_data$Apps)^2),2)
cat('MARS MSE:', paste0(mars_mse))
```

```
MARS MSE: 822261.73
```

```r
# Variable importance
variable_importance <- varImp(mars_mod)
variable_importance
```

```
earth variable importance

            Overall
Accept      100.000
Top10perc    16.090
Enroll        3.688
F.Undergrad   0.000
```

4e) Comment on the results obtained. Is there much difference among the test errors resulting from these three nonparametric approaches? [3 points]

SVM Results

```r
svm_mod
```

```
Call:
svm(formula = Apps ~ ., data = train_data, kernel = "radial")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.05555556
    epsilon:  0.1


Number of Support Vectors:  278
```

```r
cat("SVM MSE: ", svm_mse)
```

```
SVM MSE:  622415.5
```

Neural Network Results

```
model_neural_network
```

```
Neural Network

624 samples
 17 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 499, 500, 497, 500, 500
Resampling results across tuning parameters:

  size  decay  RMSE      Rsquared      MAE
   5    0.1    5069.826  0.0005407588  3091.989
   5    0.5    5069.826           NaN  3091.989
  10    0.1    5069.826  0.0706531705  3091.989
  10    0.5    5069.826  0.0893022018  3091.989

RMSE was used to select the optimal model using the smallest value.
The final values used for the model were size = 5 and decay = 0.5.
```

```
cat("Neural Network MSE: ", neural_network_mse)
```

```
Neural Network MSE:  14683009
```

MARS Results

```
mars_mod
```

```
Multivariate Adaptive Regression Spline

624 samples
 17 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 499, 500, 497, 500, 500
Resampling results across tuning parameters:

  nprune  RMSE      Rsquared   MAE
   2      1594.315  0.8500763  699.0368
   4      1270.057  0.9178074  548.5028
   6      1251.483  0.9218557  557.1920
   8      1286.650  0.9170281  597.1205
  10      1311.021  0.9134643  608.1759

Tuning parameter 'degree' was held constant at a value of 1
RMSE was used to select the optimal model using the smallest value.
```

```
The final values used for the model were nprune = 6 and degree = 1.
```

```
cat("MARS MSE: ", mars_mse)
```

```
MARS MSE:  822261.7
```

Conclusion: There is much difference among the test errors resulting from these three nonparametric approaches. The neural network approach calculated the largest test error. Additionally, it did not calculate decent R-squared values. The MARS model had the second highest test error, but the model produced R-squared values greater than 85%, indicating that it explains the variance of the data well. The support vector machine with the radial basis function calculated the lowest test error.

Problem 5 (Total: 22 Points)

A chemical manufacturing process for a pharmaceutical product was discussed in Sect. 1.4 of the textbook. In this problem, the objective is to understand the relationship between biological measurements of the raw materials (predictors), measurements of the manufacturing process (predictors), and the response of product yield. Biological predictors cannot be changed but can be used to assess the quality of the raw material before processing. On the other hand, manufacturing process predictors can be changed in the manufacturing process. Improving product yield by 1% will boost revenue by approximately one hundred thousand dollars per batch. We may start R and use these commands to load the data:

```
library(AppliedPredictiveModeling)
library(ggplot2)
library(dplyr)
library(caret)
library(e1071)
data("ChemicalManufacturingProcess")
```

The matrix processPredictors contains the 57 predictors (12 describing the input biological material and 45 describing the process predictors) for the 176 manufacturing runs. yield contains the percent yield for each run.

5a) A small percentage of cells in the predictor set contain missing values. Use an appropriate imputation function to fill in these missing values. [3 points]

```
# Identify missing values
missing_vals <- sum(is.na(ChemicalManufacturingProcess))
print(paste0("Pre-imputation missing values: ", missing_vals))
```

```
[1] "Pre-imputation missing values: 106"
```

```
# Impute missing values with k-NN
pre_process <- preProcess(ChemicalManufacturingProcess, c("knnImpute"))
df <- predict(pre_process, ChemicalManufacturingProcess)

# Double check successful imputation (i.e., 0 missing values)
impute_miss <- sum(is.na(pre_process))
```

```
impute_miss <- sum(is.na(pre_process))
print(paste0("Post-imputation missing values: ", impute_miss))
```

```
[1] "Post-imputation missing values: 0"
```

5b) Split the data into a training and a test set, pre-process the data, and build at least four different models from Chapter 6. For those models with tuning parameters (e.g., ENET), what are the optimal values of the tuning parameter(s)? [8 points]

```
# Create variables to separate predictors and response
proc_preds <- df[, -1]
resp <- df[, 1]

# Split the data
train_index <- createDataPartition(resp, p = 0.8, list = FALSE)
train_data <- proc_preds[train_index, ]
train_resp <- resp[train_index]
test_data <- proc_preds[-train_index, ]
test_resp <- resp[-train_index]

# Data normalization - center and scale
pre_process_range <- preProcess(train_data, method = c("center", "scale"))
train_transformed <- predict(pre_process_range, train_data)
test_transformed <- predict(pre_process_range, test_data)

# Create list for model training
models <- list(
  ENET = trainControl(method = "cv", number = 5),
  Ridge = trainControl(method = "cv", number = 5),
  PCR = trainControl(method = "cv", number = 5),
  PLS = trainControl(method = "cv", number = 5)
)

# Tuning grids
tuning_grids <- list(
  ENET = expand.grid(alpha = seq(0, 1, length = 10), lambda = 10^seq(4, -4, length = 100)),
  Ridge = expand.grid(alpha = 0, lambda = 10^seq(4, -4, length = 100)),
  PLS = expand.grid(ncomp = 1:10)
)

# Train the four models
set.seed(123)
enet_mod <- train(train_transformed, train_resp, method = "glmnet", trControl = models$ENET,
ridge_mod <- train(train_transformed, train_resp, method = "glmnet", trControl = models$Ridge
pls_mod <- train(train_transformed, train_resp, method = "pls", trControl = models$PLS, tuneG
pcr_mod <- train(train_transformed, train_resp, method = "pcr", trControl = models$PCR, tuneG

# Optimal tuning parameters
enet_mod$bestTune
```

```
    alpha    lambda
940     1 0.1417474
```

```
ridge_mod$bestTune
```

```
   alpha   lambda
59     0 4.862602
```

```
pls_mod$bestTune
```

```
  ncomp
3     3
```

```
pcr_mod$bestTune
```

```
   ncomp
10    10
```

5c) Which model has the best predictive ability? Is any model significantly better or worse than the others? You need to conduct a hypothesis testing to justify your choice if necessary. [5 points]

```
# Predict test set
enet_preds <- predict(enet_mod, test_transformed)
ridge_preds <- predict(ridge_mod, test_transformed)
pls_preds <- predict(pls_mod, test_transformed)
pcr_preds <- predict(pcr_mod, test_transformed)

# Calculate MSE (Mean Squared Error)
test_err_enet <- mean((enet_preds - test_resp)^2)
test_err_ridge <- mean((ridge_preds - test_resp)^2)
test_err_pls <- mean((pls_preds - test_resp)^2)
test_err_pcr <- mean((pcr_preds - test_resp)^2)

# Print MSE
cat("ENET Test Error: ", test_err_enet)
```

```
ENET Test Error:  0.4927384
```

```
cat("Ridge Test Error: ", test_err_ridge)
```

```
Ridge Test Error:  0.6256541
```

```
cat("PLS Test Error: ", test_err_pls)
```

```
PLS Test Error:  0.3802276
```

```
cat("PCR Test Error: ", test_err_pcr)
```

```
PCR Test Error:  0.4212739
```

```
# Calculate residuals for each model
enet_res <- predict(enet_mod, test_transformed) - test_resp
ridge_res <- predict(ridge_mod, test_transformed) - test_resp
pls_res <- predict(pls_mod, test_transformed) - test_resp
```

```
pcr_res <- predict(pcr_mod, test_transformed) - test_resp

# Paired t-tests comparing ENET res with other models
enet_ridge_t_test <- t.test(enet_res, ridge_res, paired = TRUE)
enet_pls_t_test <- t.test(enet_res, pls_res, paired = TRUE)
enet_pcr_t_test <- t.test(enet_res, pcr_res, paired = TRUE)

# Print p-values of the t-tests
cat("P-value for ENET vs. Ridge Regression:", enet_ridge_t_test$p.value)
```

```
P-value for ENET vs. Ridge Regression: 0.3235442
```

```
cat("P-value for ENET vs. PLS:", enet_pls_t_test$p.value)
```

```
P-value for ENET vs. PLS: 0.945268
```

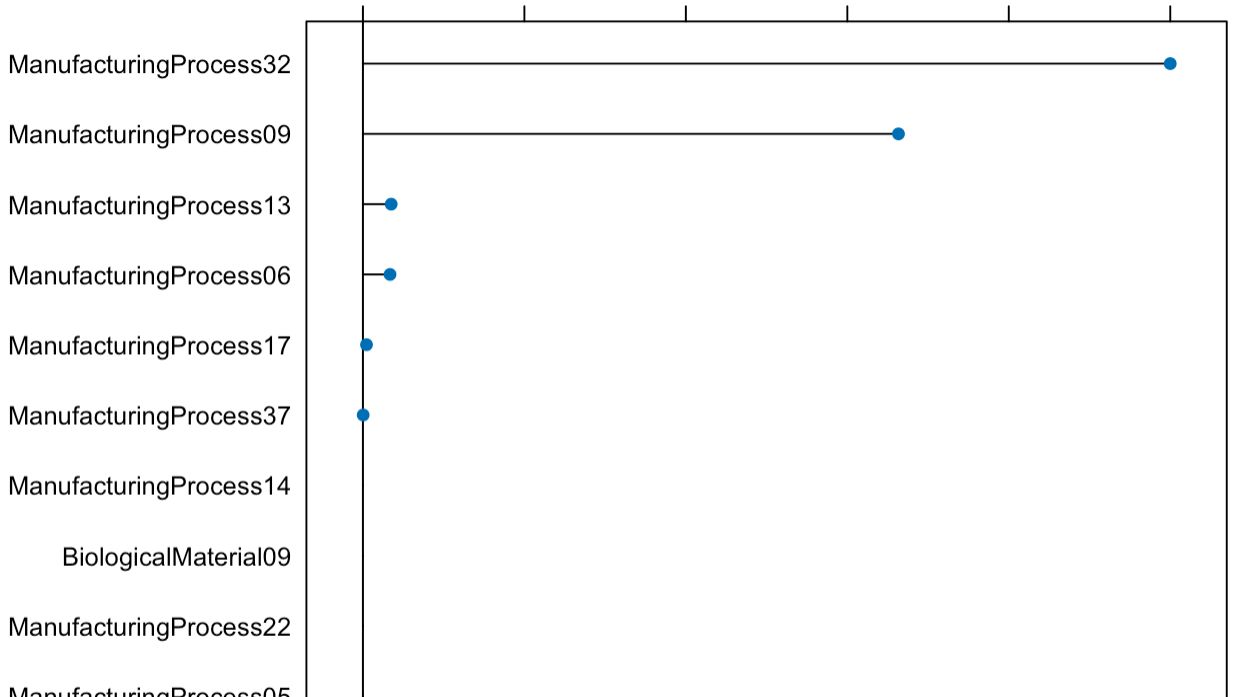
```
cat("P-value for ENET vs. PCR:", enet_pcr_t_test$p.value)
```

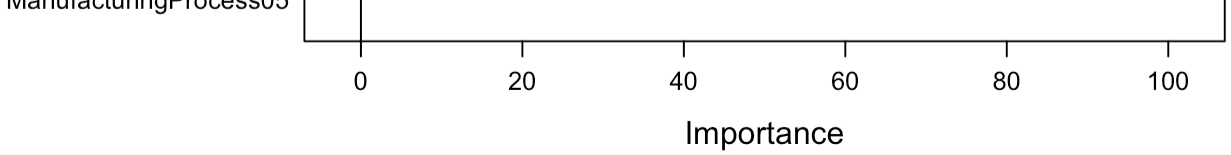
```
P-value for ENET vs. PCR: 0.8532876
```

Since our PLS model has the lowest test error, it has the best predictive ability among our four models; however, the results from our hypothesis testing indicate that there is no significant difference between the predictive abilities of our models (i.e., all p-values were greater than the significance level 0.05, so we do not reject the null hypothesis).

5d) Which predictors are most important in the model you have trained? Do either the biological or process predictors dominate the list [3 points]

```
# Variable Importance - ENET
variable_importance <- varImp(enet_mod)
plot(variable_importance, top = 10)
```





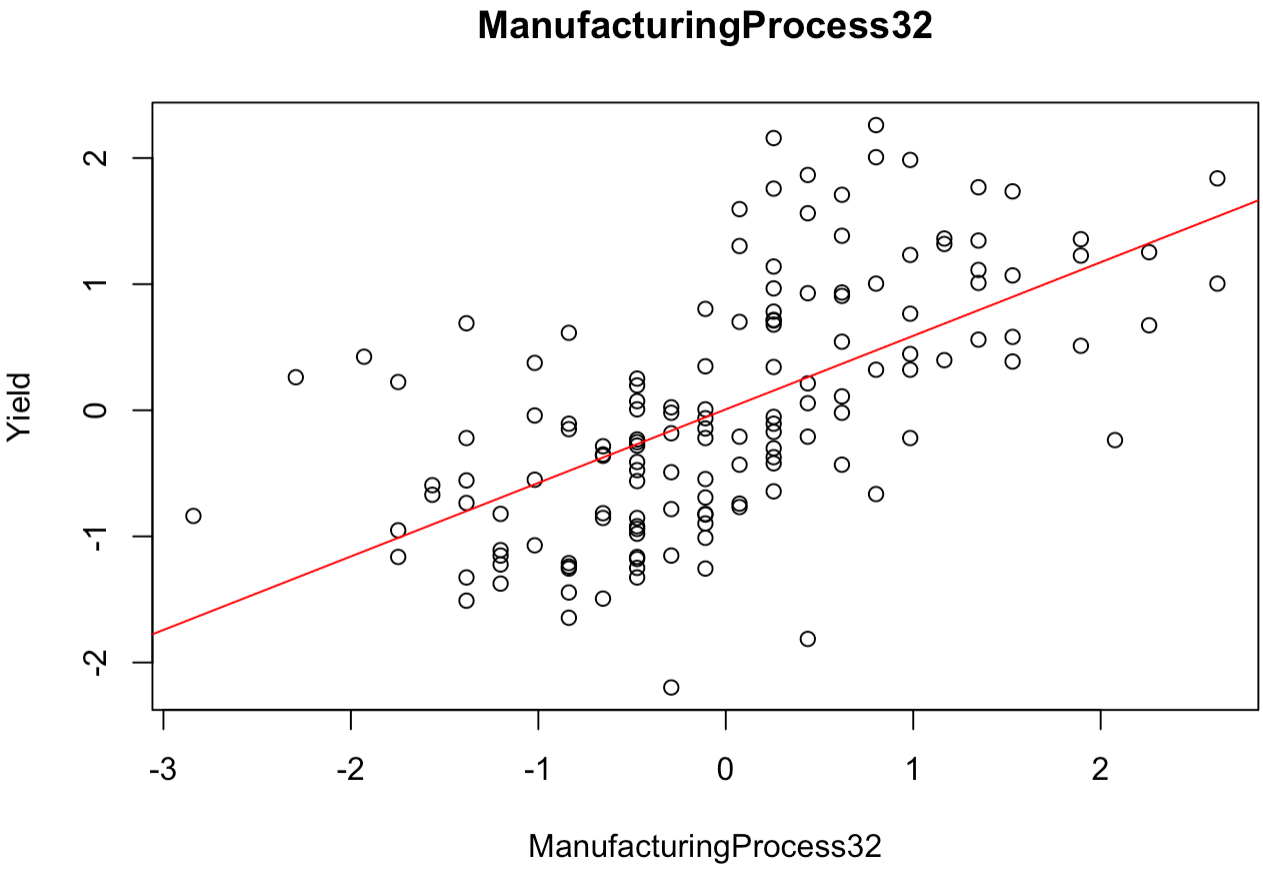

```
top_preds <- rownames(variable_importance$importance)[order(variable_importance$importance[,1
cat("Top 5 predictors: ", paste0(top_preds), sep = "\n")
```

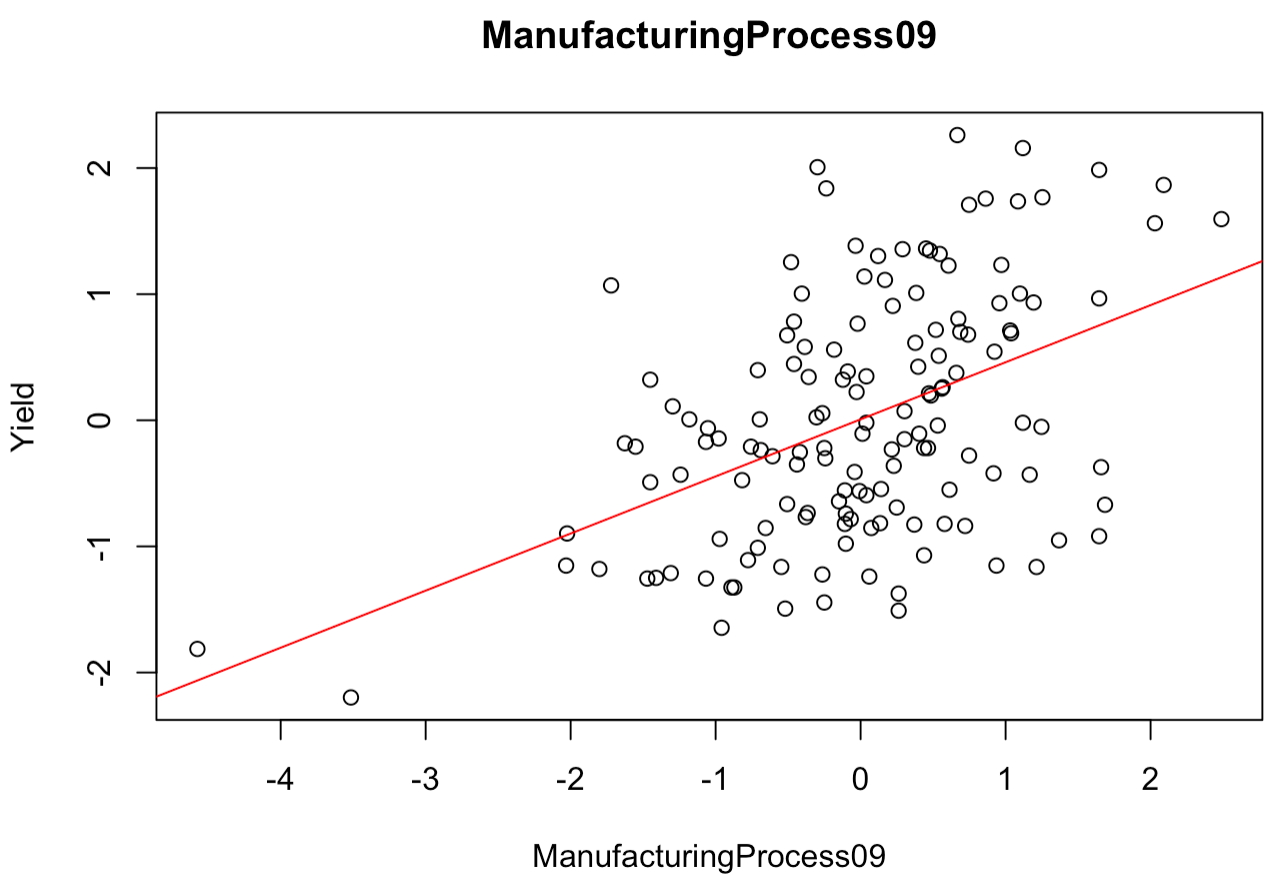
```
Top 5 predictors:
ManufacturingProcess32
ManufacturingProcess09
ManufacturingProcess13
ManufacturingProcess06
ManufacturingProcess17
```

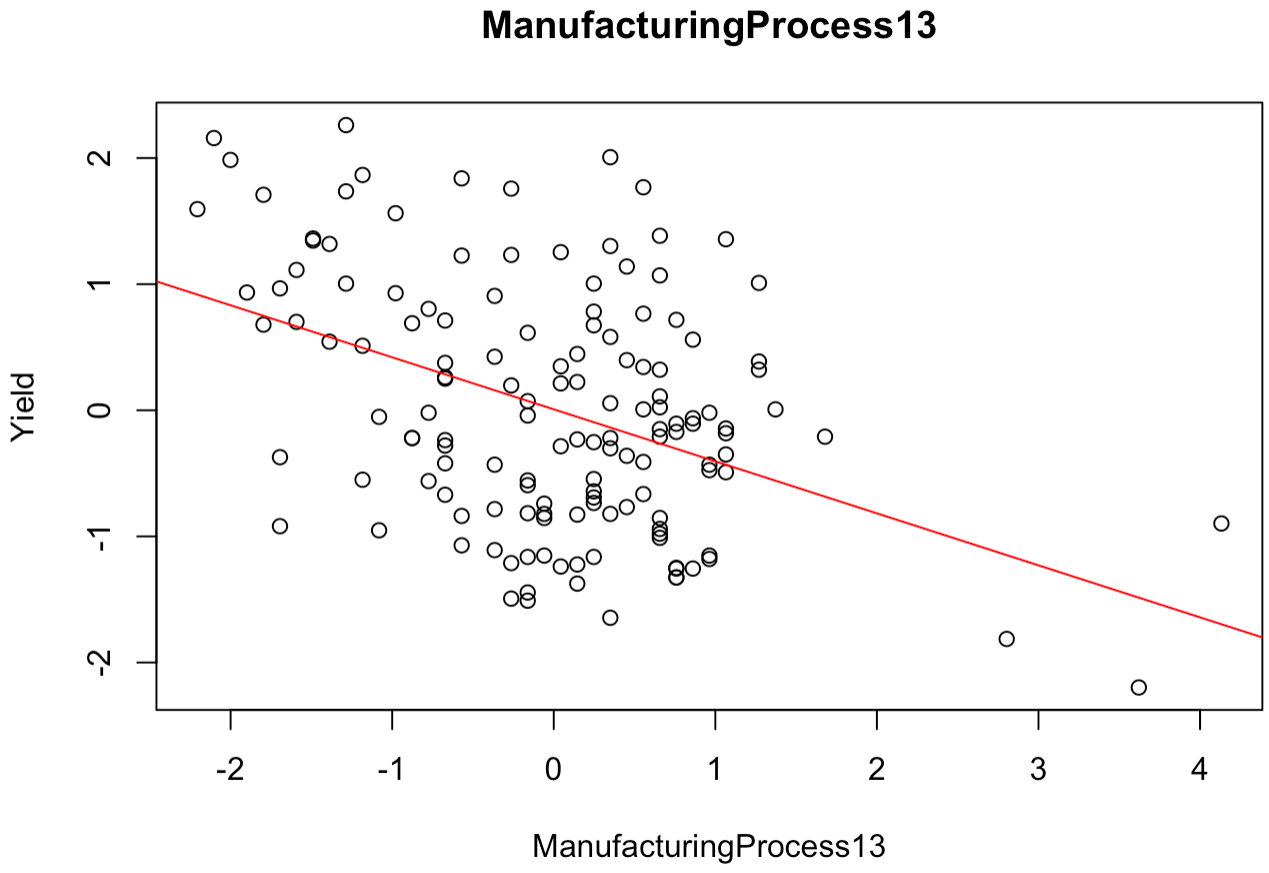
The top 10 predictors (listed above) indicate that the majority are process rather than biological predictors. ManufacturingProcess32 & ManufacturingProcess09 are the most important predictors.

5e) Explore the relationships between each of the top predictors and the response. How could this information be helpful in improving yield in future runs of the manufacturing process? [3 points]

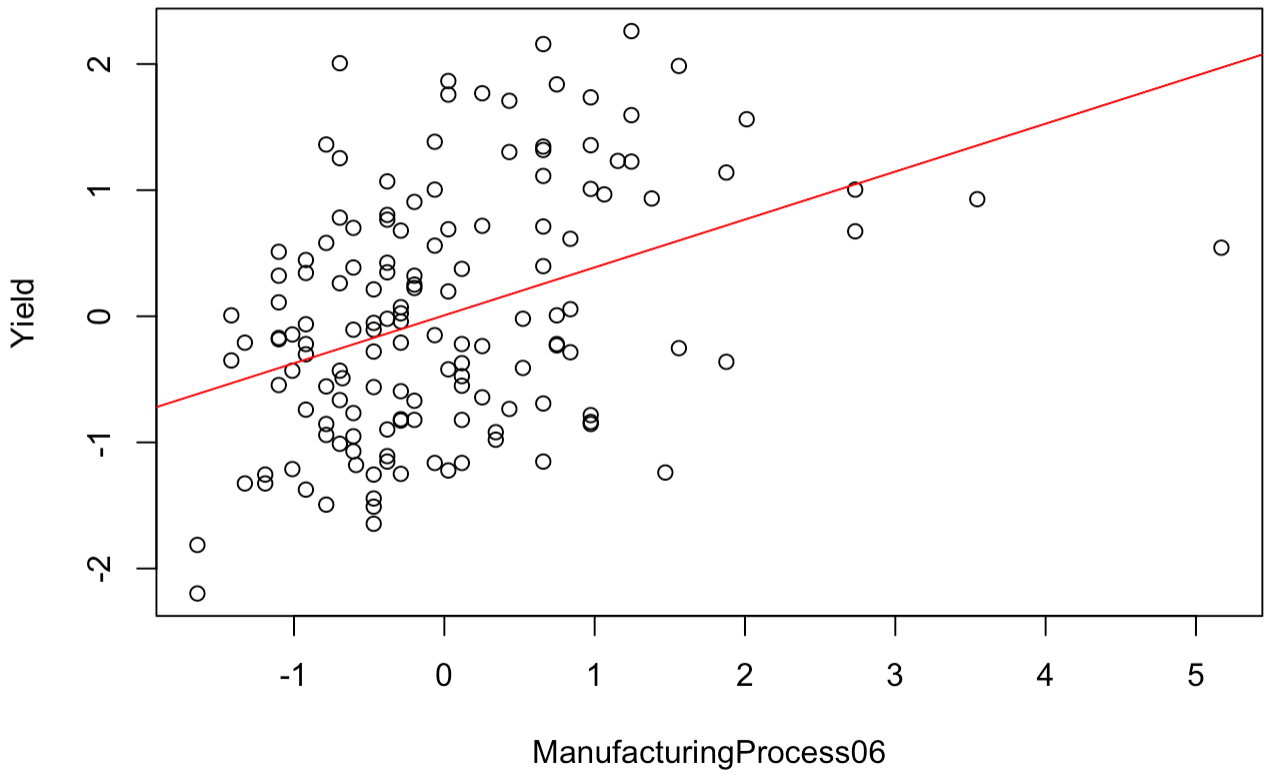
```
for (pred in top_preds) {
  plot(train_transformed[, pred], train_resp, main = pred, xlab = pred, ylab = "Yield")
  abline(lm(train_resp ~ train_transformed[, pred]), col = "red")
}
```



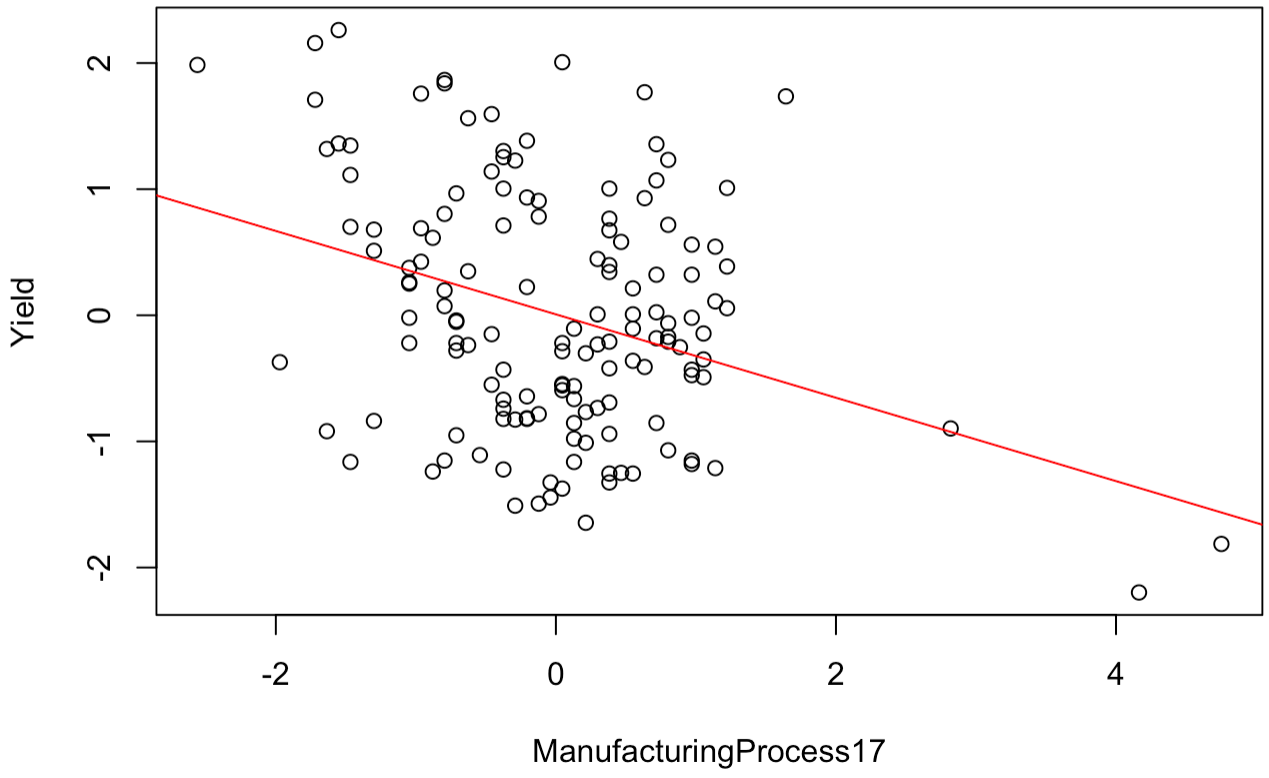




ManufacturingProcess06



ManufacturingProcess17



The relationships between our top predictors and the response variable (Yield) indicate that the process predictors are pretty evenly split between positive and negative linear relationships. By identifying these relationships between individual predictors and the response variable, the manufacturers have the opportunity to optimize their processes and potentially increase Yield.