Sta 325 Final Project

Calleigh Smith, Hannah Bogomilsky, Hugh Esterson, Maria Henriquez, Mariana Izon

11/22/2020

```
library(readr)
library(dplyr)
library(tidyverse)
library(gridExtra)
library(mgcv)
library(patchwork)
```

```
flights <- read_csv("data/flights.csv")

unique(flights$OP_CARRIER)
```

```
## [1] "AA" "DL" "B6" "AS"
```

```
unique(flights$DEST)
```

```
##  [1] "LAX" "SFO" "SJC" "SAN" "PSP" "SMF" "OAK" "LGB" "ONT" "BUR"
```

```
class(flights$CARRIER_DELAY)
```

```
## [1] "numeric"
```

```
flights <- flights %>%
  mutate(CARRIER_DELAY = case_when(CARRIER_DELAY > 0 ~ 1,
                                   TRUE ~ 0),
         WEATHER_DELAY = case_when(WEATHER_DELAY > 0 ~ 1,
                                   TRUE ~ 0),
         NAS_DELAY = case_when(NAS_DELAY > 0 ~ 1,
                               TRUE ~ 0),
         SECURITY_DELAY = case_when(SECURITY_DELAY > 0 ~ 1,
                                    TRUE ~ 0),
         LATE_AIRCRAFT_DELAY = case_when(LATE_AIRCRAFT_DELAY > 0 ~ 1,
                                         TRUE ~ 0)) %>%
  filter(!is.na(ARR_DELAY))
```

```
flights
```

```
## # A tibble: 2,033 x 34
##     YEAR MONTH DAY_OF_MONTH DAY_OF_WEEK FL_DATE    OP_CARRIER TAIL_NUM
##    <dbl> <dbl>        <dbl>       <dbl> <date>     <chr>      <chr>
##  1  2020     1            1           3 2020-01-01 AA         N110AN
##  2  2020     1            2           4 2020-01-02 AA         N111ZM
##  3  2020     1            3           5 2020-01-03 AA         N108NN
##  4  2020     1            4           6 2020-01-04 AA         N102NN
##  5  2020     1            5           7 2020-01-05 AA         N113AN
##  6  2020     1            6           1 2020-01-06 AA         N103NN
```

```
## 7  2020     1            7           2 2020-01-07 AA         N113AN
## 8  2020     1            8           3 2020-01-08 AA         N106NN
## 9  2020     1            9           4 2020-01-09 AA         N102NN
## 10 2020     1           10           5 2020-01-10 AA         N117AN
## # ... with 2,023 more rows, and 27 more variables: OP_CARRIER_FL_NUM <dbl>,
## #   ORIGIN <chr>, ORIGIN_CITY_NAME <chr>, DEST <chr>, DEST_CITY_NAME <chr>,
## #   CRS_DEP_TIME <dbl>, DEP_TIME <dbl>, DEP_DELAY <dbl>, TAXI_OUT <dbl>,
## #   WHEELS_OFF <dbl>, WHEELS_ON <dbl>, TAXI_IN <dbl>, CRS_ARR_TIME <dbl>,
## #   ARR_TIME <dbl>, ARR_DELAY <dbl>, CANCELLED <dbl>, CANCELLATION_CODE <lgl>,
## #   DIVERTED <dbl>, CRS_ELAPSED_TIME <dbl>, ACTUAL_ELAPSED_TIME <dbl>,
## #   AIR_TIME <dbl>, DISTANCE <dbl>, CARRIER_DELAY <dbl>, WEATHER_DELAY <dbl>,
## #   NAS_DELAY <dbl>, SECURITY_DELAY <dbl>, LATE_AIRCRAFT_DELAY <dbl>
```

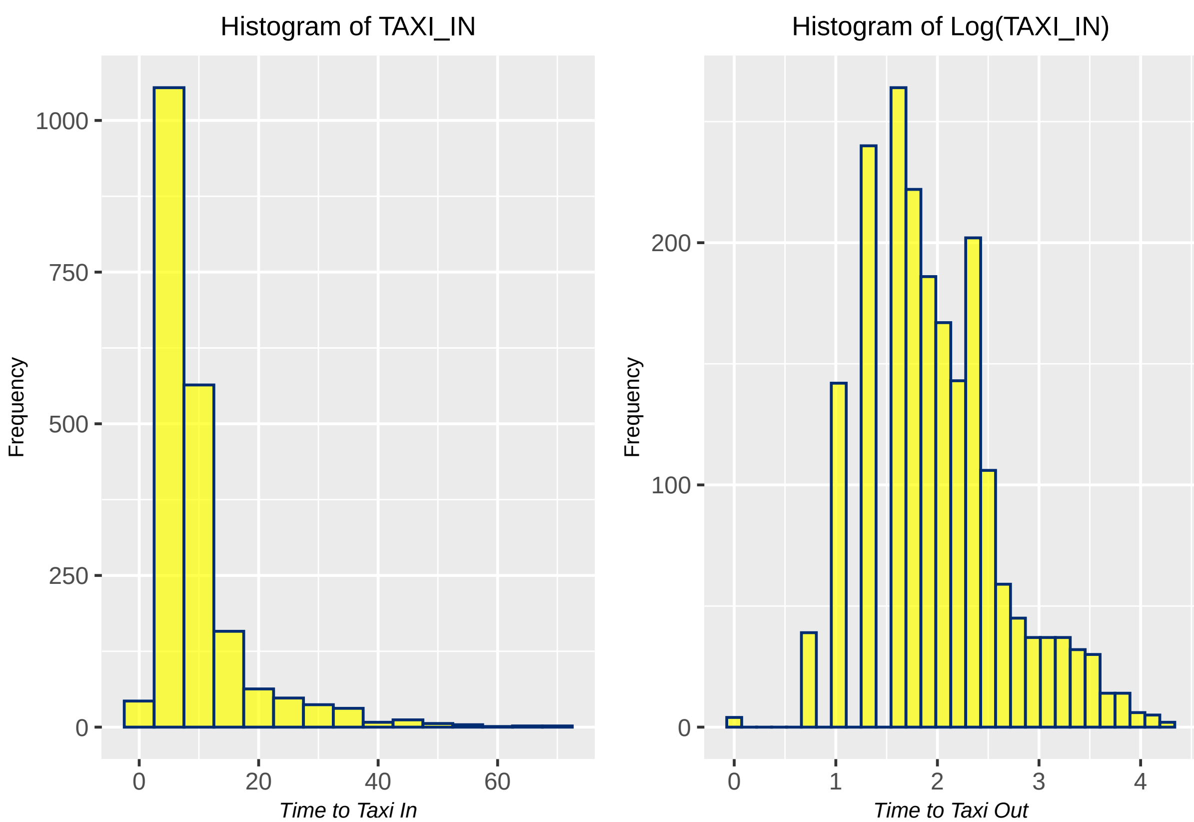
INDIVIDUAL PREDICTORS

Taxi Histograms

```
pTAXI_IN <- ggplot(data = flights, aes(x = TAXI_IN)) +
  geom_histogram(binwidth = 5, fill = "#FFFF00", color = "#002D72", alpha = .7) +
  labs(x = "Time to Taxi In",
       y = "Frequency",
       title = "Histogram of TAXI_IN") +
  theme(plot.title = element_text(size = 10,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))



 # ggplot(train_data, mapping = aes(x = St2)) +
 #   geom_histogram(binwidth =2.5, fill = "#FFFF00", color = "#002D72", alpha = .7) +
 #   labs(x = xlab(bquote('St^2')),
 #       # xlab(bquote('Assimilation ('*mu~ 'mol' ~CO[2]~ m^-2~s^-1*')')),
 #        y = "Frequency",
 #        title = "Histogram of Stokes Number, Squared") +
 #   theme(plot.title = element_text(size = 10,hjust = 0.5),
 #         plot.subtitle = element_text(hjust = 0.5),
 #         axis.title.x.bottom = element_text(size = 8, face = "italic"),
 #         axis.title.y.left = element_text(size = 8))


pTAXI_OUT <- ggplot(data = flights, aes(x = TAXI_OUT)) +
  geom_histogram(binwidth = 5, fill = "#FFFF00", color = "#002D72", alpha = .7) +
  labs(x = "Time to Taxi Out",
       y = "Frequency",
       title = "Histogram of TAXI_OUT") +
  theme(plot.title = element_text(size = 10,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))


flights$log_TAXI_OUT <- log(flights$TAXI_OUT)
```

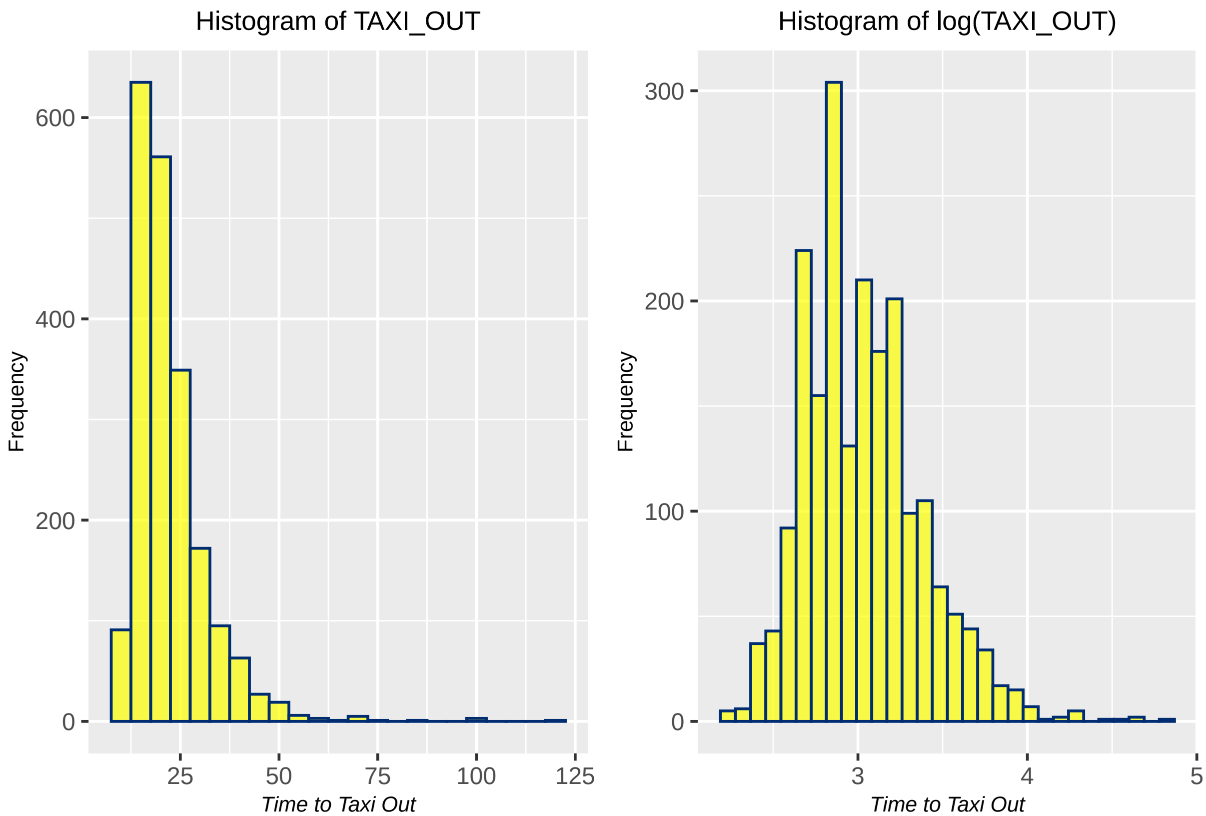
```r
flights$log_TAXI_IN <- log(flights$TAXI_IN)

plog_TAXI_OUT <- ggplot(data = flights, aes(x = log_TAXI_OUT)) +
  geom_histogram(fill = "#FFFF00", color = "#002D72", alpha = .7) +
  labs(x = "Time to Taxi Out",
       y = "Frequency",
       title = "Histogram of log(TAXI_OUT)") +
  theme(plot.title = element_text(size = 10,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))


plog_TAXI_IN <- ggplot(data = flights, aes(x = log_TAXI_IN)) +
  geom_histogram(fill = "#FFFF00", color = "#002D72", alpha = .7) +
  labs(x = "Time to Taxi Out",
       y = "Frequency",
       title = "Histogram of Log(TAXI_IN)") +
  theme(plot.title = element_text(size = 10,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))

pTAXI_IN + plog_TAXI_IN
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



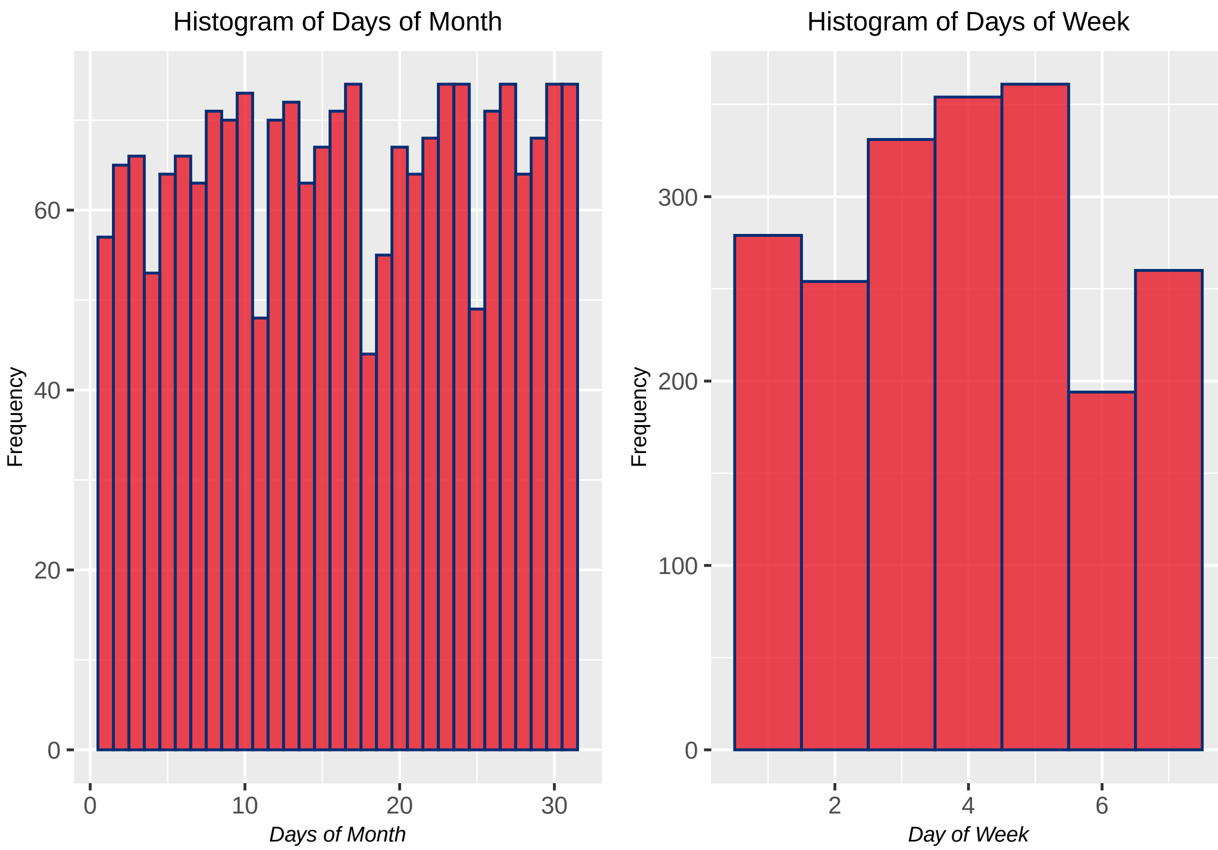
```r
pTAXI_OUT + plog_TAXI_OUT
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Days of Month and Week

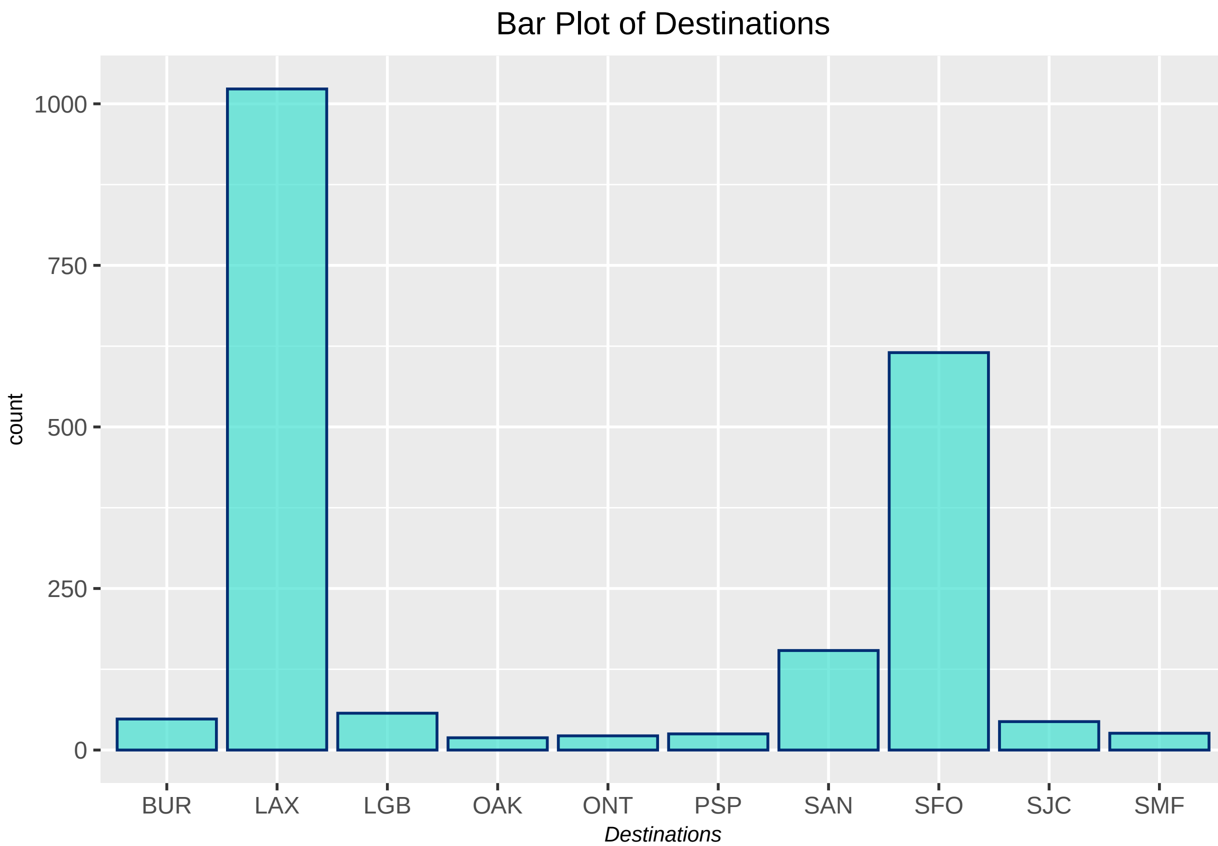
```r
p02 <- ggplot(data = flights, aes(x = DAY_OF_MONTH)) +
  geom_histogram(binwidth = 1, fill = "#E81828", color = "#002D72", alpha = .8) +
  labs(x = "Days of Month",
       y = "Frequency",
       title = "Histogram of Days of Month") +
    theme(plot.title = element_text(size = 10,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))

p03 <- ggplot(data = flights, aes(x = DAY_OF_WEEK)) +
  geom_histogram(binwidth = 1, fill = "#E81828", color = "#002D72", alpha = .8) +
  labs(x = "Day of Week",
       y = "Frequency",
       title = "Histogram of Days of Week") +
    theme(plot.title = element_text(size = 10,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))

grid.arrange(p02, p03, nrow = 1)
```



Destination Locations

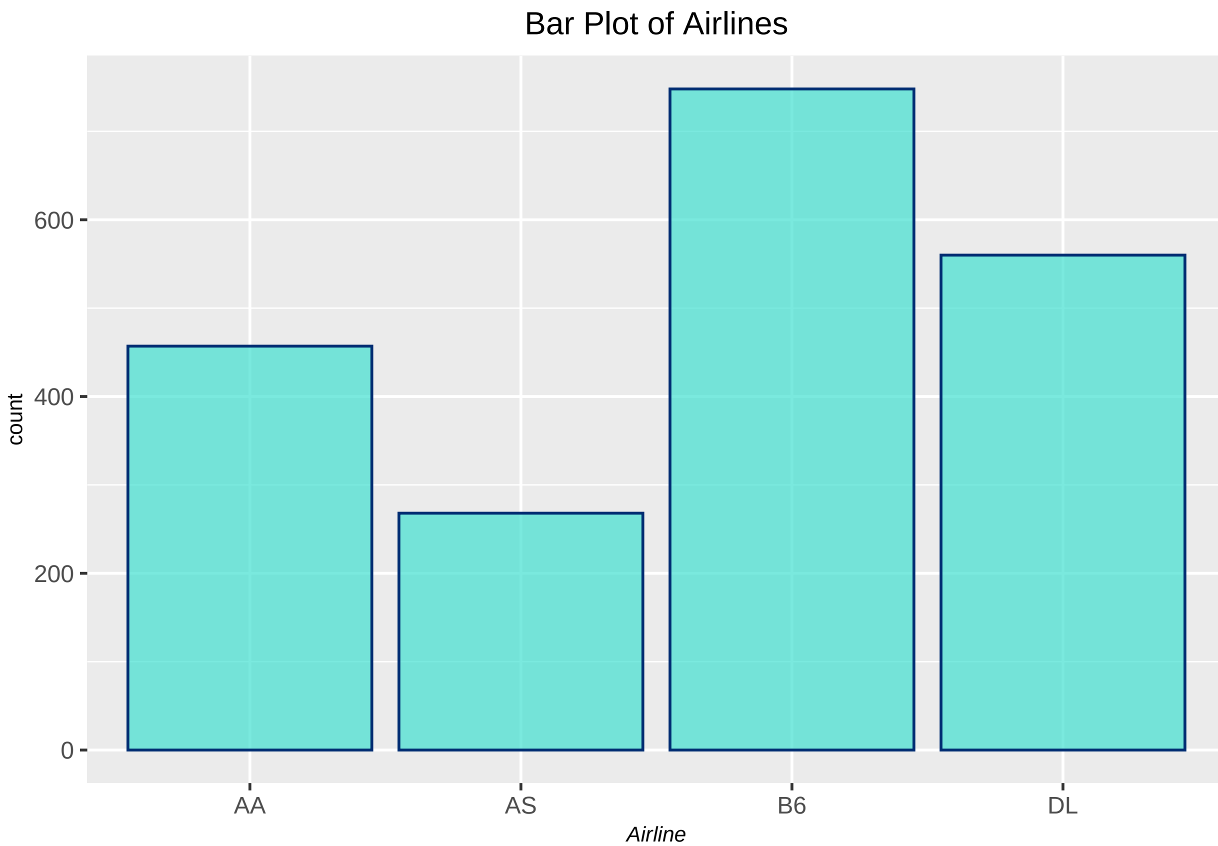
Origin is all JFK, but we could consider the different destination locations.

```
ggplot(data = flights, aes(x = DEST)) +
  geom_bar(fill = "#40E0D0", color = "#002D72", alpha = .7) +
  labs(x = "Destinations",
       title = "Bar Plot of Destinations") +
  theme(plot.title = element_text(size = 12,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))
```

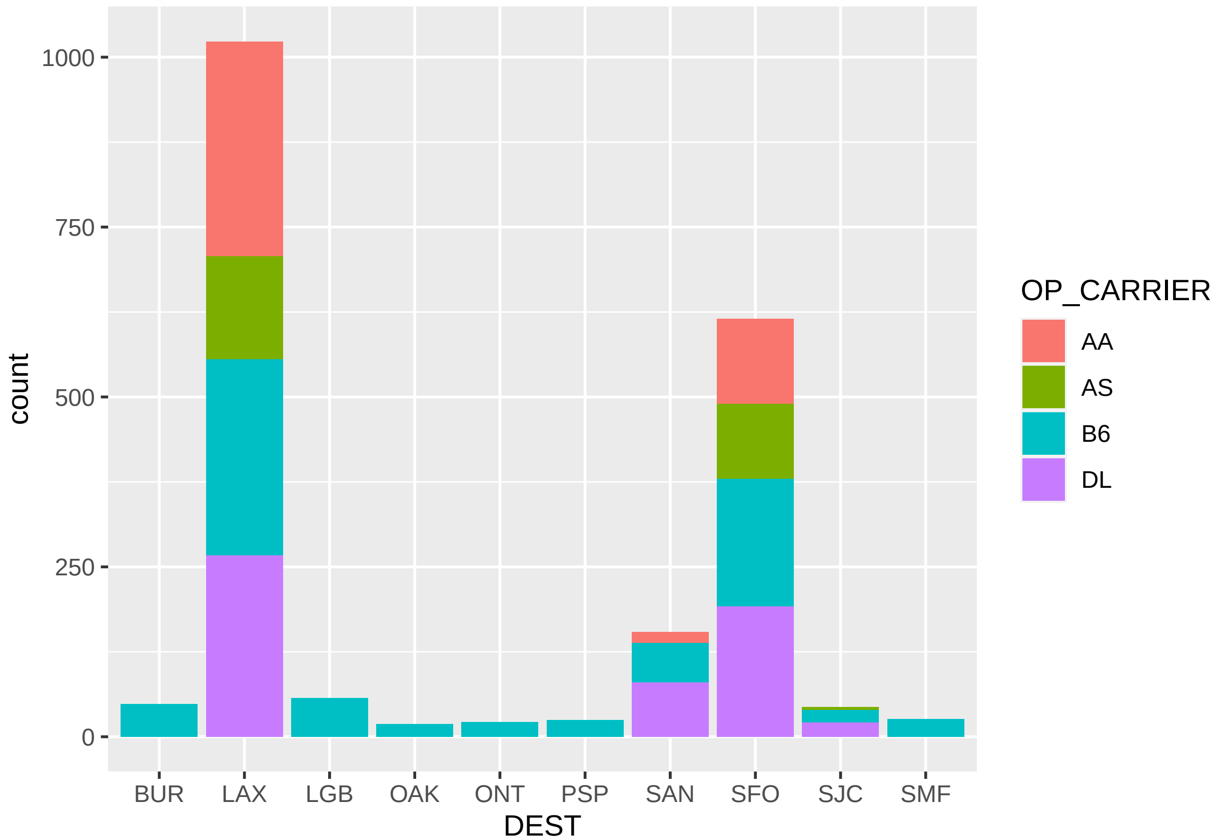


Airlines

```
ggplot(data = flights, aes(x = OP_CARRIER)) +
  geom_bar(fill = "#40E0D0", color = "#002D72", alpha = .7) +
  labs(x = "Airline",
       title = "Bar Plot of Airlines") +
  theme(plot.title = element_text(size = 12,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))
```

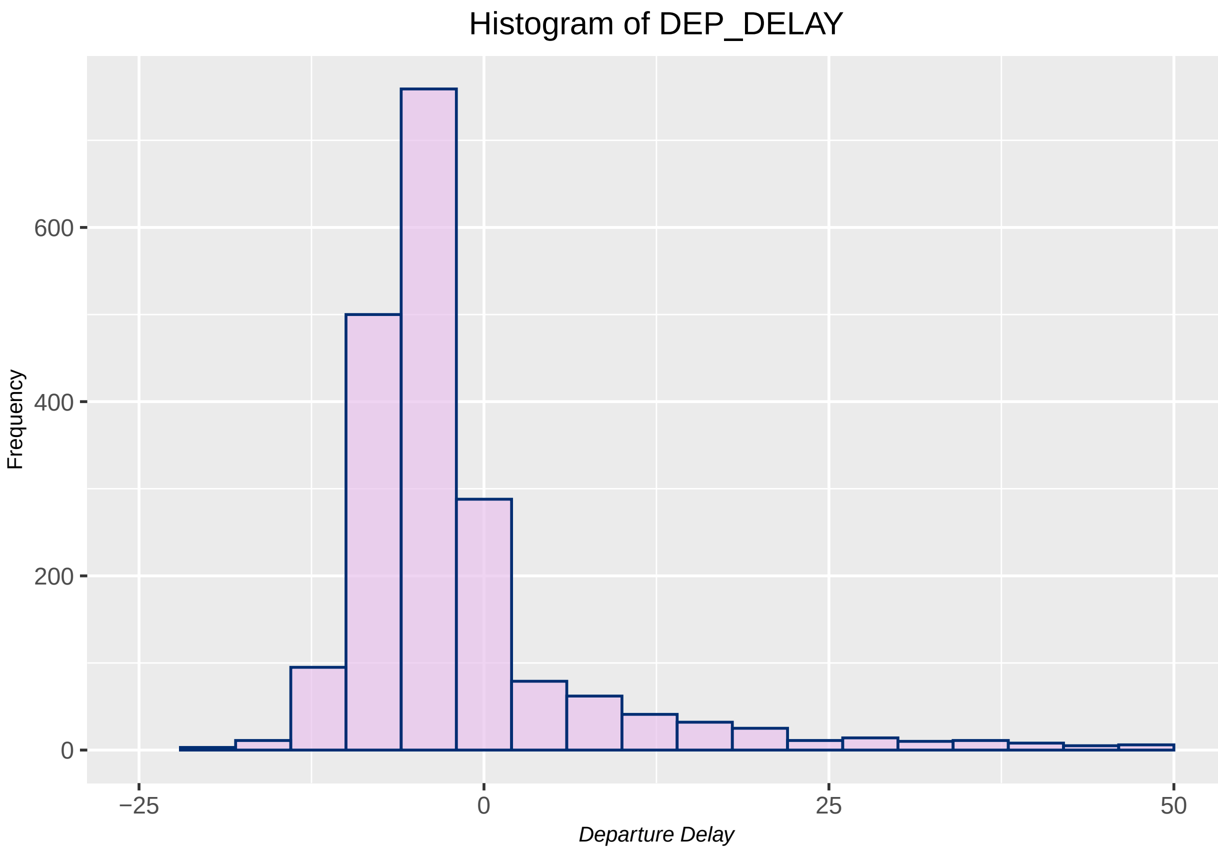



```
ggplot(data = flights, aes(x = DEST, fill = OP_CARRIER)) +
  geom_bar()
```

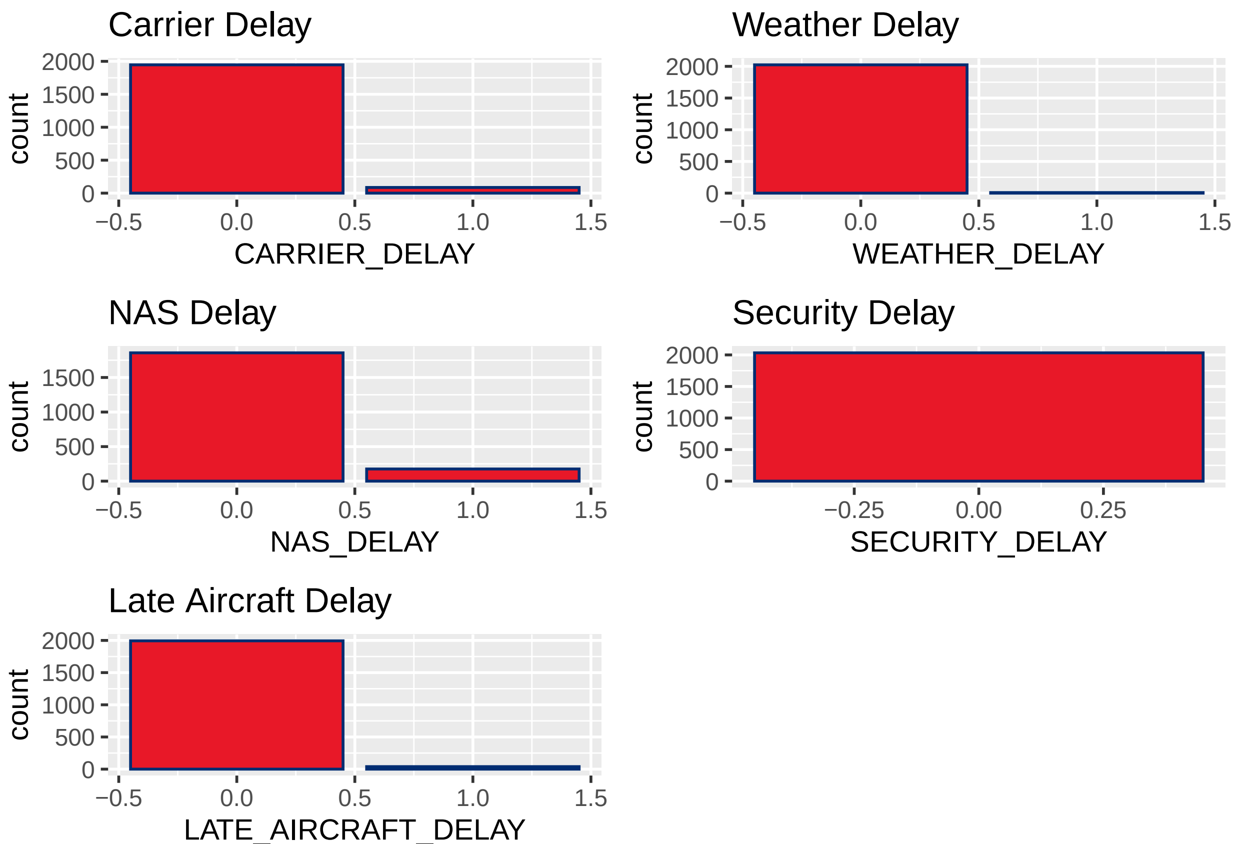


Depart Delay Histogram

```r
ggplot(data = flights, aes(x = DEP_DELAY)) +
  geom_histogram(binwidth = 4, fill = "#e9c2ed", color = "#002D72", alpha = 0.7) +
  xlim(-25, 50) +
  labs(x = "Departure Delay",
       y = "Frequency",
       title = "Histogram of DEP_DELAY") +
  theme(plot.title = element_text(size = 12,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))
```




```
p1 <- ggplot(data = flights, aes(x = CARRIER_DELAY)) +
  geom_bar(fill = "#E81828", color = "#002D72") +
  labs(title = "Carrier Delay")

p2 <- ggplot(data = flights, aes(x = WEATHER_DELAY)) +
  geom_bar(fill = "#E81828", color = "#002D72") +
  labs(title = "Weather Delay")

p3 <- ggplot(data = flights, aes(x = NAS_DELAY)) +
  geom_bar(fill = "#E81828", color = "#002D72") +
  labs(title = "NAS Delay")

p4 <- ggplot(data = flights, aes(x = SECURITY_DELAY)) +
  geom_bar(fill = "#E81828", color = "#002D72") +
  labs(title = "Security Delay")

p5 <- ggplot(data = flights, aes(x = LATE_AIRCRAFT_DELAY)) +
  geom_bar(fill = "#E81828", color = "#002D72") +
  labs(title = "Late Aircraft Delay")

grid.arrange(p1,p2,p3,p4,p5, nrow = 3)
```



From this EDA of the categorical variables, we probably should not perform analysis with `SECURITY_DELAY` since all of them are classified as 0.

```
flights %>%
  count(WEATHER_DELAY)
```

```
## # A tibble: 2 x 2
##   WEATHER_DELAY     n
##           <dbl> <int>
## 1             0  2024
## 2             1     9
```

Furthermore, only 9 flights are classified with a weather delay, so it may not be good for our model to include this as a variable for right now.

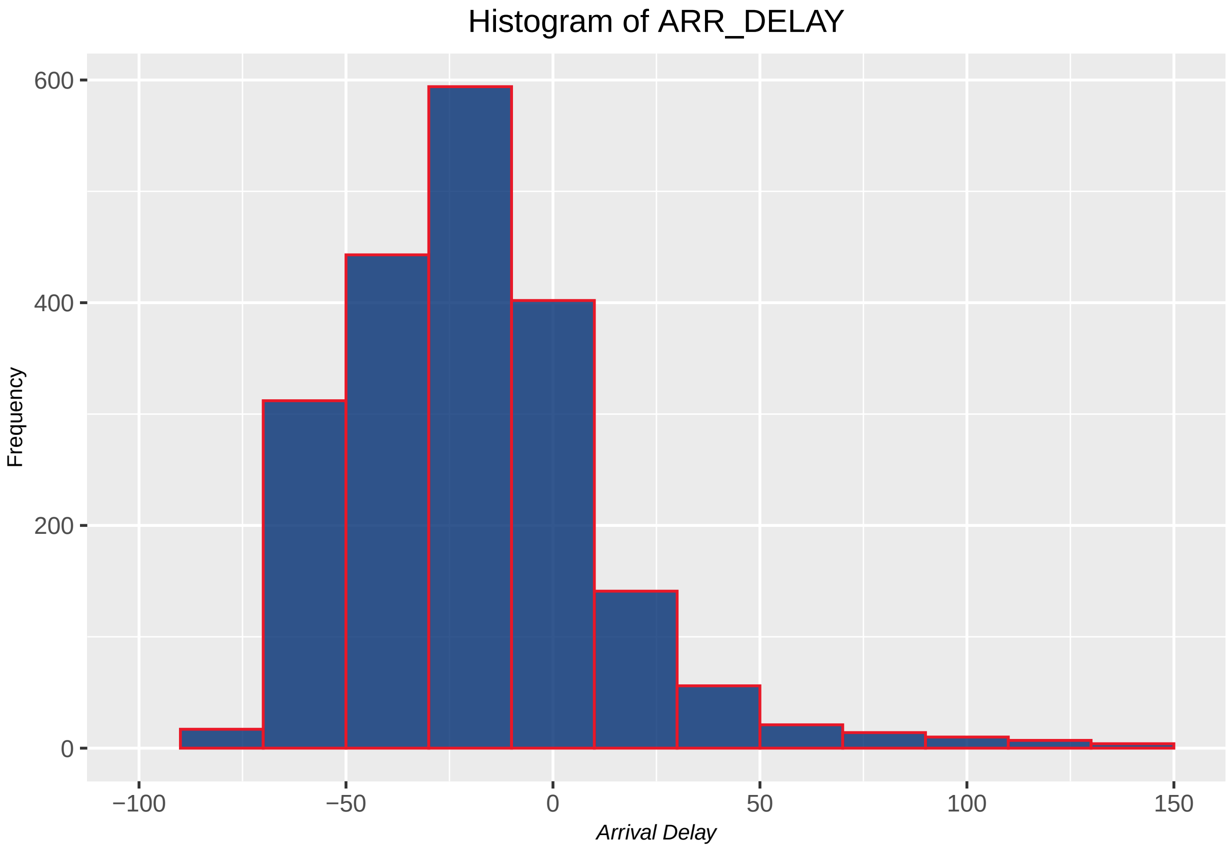
Overall, the categorical delay predictors I would think we could use are: Carrier Delay, NAS Delay, and Late Aircraft Delay

RESPONSE VARIABLE: ARRIVAL DELAY TIME

I just made it a different color so that when I scroll up to look at distributions I can easily tell the response from predictors (definitely can change at the end).

```
ggplot(data = flights, aes(x = ARR_DELAY)) +
  geom_histogram(binwidth = 20, fill = "#002D72", color = "#E81828", alpha = 0.8) +
  xlim(-100, 150) +
  labs(x = "Arrival Delay",
       y = "Frequency",
       title = "Histogram of ARR_DELAY") +
```

```r
  theme(plot.title = element_text(size = 12,hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.title.x.bottom = element_text(size = 8, face = "italic"),
        axis.title.y.left = element_text(size = 8))
```

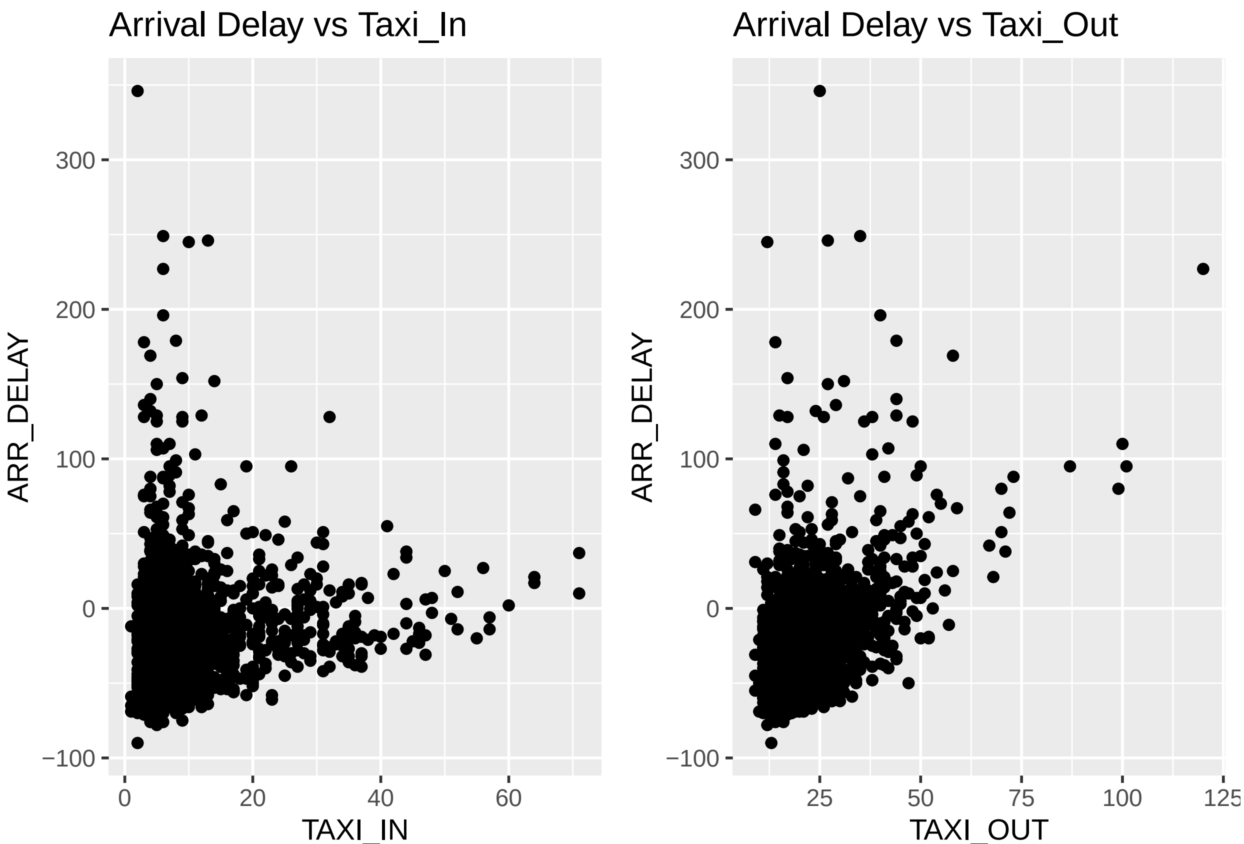



```r
# 2-parameter BC transformation
## can apply to GAM
```

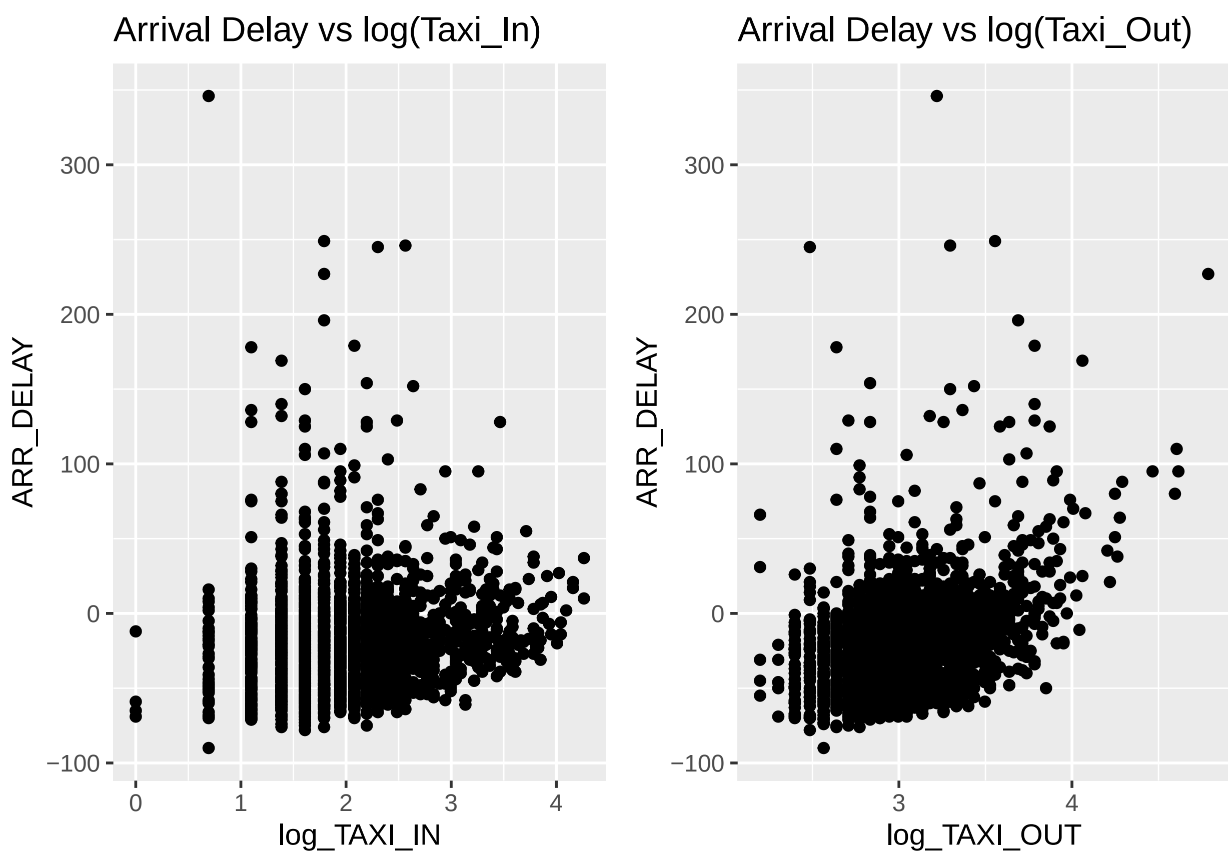
PREDICTORS VS RESPONSE

ARR_DELAY and TAXI_IN / TAXI_OUT

```r
p6 <- ggplot(data = flights, aes(y = ARR_DELAY, x = TAXI_IN)) +
  geom_point() +
  labs(title = "Arrival Delay vs Taxi_In")

p7 <- ggplot(data = flights, aes(y = ARR_DELAY, x = TAXI_OUT)) +
  geom_point() +
  labs(title = "Arrival Delay vs Taxi_Out")

grid.arrange(p6,p7, nrow = 1)
```

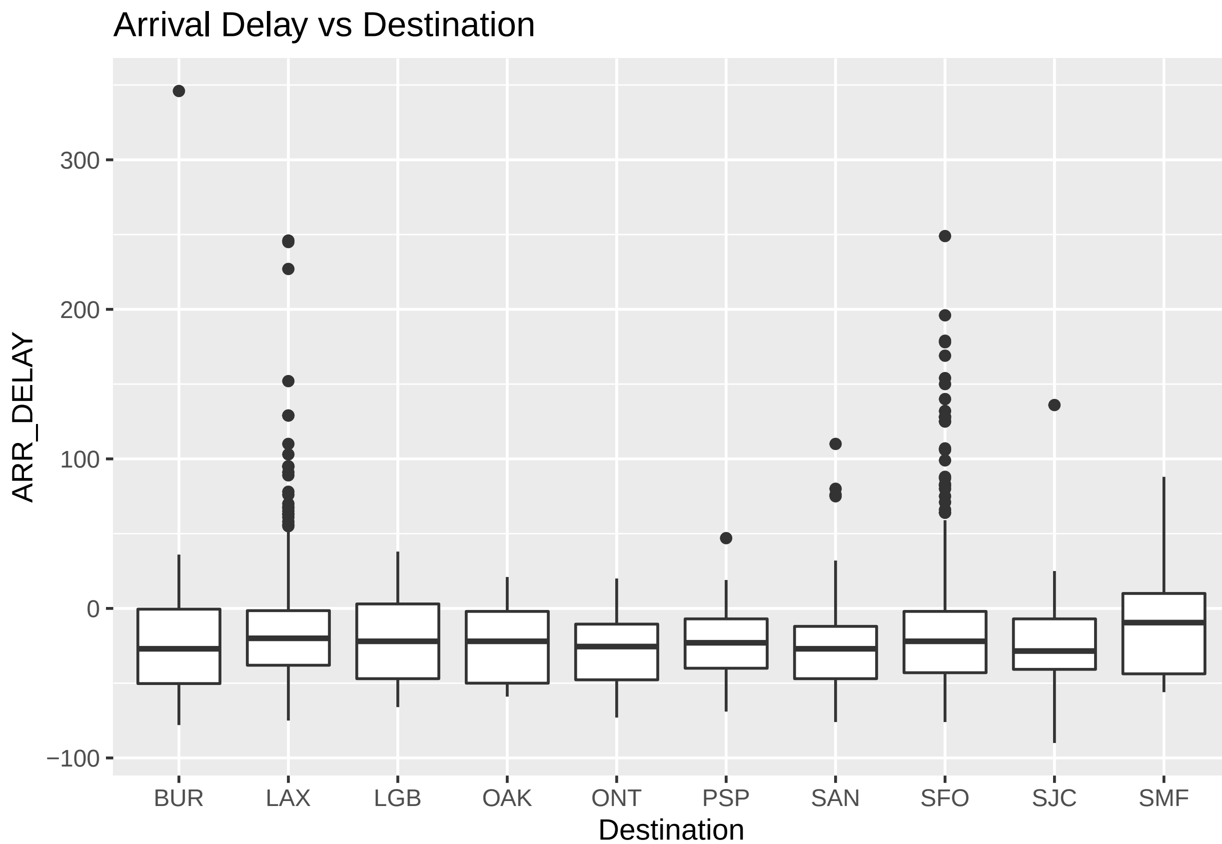



```r
plog6 <- ggplot(data = flights, aes(y = ARR_DELAY, x = log_TAXI_IN)) +
  geom_point() +
  labs(title = "Arrival Delay vs log(Taxi_In)")

plog7 <- ggplot(data = flights, aes(y = ARR_DELAY, x = log_TAXI_OUT)) +
  geom_point() +
  labs(title = "Arrival Delay vs log(Taxi_Out)")
grid.arrange(plog6,plog7, nrow = 1)
```



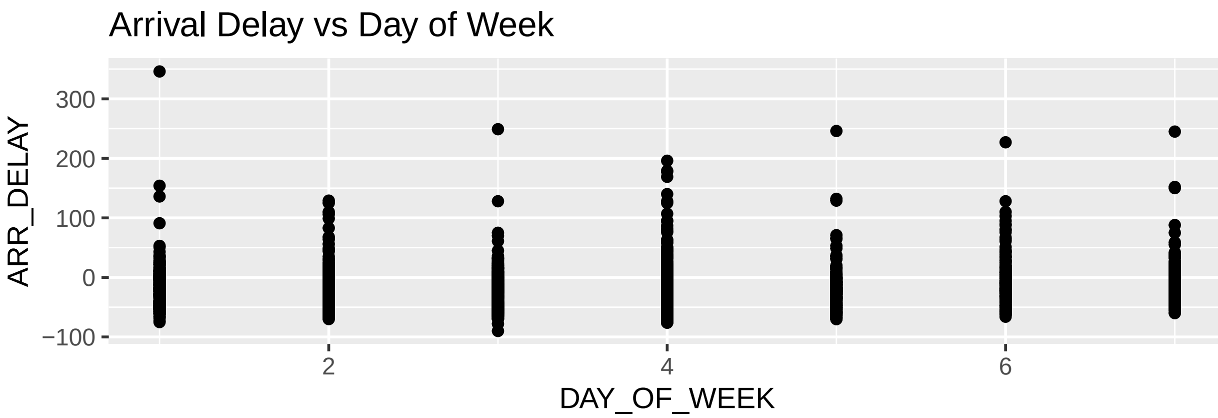
These plots above suggest that we may want to transform the variables at some point.

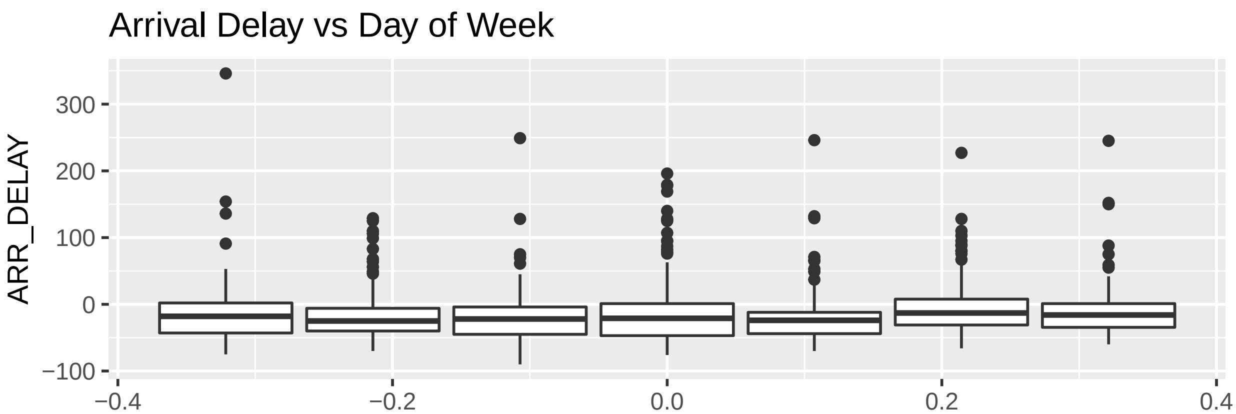
```
ggplot(data = flights, aes(y = ARR_DELAY, x = DEST)) +
  geom_boxplot() +
  labs(x = "Destination",
       title = "Arrival Delay vs Destination")
```



ARR_DELAY and DAY_OF_WEEK

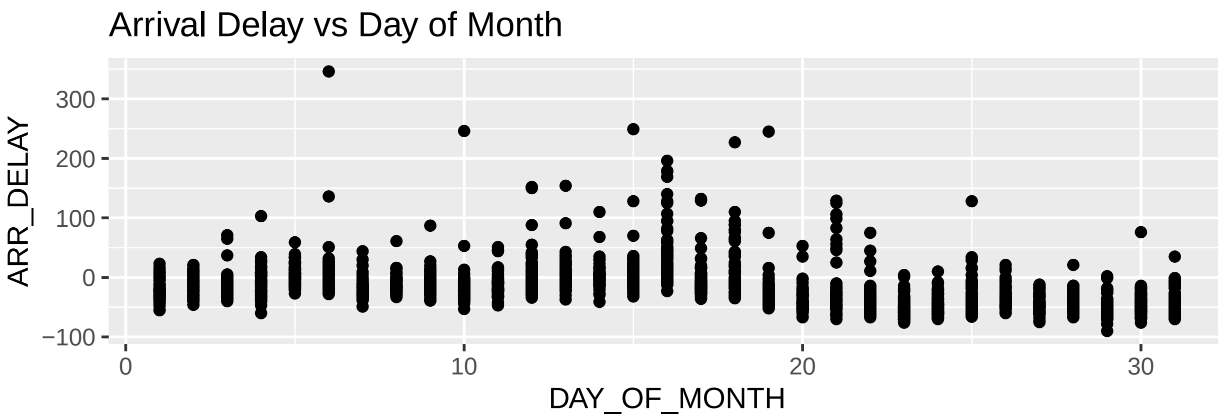
```
p8 <- ggplot(data = flights, aes(y = ARR_DELAY, x = DAY_OF_WEEK)) +
  geom_point() +
  labs(title = "Arrival Delay vs Day of Week")

p9 <- ggplot(data = flights, aes(y = ARR_DELAY, group = DAY_OF_WEEK)) +
  geom_boxplot() +
  labs(title = "Arrival Delay vs Day of Week")

grid.arrange(p8,p9, nrow = 2)
```

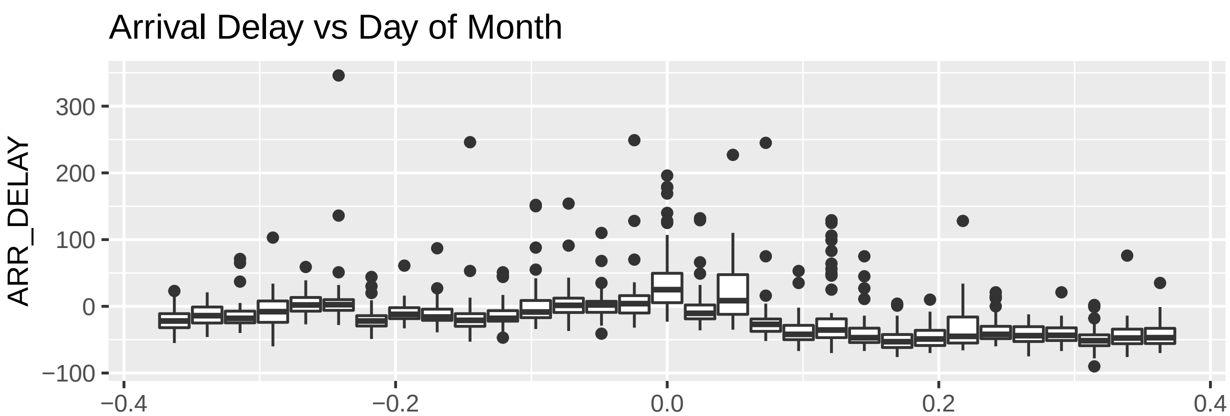




ARR_DELAY and DAY_OF_MONTH

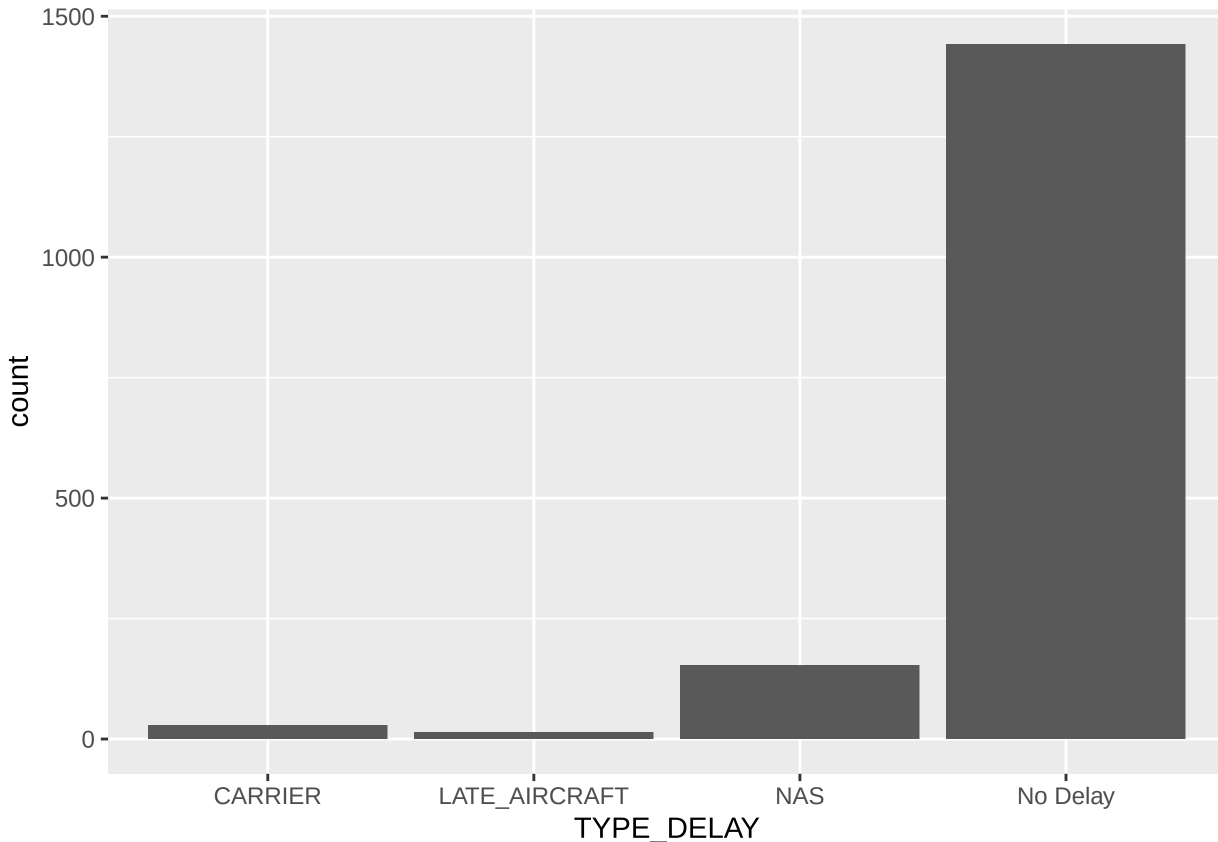
```
p10 <- ggplot(data = flights, aes(y = ARR_DELAY, x = DAY_OF_MONTH)) +
  geom_point() +
  labs(title = "Arrival Delay vs Day of Month")

p11 <- ggplot(data = flights, aes(y = ARR_DELAY, group = DAY_OF_MONTH)) +
  geom_boxplot() +
  labs(title = "Arrival Delay vs Day of Month")

grid.arrange(p10, p11, nrow = 2)
```





Further Data Cleaning

```r
# take only SFO/LAX since all 4 carriers fly there
flights <- flights %>%
  filter(DEST == "SFO" | DEST == "LAX") %>%
  mutate(TYPE_DELAY = case_when(NAS_DELAY == 1 ~ "NAS",
                                CARRIER_DELAY == 1 ~ "CARRIER",
                                LATE_AIRCRAFT_DELAY == 1 ~ "LATE_AIRCRAFT",
                                TRUE ~ "No Delay"))

ggplot(data = flights, aes(x = TYPE_DELAY)) +
  geom_bar()
```




```
unique(flights$TYPE_DELAY)
```

```
## [1] "No Delay"      "NAS"           "LATE_AIRCRAFT" "CARRIER"
```

SPLITTING DATA

```
set.seed(1234)
flights <- flights %>%
  mutate(id = row_number())
train <- flights %>%
  sample_frac(0.8)
test <- anti_join(flights, train, by = "id")
```

LINEAR MODELS

Variables that I think we could explore: department delay time, days of month, days of week, taxi-in, taxi-out, destination, Carrier Delay, NAS Delay, and Late Aircraft Delay.

Full Model

```
lm.01 <- lm(ARR_DELAY ~ DEP_DELAY + DAY_OF_WEEK + OP_CARRIER + DEST + CRS_DEP_TIME + CRS_ARR_TIME + log

#plot(lm.01)
#summary(lm.01)
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:patchwork':
##
##     area

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
step_model <- stepAIC(lm.01, direction = "backward", trace = FALSE)
#summary(step_model)

lm.02 <- lm(ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME  + log_TAXI_OUT + log_TAXI_IN + TYP
#summary(lm.02)
#anova(step_model, lm.02)

lm.03 <- lm(ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME  + log_TAXI_OUT + log_TAXI_IN + TYP

#anova(lm.02, lm.03)


log_linear_model <- lm(ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME  + log_TAXI_OUT + log_TA


anova(lm.03, log_linear_model)
```

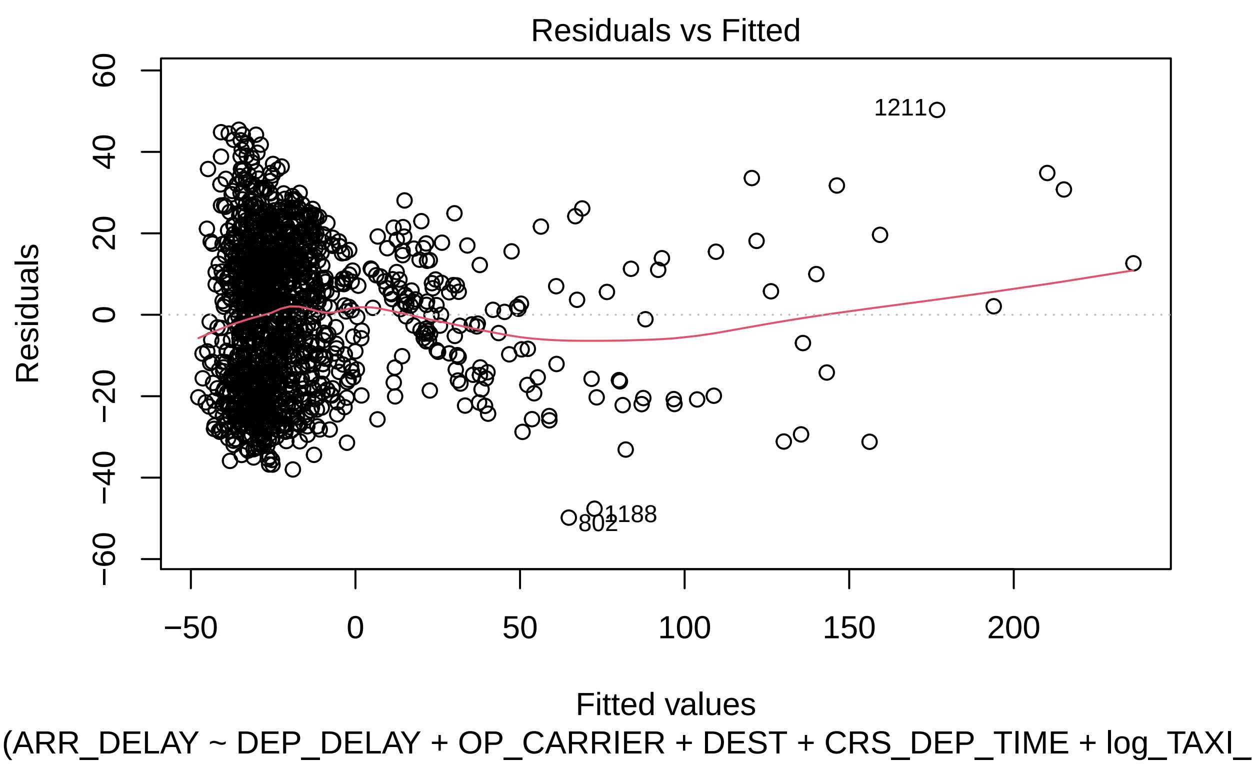
```
## Analysis of Variance Table
##
## Model 1: ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME + log_TAXI_OUT +
##     log_TAXI_IN + TYPE_DELAY + OP_CARRIER:DEST + DEST:log_TAXI_IN
## Model 2: ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME + log_TAXI_OUT +
##     log_TAXI_IN + TYPE_DELAY + OP_CARRIER:DEST + DEST:log_TAXI_IN +
##     log_TAXI_OUT:DEP_DELAY
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1   1294 427667
## 2   1293 425449  1      2218 6.7408 0.00953 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

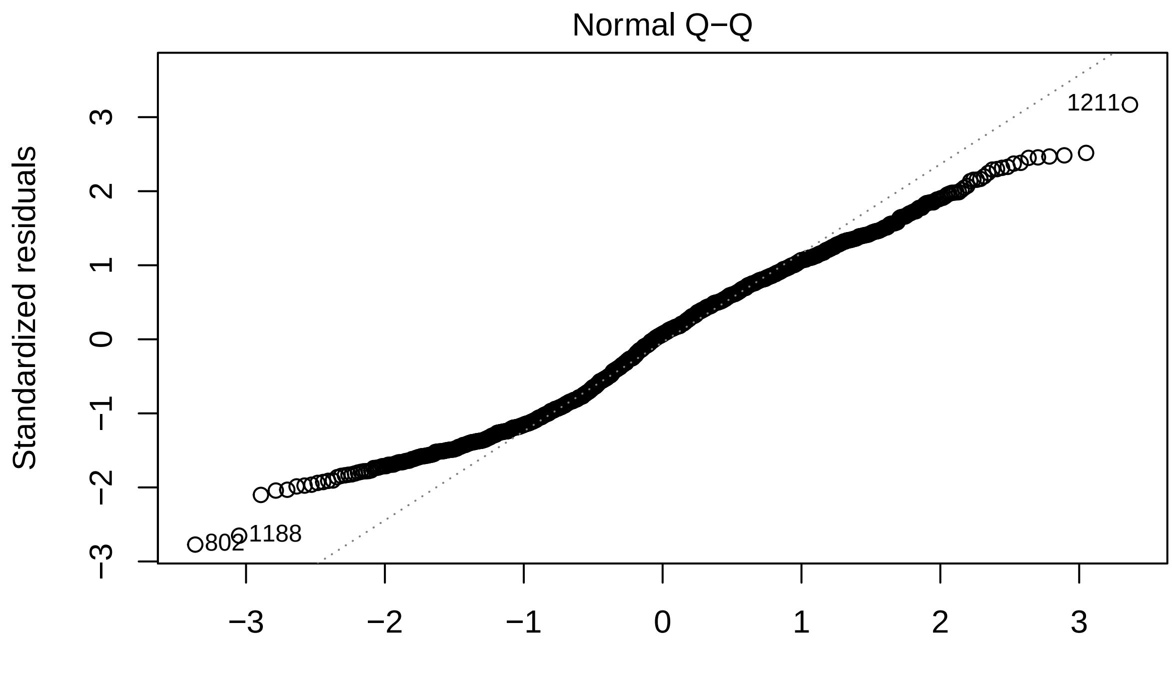
```r
summary(log_linear_model)
```

```
##
## Call:
## lm(formula = ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME +
##     log_TAXI_OUT + log_TAXI_IN + TYPE_DELAY + OP_CARRIER:DEST +
##     DEST:log_TAXI_IN + log_TAXI_OUT:DEP_DELAY, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.817 -15.330   1.198  13.897  50.301
##
## Coefficients:
```

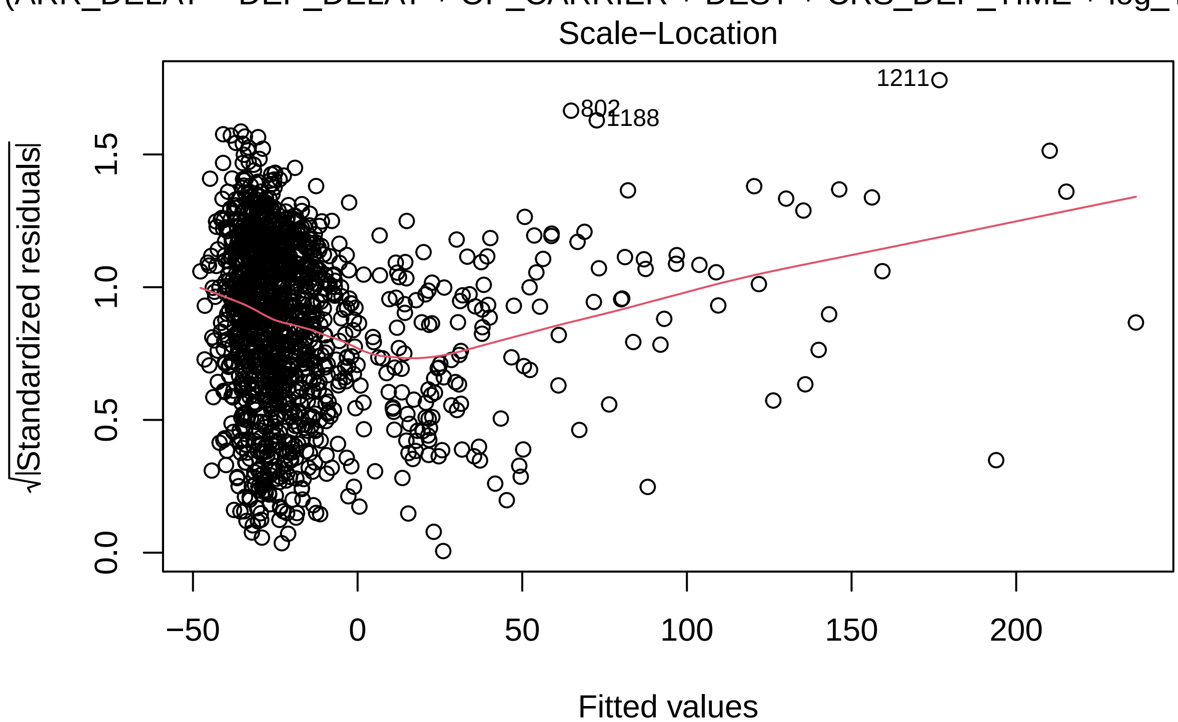
```
##                          Estimate Std. Error t value Pr(>|t|)    
## (Intercept)            -81.901399   7.262197 -11.278  < 2e-16 ***
## DEP_DELAY                0.524207   0.141256   3.711 0.000215 ***
## OP_CARRIERAS            -4.458249   2.087733  -2.135 0.032912 *  
## OP_CARRIERB6             5.045463   1.669832   3.022 0.002564 ** 
## OP_CARRIERDL            -1.493672   1.717998  -0.869 0.384775    
## DESTSFO                  9.893184   4.366882   2.266 0.023647 *  
## CRS_DEP_TIME            -0.004364   0.001070  -4.081 4.77e-05 ***
## log_TAXI_OUT            20.610508   1.617078  12.746  < 2e-16 ***
## log_TAXI_IN              8.433233   1.057392   7.976 3.32e-15 ***
## TYPE_DELAYLATE_AIRCRAFT -3.973566   6.537317  -0.608 0.543408    
## TYPE_DELAYNAS           24.019795   4.598524   5.223 2.05e-07 ***
## TYPE_DELAYNo Delay     -15.676745   4.540377  -3.453 0.000573 ***
## OP_CARRIERAS:DESTSFO     6.630276   3.374581   1.965 0.049655 *  
## OP_CARRIERB6:DESTSFO    -4.199151   2.858830  -1.469 0.142121    
## OP_CARRIERDL:DESTSFO    -1.424895   2.900122  -0.491 0.623282    
## DESTSFO:log_TAXI_IN     -5.261163   1.951509  -2.696 0.007110 ** 
## DEP_DELAY:log_TAXI_OUT   0.113332   0.043651   2.596 0.009530 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.14 on 1293 degrees of freedom
## Multiple R-squared:  0.7376, Adjusted R-squared:  0.7344 
## F-statistic: 227.2 on 16 and 1293 DF,  p-value: < 2.2e-16
```

```r
plot(log_linear_model)
```

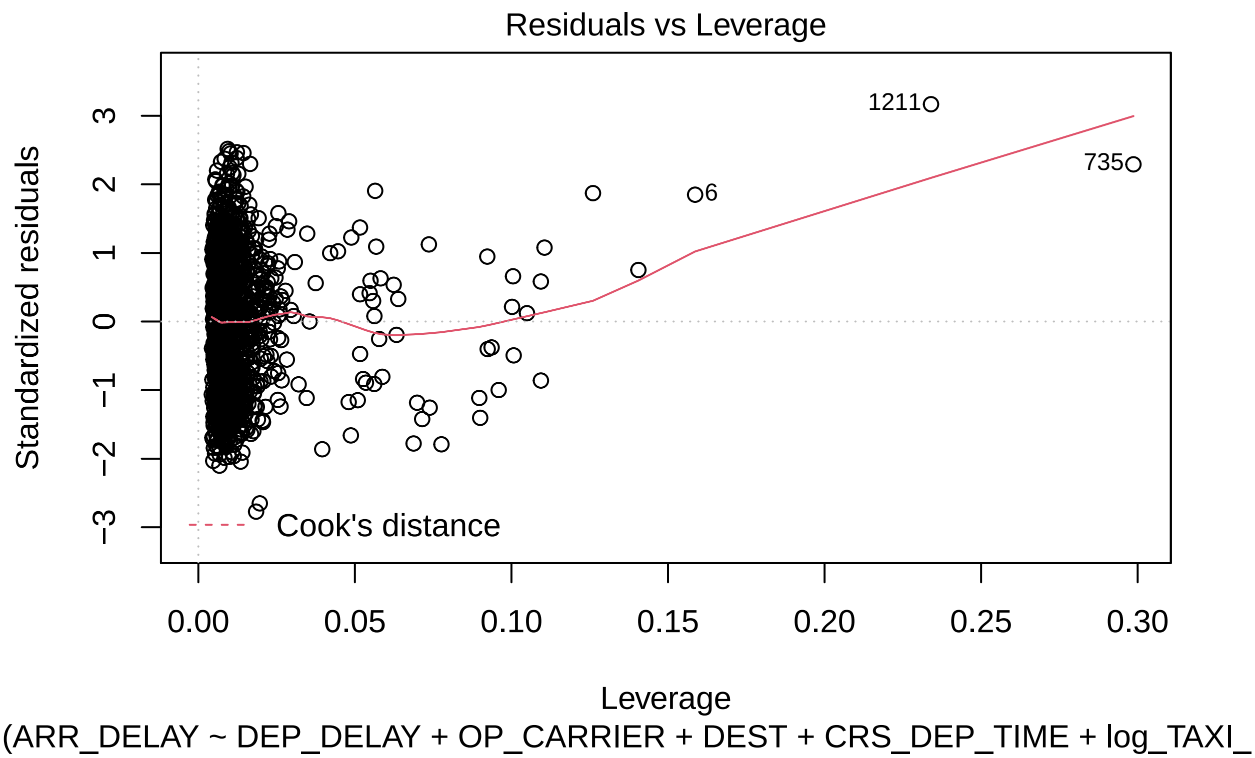




(ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME + log_TAXI_



(ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME + log_TAXI_




```r
## SIGNIFICANT INTERACTIONS
#OP_CARRIER:DEST
#DEST:log_TAXI_IN
#CRS_DEP_TIME:DEST (***** makes zero intuitive sense - might not wanna do this)
#CRS_ARR_TIME:log_TAXI_IN
#log_TAXI_OUT:DEP_DELAY

#log_TAXI_OUT:CRS_DEP_TIME (verrrrrry close to 0.05)
library(broom)
log_linear_preds <- predict(log_linear_model, test)
log_linear_MSE <- sum((log_linear_preds-test$ARR_DELAY)^2, na.rm=T)/328
log_linear_MSE
```

```
## [1] 333.8962
```

```r
min(train$ARR_DELAY, na.rm = TRUE)
```

```
## [1] -74
```

```r
train$adj_ARR_DELAY <- train$ARR_DELAY + 75
min(train$adj_ARR_DELAY, na.rm = T)
```

```
## [1] 1
```

```r
adj_log_linear_model <- lm(adj_ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME  + log_TAXI_OUT

library(EnvStats)
```

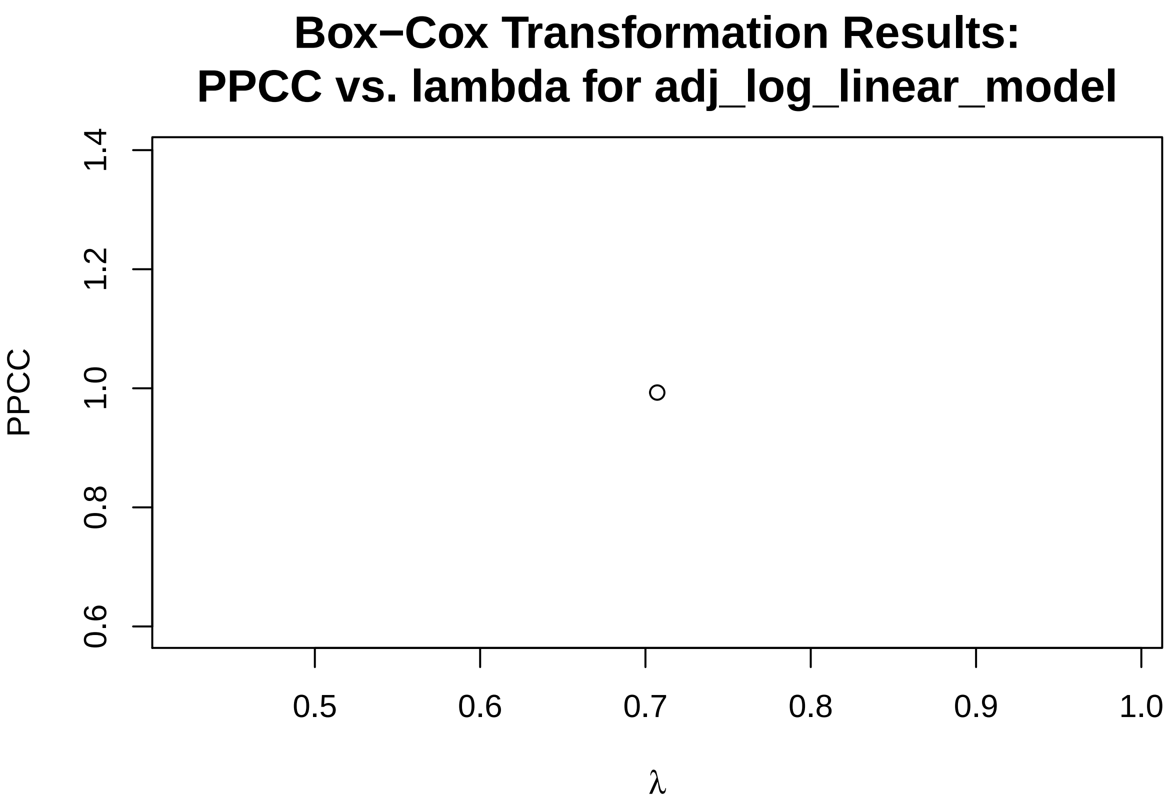
```
##
## Attaching package: 'EnvStats'

## The following object is masked from 'package:MASS':
##
```

```
##     boxcox

## The following objects are masked from 'package:stats':
##
##     predict, predict.lm

## The following object is masked from 'package:base':
##
##     print.default
```

```
bc_model <- boxcox(adj_log_linear_model, optimize = TRUE)
bc_lambda <- bc_model$lambda
bc_lambda
```

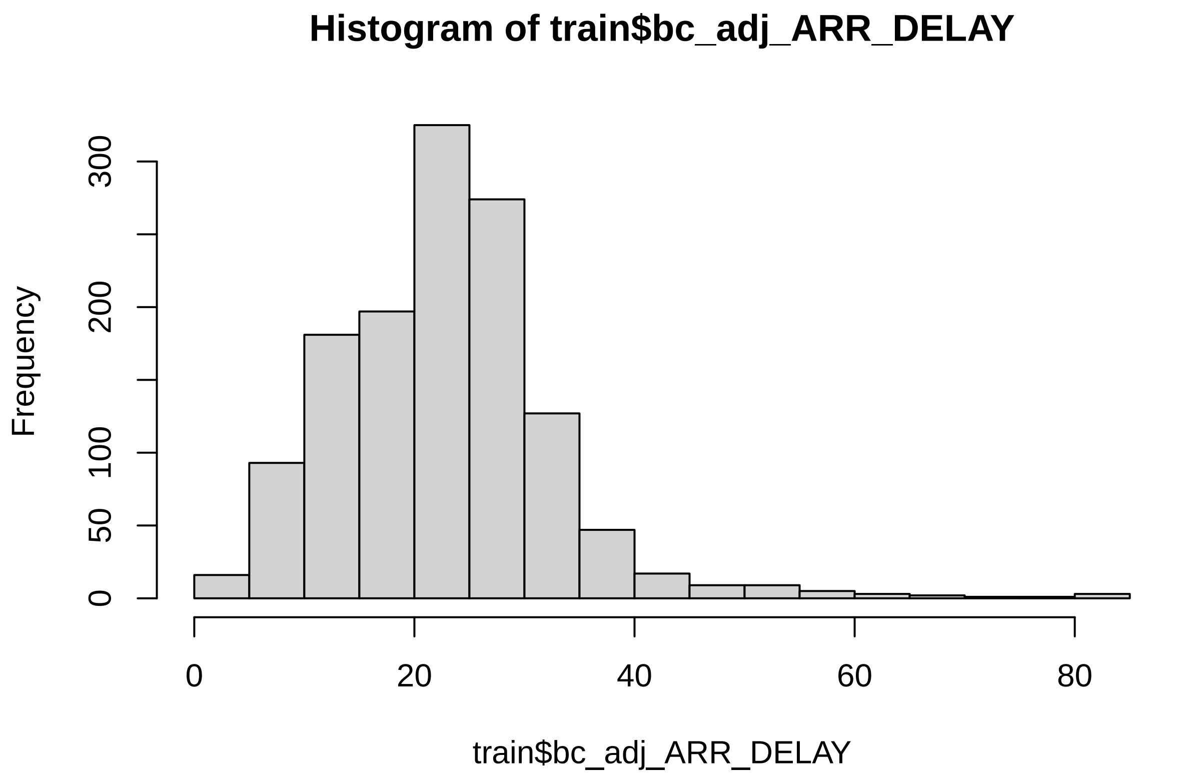
```
## [1] 0.707128
```

```
plot(bc_model)
```

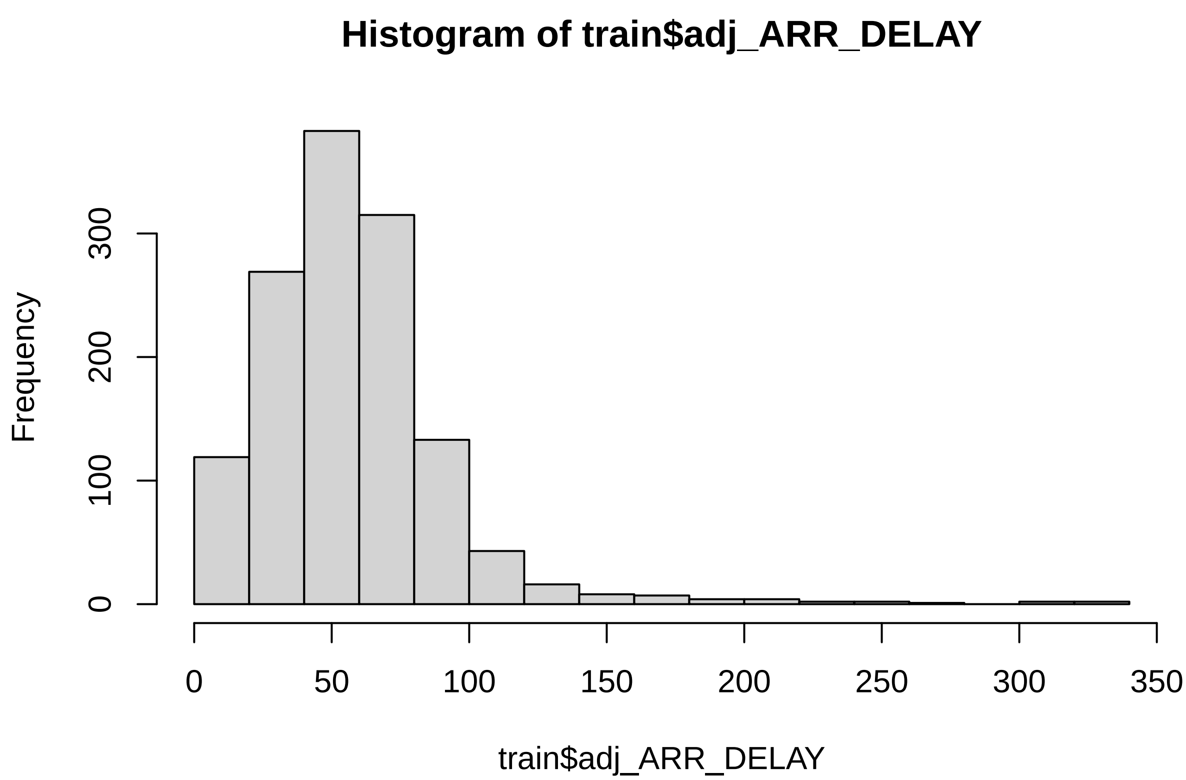



```
# library(car)
# hello <- boxCox(log_linear_model, family="yjPower", plotit = T)
# hello
```

```
train <- train %>%
  mutate(bc_adj_ARR_DELAY = ((adj_ARR_DELAY^bc_lambda) - 1)/bc_lambda)

hist(train$bc_adj_ARR_DELAY)
```




```r
hist(train$adj_ARR_DELAY)
```



First, let's just fit a full linear model with all the variables we would like to explore.

```r
full_model <- lm(ARR_DELAY ~ DAY_OF_MONTH +
                   DAY_OF_WEEK +
```

```r
                   TAXI_IN +
                   TAXI_OUT +
                   DEST +
                   DEP_DELAY +
                   CARRIER_DELAY +
                   NAS_DELAY +
                   LATE_AIRCRAFT_DELAY, data = train)

summary(full_model)
```

```
##
## Call:
## lm(formula = ARR_DELAY ~ DAY_OF_MONTH + DAY_OF_WEEK + TAXI_IN +
##     TAXI_OUT + DEST + DEP_DELAY + CARRIER_DELAY + NAS_DELAY +
##     LATE_AIRCRAFT_DELAY, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -45.009 -10.360  -1.363   9.236  46.688
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -21.90047    1.65700 -13.217   <2e-16 ***
## DAY_OF_MONTH         -1.33012    0.04462 -29.813   <2e-16 ***
## DAY_OF_WEEK          -0.14322    0.20971  -0.683    0.495
## TAXI_IN               0.57735    0.04795  12.041   <2e-16 ***
## TAXI_OUT              0.73384    0.04625  15.866   <2e-16 ***
## DESTSFO              -0.31121    0.83728  -0.372    0.710
## DEP_DELAY             0.91972    0.02068  44.477   <2e-16 ***
## CARRIER_DELAY         1.86709    2.46556   0.757    0.449
## NAS_DELAY            32.10558    1.58215  20.292   <2e-16 ***
## LATE_AIRCRAFT_DELAY   0.55130    3.05739   0.180    0.857
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.2 on 1300 degrees of freedom
## Multiple R-squared:  0.8382, Adjusted R-squared:  0.8371
## F-statistic: 748.4 on 9 and 1300 DF,  p-value: < 2.2e-16
```

```r
full_model_preds <- predict(full_model, test)
linear_MSE <- sum((full_model_preds-test$ARR_DELAY)^2, na.rm=T)/328
linear_MSE
```

```
## [1] 215.6589
```

```r
full_model2 <- lm(ARR_DELAY ~
                    DEP_DELAY +
                    DAY_OF_WEEK +
                    OP_CARRIER +
                    DEST +
                    CRS_DEP_TIME +
                    CRS_ARR_TIME +
                    TAXI_OUT +
                    TAXI_IN +
                    TYPE_DELAY, train)
```

Select Model with AIC

```
library(MASS)
step_model <- stepAIC(full_model2, trace = FALSE)
summary(step_model)
```

```
##
## Call:
## lm(formula = ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME +
##     CRS_ARR_TIME + TAXI_OUT + TAXI_IN + TYPE_DELAY, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.067 -16.149   1.368  13.672  49.316
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -2.410e+01  4.895e+00  -4.924 9.56e-07 ***
## DEP_DELAY                 8.732e-01  2.704e-02  32.294  < 2e-16 ***
## OP_CARRIERAS             -1.570e+00  1.658e+00  -0.947 0.344035
## OP_CARRIERB6              1.918e+00  1.365e+00   1.405 0.160140
## OP_CARRIERDL             -2.304e+00  1.383e+00  -1.665 0.096130 .
## DESTSFO                  -1.833e+00  1.082e+00  -1.694 0.090417 .
## CRS_DEP_TIME             -4.231e-03  1.096e-03  -3.859 0.000119 ***
## CRS_ARR_TIME             -1.525e-03  8.762e-04  -1.741 0.081976 .
## TAXI_OUT                  8.668e-01  6.090e-02  14.234  < 2e-16 ***
## TAXI_IN                   4.700e-01  6.132e-02   7.665 3.50e-14 ***
## TYPE_DELAYLATE_AIRCRAFT -2.223e+00  6.499e+00  -0.342 0.732329
## TYPE_DELAYNAS             2.509e+01  4.485e+00   5.594 2.71e-08 ***
## TYPE_DELAYNo Delay       -1.360e+01  4.441e+00  -3.063 0.002233 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.16 on 1297 degrees of freedom
## Multiple R-squared:  0.7361, Adjusted R-squared:  0.7337
## F-statistic: 301.5 on 12 and 1297 DF,  p-value: < 2.2e-16
```

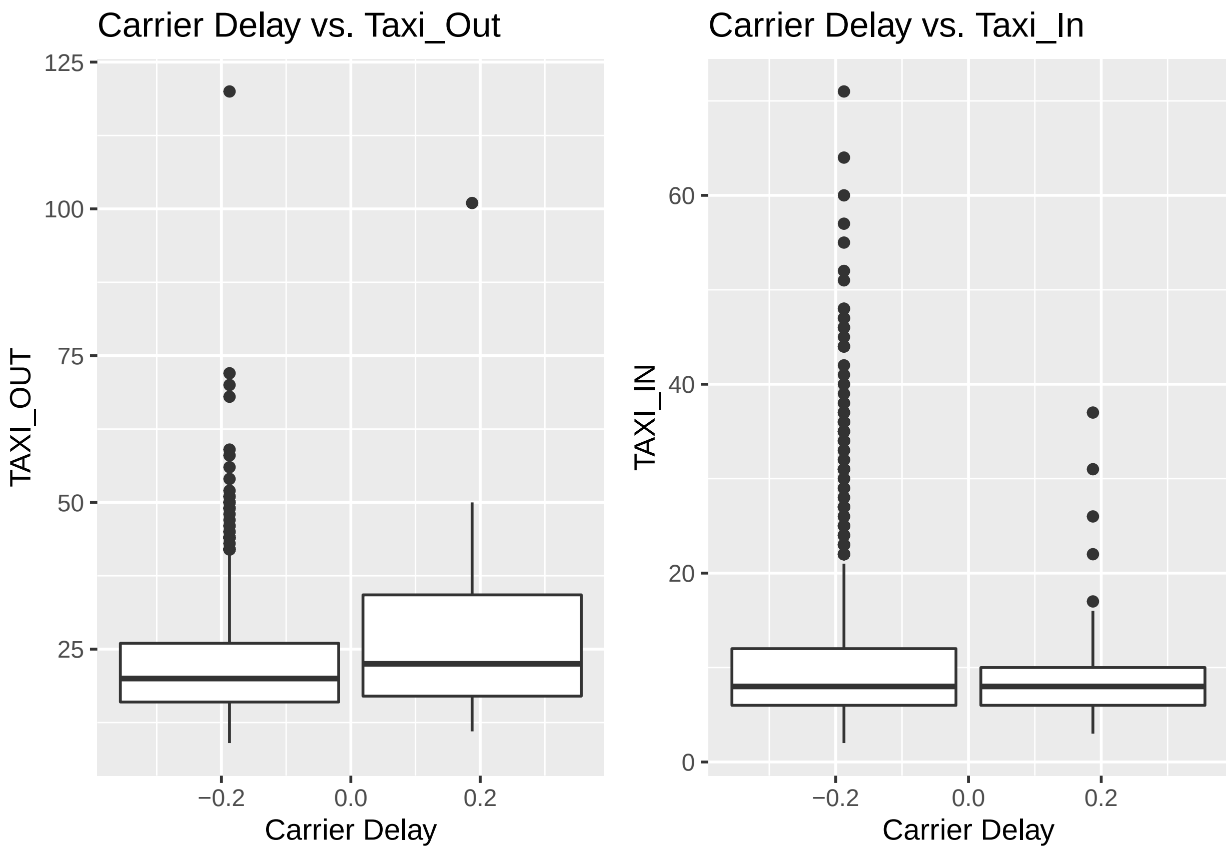
The only variables that were removed were DAY_OF_WEEK and LATE_AIRCRAFT_DELAY. Let's continue using the step_model then.

Interactions

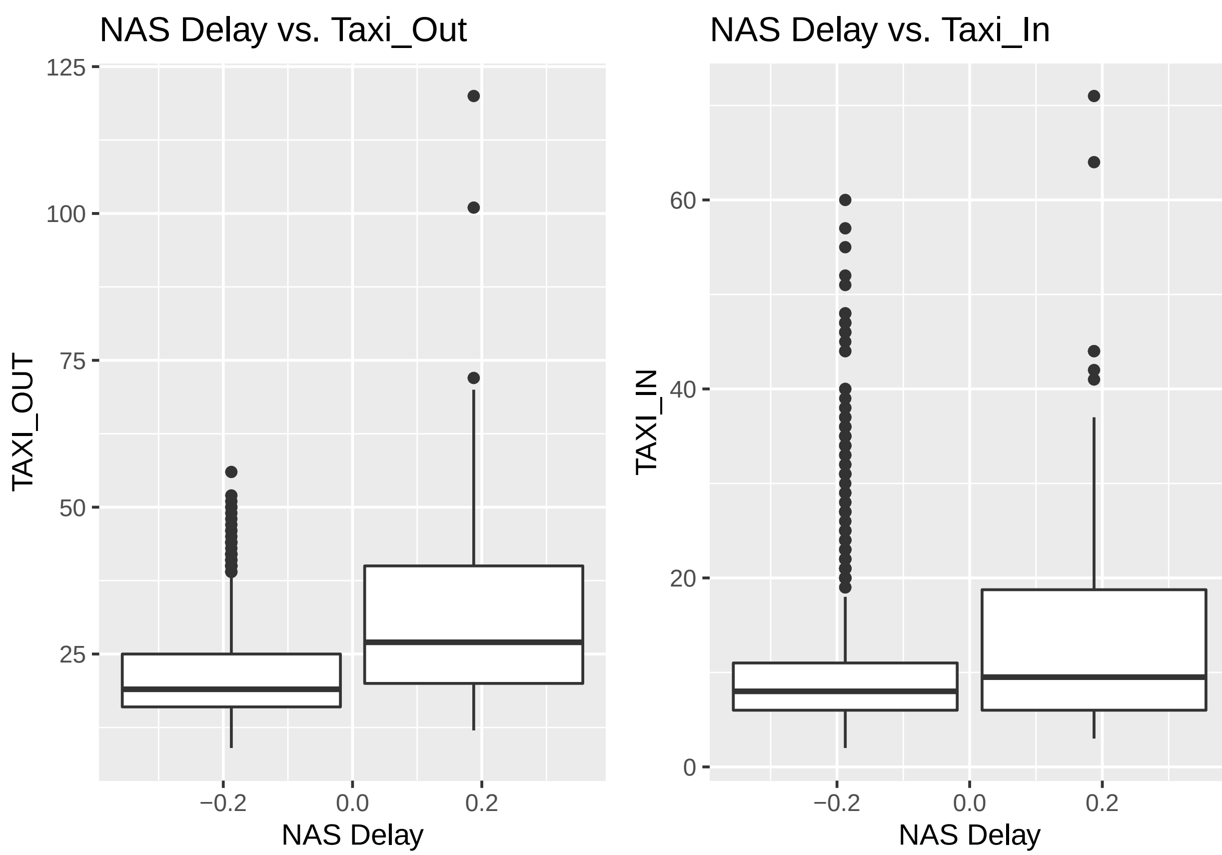
Because there are so many levels to Destination, I don't know if we should necessarily include an interaction with this categorical variable. My suggestion would be to find interactions with carrier_delay and nas_delay.

```
p12 <- ggplot(data = train, aes(group = CARRIER_DELAY, y = TAXI_OUT)) +
  geom_boxplot() +
  labs(title = "Carrier Delay vs. Taxi_Out",
       x = "Carrier Delay")

p13 <- ggplot(data = train, aes(group = CARRIER_DELAY, y = TAXI_IN)) +
  geom_boxplot() +
  labs(title = "Carrier Delay vs. Taxi_In",
       x = "Carrier Delay")
```

```r
grid.arrange(p12, p13, nrow = 1)
```




```r
p14 <- ggplot(data = train, aes(group = NAS_DELAY, y = TAXI_OUT)) +
  geom_boxplot() +
  labs(title = "NAS Delay vs. Taxi_Out",
       x = "NAS Delay")

p15 <- ggplot(data = train, aes(group = NAS_DELAY, y = TAXI_IN)) +
  geom_boxplot() +
  labs(title = "NAS Delay vs. Taxi_In",
       x = "NAS Delay")

grid.arrange(p14, p15, nrow = 1)
```




```
step_model
```

```
##
## Call:
## lm(formula = ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME +
##     CRS_ARR_TIME + TAXI_OUT + TAXI_IN + TYPE_DELAY, data = train)
##
## Coefficients:
##             (Intercept)                DEP_DELAY             OP_CARRIERAS
##              -24.104903                 0.873248                -1.569862
##            OP_CARRIERB6             OP_CARRIERDL                  DESTSFO
##                1.918141                -2.303676                -1.832585
##            CRS_DEP_TIME             CRS_ARR_TIME                 TAXI_OUT
##               -0.004231                -0.001525                 0.866771
##                 TAXI_IN  TYPE_DELAYLATE_AIRCRAFT            TYPE_DELAYNAS
##                0.469992                -2.223475                25.087061
##      TYPE_DELAYNo Delay
##              -13.604813
```

From what I'm seeing in the plots above, there could be an interaction between taxi_out and carrier_delay. There also seems to be an interaction between NAS delay and taxi_out as well as a possible one between NAS delay and taxi_in. Let's test these three interactions below.

```
# carrier vs taxi out
# interaction1 <- lm(ARR_DELAY ~
#                      DEP_DELAY +
#                      OP_CARRIER +
#                      DEST +
```

```r
#                       CRS_DEP_TIME +
#                       CRS_ARR_TIME +
#                       TAXI_OUT +
#                       TAXI_IN +
#                       TYPE_DELAY +
#
#                        , data = train)
```

```r
#anova(step_model, interaction1)
```

```r
#anova(step_model, interaction2)
```

```r
#anova(step_model, interaction3)
```

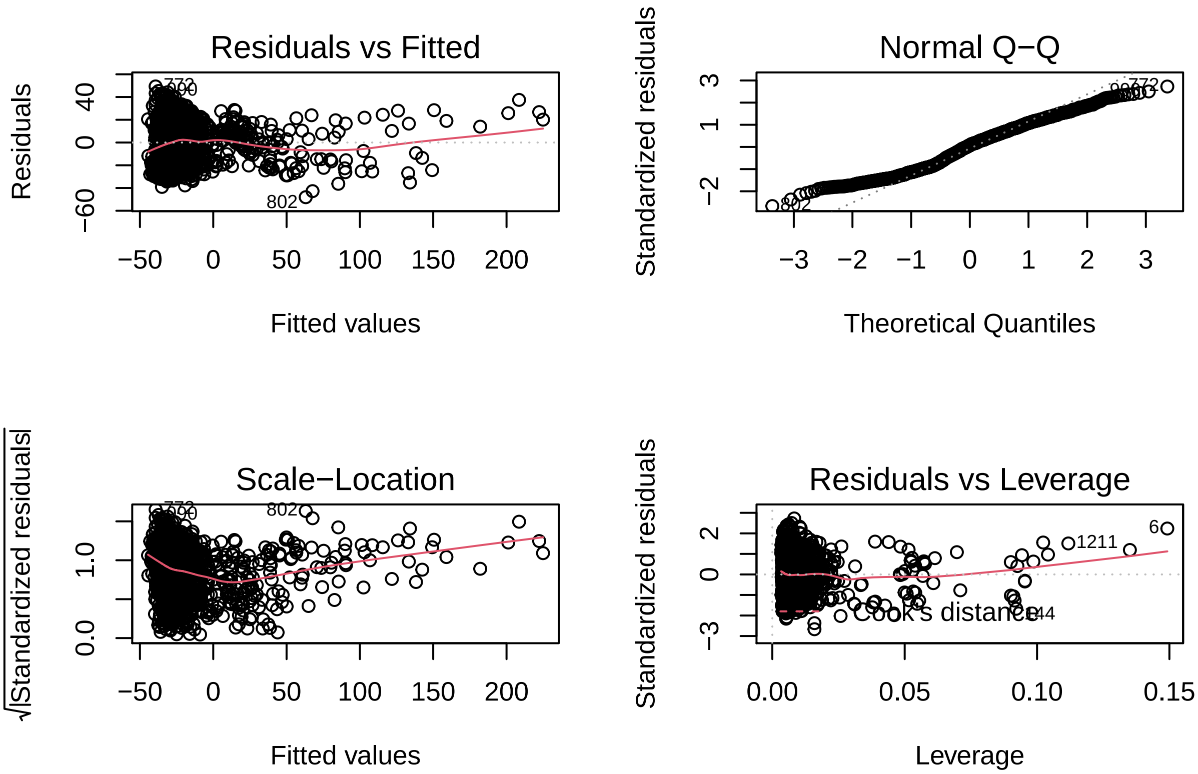
It actually seems that interaction3: NAS_DELAY and TAXI_IN is the only interaction that is statistically significant in predicting ARR_DELAY. Let's make this model our current model:

Final Linear Model

```r
#EDIT!
current_model <- step_model

summary(current_model)
```

```
##
## Call:
## lm(formula = ARR_DELAY ~ DEP_DELAY + OP_CARRIER + DEST + CRS_DEP_TIME +
##     CRS_ARR_TIME + TAXI_OUT + TAXI_IN + TYPE_DELAY, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.067 -16.149   1.368  13.672  49.316
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -2.410e+01  4.895e+00  -4.924 9.56e-07 ***
## DEP_DELAY               8.732e-01  2.704e-02  32.294  < 2e-16 ***
## OP_CARRIERAS           -1.570e+00  1.658e+00  -0.947 0.344035
## OP_CARRIERB6            1.918e+00  1.365e+00   1.405 0.160140
## OP_CARRIERDL           -2.304e+00  1.383e+00  -1.665 0.096130 .
## DESTSFO                -1.833e+00  1.082e+00  -1.694 0.090417 .
## CRS_DEP_TIME           -4.231e-03  1.096e-03  -3.859 0.000119 ***
## CRS_ARR_TIME           -1.525e-03  8.762e-04  -1.741 0.081976 .
## TAXI_OUT                8.668e-01  6.090e-02  14.234  < 2e-16 ***
## TAXI_IN                 4.700e-01  6.132e-02   7.665 3.50e-14 ***
## TYPE_DELAYLATE_AIRCRAFT -2.223e+00  6.499e+00  -0.342 0.732329
## TYPE_DELAYNAS           2.509e+01  4.485e+00   5.594 2.71e-08 ***
## TYPE_DELAYNo Delay     -1.360e+01  4.441e+00  -3.063 0.002233 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.16 on 1297 degrees of freedom
## Multiple R-squared:  0.7361, Adjusted R-squared:  0.7337
## F-statistic: 301.5 on 12 and 1297 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2,2))
plot(current_model)
```



The diagnostic plots above suggest that this model decently satisfies the necessary conditions to assume a linear regression.

Response (Box-Cox) Transformation

```r
# library(EnvStats)
#
# bc_model <- boxcox(current_model, optimize = TRUE)
# bc_lambda <- bc_model$lambda
# bc_lambda
# plot(bc_model)
```

```r
# add Box-Cox transform to data
# train_data <- train_data %>%
#   mutate(bc_R_moment_1 = ((R_moment_1^bc_lambda) - 1)/bc_lambda)
#
# hist(train_data$bc_R_moment_1)
```

Test Error

```r
lm_preds <- predict(current_model, test)
linear_model_MSE <- sum((test$ARR_DELAY - lm_preds)^2, na.rm=T)/328
linear_model_MSE
```

```
## [1] 322.4588
```

GAM MODEL

Initial Model

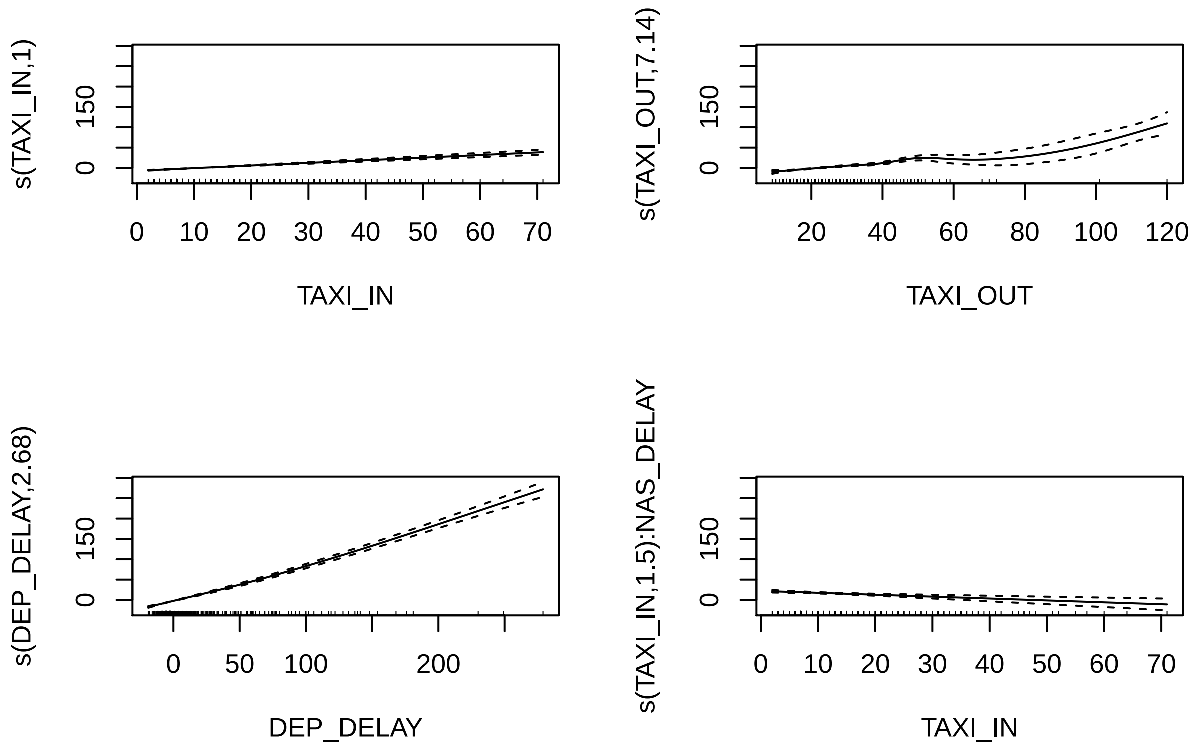
fit a gam model with numerical variables on a smoothing spline and including the interaction between NAS_DELAY and TAXI_IN

```
gam00 <- gam(ARR_DELAY ~ DAY_OF_MONTH +
                   DAY_OF_WEEK +
                   s(TAXI_IN) +
                   s(TAXI_OUT) +
                   DEST +
                   s(DEP_DELAY) +
                   CARRIER_DELAY +
                   NAS_DELAY +
                   LATE_AIRCRAFT_DELAY +
                   s(TAXI_IN, by = NAS_DELAY), data = train)

summary(gam00)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## ARR_DELAY ~ DAY_OF_MONTH + DAY_OF_WEEK + s(TAXI_IN) + s(TAXI_OUT) +
##     DEST + s(DEP_DELAY) + CARRIER_DELAY + NAS_DELAY + LATE_AIRCRAFT_DELAY +
##     s(TAXI_IN, by = NAS_DELAY)
##
## Parametric coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           2.7005     1.2425   2.173   0.0299 *
## DAY_OF_MONTH         -1.3525     0.0444 -30.462   <2e-16 ***
## DAY_OF_WEEK          -0.1165     0.2071  -0.563   0.5738
## DESTSFO              -0.2879     0.8284  -0.348   0.7282
## CARRIER_DELAY         3.7611     2.4734   1.521   0.1286
## NAS_DELAY            17.2608     0.8215  21.011   <2e-16 ***
## LATE_AIRCRAFT_DELAY   2.9740     3.0656   0.970   0.3322
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                        edf Ref.df      F p-value
## s(TAXI_IN)           1.000  1.000 156.33  <2e-16 ***
## s(TAXI_OUT)          7.143  8.101  34.41  <2e-16 ***
## s(DEP_DELAY)         2.680  3.350 560.25  <2e-16 ***
## s(TAXI_IN):NAS_DELAY 1.500  1.500 133.42  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 43/44
## R-sq.(adj) =  0.842   Deviance explained = 84.5%
## GCV = 198.05  Scale est. = 195.21    n = 1310
```

```
par(mfrow = c(2,2))
```

```r
plot.gam(gam00, se=TRUE)
```



Checking Lineartiy

TAXI_IN and the interaction between NAS_DELAY and TAXI_IN may be linear

```r
gam01 <- gam(ARR_DELAY ~ DAY_OF_MONTH +
                   DAY_OF_WEEK +
                   TAXI_IN +
                   s(TAXI_OUT) +
                   DEST +
                   s(DEP_DELAY) +
                   CARRIER_DELAY +
                   NAS_DELAY +
                   LATE_AIRCRAFT_DELAY +
                   TAXI_IN*NAS_DELAY, data = train)

anova(gam00, gam01, test = "F")
```

```
## Analysis of Deviance Table
##
## Model 1: ARR_DELAY ~ DAY_OF_MONTH + DAY_OF_WEEK + s(TAXI_IN) + s(TAXI_OUT) +
##     DEST + s(DEP_DELAY) + CARRIER_DELAY + NAS_DELAY + LATE_AIRCRAFT_DELAY +
##     s(TAXI_IN, by = NAS_DELAY)
## Model 2: ARR_DELAY ~ DAY_OF_MONTH + DAY_OF_WEEK + TAXI_IN + s(TAXI_OUT) +
##     DEST + s(DEP_DELAY) + CARRIER_DELAY + NAS_DELAY + LATE_AIRCRAFT_DELAY +
##     TAXI_IN * NAS_DELAY
##   Resid. Df Resid. Dev          Df    Deviance      F    Pr(>F)
## 1    1289.5     252048
## 2    1289.5     252048 -2.4319e-06 -0.0012137 2.5567 1.472e-05 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

based on anova test, the model without smoothing splines on TAXI_IN and the interaction term is a better fit

More Anova

DAY_OF_WEEK and DEST have very high p-values, so let's try an anova test without including them

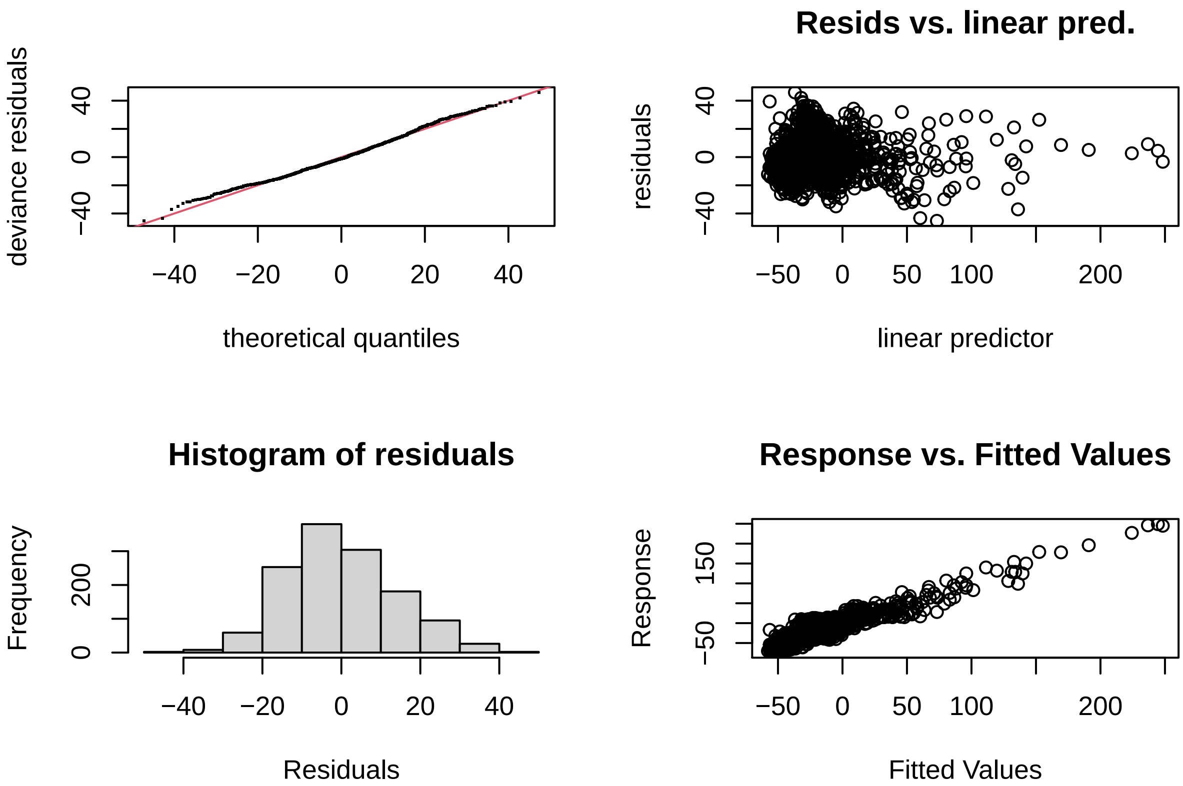
```
gam02 <- gam(ARR_DELAY ~ DAY_OF_MONTH +
                   TAXI_IN +
                   s(TAXI_OUT) +
                   s(DEP_DELAY) +
                   CARRIER_DELAY +
                   NAS_DELAY +
                   LATE_AIRCRAFT_DELAY +
                   TAXI_IN, by = NAS_DELAY, data = train)

anova(gam01, gam02, test = "F")
```

```
## Analysis of Deviance Table
##
## Model 1: ARR_DELAY ~ DAY_OF_MONTH + DAY_OF_WEEK + TAXI_IN + s(TAXI_OUT) +
##     DEST + s(DEP_DELAY) + CARRIER_DELAY + NAS_DELAY + LATE_AIRCRAFT_DELAY +
##     TAXI_IN * NAS_DELAY
## Model 2: ARR_DELAY ~ DAY_OF_MONTH + TAXI_IN + s(TAXI_OUT) + s(DEP_DELAY) +
##     CARRIER_DELAY + NAS_DELAY + LATE_AIRCRAFT_DELAY + TAXI_IN
##   Resid. Df Resid. Dev     Df Deviance      F   Pr(>F)
## 1    1289.5     252048
## 2    1293.2     255469 -3.682  -3421.1 4.7598 0.001183 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

based on the anova test, the model excluding DAY_OF_WEEK and DEST is a better fit

Model Diagnostics

```
par(mfrow = c(2,2))
gam.check(gam02)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradient at convergence was 9.566413e-05 .
## The Hessian was positive definite.
## Model rank =  24 / 24
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                 k'  edf k-index p-value
## s(TAXI_OUT)   9.00 6.36    1.03   0.820
## s(DEP_DELAY) 9.00 2.67    0.96   0.055 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Test Error

```
gam_preds <- predict.gam(gam02, newdata = test)
gam_MSE <- sum((test$ARR_DELAY - gam_preds)^2, na.rm=T)/328
gam_MSE
```

```
## [1] 216.3382
```

Boxcox Transformed GAM

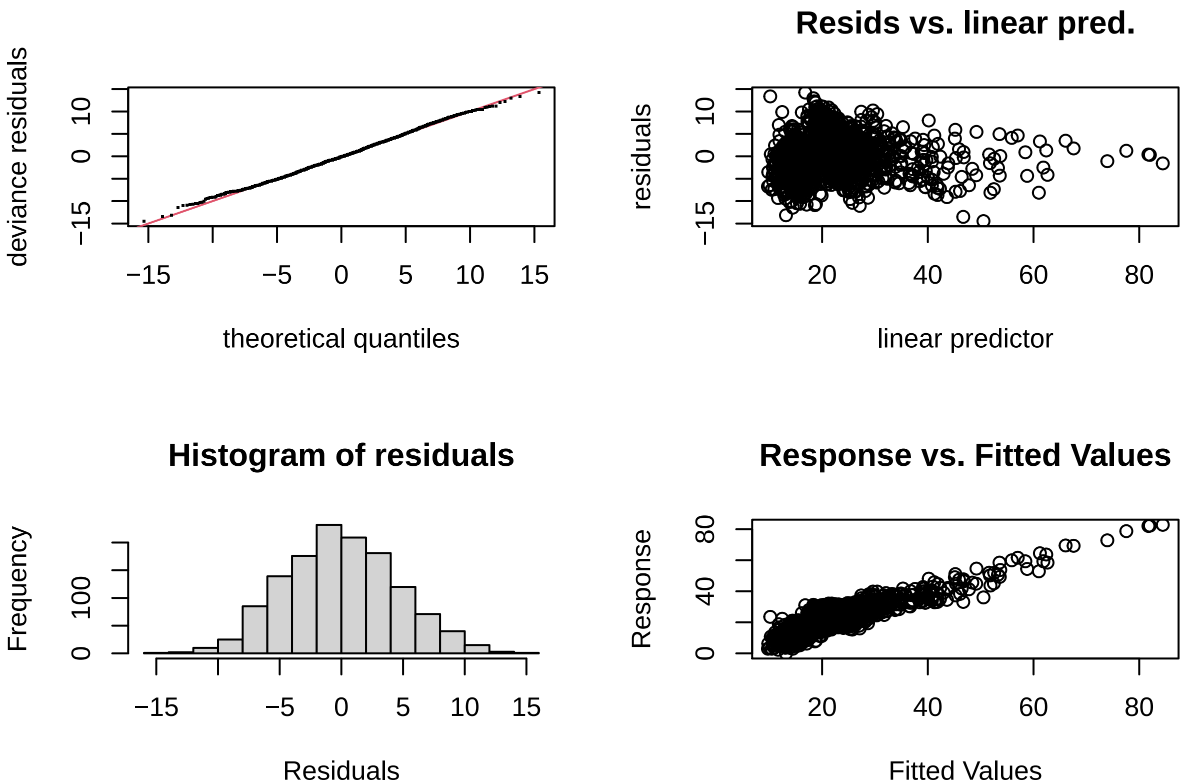
```
gambc <- gam(bc_adj_ARR_DELAY ~ DAY_OF_MONTH +
                   TAXI_IN +
```

```r
                   s(TAXI_OUT) +
                   s(DEP_DELAY) +
                   CARRIER_DELAY +
                   NAS_DELAY +
                   LATE_AIRCRAFT_DELAY +
                   TAXI_IN, by = NAS_DELAY, data = train)

summary(gambc)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## bc_adj_ARR_DELAY ~ DAY_OF_MONTH + TAXI_IN + s(TAXI_OUT) + s(DEP_DELAY) +
##     CARRIER_DELAY + NAS_DELAY + LATE_AIRCRAFT_DELAY + TAXI_IN
##
## Parametric coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          27.15996    0.30562  88.867   <2e-16 ***
## DAY_OF_MONTH         -0.44245    0.01439 -30.739   <2e-16 ***
## TAXI_IN               0.18985    0.01500  12.656   <2e-16 ***
## CARRIER_DELAY         0.83125    0.80147   1.037    0.300
## NAS_DELAY             8.74534    0.51081  17.121   <2e-16 ***
## LATE_AIRCRAFT_DELAY  1.23548    1.01060   1.223    0.222
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                edf Ref.df     F p-value
## s(TAXI_OUT)  1.000  1.000 207.9  <2e-16 ***
## s(DEP_DELAY) 4.889  5.957 200.9  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.791   Deviance explained = 79.3%
## GCV = 20.964  Scale est. = 20.774    n = 1310
```

BC Model Diagnostics

```r
par(mfrow = c(2,2))
gam.check(gambc)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 11 iterations.
## The RMS GCV score gradient at convergence was 9.854045e-07 .
## The Hessian was positive definite.
## Model rank =  24 / 24
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                k'  edf k-index p-value
## s(TAXI_OUT)  9.00 1.00    1.01    0.58
## s(DEP_DELAY) 9.00 4.89    0.96    0.06 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

BC Test Error

```
#gambc_preds <- predict.gam(gambc, newdata = test)
#gambc_MSE <- sum((test$ARR_DELAY - gambc_preds)^2, na.rm=T)/328
#gambc_MSE
```

TREES

Random Forests

```
library(tree)
library(randomForest)
```

By default, `randomForest()` uses $p/3$ variables when building a random forest of regression trees.

```
set.seed(1)
rf.delay <- randomForest(ARR_DELAY ~ DAY_OF_MONTH +
                   TAXI_IN +
                   TAXI_OUT +
                   DEST +
                   DEP_DELAY +
                   CARRIER_DELAY +
                   NAS_DELAY +
                   NAS_DELAY*TAXI_IN,
                   data = train, na.action = na.omit, importance = TRUE,
                   ntree=10000)
yhat.rf <- predict(rf.delay, newdata = test)
rf.MSE <- sum((test$ARR_DELAY - yhat.rf)^2, na.rm=T)/328
rf.MSE
```

```
## [1] 155.0148
```

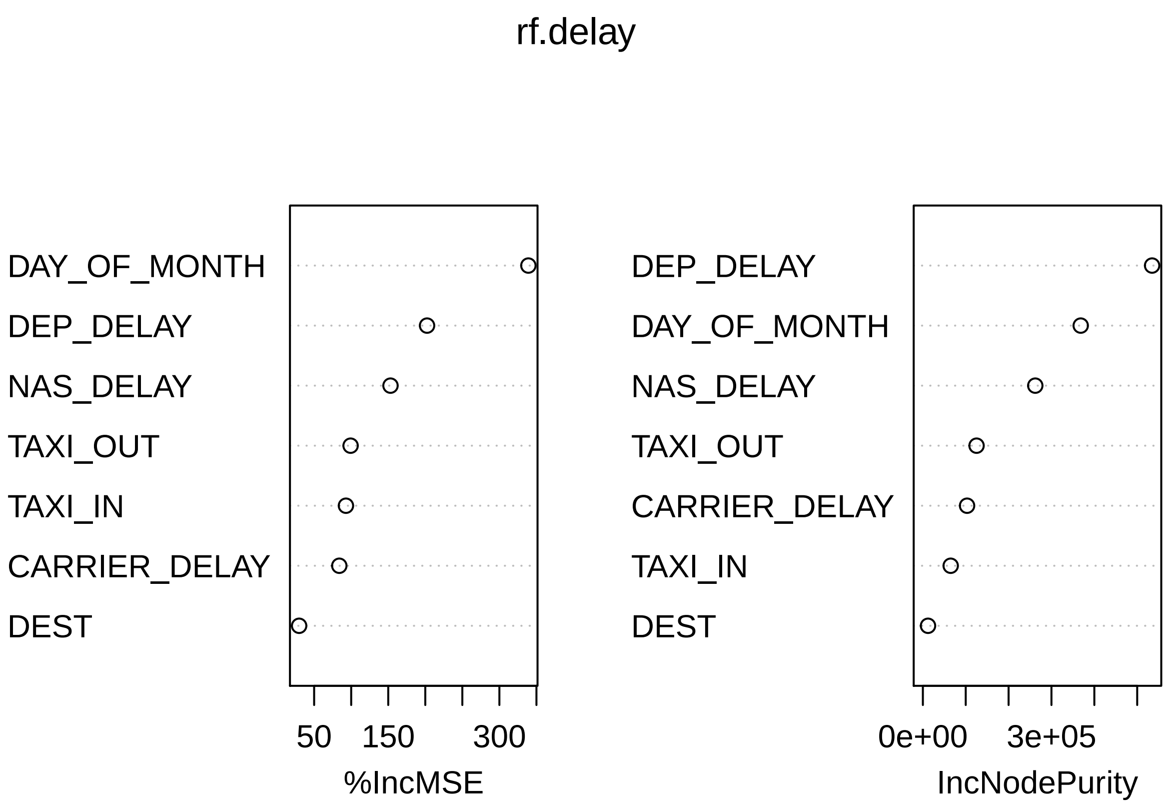
Using the `importance()` function, we can view the importance of each variable.

```
importance(rf.delay)
```

```
##                 %IncMSE IncNodePurity
## DAY_OF_MONTH  339.08627     368261.46
## TAXI_IN        92.93643      64905.19
## TAXI_OUT       99.21626     125258.87
## DEST           29.76223      12009.51
## DEP_DELAY     202.38571     534776.35
## CARRIER_DELAY  83.95478     103105.51
## NAS_DELAY     153.05251     262232.63
```

Two measures of variable importance are reported. The former is based on the mean decrease in accuracy in predictions on the out of bag samples when a given variable is excluded from the model. The latter is a measure of the total decrease in node impurity that results from splits over that variable, averaged over all trees (this was plotted in Figure 8.9 in the text). In the case of regression trees, the node impurity is measured by the training RSS and for classification trees by the deviance. Plots of these importance measures can be produced using the `varImpPlot()` function.

```
varImpPlot(rf.delay)
```



4. Boosting

Here we use the `gbm()` package, and within it the `gbm()` function, to fit boosted regression trees to the `train` data set. We run `gbm()` with the option `distribution = "gaussian"` since this is a regression problem. The argument `n.trees = 10000` indicates that we want 10000 trees, and the option `interaction.depth = 1` limits the depth of each tree.

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
set.seed(1)
train <- train %>%
  filter(!is.na(ARR_DELAY))
#boost.delay <- gbm(ARR_DELAY ~ DAY_OF_MONTH +
 #                  TAXI_IN +
  #                 TAXI_OUT +
   #               DEP_DELAY +
    #              CARRIER_DELAY +
     #             NAS_DELAY +
      #            LATE_AIRCRAFT_DELAY,
       #           data = train, distribution = "gaussian",
        #          n.trees=10000, interaction.depth=1, shrinkage=0.001, cv.folds=10)
```

The `summary()` function also provides a relative influence plot and also outputs the relative influence statistics.

```
#summary(boost.delay)
```

We see that `DEP_DELAY` and `DAY_OF_MONTH` are by far the most important variables. We can also produce *partial dependence plots* for these two variables. These plots illustrate the marginal effect of the selected variables on the response after `integrating` out the other variables.

```
#par(mfrow = c(1,2))
#plot(boost.delay, i = "DEP_DELAY")
#plot(boost.delay, i = "DAY_OF_MONTH")
```

We now use the boosted model to predict `ARR_DELAY` on the test set:

```
#yhat.boost <- predict(boost.delay, newdata =test,
#                      n.trees = 10000)
#boost_MSE <- sum((test$ARR_DELAY-yhat.boost)^2, na.rm = T)/328
#boost_MSE
```

The test MSE obtained is 11.8; similar to the test MSE for random forests and superior to that for bagging. If we want to, we can perform boosting with a different value of the shrinkage parameter λ in Equation 8.10. The default value is 0.001, but this is easily modified. Here, we take $\lambda = 0.2$.

```
# boost.boston <- gbm(medv~., data = Boston[train,],
#                     distribution = "gaussian", n.trees = 5000,
#                     interaction.depth = 4,
#                     shrinkage = 0.2,
#                     verbose =FALSE)
# yhat.boost <- predict(boost.boston, newdata = Boston[-train,],
#                      n.trees = 5000)
# mean((yhat.boost - boston.test)^2)
```