Predicting and Analyzing Causes of Delay for Flight Departures

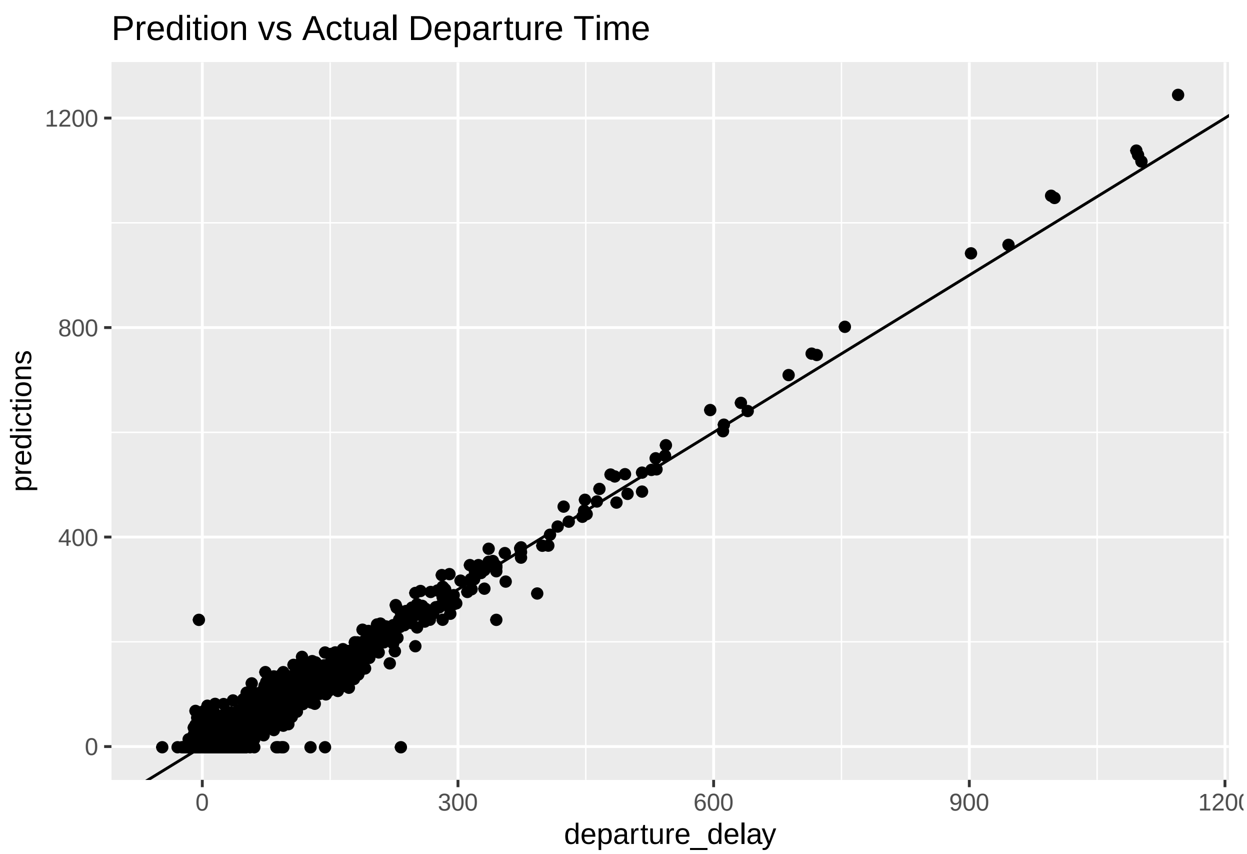
Shaun Tran

2023-04-26

```r
# Load necessary libraries
library(tidyverse)
library(caret)

# Read in data
data <- read_csv("Detailed_Statistics_Departures1.csv")

# Select relevant columns
data <- data %>%
  select(DelayCarrierMinutes, DelayWeatherMinutes, DelayNationalAviationSystemMinutes, DelaySecurityMinu

# Rename columns for convenience
colnames(data) <- c("carrier", "weather", "nas", "security", "late_aircraft", "destination", "departure_
```

```r
# Split data into training and test sets
set.seed(123)
trainIndex <- createDataPartition(data$departure_delay, p = 0.8, list = FALSE)
trainData  <- data[trainIndex, ]
testData   <- data[-trainIndex, ]
```

```r
# Create linear model using least squares method on training data
model <- lm(departure_delay ~ carrier + weather + nas + security + late_aircraft, data = trainData)

# Make predictions on test data
predictions <- predict(model, testData)

# Add predictions to test data
testData$predictions <- predictions

# Plot predictions against actual values using ggplot
ggplot(testData, aes(x = departure_delay, y = predictions)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1) + ggtitle("Predition vs Actual Departure Time")
```




```
summary(model)
```

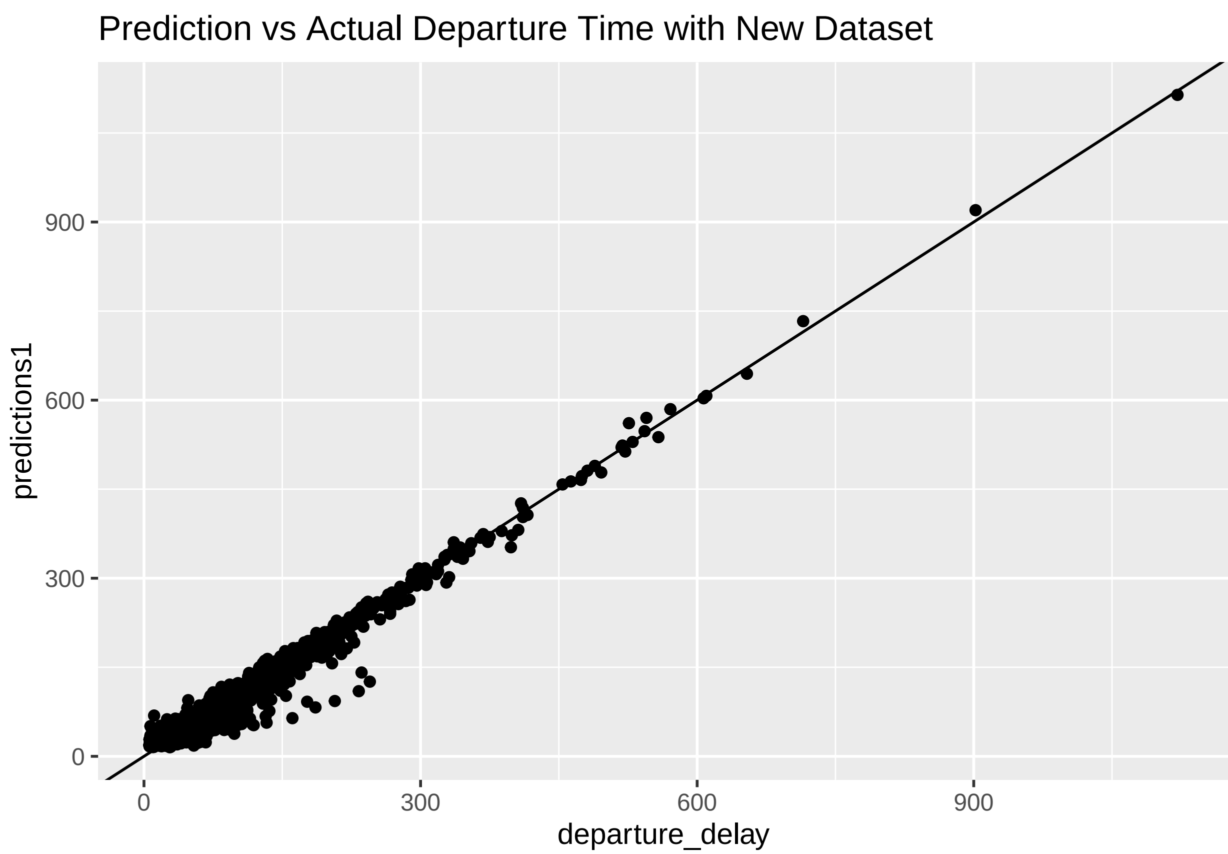
```
##
## Call:
## lm(formula = departure_delay ~ carrier + weather + nas + security +
##     late_aircraft, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -146.14   -4.69   -1.69    2.27  855.31
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.307427   0.039550  -33.06   <2e-16 ***
## carrier        1.040379   0.001411  737.47   <2e-16 ***
## weather        1.008467   0.008425  119.69   <2e-16 ***
## nas            0.762434   0.002844  268.06   <2e-16 ***
## security       1.312695   0.084342   15.56   <2e-16 ***
## late_aircraft  1.083832   0.001718  630.83   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.34 on 87942 degrees of freedom
## Multiple R-squared:  0.9247, Adjusted R-squared:  0.9247
## F-statistic: 2.159e+05 on 5 and 87942 DF,  p-value: < 2.2e-16
```

```r
#model without security
model12 <- lm(departure_delay ~ carrier + weather + nas + late_aircraft, data = trainData)
summary(model12)
```

```
##
## Call:
## lm(formula = departure_delay ~ carrier + weather + nas + late_aircraft,
##     data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -146.14   -4.70   -1.70    2.28  855.30
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.302272   0.039603  -32.88   <2e-16 ***
## carrier        1.040351   0.001413  736.44   <2e-16 ***
## weather        1.008407   0.008437  119.52   <2e-16 ***
## nas            0.762418   0.002848  267.69   <2e-16 ***
## late_aircraft  1.084068   0.001720  630.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.35 on 87943 degrees of freedom
## Multiple R-squared:  0.9245, Adjusted R-squared:  0.9245
## F-statistic: 2.691e+05 on 4 and 87943 DF,  p-value: < 2.2e-16
```

```r
#Removing 0s from the predictor column because we are only looking at when there is a delay in the fact
data1 <- trainData[trainData$carrier != 0, ]
data2 <- trainData[trainData$weather != 0, ]
data3 <- trainData[trainData$nas != 0, ]
data4 <- trainData[trainData$security != 0, ]
data5 <- trainData[trainData$late_aircraft != 0, ]
```

```r
#data6 is the new dataset that contains only the rows with a delay in at least one predictor
data6 <- data[rowSums(data[c(1, 5)] != 0) > 0,]
set.seed(123)
trainIndex1 <- createDataPartition(data6$departure_delay, p = 0.8, list = FALSE)
trainData1 <- data6[trainIndex1, ]
testData1 <- data6[-trainIndex1, ]
modelgg <- lm(departure_delay ~ carrier + weather + nas + security + late_aircraft, data = trainData1)
predictions1 <- predict(modelgg, testData1)
testData1$predictions1 <- predictions1
ggplot(testData1, aes(x = departure_delay, y = predictions1)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1) + ggtitle("Prediction vs Actual Departure Time with New Dataset
```



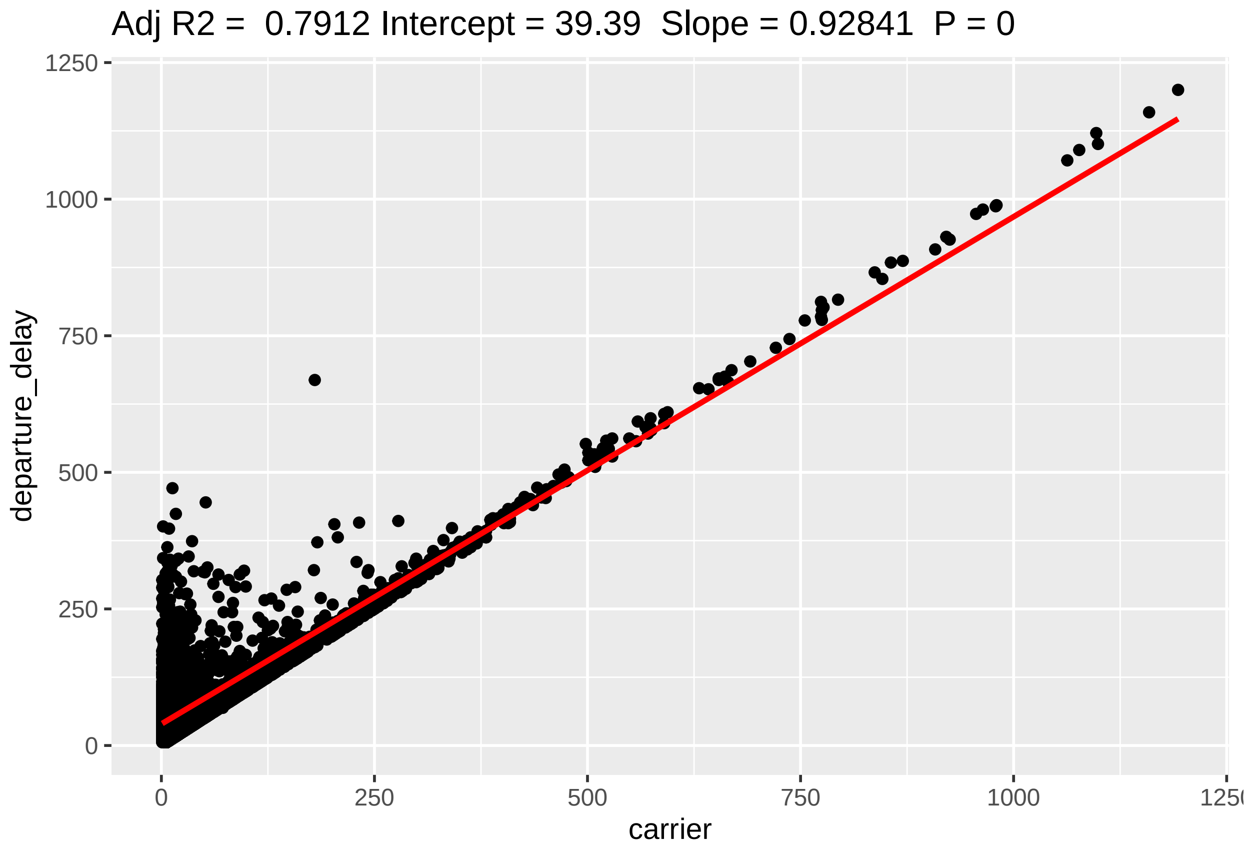

```
summary(modelgg)
```

```
##
## Call:
## lm(formula = departure_delay ~ carrier + weather + nas + security +
##     late_aircraft, data = trainData1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.128 -10.343  -1.572   8.115 117.024
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.434904   0.206251  40.896  < 2e-16 ***
## carrier       1.007931   0.001807 557.666  < 2e-16 ***
## weather       0.975164   0.022996  42.406  < 2e-16 ***
## nas           0.388759   0.009317  41.725  < 2e-16 ***
## security      1.466055   0.309048   4.744 2.13e-06 ***
## late_aircraft 1.024886   0.002561 400.260  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.21 on 9270 degrees of freedom
## Multiple R-squared:  0.9776, Adjusted R-squared:  0.9776
## F-statistic: 8.108e+04 on 5 and 9270 DF,  p-value: < 2.2e-16
```

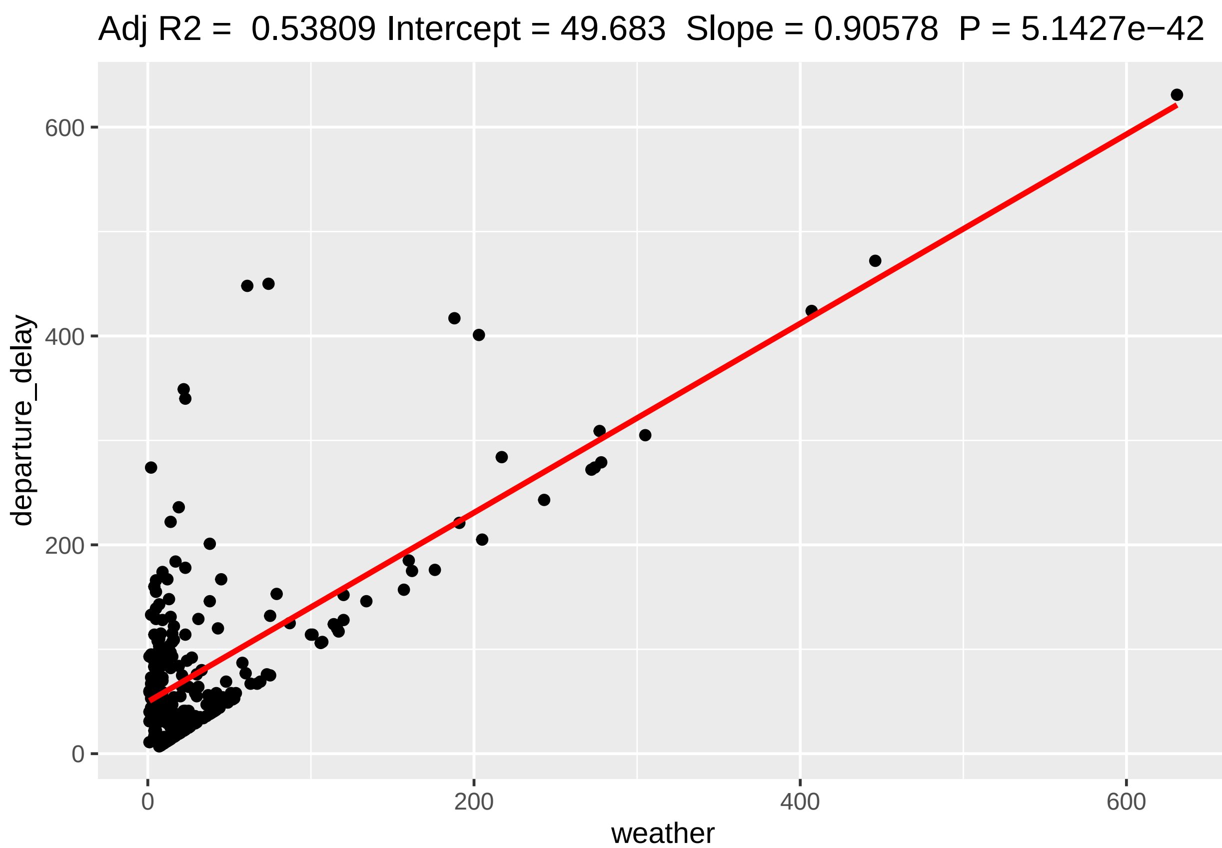
```r
#Here i did a fitted model without the predictor  security
mod1 <- lm(departure_delay ~ carrier + weather + nas + late_aircraft, data = trainData)
#anova is a function to analysis the variance, this is to see their F-statistic and P value.
anova(mod1,model)
```

```
## Analysis of Variance Table
##
## Model 1: departure_delay ~ carrier + weather + nas + late_aircraft
## Model 2: departure_delay ~ carrier + weather + nas + security + late_aircraft
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1  87943 11336579
## 2  87942 11305438  1     31141 242.23 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

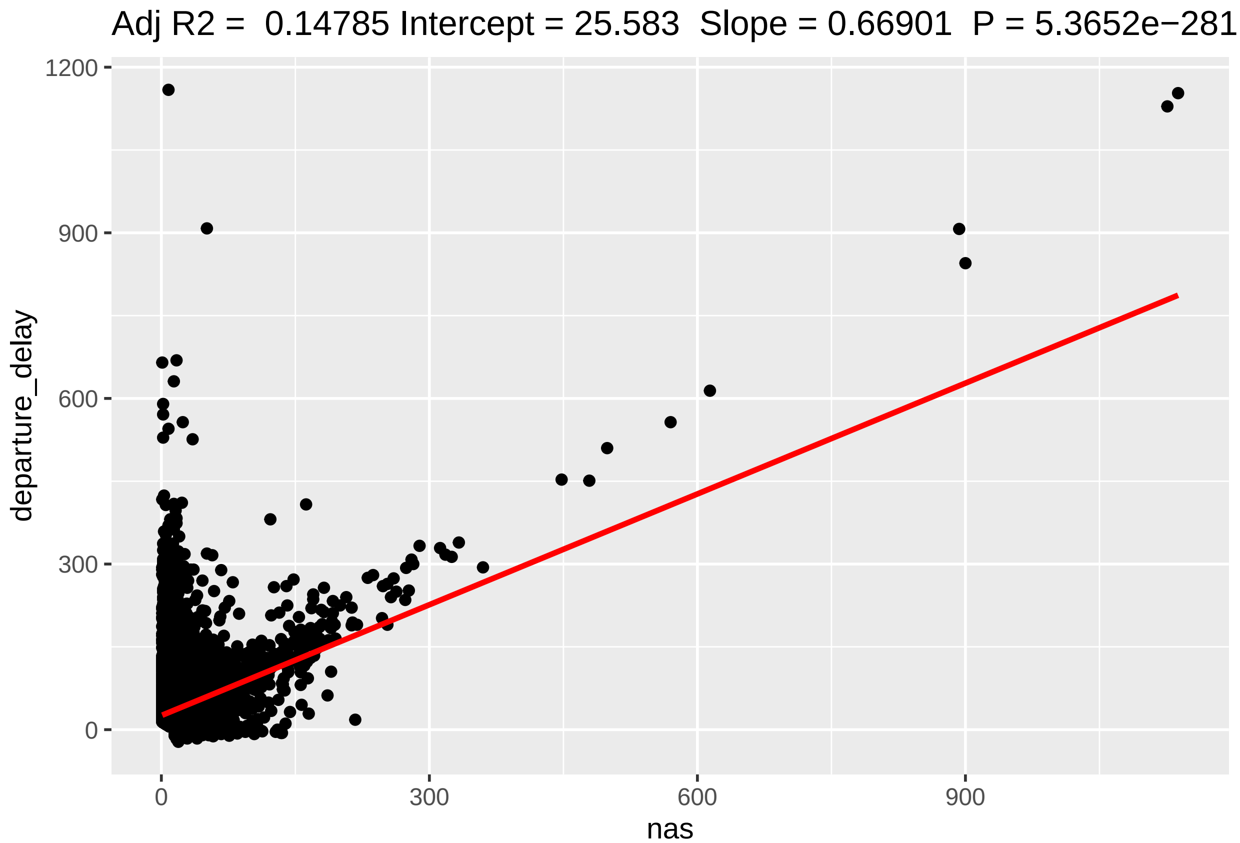
```r
#this function allow us to plot and show the adjusted r square, intercept, slope and p value
ggplotRegression <- function (fit) {

require(ggplot2)

ggplot(fit$model, aes_string(x = names(fit$model)[2], y = names(fit$model)[1])) +
  geom_point() +
  stat_smooth(se=FALSE, method = "lm", col = "red") +
  labs(title = paste("Adj R2 = ",signif(summary(fit)$adj.r.squared, 5),
                     "Intercept =",signif(fit$coef[[1]],5 ),
                     " Slope =",signif(fit$coef[[2]], 5),
                     " P =",signif(summary(fit)$coef[2,4], 5)))
}



#I plotted one predictor to the response variable at a time using the dataset that i cleaned.
ggplotRegression(lm(data = data1 , departure_delay~carrier))
```



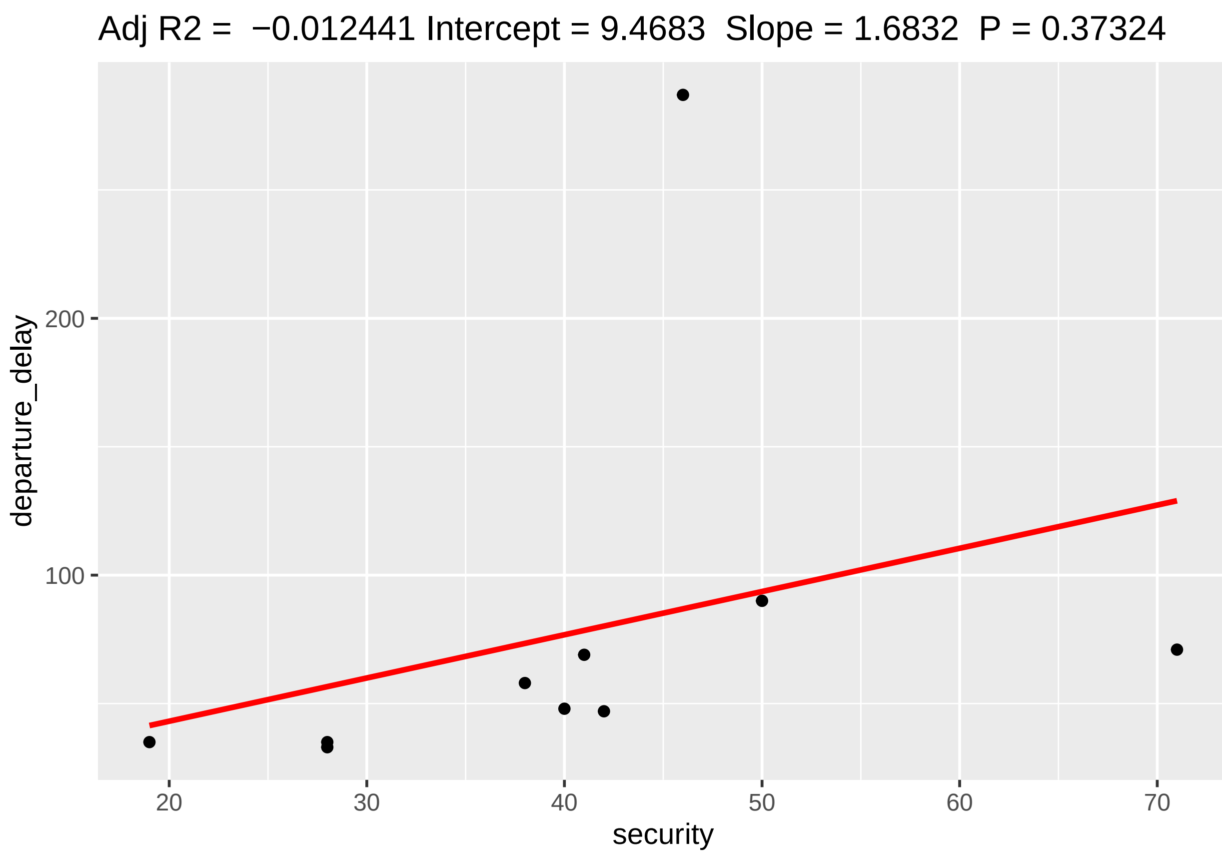

```
ggplotRegression(lm(data = data2 , departure_delay~weather))
```




```
ggplotRegression(lm(data = data3 , departure_delay~nas))
```




```
ggplotRegression(lm(data = data4 , departure_delay~security))
```




```
ggplotRegression(lm(data = data5 , departure_delay~late_aircraft))
```

Adj R2 = 0.85767 Intercept = 19.897 Slope = 1.0085 P = 0