FlightDelay

Group 6

2022-12-07

Reading in flight data from 2018 - 2022 using smaller .parquet files to limit size

```
#Data from Kaggle: https://www.kaggle.com/datasets/robikscube/flight-delay-dataset-20182022?select=Comb
con <- dbConnect(SQLite(), "flight_data.db")
```

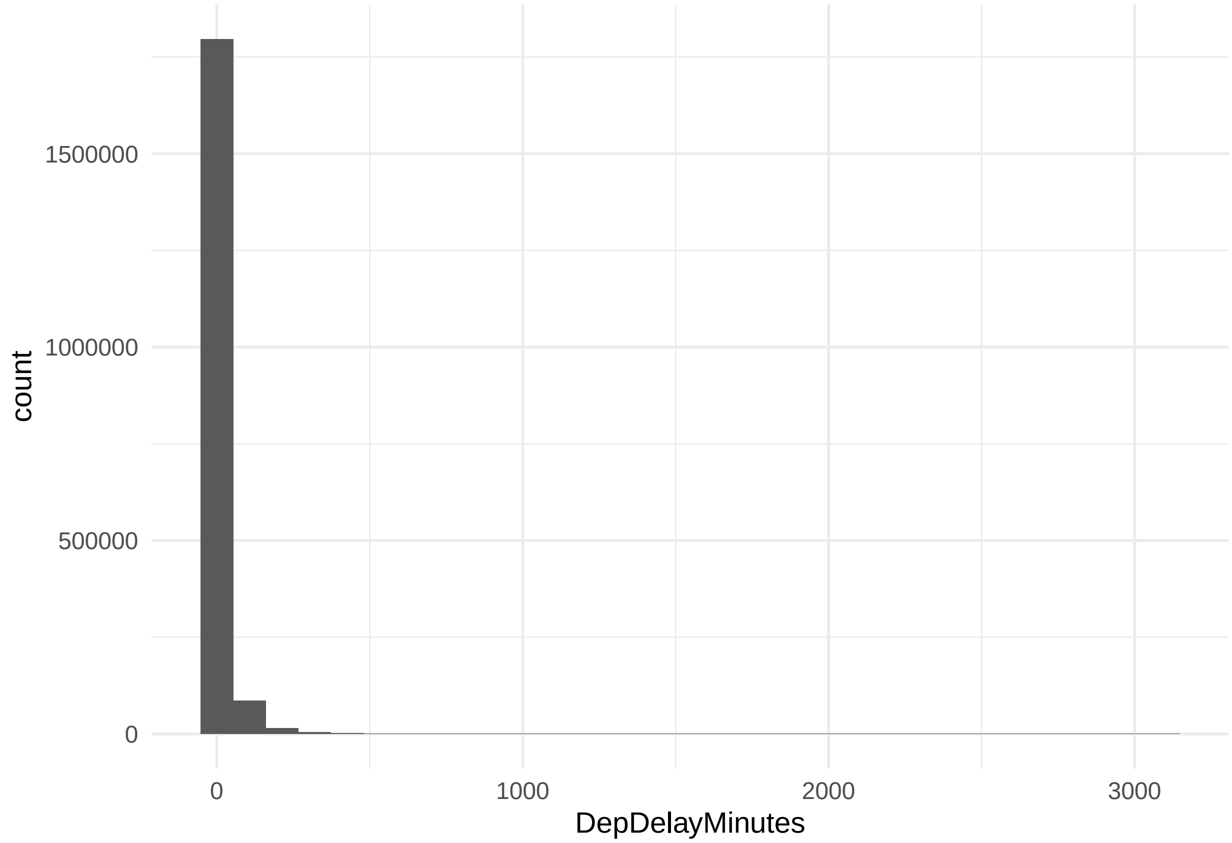
```
path <- "C:/Users/chase/OneDrive/Northeastern Code/DS 5110/Group Project/data/Combined_Flights_2021.parq
flights_2021 <- read_parquet(path, as_data_frame = TRUE) #read in flight data
dbWriteTable(con, "flights", flights_2021) #initializing flights df
rm(flights_2021) #dropping df from environment to minimize memory usage
```

```
df <- as_tibble(dbGetQuery(con,
           "SELECT FlightDate
                 , Airline
                 , Origin
                 , Dest
                 , Cancelled
                 , Diverted
                 , DepTime
                 , DepDelayMinutes
                 , Distance
                 , DistanceGroup
                 , Year
                 , Quarter
                 , Month
                 , DayofMonth
                 , DayofWeek
                 , Marketing_Airline_Network
                 , OriginAirportID
                 , OriginCityName
                 , OriginStateName
                 , OriginWAC
                 , DestAirportID
                 , DestCityName
                 , DestStateName
                 , DestWac
                 , CRSDepTime
                 , CRSElapsedTime
                 , ArrDelayMinutes
                 FROM flights
           WHERE Operating_airline in ('AA', 'UA', 'DL')
          AND OriginWac BETWEEN 1 AND 93
          AND DestWac BETWEEN 1 AND 93"))
df$OriginEncode <- as.numeric(as.factor(df$Origin))
```

```r
df$DestEncoder <- as.numeric(as.factor(df$Dest))
ggplot(df, aes(x=DepDelayMinutes)) +
  geom_histogram() +
  theme_minimal()
```

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

```
## Warning: Removed 25590 rows containing non-finite values (`stat_bin()`).
```




```r
df <- na.omit(df)
```

We can see that departure delay is very frequently zero or near zero

```r
#Compartmentalize States into Census defined regions for analysis
Northeast <- c("Connecticut", "Maine", "Massachusetts", "New Hampshire", "Rhode Island", "Vermont", "Ne
Midwest <- c("Illinois", "Indiana", "Michigan", "Ohio", "Wisconsin", "Iowa", "Kansas", "Minnesota", "Mi
South <- c("Florida", "Georgia", "North Carolina", "South Carolina", "Virginia", "West Virginia", "Alab
West <- c("Arizona", "Colorado", "Idaho", "Montana", "Nevada", "New Mexico", "Utah", "Wyoming", "Alaska
df <- df %>%
  mutate(OriginRegion = case_when(
    OriginStateName %in% Northeast ~ "Northeast",
    OriginStateName %in% Midwest ~ "Midwest",
    OriginStateName %in% South ~ "South",
    OriginStateName %in% West ~ "West"
  ), .after = OriginStateName
)
df <- df %>%
  mutate(DestRegion = case_when(
```

```r
    DestStateName %in% Northeast ~ "Northeast",
    DestStateName %in% Midwest ~ "Midwest",
    DestStateName %in% South ~ "South",
    DestStateName %in% West ~ "West"
  ), .after = DestStateName
)
#Partitioning data
df_part <- resample_partition(df,
                                    p=c(train=0.5,
                                        valid=0.25,
                                        test=0.25))
```

We re-encode region variables into quadrants based on US Census region data, in order to allow for greater interpretability.

```r
# Downsampling with 80% data.
df1 <- (df %>% filter(Airline=="Delta Air Lines Inc.", DepDelayMinutes==0))[427224:534031,]
df2 <- (df %>% filter(Airline=="United Air Lines Inc.", DepDelayMinutes==0))[225760:282200,]
df3 <- (df %>% filter(Airline=="American Airlines Inc.", DepDelayMinutes==0))[377192:471490,]
df4 <-  (df %>% filter(DepDelayMinutes!=0))
df_down <- rbind(df1,df2,df3, df4)
dim(df_down)
```

```
## [1] 870360     31
```

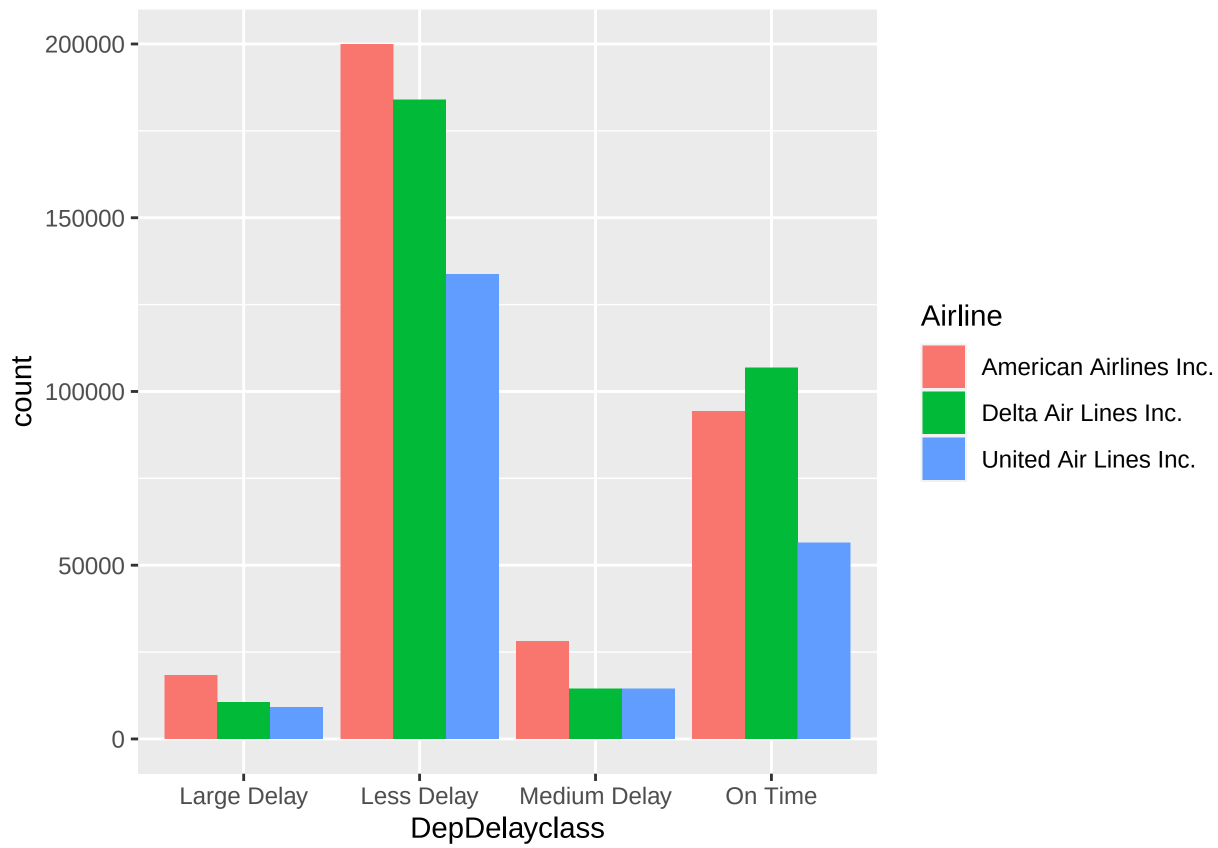
```r
interval <- function(x) {
  case_when(
    x == 0 ~ "On Time",
    between(x, 1, 60) ~ "Less Delay",
    between(x, 61, 120) ~ "Medium Delay",
    x >= 121 ~ "Large Delay"
  )
}
df_down$DepDelayclass<-interval(df_down$DepDelayMinutes)
df_down %>%
  select(Airline, DepDelayclass) %>%
  group_by(DepDelayclass, Airline) %>%
  summarise(n())
```

```
## `summarise()` has grouped output by 'DepDelayclass'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 12 x 3
## # Groups:   DepDelayclass [4]
##    DepDelayclass Airline                `n()`
##    <chr>         <chr>                  <int>
##  1 Large Delay   American Airlines Inc.  18279
##  2 Large Delay   Delta Air Lines Inc.    10561
##  3 Large Delay   United Air Lines Inc.    9175
##  4 Less Delay    American Airlines Inc. 199959
##  5 Less Delay    Delta Air Lines Inc.   183978
##  6 Less Delay    United Air Lines Inc.  133803
##  7 Medium Delay  American Airlines Inc.  28103
##  8 Medium Delay  Delta Air Lines Inc.    14478
##  9 Medium Delay  United Air Lines Inc.   14476
## 10 On Time       American Airlines Inc.  94299
```

```
## 11 On Time       Delta Air Lines Inc.   106808
## 12 On Time       United Air Lines Inc.   56441
```

```
ggplot(df_down,aes(x = DepDelayclass, fill = Airline)) + geom_bar(stat="count",position = "dodge")
```




```
#Partitioning data
df_down_part <- resample_partition(df_down,
                                    p=c(train=0.5,
                                        valid=0.25,
                                        test=0.25))
```

Downsampling here allows us to even out our significant class imbalance

```
#function to perform single step of stepwise model selection using RMSE, inspired by lecture (r markdow
step <- function(response, predictors, candidates, partition)
{
  rhs <- paste0(paste0(predictors, collapse="+"), "+", candidates)
  formulas <- lapply(paste0(response, "~", rhs), as.formula)
  rmses <- sapply(formulas,
                  function(fm) rmse(lm(fm, data=partition$train),
                                    data=partition$valid))
  names(rmses) <- candidates
  attr(rmses, "best") <- rmses[which.min(rmses)]
  rmses
}
```

Stepwise

```r
df <- df %>% filter(DepDelayMinutes!=0)
#initalizing model variable
model <- NULL
```

OriginRegion: No Region seems to strongly effect log(DepDelayMinutes) DestRegion: No Region seems to strongly effect log(DepDelayMinutes) Airline: Doesn't seem highly significant, though American probably highest Distance: Hard to tell, but appears fairly linear with log(Distance) DistanceGroup: Unclear if significant Quarter: 3 slightly higher, not significant seeming Month: 6-8 appear slightly higher DayofMonth: Cant tell if any are higher DayofWeek: Cant tell if any are higher

Using our forward stepwise function we select the most significant variable for predicting Departure Delay with each iteration

```r
preds <- "1"
cands <- c("OriginEncode", "DestEncoder", "Airline", "Distance","DistanceGroup", "Quarter", "Month", "Da
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

```
##  OriginEncode   DestEncoder       Airline      Distance DistanceGroup
##      1.645827      1.645845      1.637460      1.645620      1.645716
##       Quarter         Month    DayofMonth     DayOfWeek
##      1.645809      1.645002      1.645765      1.646035
## attr(,"best")
## Airline
## 1.63746
```

```r
preds <- c("Airline")
cands <- c("OriginEncode", "DestEncoder", "Month", "Distance","DistanceGroup", "Quarter", "DayofMonth",
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

```
##  OriginEncode   DestEncoder         Month      Distance DistanceGroup
##      1.636785      1.637376      1.636399      1.637139      1.637201
##       Quarter    DayofMonth     DayOfWeek
##      1.637171      1.636986      1.637289
## attr(,"best")
##    Month
## 1.636399
```

```r
preds <- c("Month", "Airline")
cands <- c("OriginEncode", "DestEncoder", "Distance","DistanceGroup", "Quarter", "DayofMonth", "DayOfWe
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

```
##  OriginEncode   DestEncoder      Distance DistanceGroup       Quarter
##      1.635704      1.636311      1.636057      1.636122      1.633619
##    DayofMonth     DayOfWeek
##      1.635967      1.636207
## attr(,"best")
##  Quarter
## 1.633619
```

```r
preds <- c("Month", "Airline", "Quarter")
cands <- c("Distance", "DestEncoder","DistanceGroup", "OriginEncode", "DayofMonth", "DayOfWeek")
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
```

```r
model <- c(model, attr(s1, "best"))
s1
```

```
##      Distance   DestEncoder DistanceGroup  OriginEncode    DayofMonth
##      1.633313      1.633531      1.633375      1.632910      1.633244
##     DayOfWeek
##      1.633352
## attr(,"best")
## OriginEncode
##      1.63291
```

```r
preds <- c("Month", "Airline", "Quarter", "OriginEncode")
cands <- c("DestEncoder","DistanceGroup", "Distance", "DayofMonth", "DayOfWeek")
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

```
##   DestEncoder DistanceGroup      Distance    DayofMonth     DayOfWeek
##      1.632856      1.632496      1.632436      1.632536      1.632642
## attr(,"best")
## Distance
## 1.632436
```

```r
preds <- c("Month", "Airline", "Distance", "OriginEncode", "Quarter")
cands <- c("DestEncoder","DistanceGroup", "DayOfWeek", "DayofMonth")
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

```
##   DestEncoder DistanceGroup     DayOfWeek    DayofMonth
##      1.632418      1.632431      1.632172      1.632057
## attr(,"best")
## DayofMonth
##   1.632057
```

```r
preds <- c("Month", "Airline", "Distance", "OriginEncode", "Quarter", "DayofMonth")
cands <- c("DestEncoder","DistanceGroup", "DayOfWeek")
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

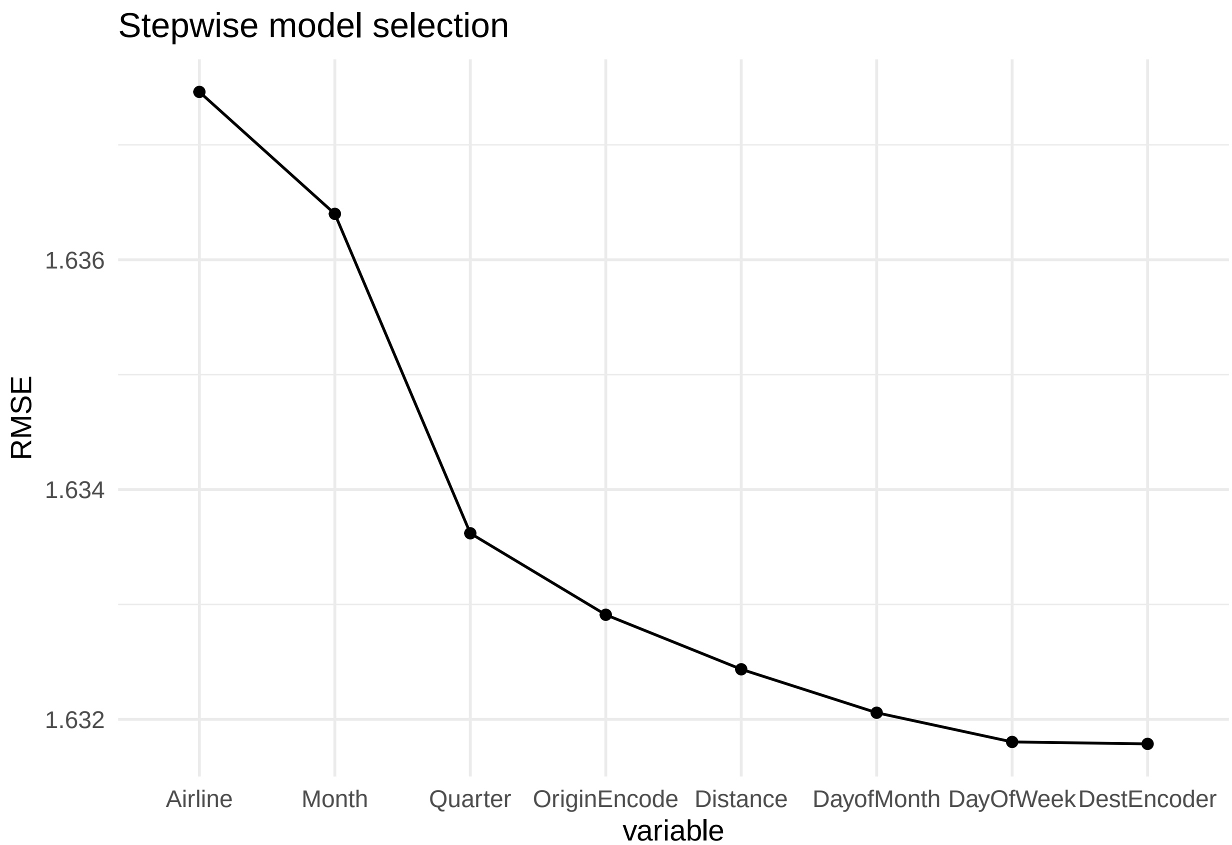
```
##   DestEncoder DistanceGroup     DayOfWeek
##      1.632040      1.632053      1.631803
## attr(,"best")
## DayOfWeek
##  1.631803
```

```r
preds <- c("Month", "Airline", "Distance", "OriginEncode", "Quarter", "DayofMonth", "DayOfWeek")
cands <- c("DestEncoder","DistanceGroup")
s1 <- step("log1p(DepDelayMinutes)", preds, cands, df_down_part)
model <- c(model, attr(s1, "best"))
s1
```

```
##   DestEncoder DistanceGroup
##      1.631786      1.631799
## attr(,"best")
## DestEncoder
```

```
## 	1.631786
```

```
step_model <- tibble(index=seq_along(model),
                     variable=factor(names(model), levels=names(model)),
                     RMSE=model)
ggplot(step_model, aes(y=RMSE)) +
  geom_point(aes(x=variable)) +
  geom_line(aes(x=index)) +
  labs(title="Stepwise model selection") +
  theme_minimal()
```



Here we plot the step-wise function to determine the decrease in RMSE each variable's inclusion gives us.

```
fitting <- lm(log1p(DepDelayMinutes) ~ OriginEncode + Airline + Month + Distance + Quarter, data =df_dow
rmse(fitting, df_down_part$test)
```

```
## [1] 1.630645
```

```
summary(fitting)
```

```
##
## Call:
## lm(formula = log1p(DepDelayMinutes) ~ OriginEncode + Airline +
##     Month + Distance + Quarter, data = df_down_part$train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.424 -1.646 -0.040  1.243  6.059
##
```

```
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                     2.052e+00  9.452e-03  217.10   <2e-16 ***
## OriginEncode                   -1.396e-03  5.534e-05  -25.22   <2e-16 ***
## AirlineDelta Air Lines Inc.    -3.777e-01  5.711e-03  -66.14   <2e-16 ***
## AirlineUnited Air Lines Inc. -8.587e-02  6.413e-03  -13.39   <2e-16 ***
## Month                           1.337e-01  3.074e-03   43.49   <2e-16 ***
## Distance                        6.006e-05  3.775e-06   15.91   <2e-16 ***
## Quarter                        -3.332e-01  8.688e-03  -38.35   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.631 on 435172 degrees of freedom
## Multiple R-squared:  0.01661,    Adjusted R-squared:  0.01659 
## F-statistic:  1225 on 6 and 435172 DF,  p-value: < 2.2e-16
```

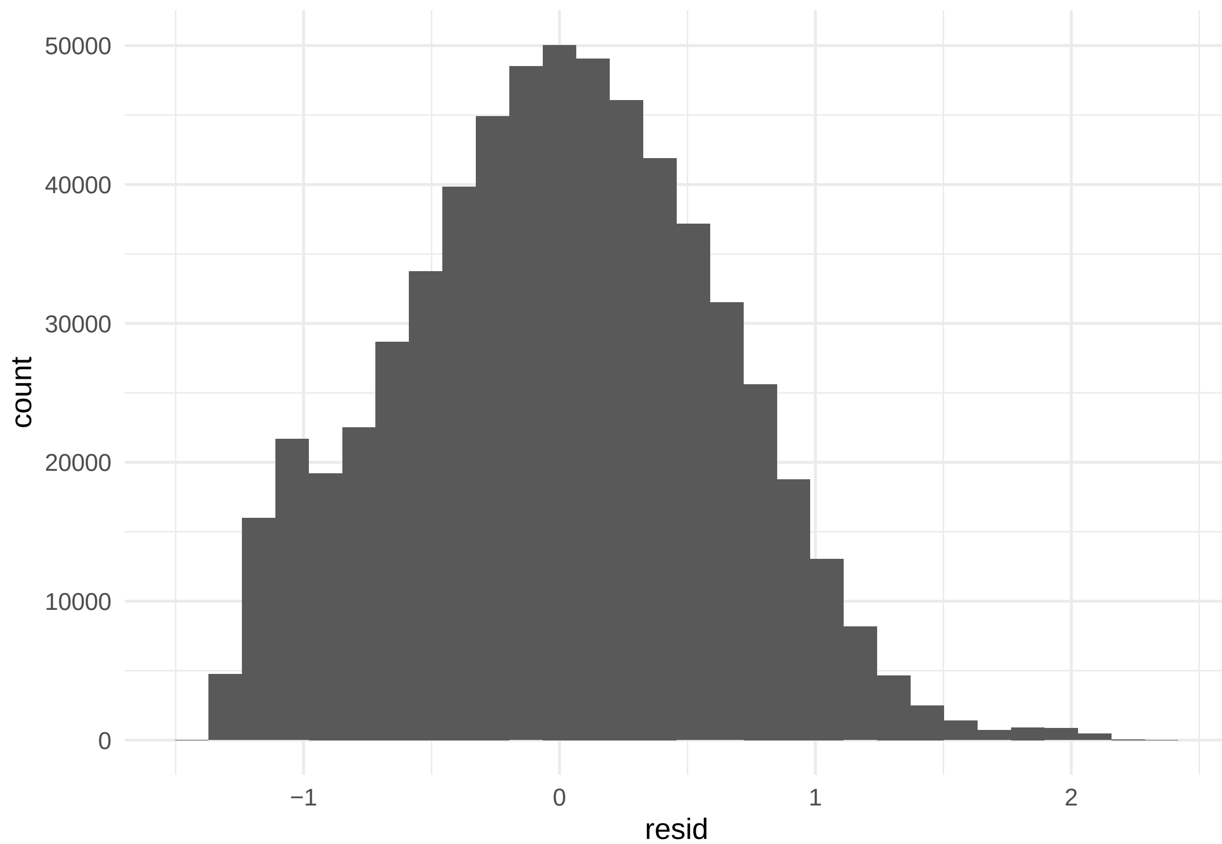
We chose to select Origin, Airline, Month, Distance, and Quarter as our variables for prediction, as including the next variables did not give large RMSE decreases, and could result in overfitting.

Code for DepDelayMinutes=0 condition.

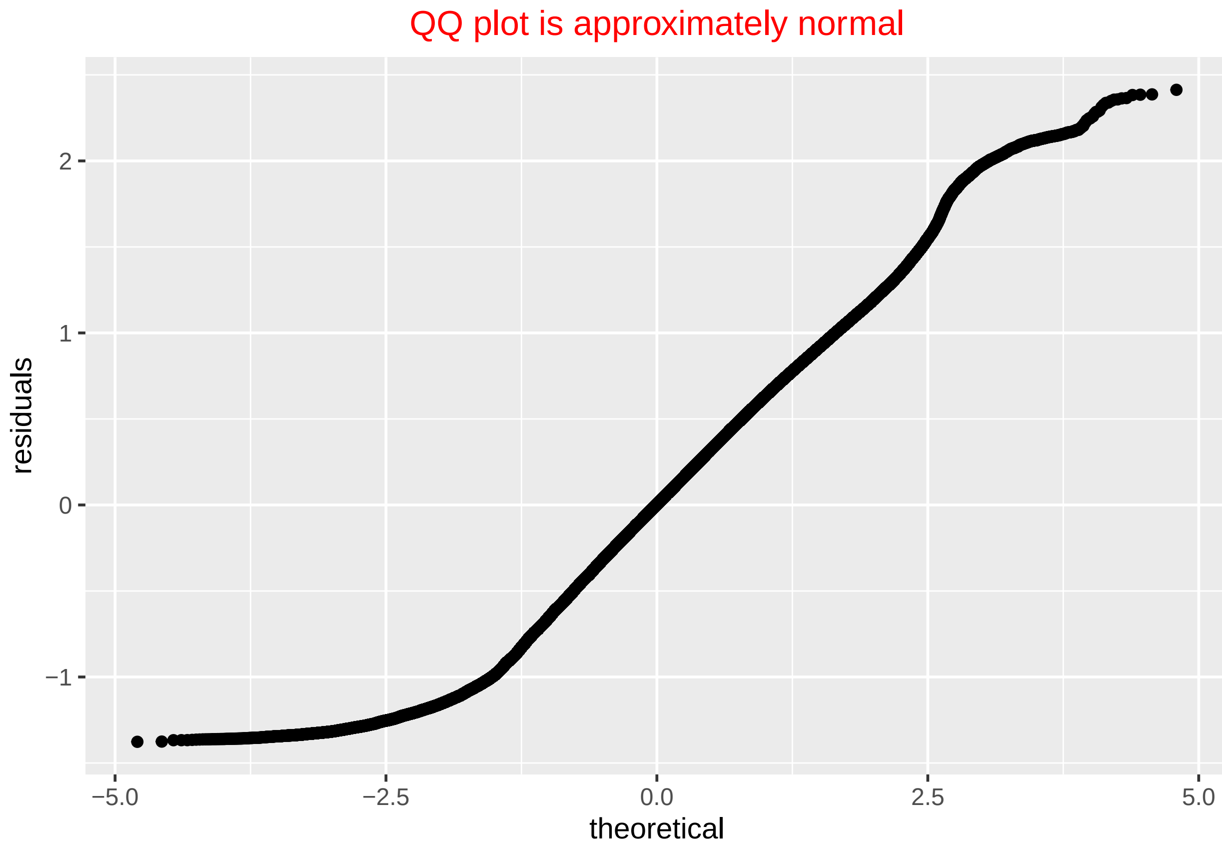
Finally, we consider the effect of removing rows where Departure Delay = 0, this gives us a much more normal distribution of residuals, and shows that it may be good to separate this analysis into two tasks, prediction of Delayed vs Not-Delayed, and separately, if delayed, by how much?

```
# Trying depDelayMinutes=0 condition.
df <- df %>% filter(DepDelayMinutes!=0)
fitting <- lm(log10(DepDelayMinutes) ~ OriginEncode + DestEncoder + Airline + Month + Distance + DayofMo
df %>% add_residuals(fitting, "resid") %>%
  ggplot(aes(x=resid)) +
  geom_histogram() +
  theme_minimal()
```

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




```
df %>%
add_residuals(fitting, "resid") %>%
ggplot(aes(sample=resid)) +
geom_qq() +
labs(title="QQ plot is approximately normal", y="residuals")+
theme(plot.title=element_text(hjust=0.5, color="red"))
```


QQ plot is approximately normal
2
1
0
−1
residuals
−5.0
−2.5
0.0
2.5
5.0
theoretical