STAT 652 PROJECT REPORT
NYC FLIGHT DATA ANALYSIS

Slavvy Coelho
(301359836)

1. INTRODUCTION

This project aims to build a prediction model for predicting flight delays for the year 2013. The project consists data of domestic flights that departed from the three major New York City airports in 2013. The data was collected over the entire year of 2013 and consists of various factors such as scheduled flight time, scheduled arrival time, weather details, carrier details, etc. This data pertains to three major airports viz, John F. Kennedy International Airport (JFK), Newark Liberty International Airport (EWR) and LaGuardia Airport (LGA).

In this project, various statistical learning frameworks for both classification and regression were explored to build efficient prediction models. The models were explored and improved by standard statistical tests and tuning and the results were compared to in turn select the best one for prediction on the held-out test set.

2. DATA

The data is taken from the 'nycflights13' package in R. The provided data is a csv file named fltrain.csv which was combined using the following 4 datasets from this package:

- Flights: all flights that departed from NYC in 2013 (schedule and logistical modifications)
- Weather: hourly meteorological data for each airport
- Airports: airport names and locations
- Planes: construction information about each plane

It contains 43 variables measured on 200,000 flights. The dataset contains information about all flights departed from NYC in 2013, hourly meteorological data for each airport, airport names and locations, construction information about each plane.

The test dataset is a held-out dataset consisting of 1336776 observations. More information about this data is available on the R documentation page[1].

3. METHODS

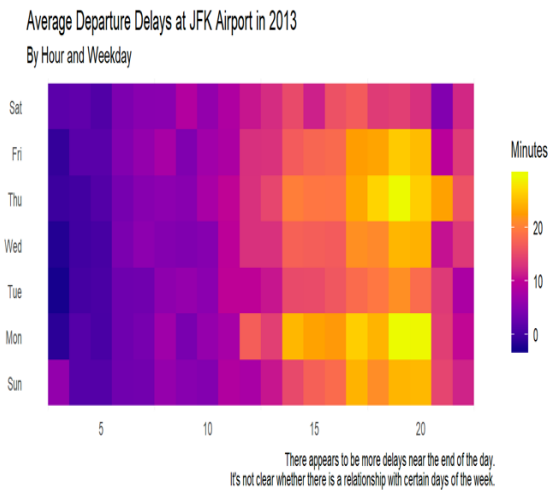
In this project, the problem can be attempted in the following two ways:

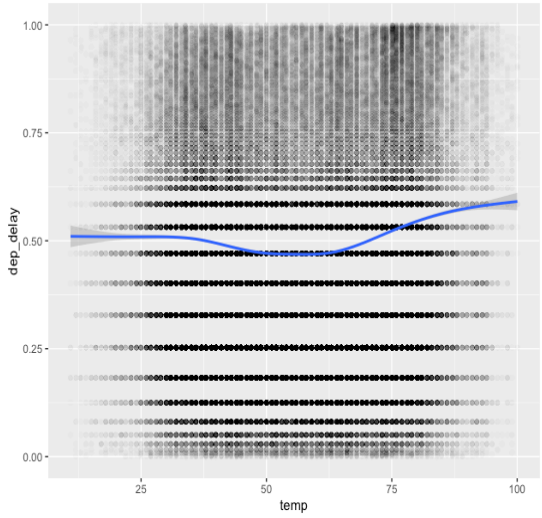
- Classification: Encoding the class label 'dep_delay' as 0 if on time and 1 if otherwise based on a threshold. (`eg: if dep_delay > 20, 0, 1`)
- Regression: Predict the actual delay in minutes using regression

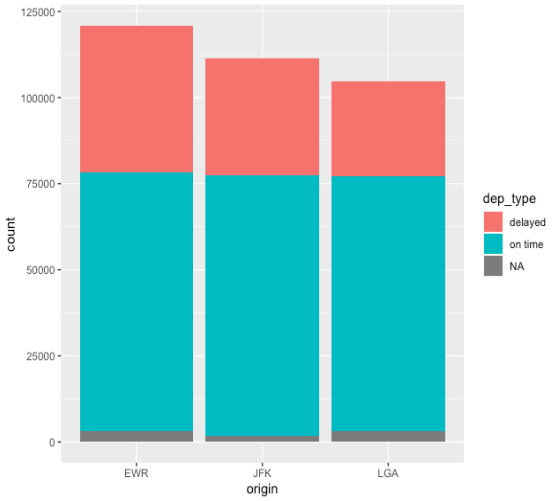
I decided to go ahead with regression analysis.

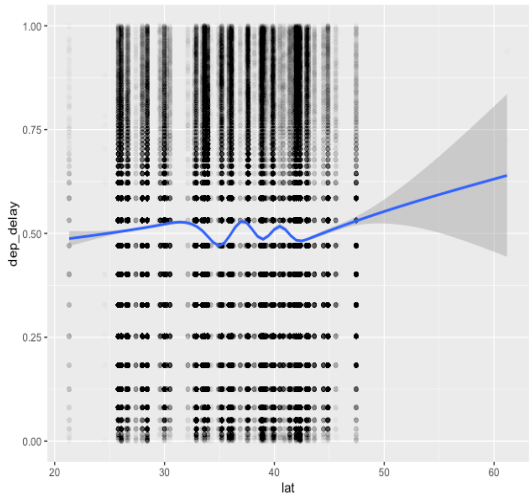
3.1. Data Preprocessing

Here, the dataset was converted to factors from characters. After this, the missing values were imputed. Few columns with negligible relative influence were discarded. The data was normalized and scaled to handle outliers (long tails). Finally, the data is ready to be worked with. Following are a few plots generated:









3.2. Statistical Modelling

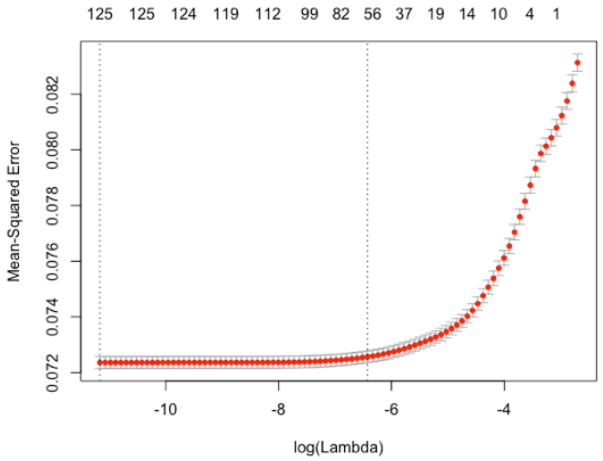
In this step, the 'dep_delay' was fit across all other variables and the error was calculated. Following were the models used for predicting the delay:

1. *GLM:*

GLM was implemented to model dichotomous outcome variables. In the logit model the log odds of the outcome were modeled as a linear combination of the predictor variables. The model here was fit for binomial family.

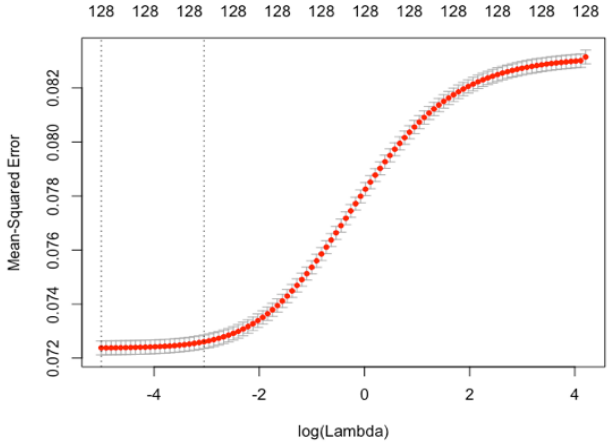
2. Lasso Regression:

Lasso is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. The best value of lambda is 1.410243e-05. Lasso regression model chose the variables sched_dep_time, few carrier names, originLGA, destBoston, dewp, humid, wind_speed, precip, log*distance*.



3. Ridge Regression:

In Ridge Regression by adding a degree of bias to the regression estimates, the standard errors are reduced. It is hoped that the net effect will be to give estimates that are more reliable. Different values of lambda are used for regularization and the best value of lambda to predict the response variable. In our case, the best λ is 0.006699799.



4. Gam Modelling:

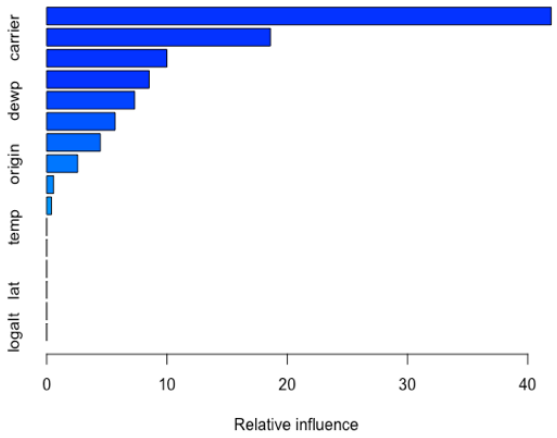
The GAM model was fit such that it is non-linear in 'dep_time', 'temp', 'dewp', etc. with 3 degrees of freedom because they are fitted using Smoothing Splines, whereas it is linear in terms of variables like 'carrier', 'origin', etc. GAMs are also a flexible and smooth technique which helps us to fit Linear Models which can be either linearly or non-linearly dependent on several predictors to capture non-linear relationships between Response and Predictors.

```
    Null Deviance: 15323.63 on 184315 degrees of freedom
Residual Deviance: 12990.38 on 184261 degrees of freedom
AIC: 34290.52

Number of Local Scoring Iterations: 3

Anova for Parametric Effects
                    Df  Sum Sq Mean Sq    F value    Pr(>F)
s(dep_date)          1     0.6    0.56     7.9064 0.0049265 **
s(sched_dep_time)    1   868.7  868.74 12322.5166 < 2.2e-16 ***
carrier             15   514.4   34.29   486.4180 < 2.2e-16 ***
origin               2   100.0   49.99   709.0436 < 2.2e-16 ***
s(logdistance)       1    19.7   19.73   279.8996 < 2.2e-16 ***
s(temp)              1     9.4    9.42   133.6031 < 2.2e-16 ***
s(dewp)              1   415.8  415.83  5898.2988 < 2.2e-16 ***
s(humid)             1    49.5   49.50   702.1529 < 2.2e-16 ***
s(wind_dir)          1    13.3   13.27   188.2044 < 2.2e-16 ***
s(wind_speed)        1    33.7   33.71   478.1157 < 2.2e-16 ***
precip               1    21.5   21.48   304.6172 < 2.2e-16 ***
s(visib)             1     0.9    0.89    12.6083 0.0003841 ***
Residuals       184261 12990.4    0.07
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Anova for Nonparametric Effects
                  Npar Df Npar F     Pr(F)
(Intercept)
s(dep_date)             3 759.55 < 2.2e-16 ***
s(sched_dep_time)       3 452.35 < 2.2e-16 ***
carrier
origin
s(logdistance)          3  75.34 < 2.2e-16 ***
s(temp)                 3 362.77 < 2.2e-16 ***
s(dewp)                 3 153.79 < 2.2e-16 ***
s(humid)                3   4.65  0.002994 **
s(wind_dir)             3  72.45 < 2.2e-16 ***
s(wind_speed)           3  13.91 4.588e-09 ***
precip
s(visib)                3  30.52 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5. Gradient Boosting Machines:

Gradient boosting involves three elements- a loss function to be optimized, a weak learner to make predictions, an additive model to add weak learners to minimize the loss function. The GBM was implemented on gaussian distribution, 1000 trees and a shrinkage factor of 0.01. Following is the plot of the influential factors.



6. SVM:

I tried fitting an SVM model but the iterations maxed out and I could not achieve the results. I even tried down-sampling the data but to no avail. I would have definitely tried computing had it been possible.

4. RESULT

Following are the results obtained from running the above models:

Methods	MSE_train	MSE_test
Lasso Regression	0.07277712	0.07233221
Ridge Regression	0.07279299	0.07235192
GAM	0.07047458	0.09557389
GBM	0.07206566	0.07181148
GLM	0.09394969	0.09359074

Table 2: MSE on Data in Part 2

The best results for training set were obtained from the GAM model with an approximate MSE of 0.070.

However, since all the models have achieved an accuracy which is nearly equal, we predict the test set on all the models to see if there is a difference.

As we can see, though GAM model gave the best result on training set, since the difference was very minor, GBM performed better on the test set.

Best MSE on test set: 0.07181148 (Gradient Boosting Machine)

5. CONCLUSION AND DISCUSSION

In this project, detailed analysis and predictions were performed on the NYC Flights 2013 data. The project aimed at predicting the departure delay in minutes. There were two approaches that could be followed: classification or regression. The regression approach was followed here. Some insights that were garnered from the data are as follows:

- LaGuardia Airport is the most punctual airport in general.
- Some airlines consistently perform poorer than others (ExpressJet (EV) has the worst delays).
- Morning flights generally have lesser delays than later part of the day.
- There is a significant amount of missing data in the dataset that could have possibly provided logical reasoning to the delays. (for example, the days showing high amounts of precipitation do not reflect in significant delays, which is a little baffling. Also, factors like precipitation should have high influence on the delay but they don't turn out to be as strong predictors).

Five types of regression models were implemented and as mentioned in the results, GBM provided the best results with an MSE of 0.07181148. The other models like Lasso and Ridge regression also performed nearly equally well. More rigorous feature selection approaches could be used to get better results. The models could also be hypertuned with early stopping to give better predictions.

6. REFERENCES

[1] https://cran.r-project.org/web/packages/nycflights13/index.html
[2] https://rna.wlu.edu/bio185/c-07-exploring-data.html
[3] http://faculty.marshall.usc.edu/gareth-james/ISL/ISLR%20Seventh%20Printing.pdf
[4] https://www.rstatisticsblog.com/data-science-in-action/lasso-regression/
[5] https://drsimonj.svbtle.com/ridge-regression-with-glmnet
[6] https://www.svm-tutorial.com/2014/10/support-vector-regression-r/

APPENDIX

Slavvy Coelho
(301359836)

1. Software Details

```
               _
platform       x86_64-apple-darwin15.6.0
arch           x86_64
os             darwin15.6.0
system         x86_64, darwin15.6.0
status
major          3
minor          6.1
year           2019
month          07
day            05
svn rev        76782
language       R
version.string R version 3.6.1 (2019-07-05)
nickname       Action of the Toes
```

2. Libraries Used

```
library(tidyverse)
library(lubridate)
library(glmnet)
library(gam)
library(magrittr)
library(gbm)
library(ISLR)
```

3. Loading Files

```
fltrain <- read_csv("/Users/slavvy/Desktop/MS/STAT/Project /Brad/fltrain.csv")
fltrain     #(20000,43)
fltest <-read_csv("/Users/slavvy/Desktop/MS/STAT/Project /Brad/fltest.csv")
fltest
```

4. Data Preprocessing – Training and Testing

```
#1. convert to factor
fl <- fltrain
for(i in 1:ncol(fl)) {
  if(typeof(fl[[i]]) == "character") {
    fl[[i]] <- factor(fl[[i]])
  }
}
```

```
#2. handling missing values
num_miss <- function(x) { sum(is.na(x)) }
sapply(fl,num_miss)

#discard few columns altogether
fl <- fl%>%
  select(-year.y,-type,-manufacturer,-model,-engines,-seats, -speed,
         -engine,-wind_gust,-pressure)
summary(fl)       #dep_delay- Max: 1301 (long tail)

#omit the rows with null values
fl <- na.omit(fl)

#3. scale the data (dep_delay)
den <- nrow(fl)+1
fl <- fl %>% mutate(dep_delay = rank(dep_delay)/den)
ggplot(fl,aes(x=dep_delay)) + geom_histogram(binwidth=.01)

#4. eliminate a few more columns
fl <- fl %>%
  mutate(dep_date = make_date(year.x,month,day)) %>%
  select(-year.x,-month,-day,-dep_time,-arr_time,-arr_delay,
         -sched_arr_time,-tailnum,-flight,-name,-air_time,
         -hour,-minute,-time_hour,-tz,-dst,-tzone) %>%
  mutate(precip = as.numeric(precip>0))
fl <- mutate(fl,logdistance = log(distance)) %>% select(-distance)
fl <- mutate(fl,logalt = log(alt)) %>% select(-alt)
```

5. Predictions

a. GAM Model

```
##1) GAM Model
form <- formula(dep_delay ~ s(dep_date) + s(sched_dep_time) +
                  carrier + origin + s(logdistance) +
                  s(temp) + s(dewp) + s(humid) + s(wind_dir) +
                  s(wind_speed) + precip + s(visib))
gam_fit <- gam(form, data=fl_tr,family=gaussian)
plot(gam_fit,se=TRUE)
gam_pred <- predict(gam_fit,newdata=fl_te)
mse_gam <- mean((fl_te$dep_delay-gam_pred)^2)  #0.07038026
abs(mse_gam - var_dd)/var_dd    #0.1534388
rmse <- function(error)
{
  sqrt(mean(error^2))
}
gam_error <- gam_fit$residuals
gam_predictionRMSE <- rmse(gam_error)   # 0.2654785
```

b. GBM Model

```
##2) Boosting

dep_date_numeric <- as.numeric(fl_tr$dep_date)
dep_date_numeric <- dep_date_numeric - mean(dep_date_numeric)
fl_tr_tem <- mutate(fl_tr,dep_date = dep_date_numeric)
gbm_fit <-gbm(dep_delay ~ .,data=fl_tr_tem,distribution="gaussian",
              n.trees = 1000, shrinkage = 0.01)
summary(gbm_fit)
dep_date_numeric <- as.numeric(fl_te$dep_date)
dep_date_numeric <- dep_date_numeric - mean(dep_date_numeric)
fl_te_tem <- mutate(fl_te,dep_date = dep_date_numeric)
gbm_pred <- predict(gbm_fit,newdata=fl_te_tem,n.trees = 1000)
mse_gbm <- mean((fl_te$dep_delay-gbm_pred)^2)  #0.07181148
abs(mse_gbm - var_dd)/var_dd   #0.1362235
gbm_predictionRMSE <- mse_gbm^0.5   # 0.2679766
```

c. GLM Model

```
##3) Logistic regression

glm_fit <- glm(dep_delay ~., data=fl_tr, family=binomial())
glm_pred <- predict(glm_fit, type="response")
mse_glm <- mean((fl_te$dep_delay-glm_pred)^2)
mse_glm       #0.09359074
abs(mse_glm - var_dd)/var_dd    #0.1257459
glm_error <- glm_fit$residuals  # same as data$Y - predictedY
glm_predictionRMSE <- mse_glm^0.5   # 0.305926
```

d. Ridge Regression

```
##4) Ridge
set.seed(1)
grid=10^seq(10,-2,length=100)
x_train <- model.matrix(dep_delay ~ ., data = fl_tr)[,-1]
x_test <- model.matrix(dep_delay ~ ., data = fl_te)[,-1]
y_train<-fl_tr$dep_delay
cv.ridge <- cv.glmnet(x_train, y_train, alpha = 0)
plot(cv.ridge)
bestlam=cv.ridge$lambda.min
bestlam      #0.006656471
fit.ridge <- glmnet(x_train, y_train, alpha = 0, lambda = grid, thresh = 1e-12)
pred.ridge <- predict(fit.ridge, s = bestlam, newx = x_test)
mse_ridge = mean((pred.ridge - fl_te$dep_delay)^2)    #0.07235192
abs(mse_ridge - var_dd)/var_dd       #0.1297229
ridge_predictionRMSE <- (mse_ridge)^0.5   #0.2689831
```

e. Lasso Regression

```
##5) Lasso
set.seed(23)
grid=10^seq(10,-2,length=100)
xmat <- model.matrix(dep_delay ~ ., data=fl_tr)[, -1]
cv.lasso <- cv.glmnet(xmat, fl_tr$dep_delay, alpha = 1)
plot(cv.lasso)
bestlam <- cv.lasso$lambda.min
bestlam     #1.687658e-05
fit.lasso <- glmnet(xmat, fl_tr$dep_delay, alpha = 1,lambda = bestlam)
pred_lasso <- predict(fit.lasso, s = bestlam, newx = x_test)
mse_lasso = mean((pred_lasso - fl_te$dep_delay)^2)   # 0.07233221
abs(mse_lasso - var_dd)/var_dd      #0.1299601
lasso_predictionRMSE <- (mse_lasso)^0.5     #0.2689465
```