knit

Vincent Chiu

4/1/2019

Contents

1. Introduction (brief)
2. Data (brief)

3. Methods
4. Results
5. Conclusions and Discussion

1 Introduction

The goal of this project is to predict the response variable, departure delays for a particular flight given the explanatory variables.

2 Data

The dataset consists of information about all the flights leaving from New York City in 2013. The dataset contains 43 variables in total. The dataset is an algamation of several datasets including datasets containing information on weather, the airports, the flights, and the models of airplanes. The training dataset provided to us contains 200,000 observations.

3 Methods:

3.1 Data Preprocessing

I performed data preprocessing. My data preprocessing steps include the following: 1. Dropping columns that contain data from after the planes' departure which may leak information about the response variable dep_delay. 2. Dropping columns with too many NAs. 3. Impute NAs for the remaining columns. 4. Scaling the data to work well with methods like lasso regression. 5. Only kept data which had a departure delay of less than 30 minutes late, which reduced the dataset from 200,000 rows to approximately 170,000.

3.2 Modelling

Initially, I used the most basic cross validation technique where I have a training dataset and a holdout test dataset. I split the original data into a ratio of 2/3 train and 1/3 of the data for test. I believe that this split gives enough data for the models to learn while 1/3 is enough data for me to get an accurate assessment of the error. k-folds cross validation was not initially used in order to save on compute time as I was only exploring the models. k-folds cross validation would increase training time for the models by a factor of k.

3.3 Basic Models

dep_delay is the number of minutes that the plane either departs early or late. Negative numbers are for early departures and positive numbers are for the number of minutes the plane is late. First, I used a basic model of simply predicting the dep_delay to always be 0. This was done to establish baseline performance. This model had an root mean squared error (RMSE) of 8.30571. TODO The model in which I predicted the mean for all the predictions had an RMSE of TODO.

3.4 Linear Regression

Then I tried linear regression with dep_delay as the response variables and all the other remaining variables as the explanatory variables. This model was better than predicting the mean with an RMSE of TODO. This suggests that there is some relationship between the dep_delay and the explanatory variables.

3.5 GBM

Aftewards, I tried a Generalized Boosted Regression Model (GBM). This model had the lowest RMSE on the test dataset after I tuned it to have a shrinkage of 0.01 and around 16,000 trees. Shrinkage is proportional to the learning rate. 16,000 trees is the number of trees used in the model. Each iteration uses 1 tree, so 16,000 trees also refers to the number of iterations. According to the vignette, the rmse can always be improved by decreasing shrinkage but this provides diminishing returns. A good strategy would be to pick a small shrinkage that balances performance and compute time. Then with this fixed shrinkage value, increase the number of trees until you get diminishing returns.

4 Results

In regression and gbm, I found different features to be important. For the best gbm model, dest which refers to which airport a given plane is going to was the most important feature. However, the one hot encoding versions of carrier were the most important features for regression. On the other hand, dest does appear as an important feature in linear regression as well but it is not the most important feature. I surmise that if we can somehow sum up all the contributions from each of the one hot encoded variables derived from dest then, it might appear as the most important feature for linear regression as well. We can try Anova in order to measure the statistical significance of dest. Performing anova on comparing linear regression model with and without dest, it was determined that due to the low p-value of 0.0001863 associated with having dest that keeping at least one of the one hot categorical variables derived from dest is beneficial for the linear regression model.

TODO: try interaction terms , try anova.

5 Conclusion and Discussion

Conclusion: In conclusion, out of the methods that we covered in class, I found gradient boosted models to provide the best performance based on having the lowest root mean squared error on the hold out test set.

Based on the relative influence scores provided by the gbm, some of the most important feature variables include dest, model, and sched_dep_time_num_minute.

The dest column contains the airport code for where a given flight is flying to. Based on my run of gbm with a shrinkage of 0.01 and 16834 trees, dest was the most important feature with 49.56 relative influence.("Gradient Boosting Machines · UC Business Analytics R Programming Guide" 2019).

TODO; think about removing points that are outliers aka points with high cook's distance consider removing outliers in train but not in test, then use k-folds cross validation on test.

TODO: remove points that are outliers ie dep_delay > 200 or 300 etc. or remove less than x number of points. then use k-folds cross validation on cross validation set where no points were removed. can repeat k-folds for different seeds. can just try this on my quickest model, ie linear regression. should be bowl shape vs rmse vs. number of points removed. theoretically

I also considered removing based on cook's distance but this took too long to compute.

5 folds with 10 different random seeds

6 have train, CV and test setå

7 1/3 train, 1/3 CV, 1/3 test

8 2/3% train, 1/3%CV, wait for prof test set

try lasso regression

9 Code

9.1 Loading Libraries

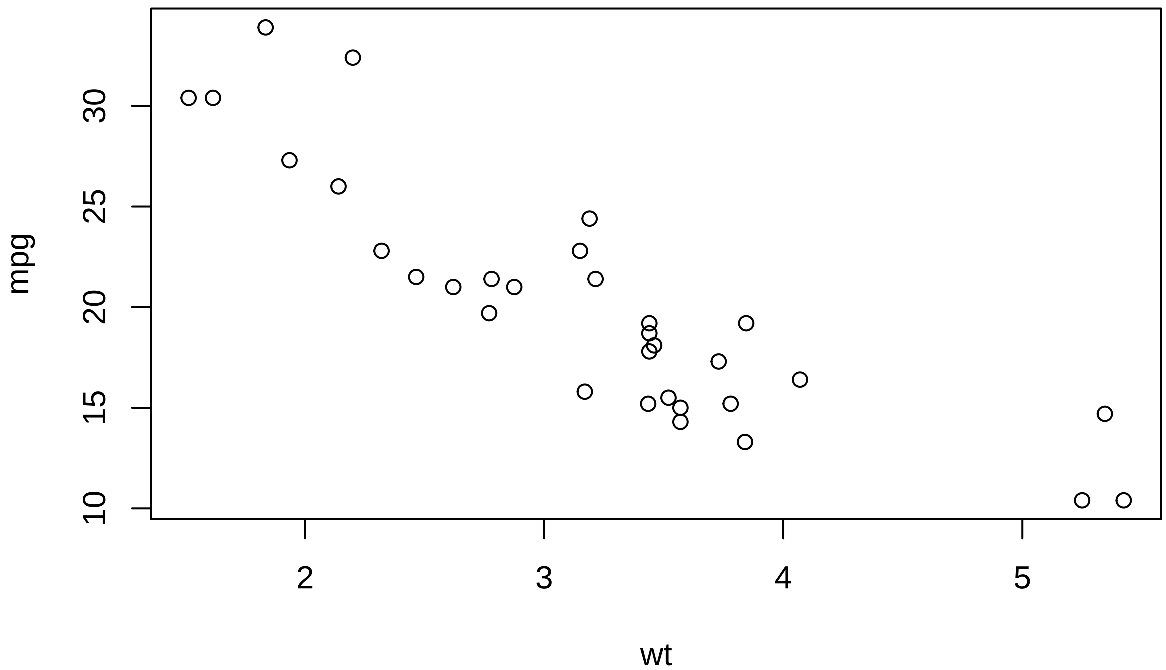
```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------------- tidyverse 1.2.1 -

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------ tidyverse_conflicts() -
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
attach(mtcars)
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
plot(wt, mpg)
```



9.2 Loading the data

```
library(nycflights13)
library(Hmisc)
```

```
## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
set.seed(42)
original_data <- read_csv("fltrain.csv.gz")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   carrier = col_character(),
##   tailnum = col_character(),
##   origin = col_character(),
##   dest = col_character(),
##   time_hour = col_datetime(format = ""),
##   name = col_character(),
```

```
##   dst = col_character(),
##   tzone = col_character(),
##   type = col_character(),
##   manufacturer = col_character(),
##   model = col_character(),
##   engine = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
DF <- original_data
```

10 turning all columns with datatype characters to factors.

```
DF[sapply(DF, is.character)] <- lapply(DF[sapply(DF, is.character)],
                                       as.factor)
DF$flight <- as.factor(DF$flight)
str(DF)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 200000 obs. of  43 variables:
##  $ year.x        : num  2013 2013 2013 2013 2013 ...
##  $ month         : num  11 10 12 11 10 11 9 12 11 3 ...
##  $ day           : num  7 30 18 20 21 7 29 21 7 31 ...
##  $ dep_time      : num  600 1252 1723 2029 1620 ...
##  $ sched_dep_time: num  600 1250 1715 2030 1625 ...
##  $ dep_delay     : num  0 2 8 -1 -5 -8 -10 -4 0 -8 ...
##  $ arr_time      : num  826 1356 2008 2141 1818 ...
##  $ sched_arr_time: num  825 1400 2020 2205 1831 ...
##  $ arr_delay     : num  1 -4 -12 -24 -13 -18 -10 -16 4 -11 ...
##  $ carrier       : Factor w/ 16 levels "9E","AA","AS",..: 15 2 5 15 5 4 6 6 1 13 ...
##  $ flight        : Factor w/ 3672 levels "1","2","3","4",..: 1525 147 1400 2343 1860 24 3083 3351 20
##  $ tailnum       : Factor w/ 3957 levels "D942DN","N0EGMQ",..: 1437 1226 836 565 756 2459 204 2890 6
##  $ origin        : Factor w/ 3 levels "EWR","JFK","LGA": 3 2 3 1 3 1 1 3 2 3 ...
##  $ dest          : Factor w/ 104 levels "ABQ","ACK","ALB",..: 5 12 54 55 33 54 59 59 27 29 ...
##  $ air_time      : num  123 44 133 107 90 136 110 118 101 47 ...
##  $ distance      : num  762 187 950 711 502 937 725 738 589 214 ...
##  $ hour          : num  6 12 17 20 16 9 15 15 16 17 ...
##  $ minute        : num  0 50 15 30 25 0 29 30 50 0 ...
##  $ time_hour     : POSIXct, format: "2013-11-07 11:00:00" "2013-10-30 16:00:00" ...
##  $ temp          : num  63 59 34 37 63 ...
##  $ dewp          : num  55.9 46.9 17.1 18 41 ...
##  $ humid         : num  77.8 64.2 49.5 45.6 44.5 ...
##  $ wind_dir      : num  210 240 270 20 160 240 180 190 320 140 ...
##  $ wind_speed    : num  13.81 9.21 17.26 5.75 13.81 ...
##  $ wind_gust     : num  NA NA 21.9 NA NA ...
##  $ precip        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ pressure      : num  1011 1025 1020 1036 1017 ...
##  $ visib         : num  10 10 10 10 10 10 10 10 10 10 ...
##  $ name          : Factor w/ 100 levels "Akron Canton Regional Airport",..: 37 31 67 17 26 67 32 32
##  $ lat           : num  33.6 42.4 28.4 41.8 42.2 ...
##  $ lon           : num  -84.4 -71 -81.3 -87.8 -83.4 ...
##  $ alt           : num  1026 19 96 620 645 ...
##  $ tz            : num  -5 -5 -5 -6 -5 -5 -6 -6 -5 -5 ...
```

```
##  $ dst          : Factor w/ 2 levels "A","N": 1 1 1 1 1 1 1 1 1 1 ...
##  $ tzone        : Factor w/ 7 levels "America/Anchorage",..: 5 5 5 2 5 5 2 2 5 5 ...
##  $ year.y       : num  2001 NA 2002 2006 1992 ...
##  $ type         : Factor w/ 3 levels "Fixed wing multi engine",..: 1 NA 1 1 1 1 1 1 1 1 ...
##  $ manufacturer : Factor w/ 35 levels "AGUSTA SPA","AIRBUS",..: 10 NA 2 10 3 2 18 11 11 3 ...
##  $ model        : Factor w/ 126 levels "150","172E","172M",..: 37 NA 80 37 84 88 106 98 99 79 ...
##  $ engines      : num  2 NA 2 2 2 2 2 2 2 2 ...
##  $ seats        : num  140 NA 145 140 182 200 55 80 95 179 ...
##  $ speed        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ engine       : Factor w/ 6 levels "4 Cycle","Reciprocating",..: 3 NA 3 3 4 3 3 3 3 3 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   year.x = col_double(),
##   ..   month = col_double(),
##   ..   day = col_double(),
##   ..   dep_time = col_double(),
##   ..   sched_dep_time = col_double(),
##   ..   dep_delay = col_double(),
##   ..   arr_time = col_double(),
##   ..   sched_arr_time = col_double(),
##   ..   arr_delay = col_double(),
##   ..   carrier = col_character(),
##   ..   flight = col_double(),
##   ..   tailnum = col_character(),
##   ..   origin = col_character(),
##   ..   dest = col_character(),
##   ..   air_time = col_double(),
##   ..   distance = col_double(),
##   ..   hour = col_double(),
##   ..   minute = col_double(),
##   ..   time_hour = col_datetime(format = ""),
##   ..   temp = col_double(),
##   ..   dewp = col_double(),
##   ..   humid = col_double(),
##   ..   wind_dir = col_double(),
##   ..   wind_speed = col_double(),
##   ..   wind_gust = col_double(),
##   ..   precip = col_double(),
##   ..   pressure = col_double(),
##   ..   visib = col_double(),
##   ..   name = col_character(),
##   ..   lat = col_double(),
##   ..   lon = col_double(),
##   ..   alt = col_double(),
##   ..   tz = col_double(),
##   ..   dst = col_character(),
##   ..   tzone = col_character(),
##   ..   year.y = col_double(),
##   ..   type = col_character(),
##   ..   manufacturer = col_character(),
##   ..   model = col_character(),
##   ..   engines = col_double(),
##   ..   seats = col_double(),
##   ..   speed = col_double(),
```

```
##   ..   engine = col_character()
##   .. )
```

11 Methods

11.1 Preprocessing

Data preprocessing steps include the following: - Dropping columns that contain data from after the planes' departure which may leak information about the response variable dep_delay. - Dropping columns with too many NAs. - Impute NAs for the remaining columns. - Scaling the data to work well with methods like lasso regression.

11.2 - Dropping columns that contain data from after the planes' departure which may leak information about the response variable dep_delay.

dropping the columns "dep_time", "arr_time", "air_time", "arr_delay", because that leaks the response variable. dropping column "year.x" because all the values are 2013 dropping tailnum because it produces too many dummy variable columns for one hot encoding.

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
DF$sched_arr_time_posix <- as.POSIXct(str_pad(as.character(DF$sched_arr_time), 4, pad="0"),format="%H%M
DF$sched_arr_time_hour <- hour(DF$sched_arr_time_posix)
DF$sched_arr_time_minute <- minute(DF$sched_arr_time_posix)

#num minute is number of minutes since start of day for scheduled arrival time
DF$sched_arr_time_num_minute <- 60*DF$sched_arr_time_hour + DF$sched_arr_time_minute

DF$sched_dep_time_posix <- as.POSIXct(str_pad(as.character(DF$sched_dep_time),4 , pad="0"),format="%H%M
DF$sched_dep_time_hour <- hour(DF$sched_dep_time_posix)
DF$sched_dep_time_minute <- minute(DF$sched_dep_time_posix)
#num minute is number of minutes since start of day for scheduled depival time
DF$sched_dep_time_num_minute <- 60*DF$sched_dep_time_hour + DF$sched_dep_time_minute
```

```
select(original_data, time_hour, sched_dep_time, sched_arr_time, tz, tzone)
```

```
## # A tibble: 200,000 x 5
##    time_hour           sched_dep_time sched_arr_time    tz tzone
##    <dttm>                       <dbl>          <dbl> <dbl> <chr>
##  1 2013-11-07 11:00:00            600            825    -5 America/New_York
##  2 2013-10-30 16:00:00           1250           1400    -5 America/New_York
##  3 2013-12-18 22:00:00           1715           2020    -5 America/New_York
##  4 2013-11-21 01:00:00           2030           2205    -6 America/Chicago
##  5 2013-10-21 20:00:00           1625           1831    -5 America/New_York
##  6 2013-11-07 14:00:00            900           1157    -5 America/New_York
##  7 2013-09-29 19:00:00           1529           1649    -6 America/Chicago
```

```
##  8 2013-12-21 20:00:00           1530           1710    -6 America/Chicago
##  9 2013-11-07 21:00:00           1650           1906    -5 America/New_York
## 10 2013-03-31 21:00:00           1700           1821    -5 America/New_York
## # ... with 199,990 more rows
```

```
select(DF, sched_arr_time, sched_arr_time_hour)
```

```
## # A tibble: 200,000 x 2
##    sched_arr_time sched_arr_time_hour
##             <dbl>               <int>
##  1            825                   8
##  2           1400                  14
##  3           2020                  20
##  4           2205                  22
##  5           1831                  18
##  6           1157                  11
##  7           1649                  16
##  8           1710                  17
##  9           1906                  19
## 10           1821                  18
## # ... with 199,990 more rows
```

```
DF$sched_air_time <- DF$sched_arr_time_posix - DF$sched_dep_time_posix
drops <- c('sched_arr_time_posix', 'sched_arr_time_hour', 'sched_dep_time_posix', 'sched_dep_time_hour'
DF <- DF[ , !(names(DF) %in% drops)]
```

```
drops <- c("dep_time", "arr_time", "air_time", "arr_delay", "year.x", 'tailnum')
DF <- DF[ , !(names(DF) %in% drops)]
```

```
DF
```

```
## # A tibble: 200,000 x 37
##    month   day dep_delay carrier flight origin dest  distance  temp  dewp humid
##    <dbl> <dbl>     <dbl> <fct>   <fct>  <fct>  <fct>    <dbl> <dbl> <dbl> <dbl>
##  1    11     7         0 WN      1716   LGA    ATL        762  63.0  55.9  77.8
##  2    10    30         2 AA      178    JFK    BOS        187  59    46.9  64.2
##  3    12    18         8 DL      1585   LGA    MCO        950  34.0  17.1  49.5
##  4    11    20        -1 WN      3494   EWR    MDW        711  37.0  18.0  45.6
##  5    10    21        -5 DL      2231   LGA    DTW        502  63.0  41    44.5
##  6    11     7        -8 B6      27     EWR    MCO        937  64.4  55.4  77.3
##  7     9    29       -10 EV      4580   EWR    MKE        725  69.1  53.1  56.7
##  8    12    21        -4 EV      5207   LGA    MKE        738  57.9  46.0  64.5
##  9    11     7         0 9E      2910   JFK    CVG        589  53.6  48.2  81.9
## 10     3    31        -8 US      2183   LGA    DCA        214  51.1  36.0  56.0
## # ... with 199,990 more rows, and 26 more variables: wind_dir <dbl>,
## #   wind_speed <dbl>, wind_gust <dbl>, precip <dbl>, pressure <dbl>,
## #   visib <dbl>, name <fct>, lat <dbl>, lon <dbl>, alt <dbl>, tz <dbl>,
## #   dst <fct>, tzone <fct>, year.y <dbl>, type <fct>, manufacturer <fct>,
## #   model <fct>, engines <dbl>, seats <dbl>, speed <dbl>, engine <fct>,
## #   sched_arr_time_minute <int>, sched_arr_time_num_minute <dbl>,
## #   sched_dep_time_minute <int>, sched_dep_time_num_minute <dbl>,
## #   sched_air_time <drtn>
```

```
## Remove columns with more than 50% NA
DF <- DF[, -which(colMeans(is.na(DF)) > 0.5)]
```

```r
DF$sched_air_time <- as.numeric(DF$sched_air_time)
library(imputeMissings)
```

```
##
## Attaching package: 'imputeMissings'
## The following object is masked from 'package:Hmisc':
##
##     impute
## The following object is masked from 'package:dplyr':
##
##     compute
```

```r
impute_model <- imputeMissings::compute(DF, method="median/mode")
impute_model
```

```
## $month
## [1] 7
##
## $day
## [1] 16
##
## $dep_delay
## [1] -2
##
## $carrier
## [1] "UA"
##
## $flight
## [1] "15"
##
## $origin
## [1] "EWR"
##
## $dest
## [1] "ATL"
##
## $distance
## [1] 872
##
## $temp
## [1] 57.2
##
## $dewp
## [1] 42.8
##
## $humid
## [1] 57.69
##
## $wind_dir
## [1] 220
##
## $wind_speed
## [1] 10.35702
```

```
##
## $precip
## [1] 0
##
## $pressure
## [1] 1017.5
##
## $visib
## [1] 10
##
## $name
## [1] "Hartsfield Jackson Atlanta Intl"
##
## $lat
## [1] 36.09775
##
## $lon
## [1] -83.35339
##
## $alt
## [1] 433
##
## $tz
## [1] -5
##
## $dst
## [1] "A"
##
## $tzone
## [1] "America/New_York"
##
## $year.y
## [1] 2002
##
## $type
## [1] "Fixed wing multi engine"
##
## $manufacturer
## [1] "BOEING"
##
## $model
## [1] "A320-232"
##
## $engines
## [1] 2
##
## $seats
## [1] 149
##
## $engine
## [1] "Turbo-fan"
##
## $sched_arr_time_minute
## [1] 30
```

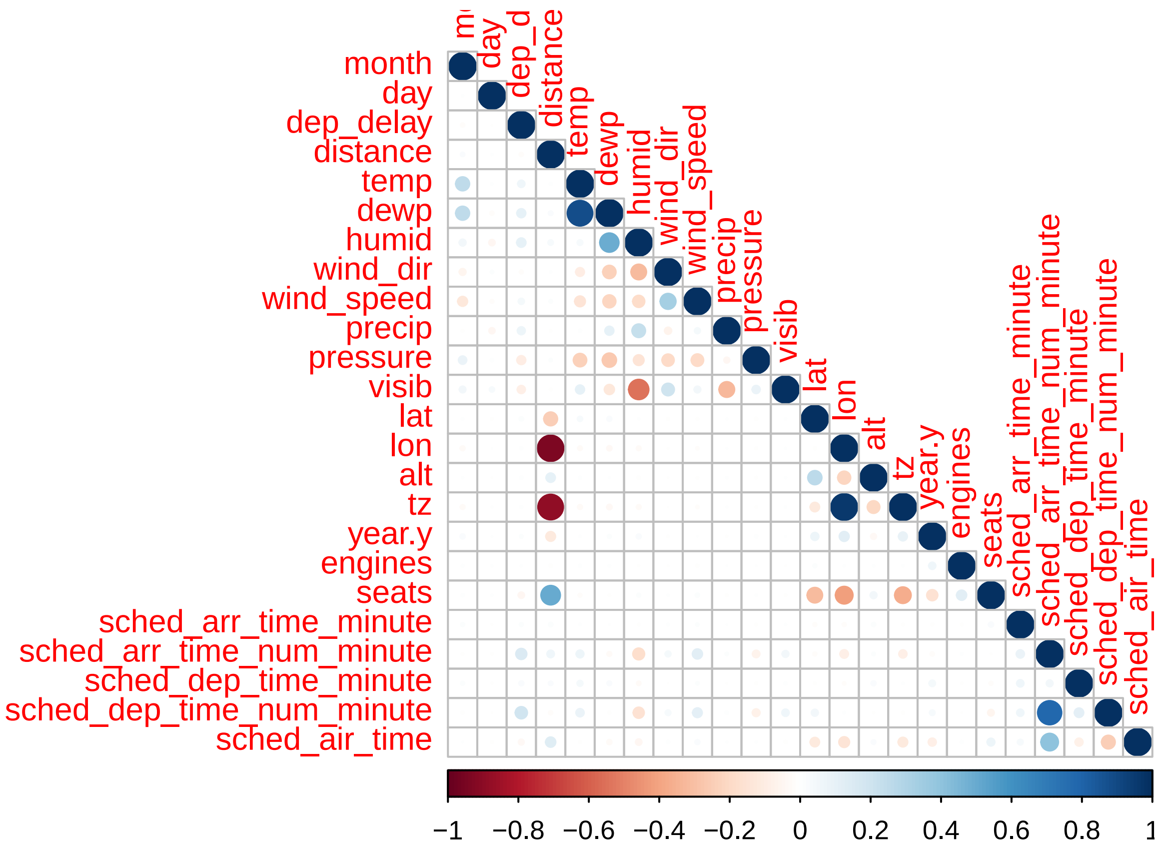
```
##
## $sched_arr_time_num_minute
## [1] 957
##
## $sched_dep_time_minute
## [1] 29
##
## $sched_dep_time_num_minute
## [1] 839
##
## $sched_air_time
## [1] 139
```

```
DF <- impute(DF, object=impute_model, flag=TRUE)
DF <- DF[!duplicated(as.list(DF))]  #remove all redundant flag columns that are identical to each other
```

```
numeric_only_df <- dplyr::select_if(DF, is.numeric)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
corrplot(cor(numeric_only_df), type = 'lower')
```



12 try features scaling

```r
dep_delay_vec <- DF$dep_delay
DF$dep_delay <- NULL
head(DF)
```

```
##   month day carrier flight origin dest distance  temp  dewp humid wind_dir
## 1    11   7      WN   1716    LGA  ATL      762 62.96 55.94 77.83      210
## 2    10  30      AA    178    JFK  BOS      187 59.00 46.94 64.22      240
## 3    12  18      DL   1585    LGA  MCO      950 33.98 17.06 49.51      270
## 4    11  20      WN   3494    EWR  MDW      711 37.04 17.96 45.58       20
## 5    10  21      DL   2231    LGA  DTW      502 62.96 41.00 44.47      160
## 6    11   7      B6     27    EWR  MCO      937 64.40 55.40 77.29      240
##   wind_speed precip pressure visib                              name      lat
## 1   13.80936      0   1011.0    10    Hartsfield Jackson Atlanta Intl 33.63672
## 2    9.20624      0   1024.9    10 General Edward Lawrence Logan Intl 42.36435
## 3   17.26170      0   1019.8    10                       Orlando Intl 28.42939
## 4    5.75390      0   1035.6    10                Chicago Midway Intl 41.78597
## 5   13.80936      0   1016.9    10             Detroit Metro Wayne Co 42.21244
## 6   16.11092      0   1017.5    10                       Orlando Intl 28.42939
##         lon  alt tz dst            tzone year.y                    type
## 1 -84.42807 1026 -5   A America/New_York   2001 Fixed wing multi engine
## 2 -71.00518   19 -5   A America/New_York   2002 Fixed wing multi engine
## 3 -81.30899   96 -5   A America/New_York   2002 Fixed wing multi engine
## 4 -87.75242  620 -6   A  America/Chicago   2006 Fixed wing multi engine
## 5 -83.35339  645 -5   A America/New_York   1992 Fixed wing multi engine
## 6 -81.30899   96 -5   A America/New_York   2006 Fixed wing multi engine
##       manufacturer    model engines seats    engine sched_arr_time_minute
## 1           BOEING  737-7H4       2   140 Turbo-fan                    25
## 2           BOEING A320-232       2   149 Turbo-fan                     0
## 3           AIRBUS A319-114       2   145 Turbo-fan                    20
## 4           BOEING  737-7H4       2   140 Turbo-fan                     5
## 5 AIRBUS INDUSTRIE A320-211       2   182 Turbo-jet                    31
## 6           AIRBUS A320-232       2   200 Turbo-fan                    57
##   sched_arr_time_num_minute sched_dep_time_minute sched_dep_time_num_minute
## 1                       505                     0                       360
## 2                       840                    50                       770
## 3                      1220                    15                      1035
## 4                      1325                    30                      1230
## 5                      1111                    25                       985
## 6                       717                     0                       540
##   sched_air_time dep_delay_flag temp_flag wind_dir_flag wind_speed_flag
## 1            145              0         0             0               0
## 2             70              0         0             0               0
## 3            185              0         0             0               0
## 4             95              0         0             0               0
## 5            126              0         0             0               0
## 6            177              0         0             0               0
##   precip_flag pressure_flag name_flag year.y_flag type_flag
## 1           0             0         0           0         0
## 2           0             0         0           1         1
## 3           0             0         0           0         0
## 4           0             0         0           0         0
## 5           0             0         0           0         0
## 6           0             1         0           0         0
```

```r
library(dplyr)
DF <- DF %>% mutate_if(is.numeric, scale)
head(DF)
```

```
##     month        day carrier flight origin dest   distance       temp
## 1 1.30322 -0.9929373      WN   1716    LGA  ATL -0.3777852  0.3339858
## 2 1.01019  1.6325235      AA    178    JFK  BOS -1.1644742  0.1127815
## 3 1.59625  0.2627179      DL   1585    LGA  MCO -0.1205721 -1.2848272
## 4 1.30322  0.4910188      WN   3494    EWR  MDW -0.4475611 -1.1138966
## 5 1.01019  0.6051693      DL   2231    LGA  DTW -0.7335054  0.3339858
## 6 1.30322 -0.9929373      B6     27    EWR  MCO -0.1383581  0.4144237
##         dewp      humid    wind_dir wind_speed     precip    pressure     visib
## 1  0.7418623  0.9315583  0.07717317  0.4871566 -0.1492223 -0.96830167 0.3664282
## 2  0.2753242  0.2375566  0.36735789 -0.3415806 -0.1492223  1.01596006 0.3664282
## 3 -1.2735821 -0.5125364  0.65754261  1.1087096 -0.1492223  0.28792159 0.3664282
## 4 -1.2269283 -0.7129351 -1.76066338 -0.9631336 -0.1492223  2.54341334 0.3664282
## 5 -0.0325909 -0.7695363 -0.40646802  0.4871566 -0.1492223 -0.12606108 0.3664282
## 6  0.7138700  0.9040226  0.36735789  0.9015253 -0.1492223 -0.04040949 0.3664282
##                                  name        lat       lon         alt
## 1    Hartsfield Jackson Atlanta Intl -0.4207546 0.3298674  0.48364704
## 2 General Edward Lawrence Logan Intl  1.1190951 1.2378347 -0.60619417
## 3                       Orlando Intl -1.3395034 0.5408516 -0.52285973
## 4                 Chicago Midway Intl  1.0170501 0.1049977  0.04424731
## 5              Detroit Metro Wayne Co  1.0922942 0.4025621  0.07130394
## 6                       Orlando Intl -1.3395034 0.5408516 -0.52285973
##           tz dst            tzone      year.y                    type
## 1  0.6826595   A America/New_York -0.08500492 Fixed wing multi engine
## 2  0.6826595   A America/New_York  0.08617407 Fixed wing multi engine
## 3  0.6826595   A America/New_York  0.08617407 Fixed wing multi engine
## 4 -0.2514221   A  America/Chicago  0.77089000 Fixed wing multi engine
## 5  0.6826595   A America/New_York -1.62561576 Fixed wing multi engine
## 6  0.6826595   A America/New_York  0.77089000 Fixed wing multi engine
##       manufacturer    model    engines      seats    engine
## 1           BOEING  737-7H4 0.05879311 0.02232546 Turbo-fan
## 2           BOEING A320-232 0.05879311 0.15869100 Turbo-fan
## 3           AIRBUS A319-114 0.05879311 0.09808410 Turbo-fan
## 4           BOEING  737-7H4 0.05879311 0.02232546 Turbo-fan
## 5 AIRBUS INDUSTRIE A320-211 0.05879311 0.65869797 Turbo-jet
## 6           AIRBUS A320-232 0.05879311 0.93142905 Turbo-fan
##   sched_arr_time_minute sched_arr_time_num_minute sched_dep_time_minute
## 1            -0.2348938                -1.4325947           -1.36042229
## 2            -1.6716145                -0.3129587            1.23408583
## 3            -0.5222379                 0.9570761           -0.58206985
## 4            -1.3842703                 1.3080068            0.19628258
## 5             0.1099192                 0.5927766           -0.06316823
## 6             1.6041087                -0.7240489           -1.36042229
##   sched_dep_time_num_minute sched_air_time dep_delay_flag temp_flag
## 1                -1.6236293     0.14883297              0         0
## 2                -0.1673447    -0.24317221              0         0
## 3                 0.7739125     0.35790240              0         0
## 4                 1.4665357    -0.11250381              0         0
## 5                 0.5963168     0.04952499              0         0
## 6                -0.9842849     0.31608852              0         0
##   wind_dir_flag wind_speed_flag precip_flag pressure_flag name_flag year.y_flag
```

```
## 1              0                0           0            0         0           0
## 2              0                0           0            0         0           1
## 3              0                0           0            0         0           0
## 4              0                0           0            0         0           0
## 5              0                0           0            0         0           0
## 6              0                0           0            1         0           0
##   type_flag
## 1         0
## 2         1
## 3         0
## 4         0
## 5         0
## 6         0
```

```
DF$dep_delay <- dep_delay_vec
```

#take out extreme departure delays

```
DF<-DF[DF$dep_delay < 30,]
```

```
set.seed(42)
DF$flight <- NULL
train_index <- sample(1:nrow(DF),size=2*nrow(DF)/3,replace=FALSE)
train_df <- DF[train_index,]
test_df <- DF[-train_index,]
```

13 predicting 0

```
rmse = mean((test_df$dep_delay-0)^2) %>% sqrt()
rmse
```

```
## [1] 8.30571
```

14 predicting the mean

```
rmse = mean((test_df$dep_delay-mean(train_df$dep_delay))^2)%>% sqrt()
rmse
```

```
## [1] 8.299767
```

15 predicting the median

```
rmse = mean((test_df$dep_delay-median(train_df$dep_delay))^2)%>% sqrt()
rmse
```

```
## [1] 8.469257
```

16 linear regression with dep

```
model <- lm(dep_delay ~ ., data=train_df)
model_without_dep <-  lm(dep_delay ~ .-dest, data=train_df)
anova(model, model_without_dep)
```

```
## Analysis of Variance Table
##
## Model 1: dep_delay ~ month + day + carrier + origin + dest + distance +
##     temp + dewp + humid + wind_dir + wind_speed + precip + pressure +
##     visib + name + lat + lon + alt + tz + dst + tzone + year.y +
##     type + manufacturer + model + engines + seats + engine +
##     sched_arr_time_minute + sched_arr_time_num_minute + sched_dep_time_minute +
##     sched_dep_time_num_minute + sched_air_time + dep_delay_flag +
##     temp_flag + wind_dir_flag + wind_speed_flag + precip_flag +
##     pressure_flag + name_flag + year.y_flag + type_flag
## Model 2: dep_delay ~ (month + day + carrier + origin + dest + distance +
##     temp + dewp + humid + wind_dir + wind_speed + precip + pressure +
##     visib + name + lat + lon + alt + tz + dst + tzone + year.y +
##     type + manufacturer + model + engines + seats + engine +
##     sched_arr_time_minute + sched_arr_time_num_minute + sched_dep_time_minute +
##     sched_dep_time_num_minute + sched_air_time + dep_delay_flag +
##     temp_flag + wind_dir_flag + wind_speed_flag + precip_flag +
##     pressure_flag + name_flag + year.y_flag + type_flag) - dest
##   Res.Df     RSS Df Sum of Sq     F    Pr(>F)
## 1 113488 7240975
## 2 113491 7242239 -3   -1263.7 6.602 0.0001863 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary <- round(summary(model)$coefficients,6)
sorteddf <- summary[order(summary[,ncol(summary)]),]
head(sorteddf)
```

```
##              Estimate Std. Error    t value Pr(>|t|)
## carrierFL    4.615319   0.887510   5.200301        0
## carrierUA    2.440853   0.484369   5.039237        0
## originJFK    1.037568   0.117779   8.809419        0
## wind_speed   0.291278   0.027929  10.429081        0
## precip       0.206814   0.029257   7.068777        0
## pressure    -0.294549   0.027059 -10.885462        0
```

```
head(sorteddf)
```

```
##              Estimate Std. Error    t value Pr(>|t|)
## carrierFL    4.615319   0.887510   5.200301        0
## carrierUA    2.440853   0.484369   5.039237        0
## originJFK    1.037568   0.117779   8.809419        0
## wind_speed   0.291278   0.027929  10.429081        0
## precip       0.206814   0.029257   7.068777        0
## pressure    -0.294549   0.027059 -10.885462        0
```

```
lm_test_df <- test_df

in_test_but_not_train <- setdiff(unique(lm_test_df$model), unique(train_df$model))
```

```r
lm_test_df <- lm_test_df[ !lm_test_df$model %in% in_test_but_not_train, ]

in_test_but_not_train <- setdiff(unique(lm_test_df$dest), unique(train_df$dest))
lm_test_df <- lm_test_df[ !lm_test_df$dest %in% in_test_but_not_train, ]

preds = predict(model, newdata=lm_test_df)
```

```
## Warning in predict.lm(model, newdata = lm_test_df): prediction from a rank-
## deficient fit may be misleading
```

```r
rmse = sqrt(mean((lm_test_df$dep_delay - preds)^2))
rmse
```

```
## [1] 7.989994
```

17 gbm

```r
set.seed(42)
library(gbm)
```

```
## Loaded gbm 2.1.5
```

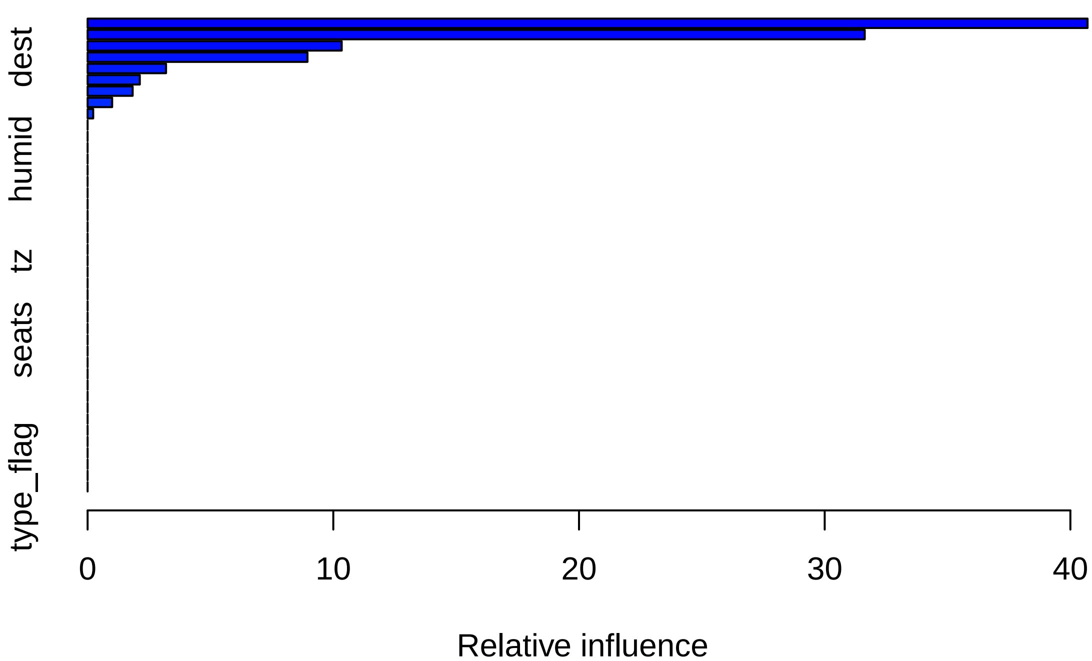
```r
model <- gbm(dep_delay ~ ., data=train_df,
              n.trees=1000, shrinkage=0.003) # default shrinkage = 0.1
```

```
## Distribution not specified, assuming gaussian ...
```

```r
preds = predict(model, newdata=test_df, n.trees=1000)
rmse = sqrt(mean((test_df$dep_delay - preds)^2))
rmse
```

```
## [1] 8.087401
```

```r
summary(model)
```



```
##                                               var    rel.inf
## sched_dep_time_num_minute sched_dep_time_num_minute 40.6986232
## model                                           model 31.6301128
## carrier                                       carrier 10.3426814
## dest                                             dest  8.9487938
## sched_arr_time_num_minute sched_arr_time_num_minute  3.1929834
## origin                                         origin  2.1249675
## month                                           month  1.8339690
## dewp                                             dewp  0.9995901
## precip                                         precip  0.2282788
## day                                               day  0.0000000
## distance                                     distance  0.0000000
## temp                                             temp  0.0000000
## humid                                           humid  0.0000000
## wind_dir                                     wind_dir  0.0000000
## wind_speed                                 wind_speed  0.0000000
## pressure                                     pressure  0.0000000
## visib                                           visib  0.0000000
## name                                             name  0.0000000
## lat                                               lat  0.0000000
## lon                                               lon  0.0000000
## alt                                               alt  0.0000000
## tz                                                 tz  0.0000000
## dst                                               dst  0.0000000
## tzone                                           tzone  0.0000000
## year.y                                         year.y  0.0000000
## type                                             type  0.0000000
## manufacturer                             manufacturer  0.0000000
## engines                                       engines  0.0000000
## seats                                           seats  0.0000000
## engine                                         engine  0.0000000
## sched_arr_time_minute         sched_arr_time_minute  0.0000000
## sched_dep_time_minute         sched_dep_time_minute  0.0000000
## sched_air_time                       sched_air_time  0.0000000
## dep_delay_flag                       dep_delay_flag  0.0000000
## temp_flag                                 temp_flag  0.0000000
## wind_dir_flag                         wind_dir_flag  0.0000000
## wind_speed_flag                     wind_speed_flag  0.0000000
## precip_flag                             precip_flag  0.0000000
## pressure_flag                         pressure_flag  0.0000000
## name_flag                                 name_flag  0.0000000
## year.y_flag                             year.y_flag  0.0000000
## type_flag                                 type_flag  0.0000000
```

Here, you can see the relative influence for each variable for gbm.
For a gbm, the improvement in the splitting criterion (which is mean squared error for regression) for a given variable is calculated at each step. The relative influence for a given variable is the average of these improvements over all the trees where the aforementioned variable is used.

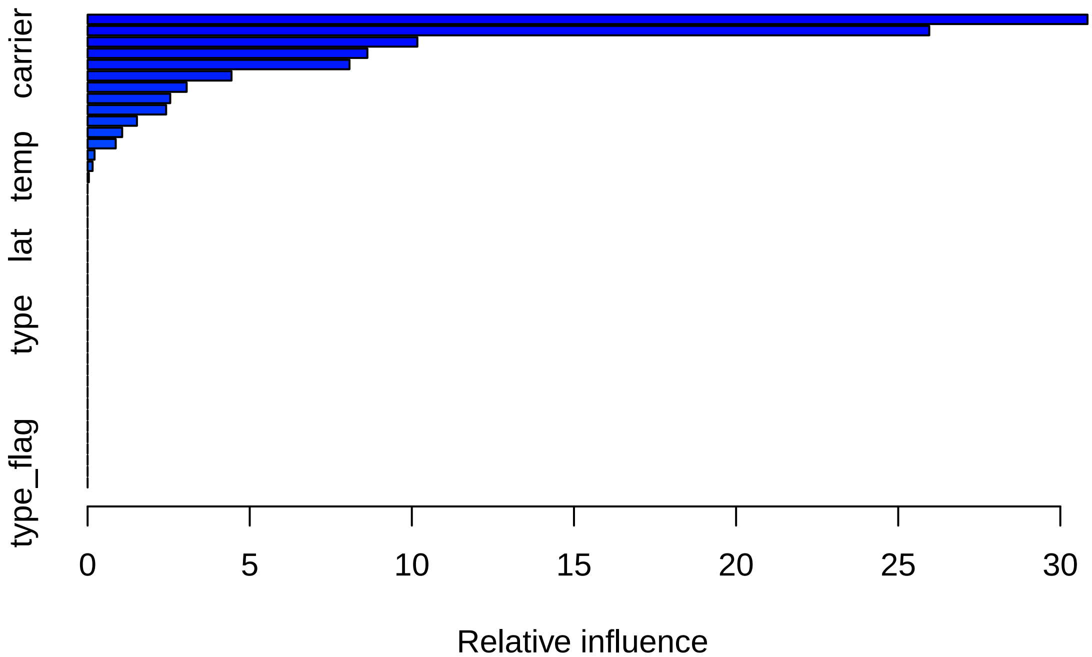
```r
model <- gbm(dep_delay ~ ., data=train_df,
              n.trees=1000, shrinkage=0.01) # default shrinkage = 0.1
```

```
## Distribution not specified, assuming gaussian ...
```

```r
preds = predict(model, newdata=test_df, n.trees=1000)
rmse = sqrt(mean((test_df$dep_delay - preds)^2))
rmse
```

```
## [1] 7.980812
```

```r
summary(model)
```




```
##                                                 var     rel.inf
## sched_dep_time_num_minute sched_dep_time_num_minute 30.83853917
## model                                         model 25.95730577
## dest                                           dest 10.17043661
## carrier                                     carrier  8.62897727
## month                                         month  8.07615687
## dewp                                           dewp  4.43848169
## origin                                       origin  3.05562665
## sched_arr_time_num_minute sched_arr_time_num_minute  2.54883378
## precip                                       precip  2.42172931
## pressure                                   pressure  1.52135832
## humid                                         humid  1.06685951
## dep_delay_flag                       dep_delay_flag  0.86763349
## pressure_flag                         pressure_flag  0.21490352
## temp                                           temp  0.15460862
## day                                             day  0.03854943
## distance                                   distance  0.00000000
## wind_dir                                   wind_dir  0.00000000
## wind_speed                               wind_speed  0.00000000
## visib                                         visib  0.00000000
## name                                           name  0.00000000
## lat                                             lat  0.00000000
## lon                                             lon  0.00000000
## alt                                             alt  0.00000000
## tz                                               tz  0.00000000
```

```
## dst                                                     dst  0.00000000
## tzone                                                 tzone  0.00000000
## year.y                                               year.y  0.00000000
## type                                                   type  0.00000000
## manufacturer                                   manufacturer  0.00000000
## engines                                             engines  0.00000000
## seats                                                 seats  0.00000000
## engine                                               engine  0.00000000
## sched_arr_time_minute               sched_arr_time_minute  0.00000000
## sched_dep_time_minute               sched_dep_time_minute  0.00000000
## sched_air_time                             sched_air_time  0.00000000
## temp_flag                                       temp_flag  0.00000000
## wind_dir_flag                               wind_dir_flag  0.00000000
## wind_speed_flag                           wind_speed_flag  0.00000000
## precip_flag                                   precip_flag  0.00000000
## name_flag                                       name_flag  0.00000000
## year.y_flag                                   year.y_flag  0.00000000
## type_flag                                       type_flag  0.00000000
```

```
rmse = sqrt(mean((test_df$dep_delay - preds)^2))
rmse
```

```
## [1] 7.980812
```

set.seed(42)

x <- 2^seq(5,14, by=1) rmse_vec <- numeric(length(x)) count <- 1 for (val in x) { hboost <- gbm(dep_delay ~ ., data = train_df, n.trees = val, distribution = 'gaussian', shrinkage = 0.01) preds = predict(hboost, n.trees = val, newdata = test_df) mse = mean((test_df$dep_delay - preds) ^ 2) rmse <- sqrt(mse) rmse_vec[count] <- rmse

print(val) print(rmse) count = count + 1 }

plot(x, rmse_vec)

summary(hboost) class(summary(hboost)) summary <- summary(hboost) write.csv(summary,'16384trees_gbm.csv')

```
gbm_benchmark<-read_csv('shrinkage_0point01_numtrees_32_to_16384_gbm_benchmark.csv')
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

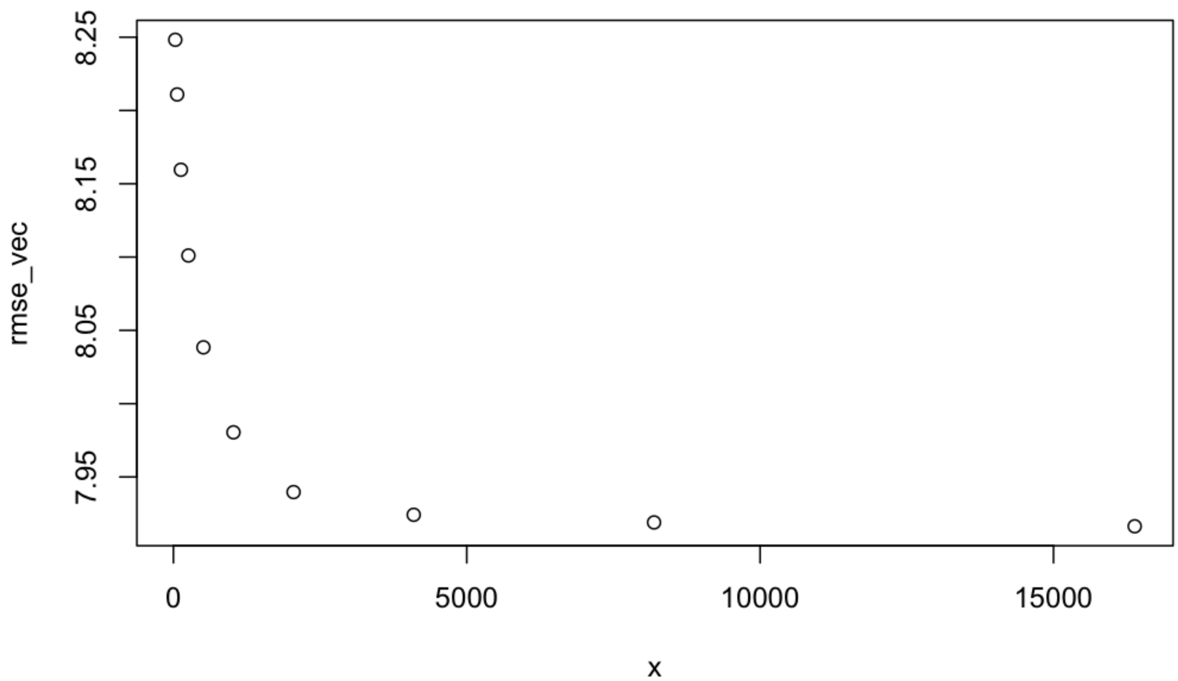
```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   num_trees = col_double(),
##   rmse = col_double()
## )
```

```
gbm_benchmark
```

```
## # A tibble: 10 x 3
##       X1 num_trees  rmse
##    <dbl>     <dbl> <dbl>
##  1     1        32  8.25
##  2     2        64  8.21
##  3     3       128  8.16
##  4     4       256  8.10
##  5     5       512  8.04
##  6     6      1024  7.98
```

```
## 7     7      2048  7.94
## 8     8      4096  7.92
## 9     9      8192  7.92
## 10    10    16384  7.92
```

Above values are for gbm with shrinkage of 0.01 Analysis:



Tuning gbm

Here I plotted root mean squared error (rmse) vs the number of trees for shrinkage of 0.01 and all other variables as default for gbm. You can see that after around 5000 trees, increasing the number of trees further gives diminishing returns.

library(EZtune) response <- DF$dep_delayeztune_df < -DFeztune_df$dep_delay <- NULL eztune_obj <- eztune(eztune_df, response, method = “gbm”, optimizer = “hjn”, fast = TRUE, cross = NULL)

eztune_obj $n [1] 200

$n.trees [1] 2001

$interaction.depth [1] 10

$n.minobsinnode [1] 7

$shrinkage [1] 0.001

$mse [1] 72.68835

$model gbm :: gbm(formula = y \sim ., distribution = "gaussian", data = dat, n.trees = results$n.trees, interaction.depth = results$interaction.depth, n.minobsinnode = results$n.minobsinnode, shrinkage = results$shrinkage) A gradient boosted model with gaussian loss function. 2001 iterations were performed. There were 42 predictors of which 24 had non-zero influence.

sqrt(72) [1] 8.485281 sqrt(72.68) [1] 8.525257

References

"Gradient Boosting Machines · UC Business Analytics R Programming Guide." 2019. http://uc-r.github.io/gbm_regression#h2o.