

SmartFly: Exploratory Data Analysis

Cindy Lamm

January 14, 2015

First load variable names and types of historic data (prepared in an additional csv file):

```
nameTypeDataFile <- "resources/raw_variables.csv"
variableNames <- read.csv(nameTypeDataFile, header=TRUE, stringsAsFactors=FALSE)
variableNames

##           name      type
## 1           id character
## 2          year    factor
## 3          month    factor
## 4    day_of_month    factor
## 5    day_of_week    factor
## 6 scheduled_departure_time    factor
## 7  scheduled_arrival_time    factor
## 8          airline    factor
## 9    flight_number    factor
## 10         tail_number    factor
## 11         plane_model    factor
## 12   seat_configuration    factor
## 13    departure_delay    numeric
## 14    origin_airport    factor
## 15 destination_airport    factor
## 16    distance_travelled    numeric
## 17         taxi_time_in    numeric
## 18         taxi_time_out    numeric
## 19          cancelled    integer
## 20    cancellation_code    factor

factor_idx <- which(variableNames$type=="factor")
```

Then load historic data into R. I set empty strings to NA (because of variables tail_number and cancellation_code).

```
historicDataFile <- "../data/smartyfly_historic.csv"
# historicDataFile <- "../data/mod4000.csv"
trainDataTyped <- read.csv(historicDataFile, header=FALSE, stringsAsFactors=FALSE,
                           col.names=variableNames$name, colClasses=variableNames$type,
                           na.strings=c("NA", ""))
# convert integer to logical
trainDataTyped$cancelled <- as.logical(trainDataTyped$cancelled)
```

Checkout data content:

```
str(trainDataTyped)

## 'data.frame': 7374365 obs. of 20 variables:
## $ id : chr "4982598272866526024" "5074130684343212714" "8872634703988349126" ...
## $ year : Factor w/ 2 levels "2013","2014": 1 1 1 1 1 1 1 1 1 1 ...
## $ month : Factor w/ 12 levels "1","10","11",...: 11 11 11 11 11 11 11 11 11 11 ...
## $ day_of_month : Factor w/ 31 levels "1","10","11",...: 3 9 10 17 18 25 1 12 23 26 ...
## $ day_of_week : Factor w/ 7 levels "1","2","3","4",...: 7 6 7 6 7 6 4 5 6 7 ...
## $ scheduled_departure_time: Factor w/ 1190 levels "0","10","100",...: 20 20 20 20 20 20 1041 1041 1041 ...
## $ scheduled_arrival_time : Factor w/ 1323 levels "0","1","10","100",...: 111 111 111 111 111 111 123 123 ...
## $ airline : Factor w/ 17 levels "AA","AS","B6",...: 15 15 15 15 15 15 15 15 15 15 ...
## $ flight_number : Factor w/ 6889 levels "1","10","100",...: 6744 6744 6744 6744 6744 6744 6744 6744 ...
## $ tail_number : Factor w/ 5035 levels "0","000000","N050AA",...: 3898 3963 3806 3810 4008 4008 ...
## $ plane_model : Factor w/ 6 levels "737","747","757",...: 3 3 5 2 5 2 2 3 2 6 ...
## $ seat_configuration : Factor w/ 6 levels "Standard","Three Class",...: 2 1 4 5 4 5 2 1 5 2 ...
## $ departure_delay : num -5 5 -4 -6 -3 -8 0 -2 14 -6 ...
## $ origin_airport : Factor w/ 279 levels "ABE","ABI","ABQ",...: 46 46 46 46 46 46 133 133 133 133 ...
## $ destination_airport : Factor w/ 279 levels "ABE","ABI","ABQ",...: 61 61 61 61 61 61 61 61 61 61 ...
## $ distance_travelled : num 361 361 361 361 361 361 185 185 185 185 ...
## $ taxi_time_in : num 9 7 6 15 7 5 9 3 5 5 ...
## $ taxi_time_out : num 11 7 9 11 12 15 8 8 16 9 ...
## $ cancelled : logi FALSE FALSE FALSE FALSE FALSE FALSE ...
## $ cancellation_code : Factor w/ 4 levels "A","B","C","D": NA NA NA NA NA NA NA NA NA NA ...
```

Specifically note the factor levels for the different variables¹. We see that `scheduled_departure_time` and `scheduled_arrival_time` need to be reformatted to have for all observations values that are 4 characters long (assuming "100" means "0100" and thus a time of 01h00):

```
trainDataTyped$scheduled_departure_time <- as.factor(
  sprintf("%04s", as.character(trainDataTyped$scheduled_departure_time)))
trainDataTyped$scheduled_arrival_time <- as.factor(
  sprintf("%04s", as.character(trainDataTyped$scheduled_arrival_time)))
```

In addition we truncate the scheduled times to the hour:

```
trainDataTyped$scheduled_departure_time <- as.factor(
  substr(as.character(trainDataTyped$scheduled_departure_time),1,2))
trainDataTyped$scheduled_arrival_time <- as.factor(
  substr(as.character(trainDataTyped$scheduled_arrival_time),1,2))
```

We also reformat the variables `day_of_month` and `month` (so that they're ordered automatically in graphs):

```
trainDataTyped$month <- as.factor(
  sprintf("%02s", as.character(trainDataTyped$month)))
trainDataTyped$day_of_month <- as.factor(
  sprintf("%02s", as.character(trainDataTyped$day_of_month)))
```

¹The number of levels matters if we would want to create a dummy variable for each level. With lots of levels the number of variables would be HUGE and so would be the sparsity of the design matrix.

See summary of descriptive statistics of the historic data:

```
summary(trainDataTyped)

##          id          year          month          day_of_month          day_of_week
## Length:7374365    2013:2185499    08      :1023748    13      : 252615    1:1079862
## Class :character    2014:5188866    09      : 957710    06      : 252560    2:1063516
## Mode  :character          10      : 782952    03      : 252160    3:1069847
##          03      : 559342    17      : 251944    4:1096825
##          07      : 558568    16      : 250869    5:1096417
##          01      : 552109    02      : 250647    6: 935465
##          (Other):2939936    (Other):5863570    7:1032433
## scheduled_departure_time scheduled_arrival_time airline flight_number
## 17      : 539987          16      : 534524      WN      :1171236    192      : 5702
## 08      : 511491          17      : 492008      AA      : 960866    64       : 5639
## 09      : 500448          19      : 490631      DL      : 825543    706      : 5409
## 07      : 499763          18      : 488335      UA      : 686409    186      : 5373
## 13      : 490531          20      : 484248      NW      : 619091    751      : 5209
## 12      : 487784          11      : 482204      US      : 529032    340      : 5060
## (Other):4344361          (Other):4402415    (Other):2582188 (Other):7341973
## tail_number plane_model seat_configuration departure_delay
## 0           : 17138    737 :2317735    Standard :2130560    Min.      : -1410.00
## 000000      : 10157    747 :1579936    Three Class: 779700    1st Qu.:  -4.00
## N183UW      : 4694    757 : 999512    Two Class  : 779964    Median :   0.00
## N80         : 4290    777 : 634170    V1          :1430984    Mean      :  4.87
## N96         : 4269    787 : 633182    V2          :1105044    3rd Qu.:   2.00
## (Other):7291604    A320:1209830    V3          :1148113    Max.      : 2119.00
## NA's       : 42213                                NA's       :104127
## origin_airport destination_airport distance_travelled taxi_time_in
## ORD      : 431004    ORD      : 431004    Min.      : 11      Min.      : 0.000
## ATL      : 389963    ATL      : 389886    1st Qu.: 308      1st Qu.:  4.000
## DFW      : 382123    DFW      : 382349    Median : 569      Median :  5.000
## LAX      : 255642    LAX      : 255786    Mean      : 726      Mean      :  6.808
## PHX      : 209831    PHX      : 209839    3rd Qu.: 964      3rd Qu.:  7.000
## IAH      : 195923    IAH      : 195926    Max.      :4962      Max.      :1495.000
## (Other):5509879    (Other):5509575
## taxi_time_out cancelled cancellation_code
## Min.      : 0.00    Mode :logical    A      : 14587
## 1st Qu.: 10.00    FALSE:7270238    B      : 8072
## Median : 13.00    TRUE :104127     C      : 8309
## Mean      : 15.05    NA's :0          D      : 179
## 3rd Qu.: 18.00                                NA's:7343218
## Max.      :1439.00
##
```

Save data frame for next step:

```
save(trainDataTyped, file="trainDataTyped.Rdata")
```

Since I want to predict whether a flight is delayed or not I create a specific variable `is_delayed` based on `departure_delay` using the definition that only positive delay and non-cancelled flights count as "delayed":

```

trainDataTyped$is_delayed <- factor(trainDataTyped$departure_delay > 0
                                   & trainDataTyped$cancelled==FALSE,
                                   labels= c("on_time", "delayed"))

summary(trainDataTyped$is_delayed)

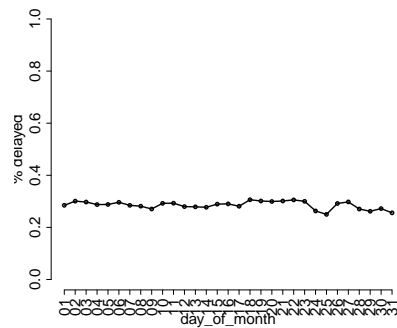
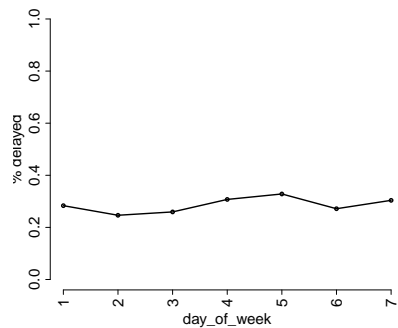
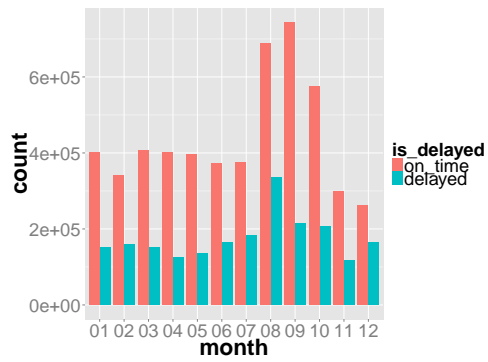
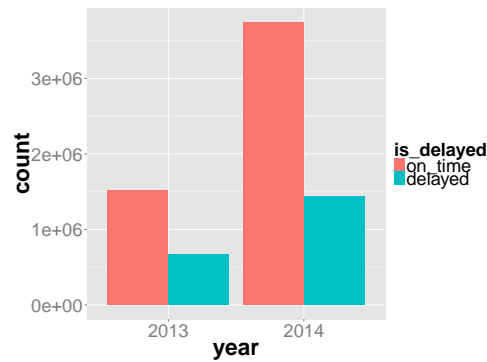
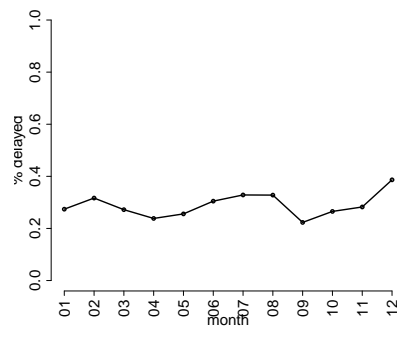
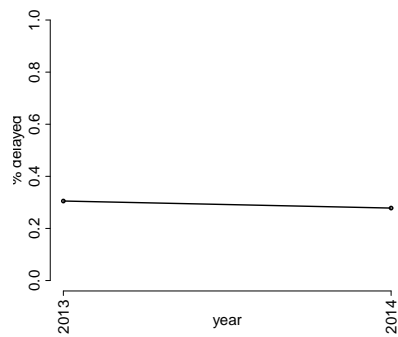
## on_time delayed
## 5263866 2110499

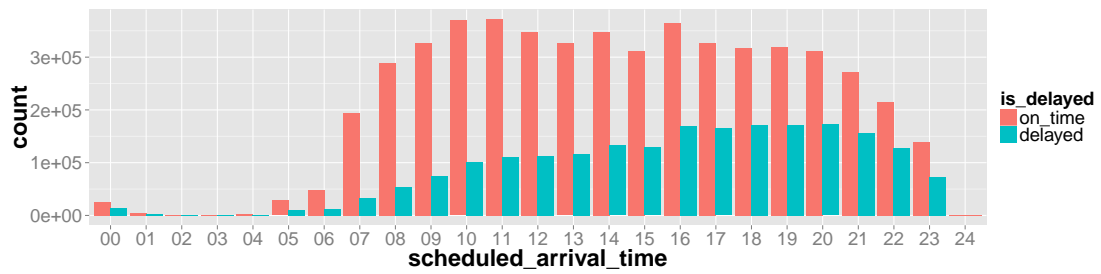
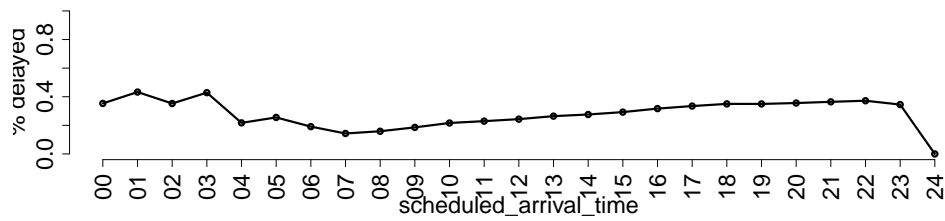
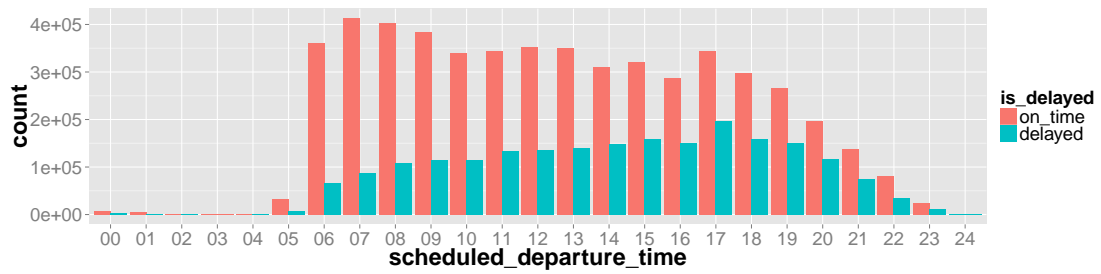
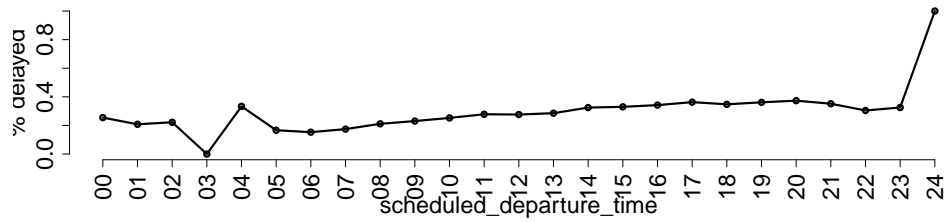
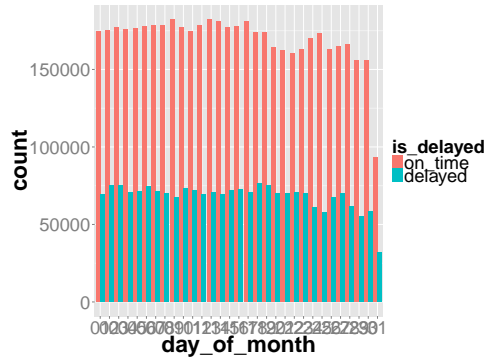
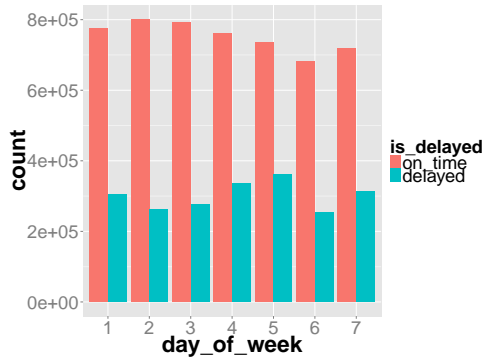
summary(trainDataTyped$is_delayed) / length(trainDataTyped$is_delayed)

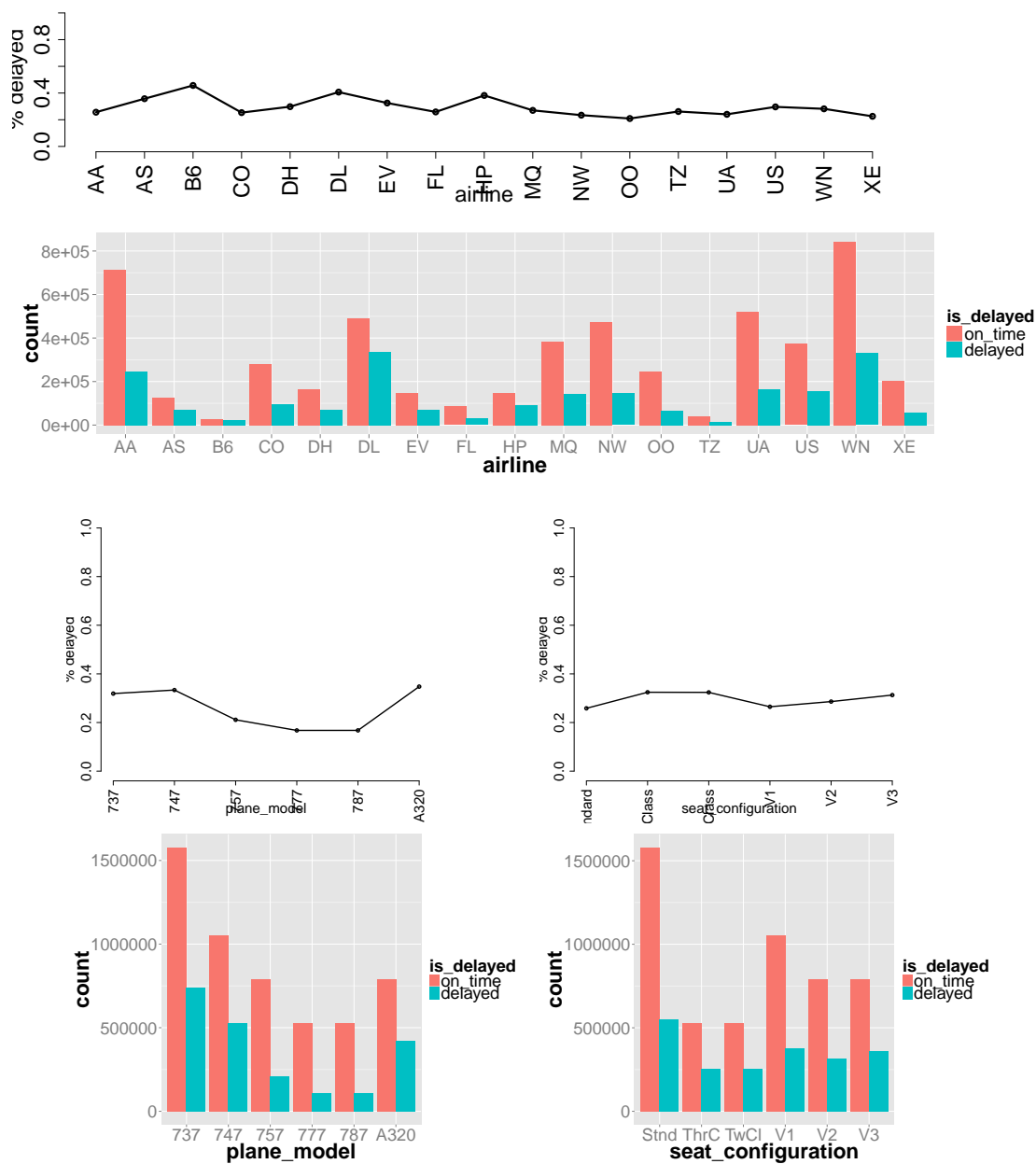
## on_time delayed
## 0.713806 0.286194

```

Plot the data:

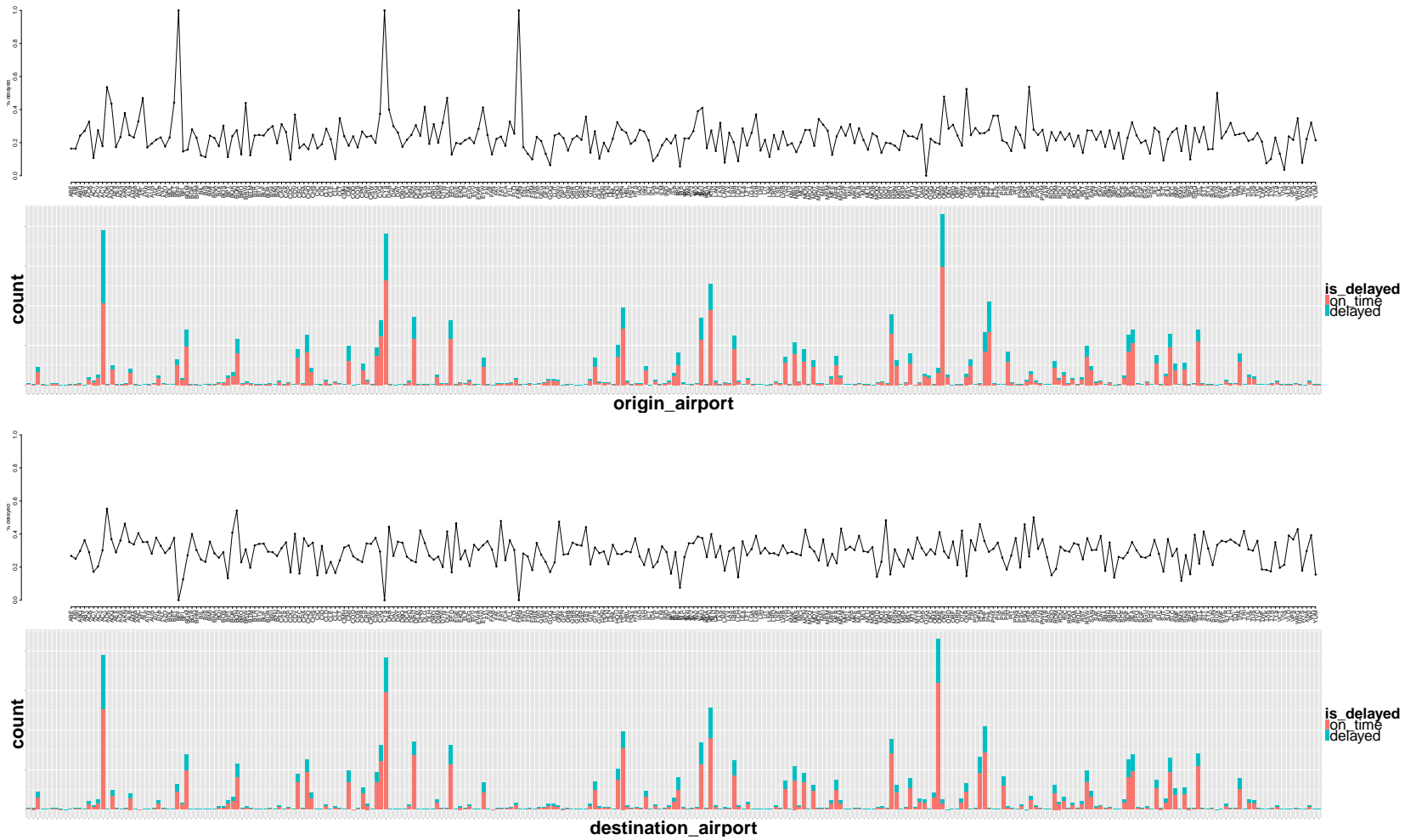


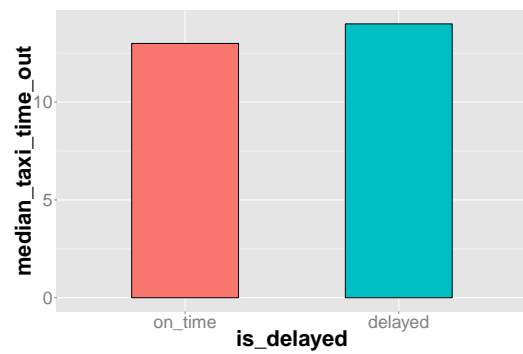
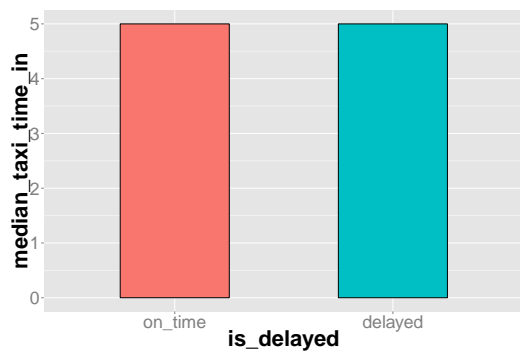
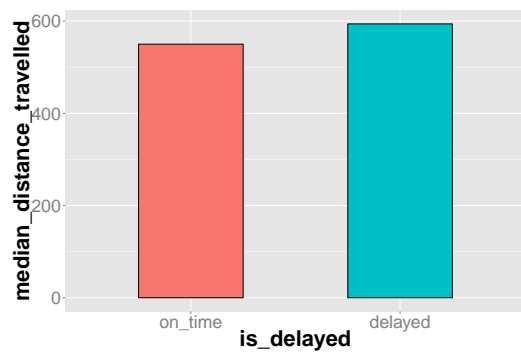
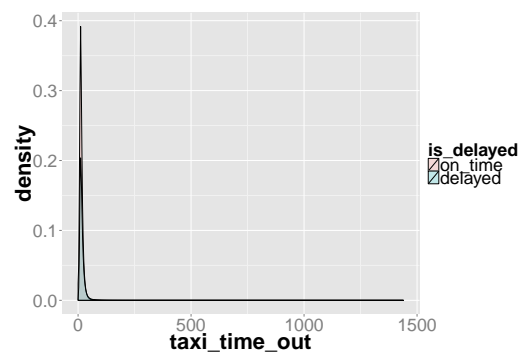
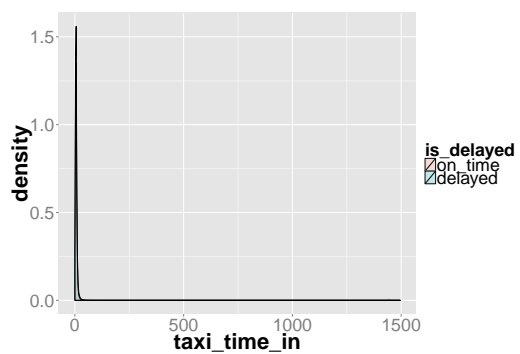
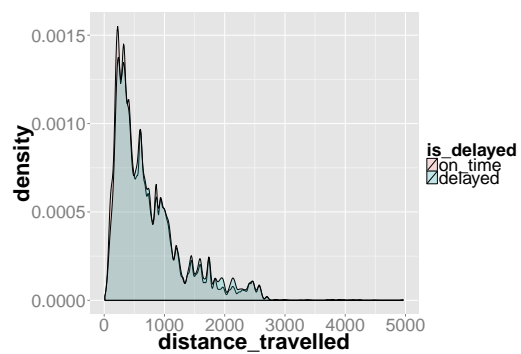




The variables flight_number and tail_number don't produce any valuable plots due to their large number in levels.

7





Look at correlations between continuous variables:

```
cor(trainDataTyped$departure_delay, trainDataTyped$distance_travelled, use="pairwise.complete.obs")
## [1] -0.0007718446

cor(trainDataTyped$departure_delay, trainDataTyped$taxi_time_in, use="pairwise.complete.obs")
## [1] 0.03345877

cor(trainDataTyped$departure_delay, trainDataTyped$taxi_time_out, use="pairwise.complete.obs")
## [1] 0.06387488
```

Look at some dependency between the binary target variable and other factor variables (with reasonably few levels) using the Chi-Square test of independence. The null hypothesis is that the two variables are independent, which we reject if the p-value is smaller than $\alpha = 0.001$ (chosen so small due to large sample size):

```
dependent_with_target

##          year          month      day_of_week      airline
##          TRUE          TRUE          TRUE          TRUE
## plane_model seat_configuration cancellation_code
##          TRUE          TRUE          FALSE
```