Exploration Analysis Taxi Data

In this document, we have subsampled data by taking 100K registers from each month in order to speed up computations in the exploratory analysis. The full dataset will be used in the learning part.

Preprocessing: coercing features

Begin with the preprocessing step. Firstly, we coerce categorical features to be factor-like, and time-based features to datetime ones.

```
# coerce categorical features
categ_features <- c("VendorID", "RatecodeID", "PULocationID", "DOLocationID", "payment_type")
raw_data[categ_features] <- lapply(raw_data[categ_features], as.factor)
sapply(raw_data, class)
```

```
##              VendorID  tpep_pickup_datetime tpep_dropoff_datetime
##              "factor"              "factor"              "factor"
##       passenger_count         trip_distance            RatecodeID
##             "integer"             "numeric"              "factor"
##    store_and_fwd_flag          PULocationID          DOLocationID
##              "factor"              "factor"              "factor"
##          payment_type           fare_amount                 extra
##              "factor"             "numeric"             "numeric"
##               mta_tax            tip_amount          tolls_amount
##             "numeric"             "numeric"             "numeric"
## improvement_surcharge          total_amount
##             "numeric"             "numeric"
```

```
  raw_data$tpep_pickup_datetime <- as.POSIXct(as.character(raw_data$tpep_pickup_datetime), format = "%Y-
  raw_data$tpep_dropoff_datetime <- as.POSIXct(as.character(raw_data$tpep_dropoff_datetime), format = "
```

Brief summary of data

We get the first sight of our data, the most basic statistical measurements are displayed: min-max values, median, mean for numerical values; and number of values for each category in factor features.

```
summary(raw_data)
```

```
##  VendorID   tpep_pickup_datetime          tpep_dropoff_datetime
##  1:136594   Min.   :2017-03-01 00:01:11   Min.   :2017-03-01 00:11:25
##  2:163406   1st Qu.:2017-03-24 11:09:40   1st Qu.:2017-03-24 11:24:26
##             Median :2017-06-15 17:23:52   Median :2017-06-15 17:46:09
##             Mean   :2017-07-06 06:12:19   Mean   :2017-07-06 06:29:12
##             3rd Qu.:2017-11-08 07:48:19   3rd Qu.:2017-11-08 08:02:52
##             Max.   :2017-12-01 00:01:54   Max.   :2017-12-01 19:10:13
##
##  passenger_count trip_distance     RatecodeID  store_and_fwd_flag
##  Min.   :0.000   Min.   :  0.000   1 :291529   N:298715
##  1st Qu.:1.000   1st Qu.:  0.970   2 :  6678   Y:  1285
##  Median :1.000   Median :  1.600   3 :   652
##  Mean   :1.615   Mean   :  2.916   4 :   170
##  3rd Qu.:2.000   3rd Qu.:  3.010   5 :   964
```

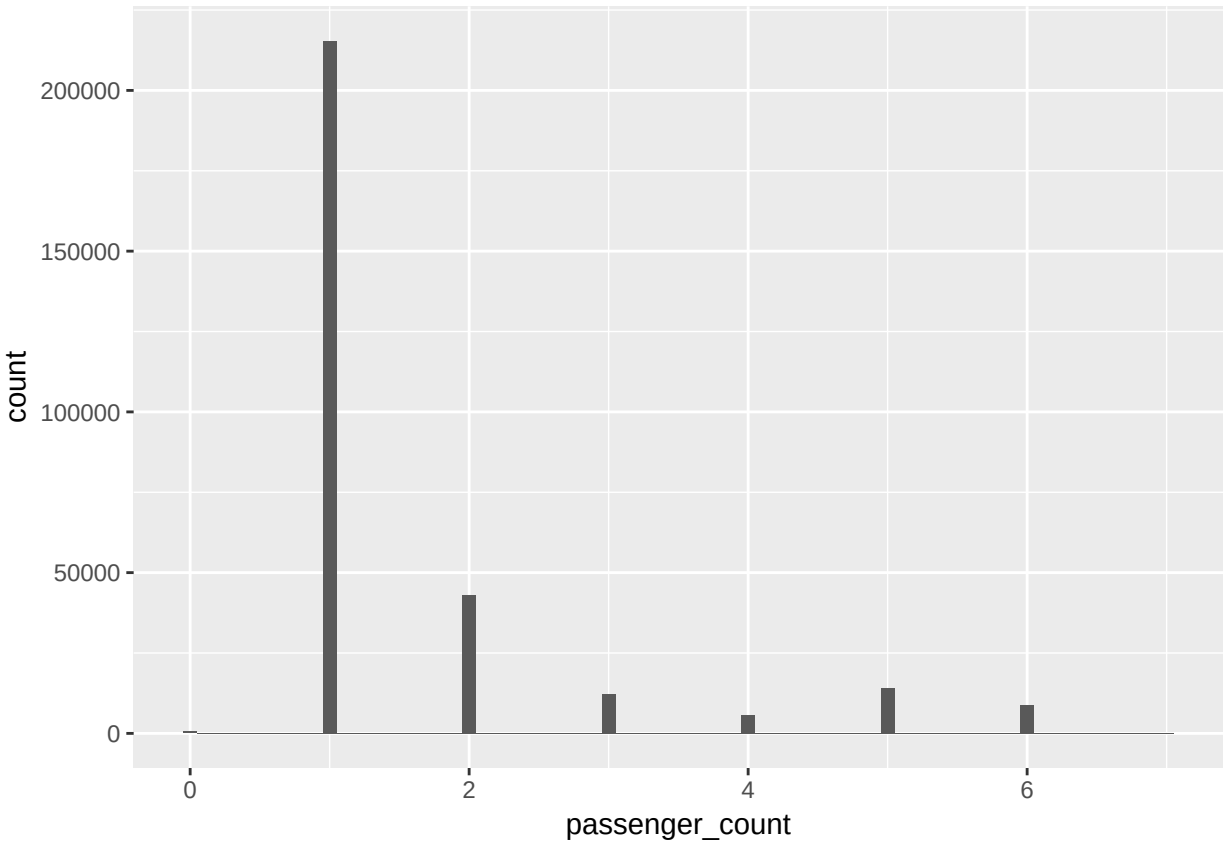
```
##  Max.   :7.000   Max.   :101.710   6 :     4
##                                    99:     3
##   PULocationID     DOLocationID    payment_type  fare_amount
##  237    : 11962   161    : 11243   1:203803     Min.   :-70.00
##  161    : 11264   236    : 11186   2: 94115     1st Qu.:  6.50
##  236    : 10804   237    : 10438   3:  1649     Median :  9.50
##  162    : 10366   170    :  9670   4:   433     Mean   : 13.05
##  186    : 10358   230    :  9269                3rd Qu.: 14.50
##  230    : 10206   162    :  9026                Max.   :825.00
##  (Other):235040   (Other):239168
##      extra            mta_tax          tip_amount      tolls_amount
##  Min.   :-4.5000   Min.   :-0.5000   Min.   : -0.99   Min.   :  0.000
##  1st Qu.: 0.0000   1st Qu.: 0.5000   1st Qu.:  0.00   1st Qu.:  0.000
##  Median : 0.0000   Median : 0.5000   Median :  1.36   Median :  0.000
##  Mean   : 0.3326   Mean   : 0.4972   Mean   :  1.87   Mean   :  0.333
##  3rd Qu.: 0.5000   3rd Qu.: 0.5000   3rd Qu.:  2.46   3rd Qu.:  0.000
##  Max.   : 4.5000   Max.   :11.3000   Max.   :120.00   Max.   :975.600
##
##  improvement_surcharge  total_amount
##  Min.   :-0.3000       Min.   :-70.30
##  1st Qu.: 0.3000       1st Qu.:  8.75
##  Median : 0.3000       Median : 11.80
##  Mean   : 0.2996       Mean   : 16.39
##  3rd Qu.: 0.3000       3rd Qu.: 17.86
##  Max.   : 0.3000       Max.   :982.90
##
```

In this first sight, we can notice that the dataset is prone to the presence of outliers. For instance, there exist trips with 0 passengers, which means taxi is empty. In next steps, we should check if this kind of rides are free-charge (the natural scenario), or on the contrary, the amount features reflects some income. Other negative values in features (e.g.: fare_amount, extra) seems suspicious as well.

Univariate analysis

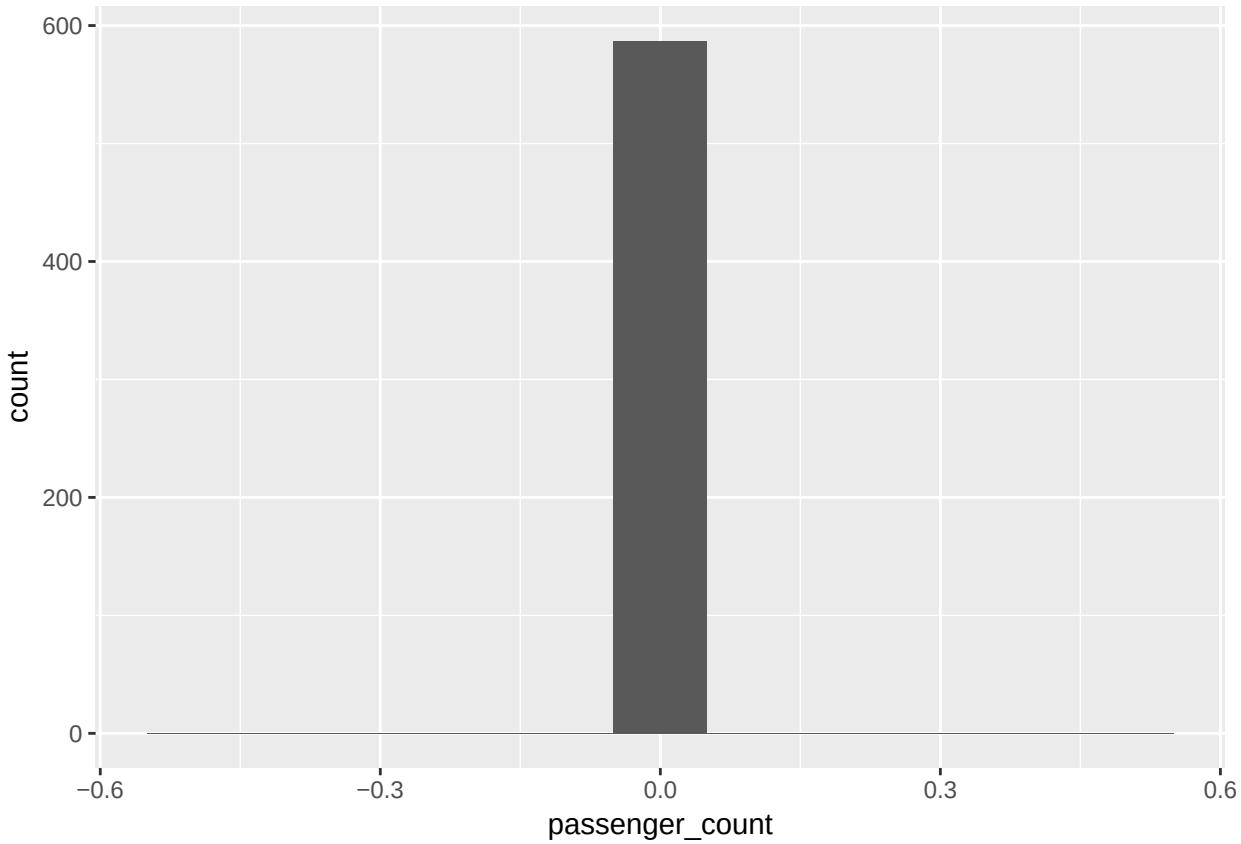
Number of passengers

Let's check first passenger_count:



Most of yellow taxi rides consist of a single passenger, which is normal in business (and stressful) cities like the great New York :D Additionally, we have also checked that no floating point values are present in this tipically integer feature.

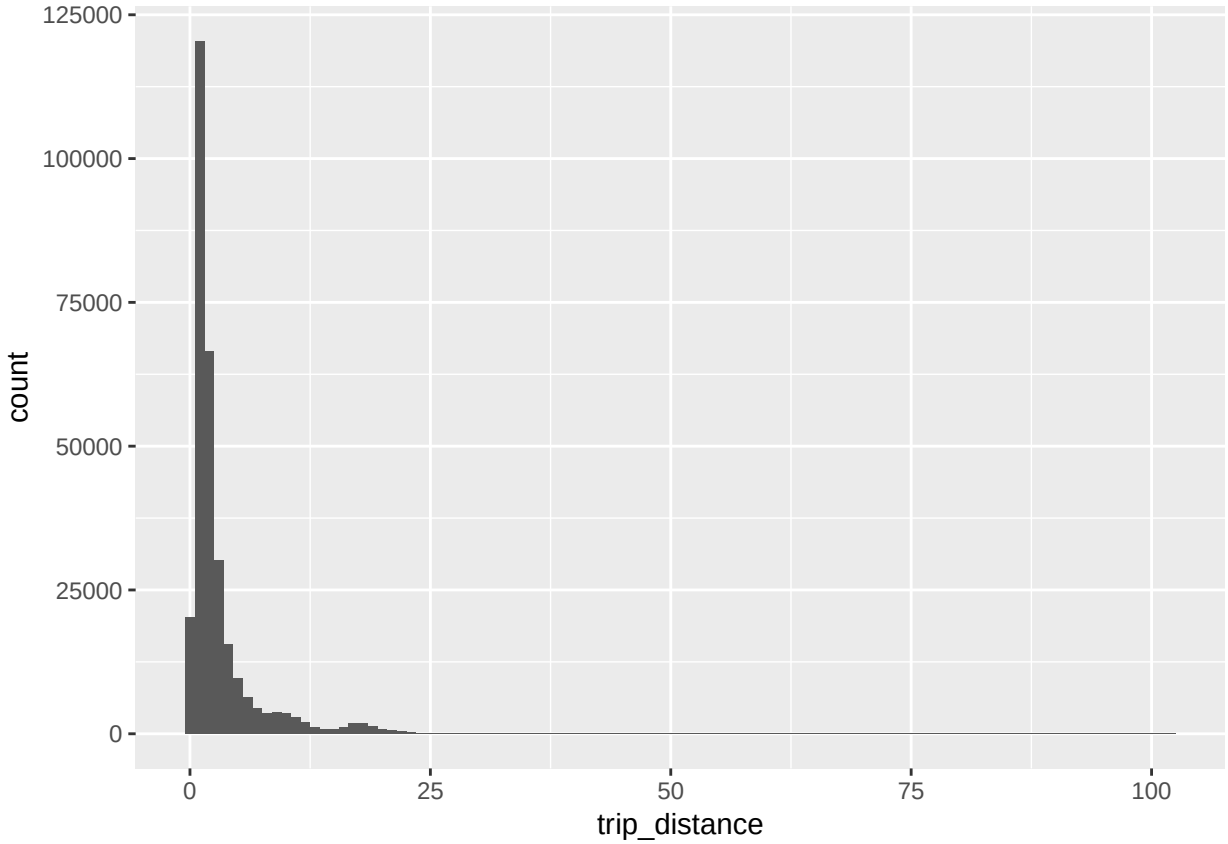
Although zero-passenger rides looks like negligible, lets zoom in in this part to check the real value:



Bingo! It seems that "ghostly" rides in NY are more common than expected. However, 600 rides from 300,000 represents a small fraction (0.002) in the sampled set. If this proportion maintains in the general set, maybe it is safe to set the passenger_count in these rides as NaN or remove them directly. It depends on the prediction algorithm used to detect patterns in data. In our case, we will rely on Gradient Boosted Trees (concretely, XGBoost) which elegantly deals with NaN values. Please, refer to the original paper to get further knowledge about this aspect.

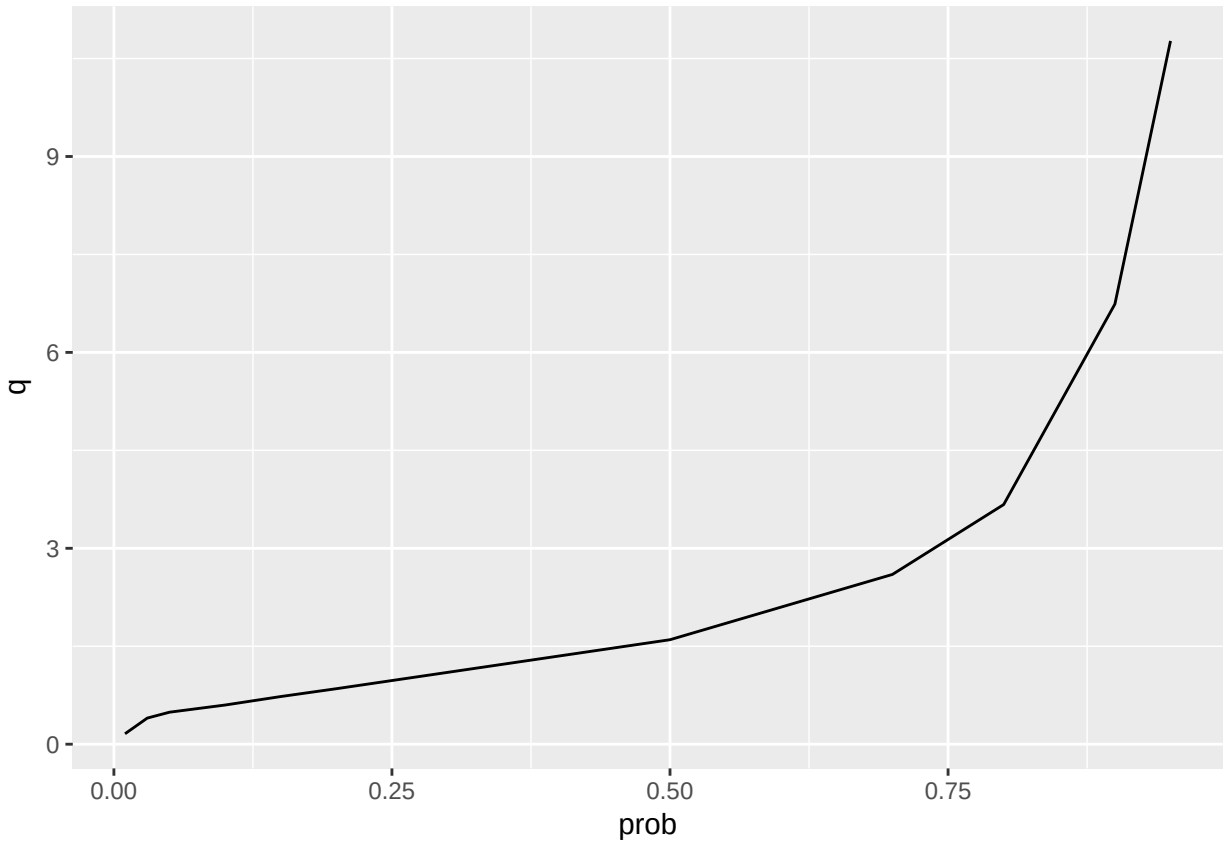
Trip distance

Let's move now to trip distance:



The plot above shows that, in general, rides are short; most of them below 5 miles. Low prices may indicate too high prices for yellow cabs or main users are people in a hurry for nearly due meetings.

Let's check percentiles in this feature:




```
distance_quantiles
```

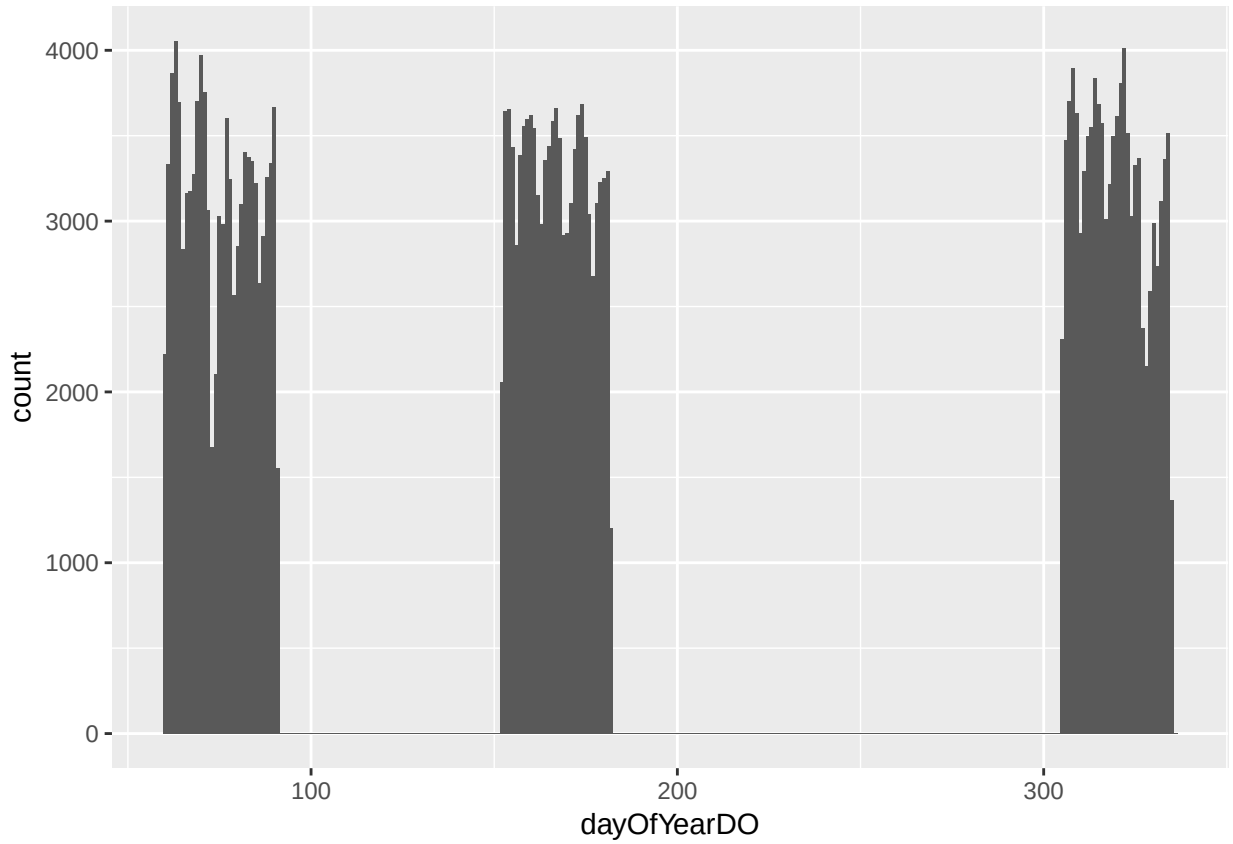
```
##         q prob
## 1%   0.16 0.01
## 3%   0.40 0.03
## 5%   0.49 0.05
## 10%  0.60 0.10
## 15%  0.73 0.15
## 20%  0.85 0.20
## 50%  1.60 0.50
## 70%  2.60 0.70
## 80%  3.67 0.80
## 90%  6.74 0.90
## 95% 10.77 0.95
```

80% of trips are below 4 miles, and 50% are below 1.60 miles. This, and the fact that the approximate longitude of Manhattan is 12.6 milles, might confirm our previous hypothesis.

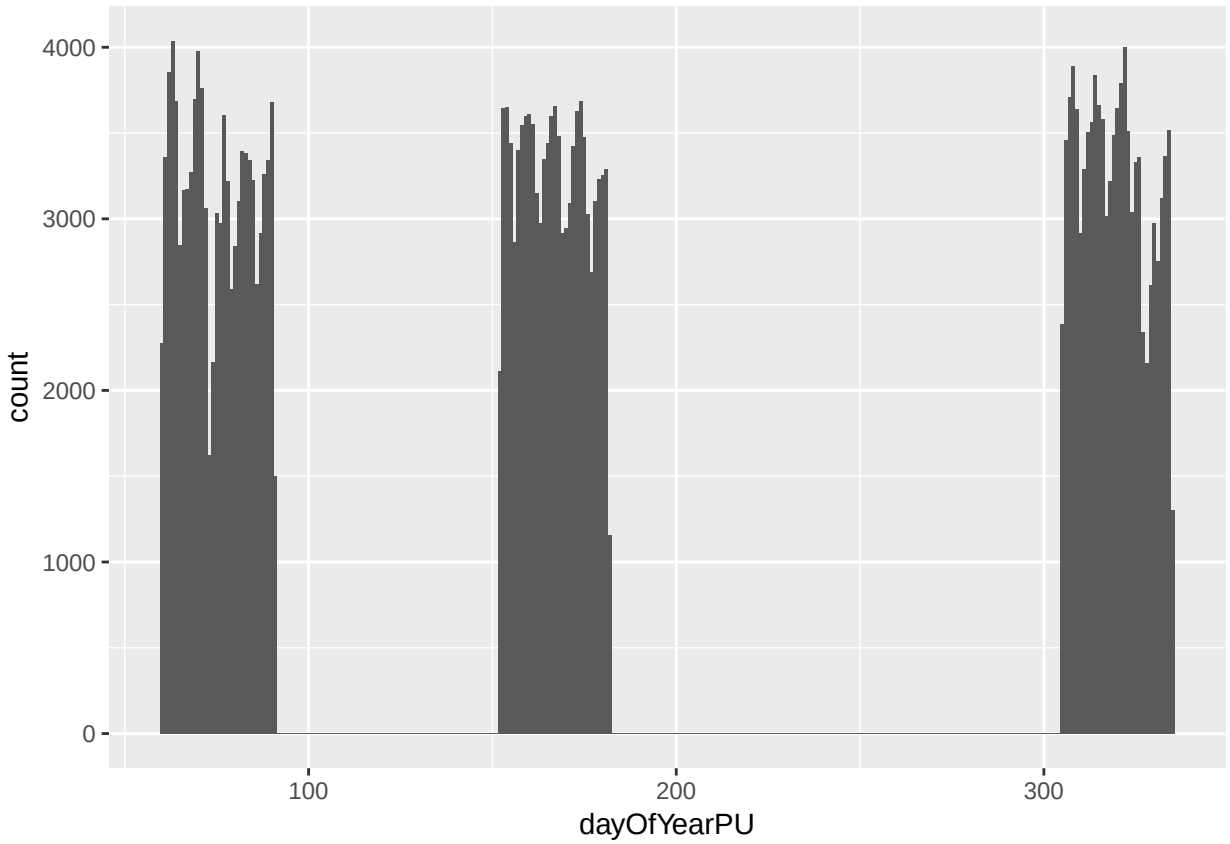
Time-based features

Now we move to check time-based features, which usually are the most conflictive in data preprocessing. First, we are gonna visualize if there are some big gaps in dropin and dropoff features. To do that we create two new variables that indicates the day of year in which ocurrs each ride (both dropoff and pickup actions).

```
raw_data <- raw_data %>% mutate(dayOfYearDO = as.numeric(strftime(tpep_dropoff_datetime, format = "%j")))
ggplot(data=raw_data, aes(dayOfYearDO)) + geom_histogram(binwidth = 1.0)
```



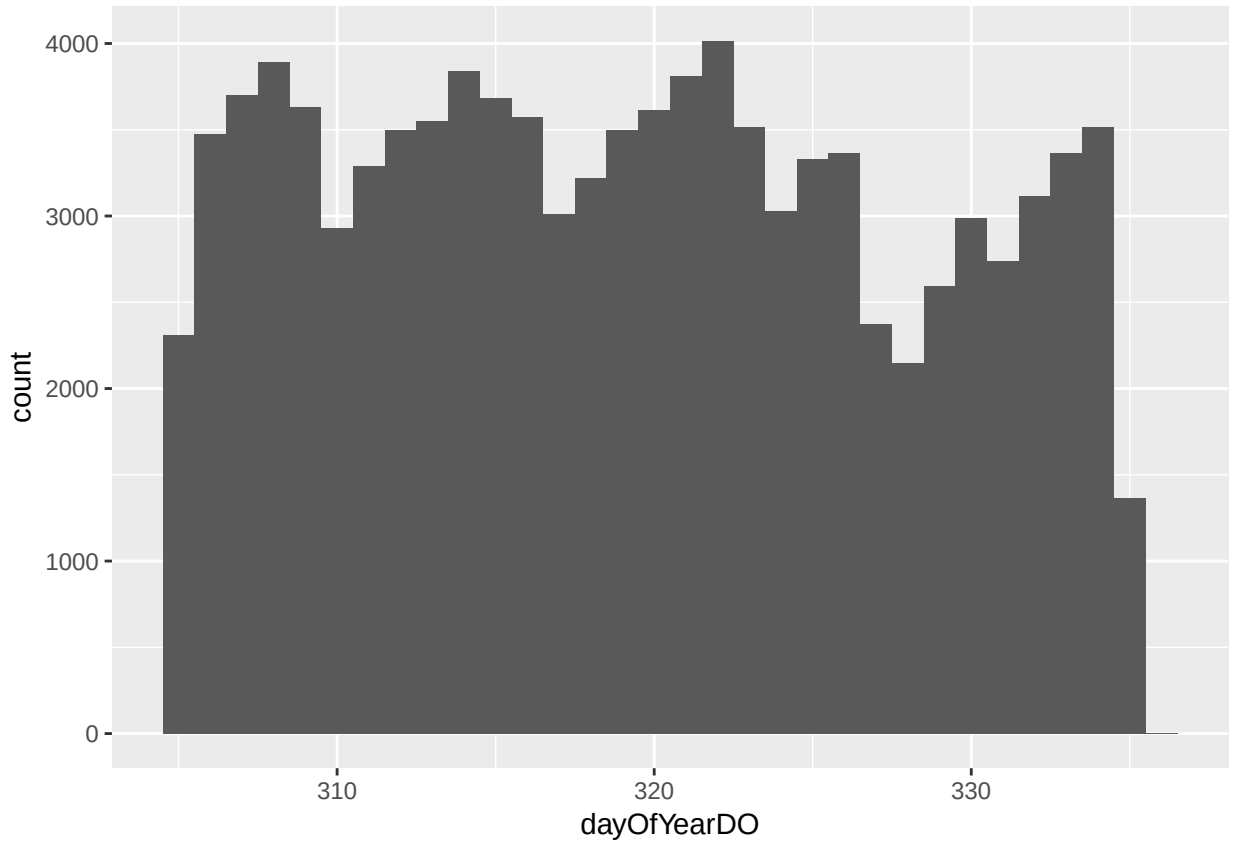

```
ggplot(data=raw_data, aes(dayOfYearPU)) + geom_histogram(binwidth = 1.0)
```



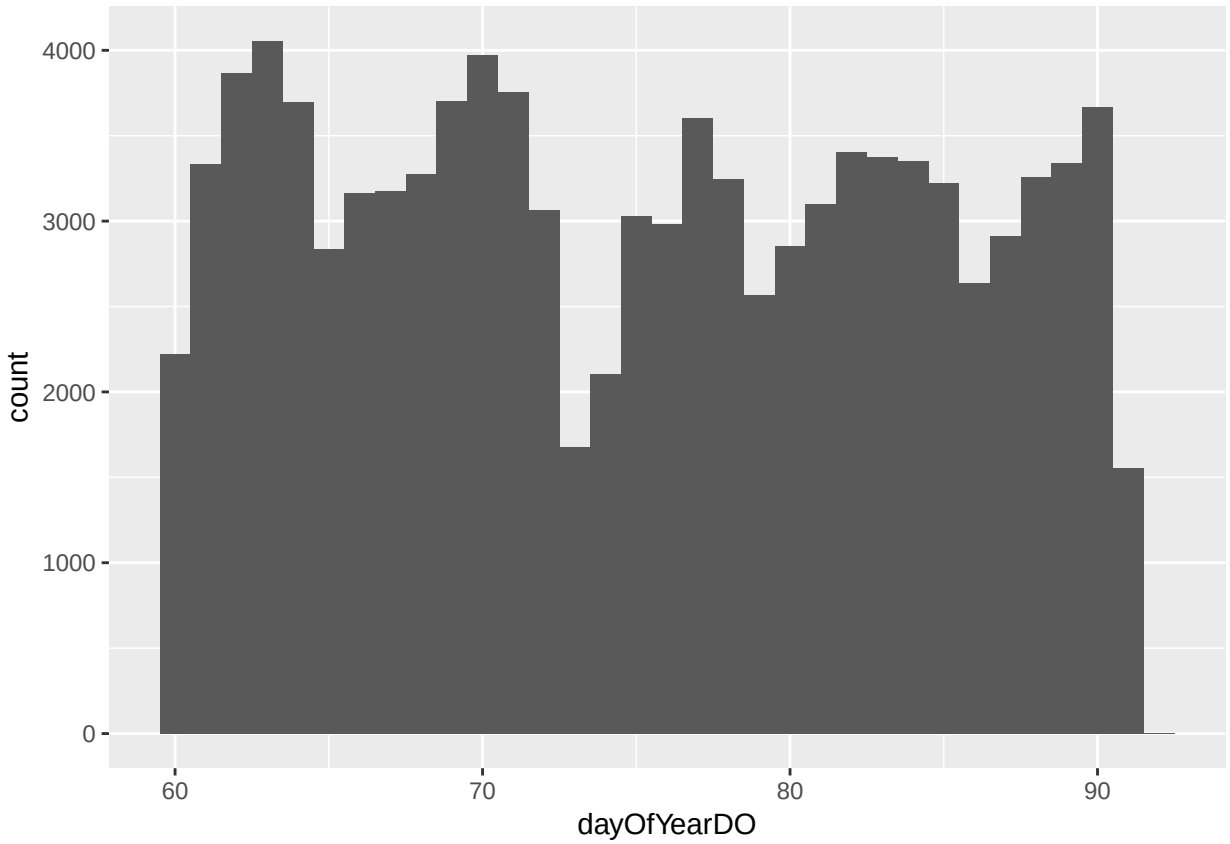
There is a considerable downtrend in dropoff (DO) timestamps in the middle of March, which matches with the correspondent day for pickup (PU) timestamps. This fact may be explained by strikes, or some relevant events in the city. However, there is a weird gap in November for DO which does not match with the same period in PU timestamp data.

Let's zoom in the data from November and March:

```
ggplot(data=raw_data %>% filter(dayOfYearDO > 300), aes(dayOfYearDO)) + geom_histogram(binwidth = 1.0)
```




```
ggplot(data=raw_data %>% filter(dayOfYearDO < 100), aes(dayOfYearDO)) + geom_histogram(binwidth = 1.0)
```

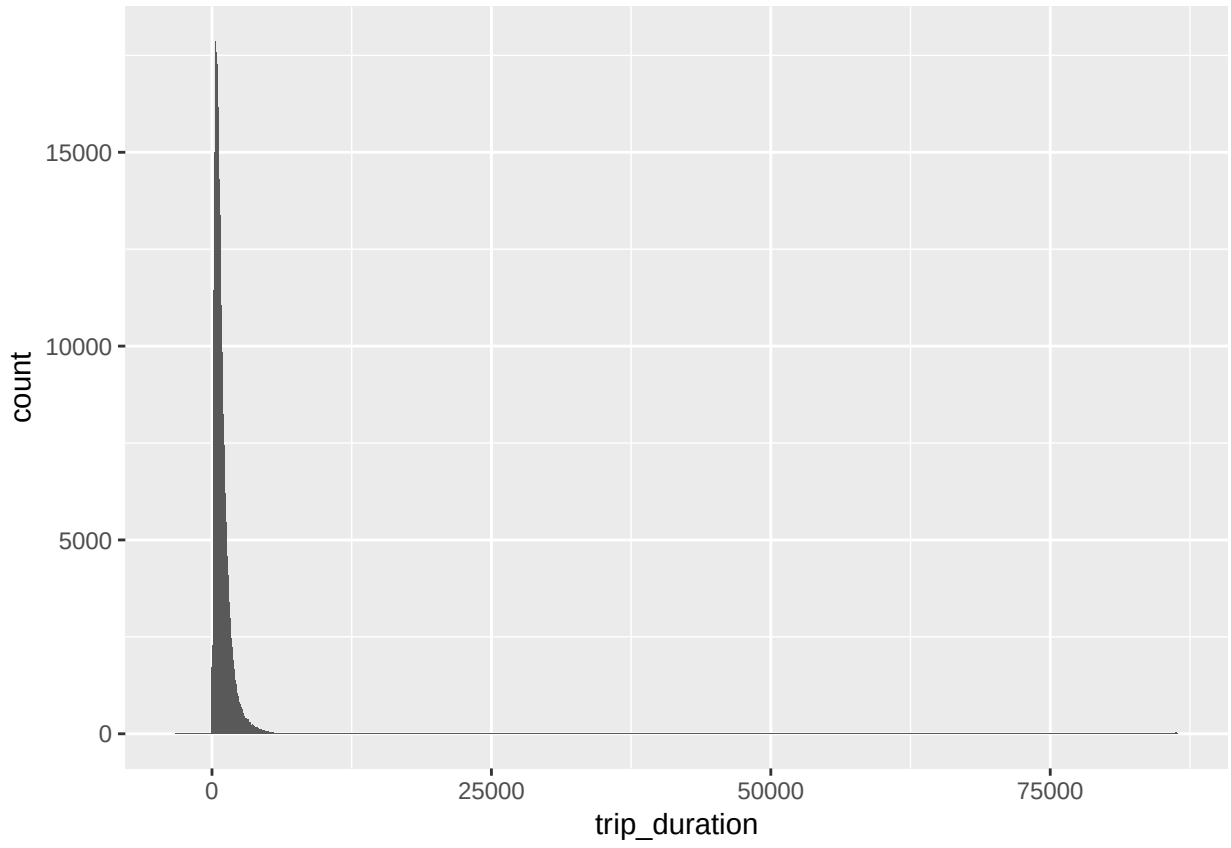


It was a false alarm. The large gap in DO was due to the granularity esblished in the previous histogram plot. However, it is rare the lack of data in the upper and lower limits in all months studied.

Trip duration

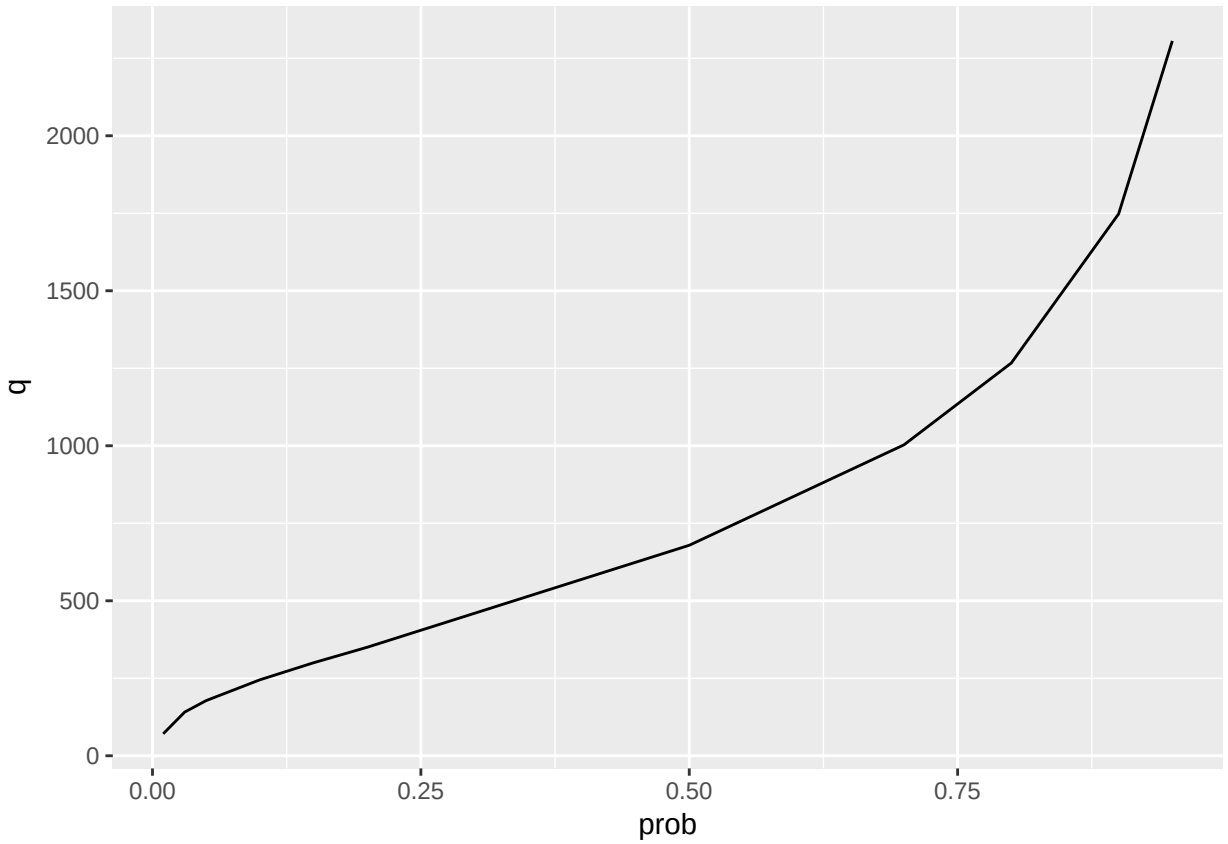
Let's create a new variable from timestamps which indicates the trip duration in rides. This numeric features will be much more interesting for our algorithm than raw datetimes.

```
raw_data <- raw_data %>% mutate(trip_duration = as.numeric(difftime(tpep_dropoff_datetime, tpep_pickup_
ggplot(data=raw_data, aes(trip_duration)) + geom_histogram(binwidth = 60.0)
```



It seems that there are some unrealist trips with too long rides. Let's observe if they represent a real nuance.

```
p = c(.01,.03,.05,.1,.15,.20,.50,.70,.80,.90,.95)
duration_quantiles = data.frame(q = quantile(raw_data[,"trip_duration"], probs = p),
                  prob = p)
ggplot(aes(x = prob, y = q), data = duration_quantiles) + geom_line()
```




```
duration_quantiles
```

```
##        q prob
## 1%    71 0.01
## 3%   141 0.03
## 5%   178 0.05
## 10%  245 0.10
## 15%  300 0.15
## 20%  350 0.20
## 50%  679 0.50
## 70% 1003 0.70
## 80% 1267 0.80
## 90% 1748 0.90
## 95% 2306 0.95
```

95% of values are below the natural threshold of 3600 s (1h). Actually, 50% of rides are below the 30-min limit.

It seems to me 2 hours in a taxi ride is too much, specially if the taxi cab smells bad, so it is a good limit to detect outliers. Instant trips seems unrealistic to me as well. So let's count how many outliers in this new feature:

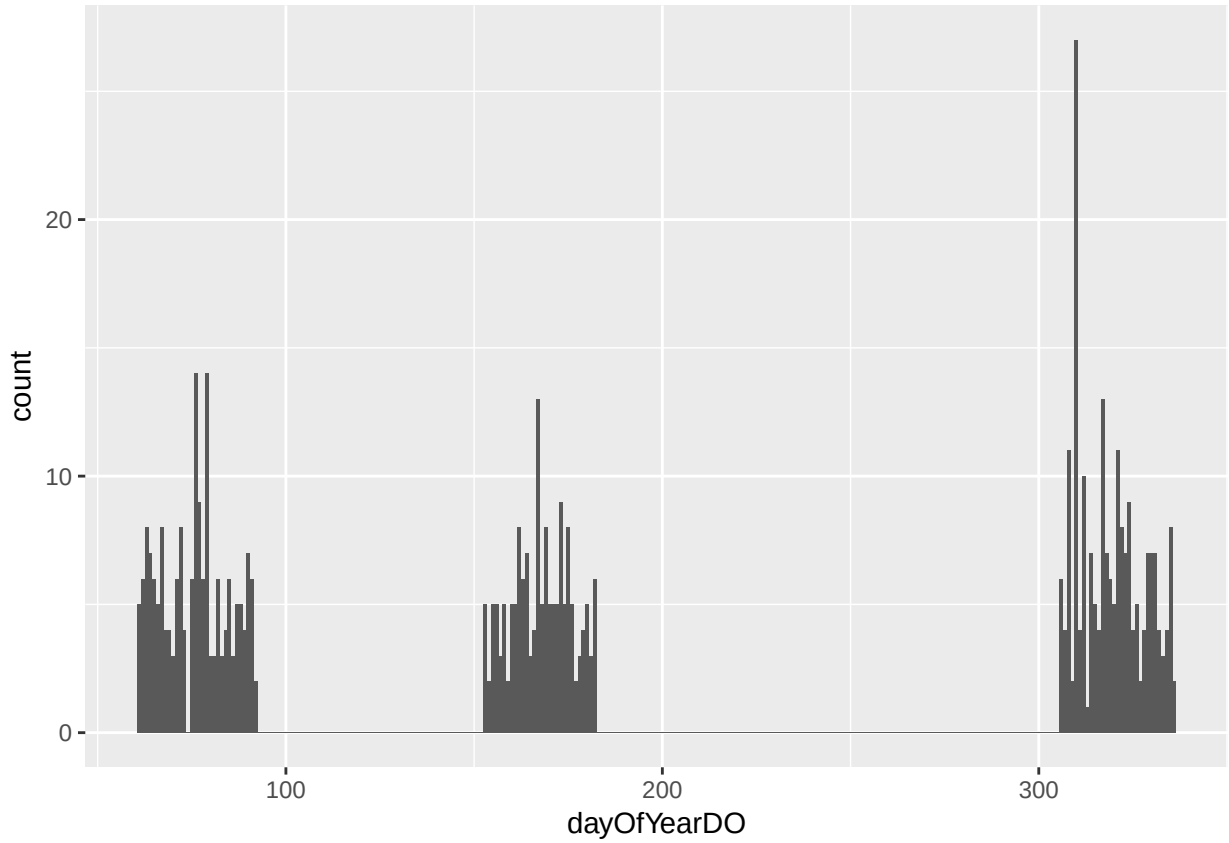
```
raw_data %>%  filter(trip_duration > 3600 * 2 | trip_duration <= 0) %>% count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   849
```

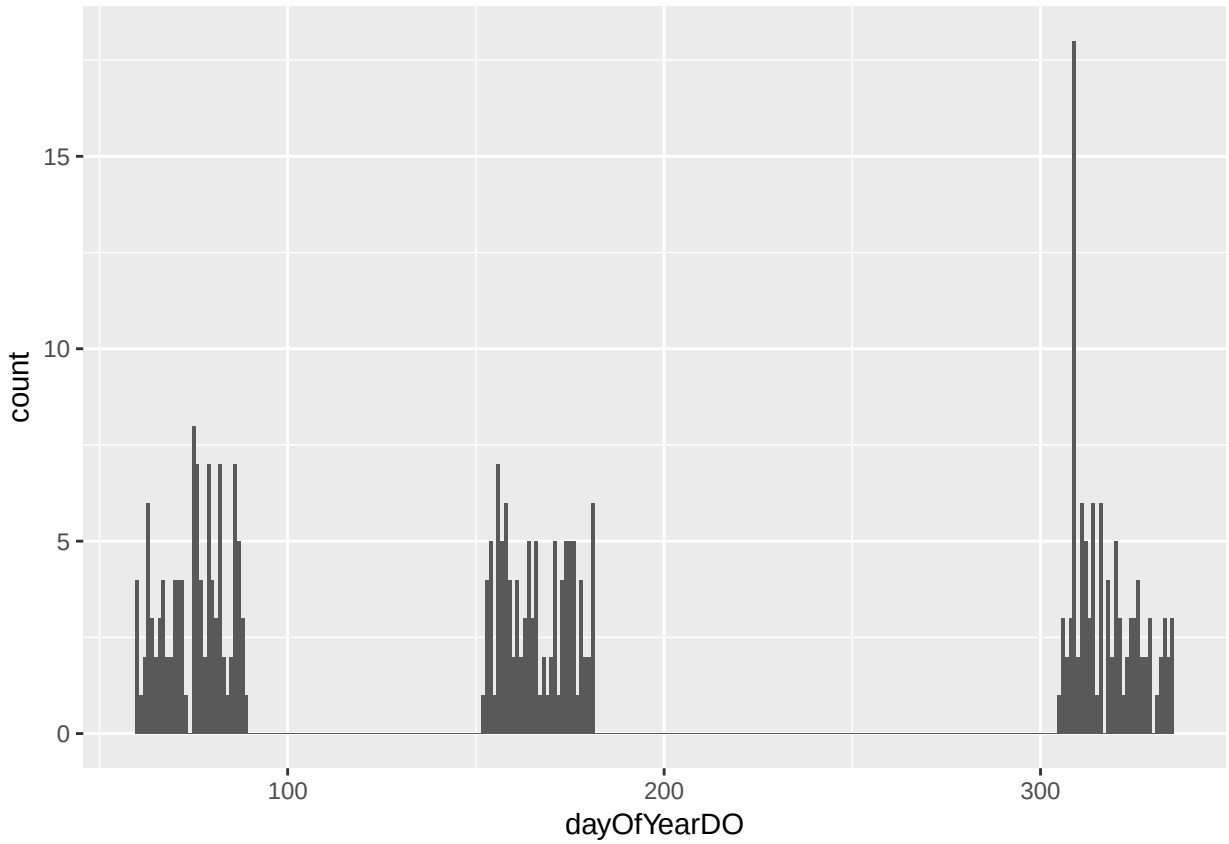
The proportion of eternal and lightspeed rides is under one percent, so we can move forward and saying it is safe to obviate these rows. We will set them to NaN in the modeling section.

Now let's draw these outliers in a temporal line to see if they come from the same period.

```
ggplot(data=raw_data %>%  filter(trip_duration > 3600 * 2), aes(dayOfYearDO)) + geom_histogram(binwidth
```




```
ggplot(data=raw_data %>%  filter(trip_duration <= 0), aes(dayOfYearDO)) + geom_histogram(binwidth = 1.0)
```



They seem well distributed among the temporal line. There are only a couple of days with no outliers though. a single day with no

```
raw_data %>% filter(trip_duration > 3600 * 2) %>% group_by(dayOfYearDO) %>%
  count() %>% arrange(desc(n))
```

```
## # A tibble: 92 x 2
## # Groups:   dayOfYearDO [92]
##    dayOfYearDO     n
##          <dbl> <int>
##  1         310    27
##  2          76    14
##  3          79    14
##  4         167    13
##  5         317    13
##  6         308    11
##  7         321    11
##  8         312    10
##  9          77     9
## 10         173     9
## # ... with 82 more rows
```

```
raw_data %>% filter(trip_duration <= 0) %>% group_by(dayOfYearDO) %>%
  count() %>% arrange(desc(n))
```

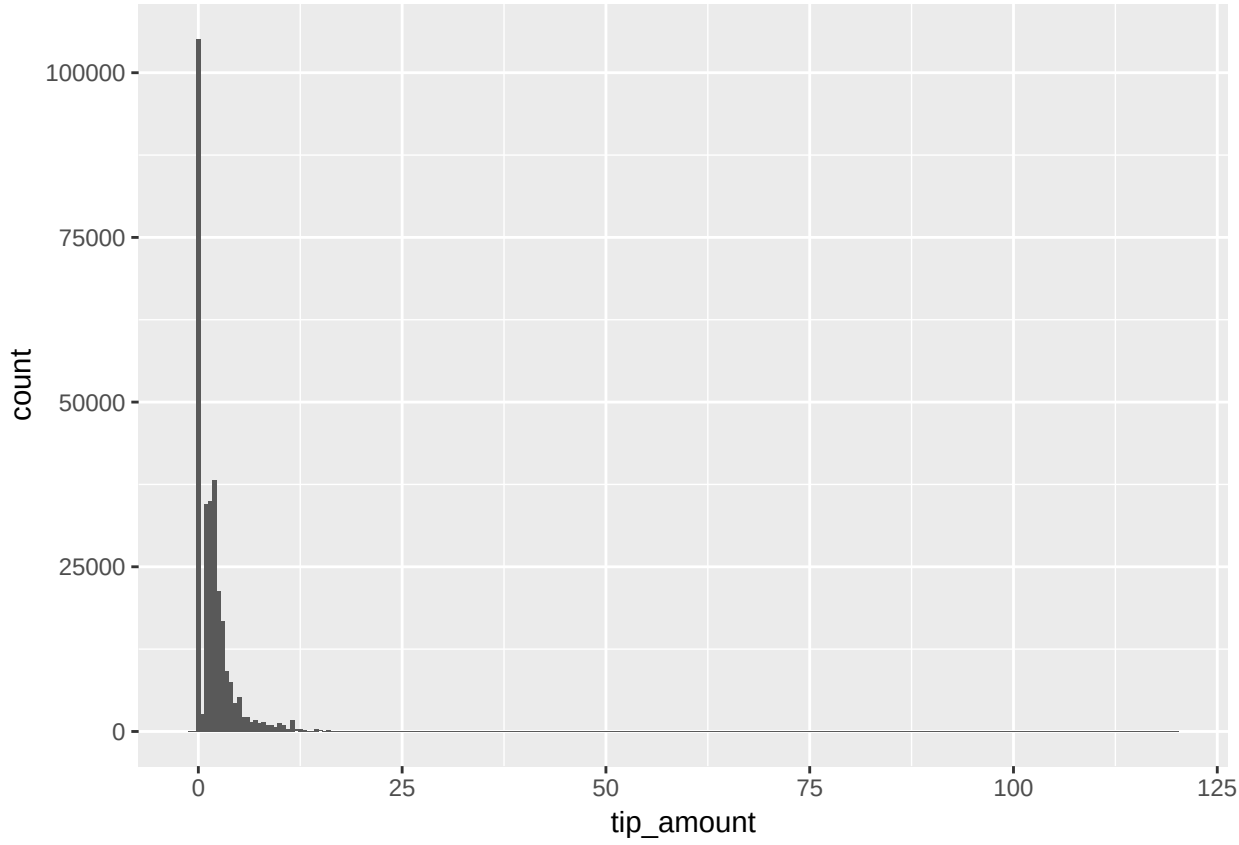
```
## # A tibble: 88 x 2
## # Groups:   dayOfYearDO [88]
##    dayOfYearDO     n
##          <dbl> <int>
```

```
##  1         309    18
##  2          75     8
##  3          76     7
##  4          79     7
##  5          82     7
##  6          86     7
##  7         156     7
##  8          63     6
##  9         158     6
## 10         181     6
## # ... with 78 more rows
```

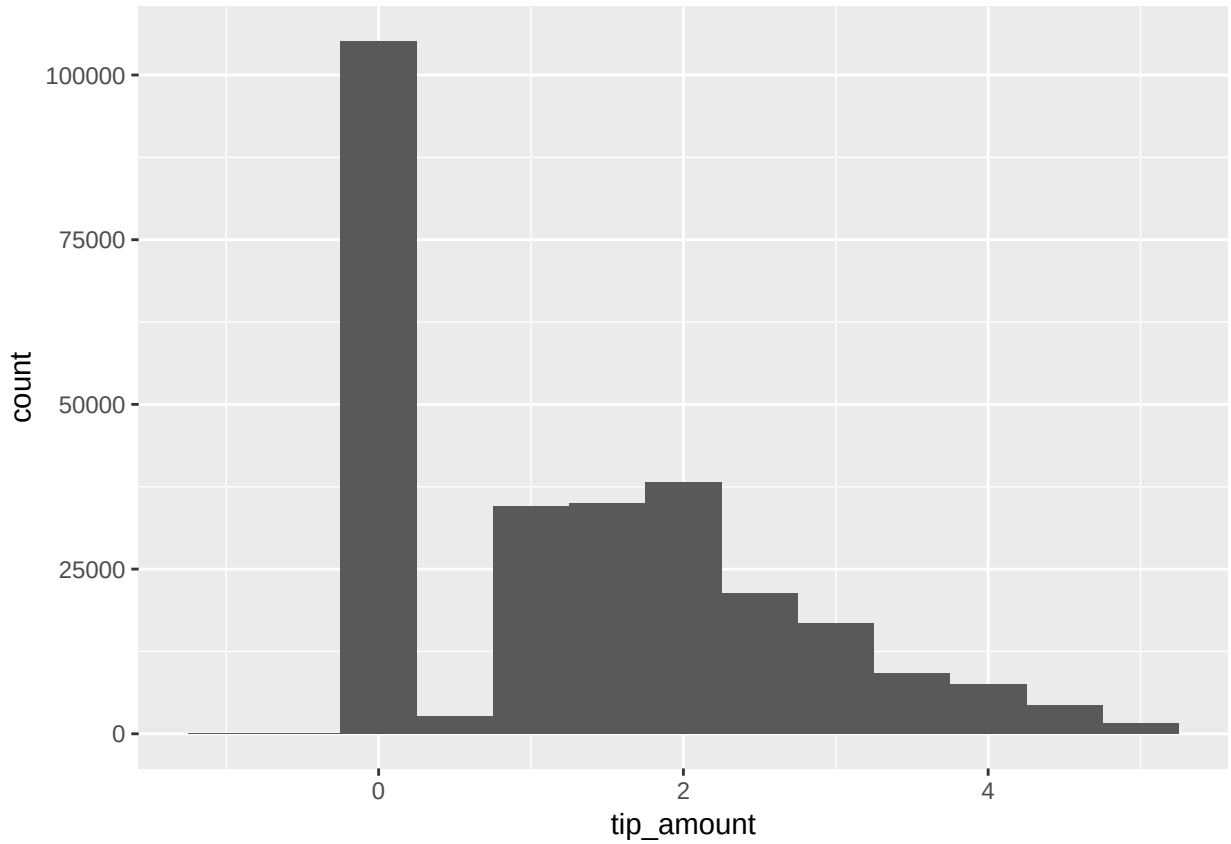
Day 309 and 310 seems noisy as they introduce most of outliers in our data, so maybe it is safe to remove them before applying any learning process.

Finally, we are gonna inspect the shape of the output variable: tip_amount.

```
ggplot(data=raw_data, aes(tip_amount)) + geom_histogram(binwidth = 0.5)
```




```
ggplot(data=raw_data %>% filter(tip_amount < 5), aes(tip_amount)) + geom_histogram(binwidth = 0.5)
```




```
raw_data %>% filter(tip_amount < 0) %>% count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     4
```

This important variables seems well-defined and with almost no outlier values, so lucky us!

Bivariate analysis

Let's check how strong/weak is the relationship among each input feature and our output objective (tip_amount) # Matrix of correlation

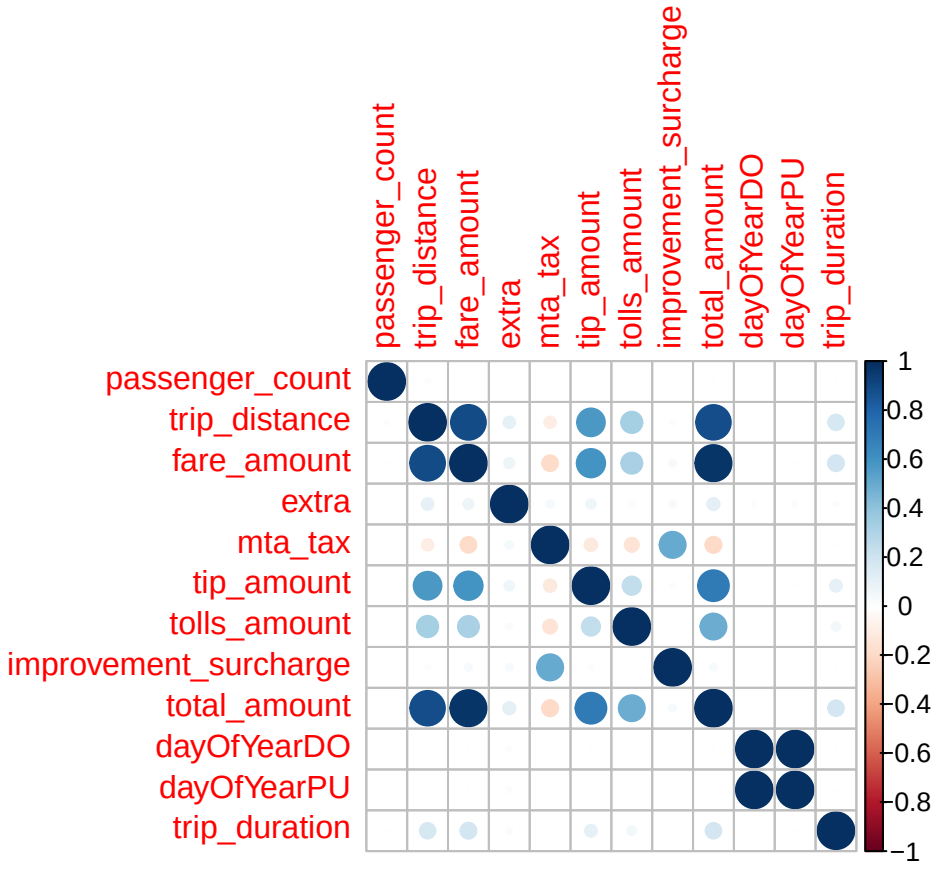
```
numeric.columns <-c("trip_duration", "passenger_count","trip_distance","fare_amount","extra","mta_tax",'
data.numeric <- raw_data[, which(names(raw_data) %in% numeric.columns)]

sapply(data.numeric, class)
```

```
##       passenger_count         trip_distance           fare_amount
##             "integer"             "numeric"             "numeric"
##                 extra              mta_tax            tip_amount
##             "numeric"             "numeric"             "numeric"
##          tolls_amount improvement_surcharge          total_amount
##             "numeric"             "numeric"             "numeric"
##           dayOfYearDO           dayOfYearPU         trip_duration
##             "numeric"             "numeric"             "numeric"
```

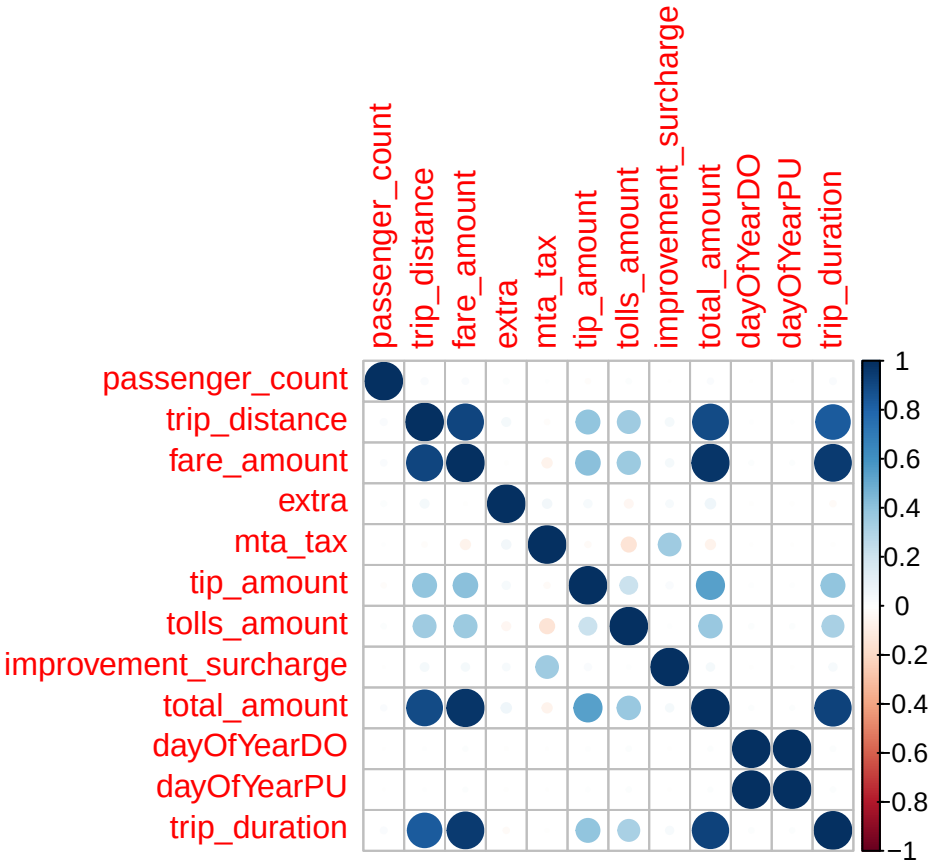
```
M <- cor(data.numeric) # get correlations
M.spear <- cor(data.numeric, method ="spearman")
library('corrplot') #package corrplot
```

```
## corrplot 0.84 loaded
```

```
corrplot(M, method = "circle") #plot matrix
```



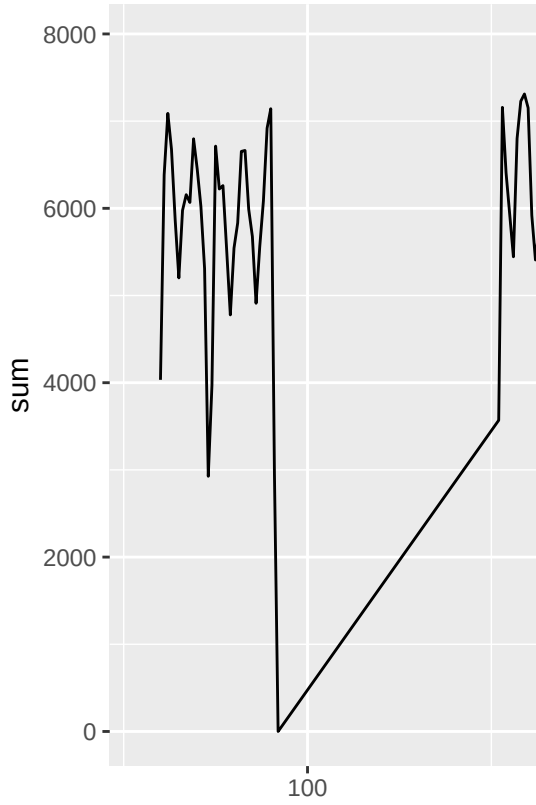

```
corrplot(M.spear, method = "circle") #plot matrix
```



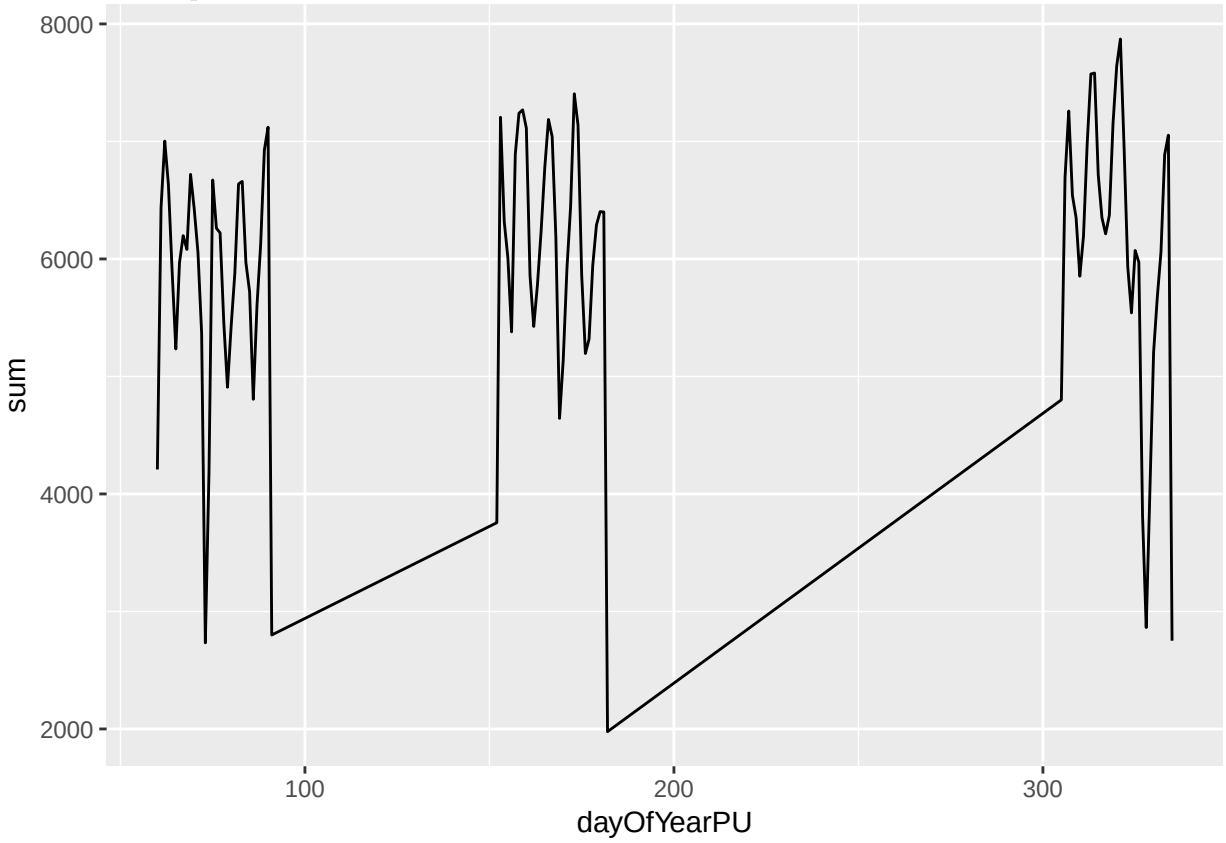
According to the graph above, tip_amount is correlated with total_amount and fare_amount, as well as trip_distance and trip_duration (our newly created feature) which at the same time is related to fare_amount.

It seems not to be a relationship among time and monetary-based features, at least in a daily basis. However, by adding more time-based features (such as, weekOfYear, dayOfWeek, hour, etc.) we may detect new correlations.

Time-tips



Let's check in depth relation with time, and then, with the most relevant features:

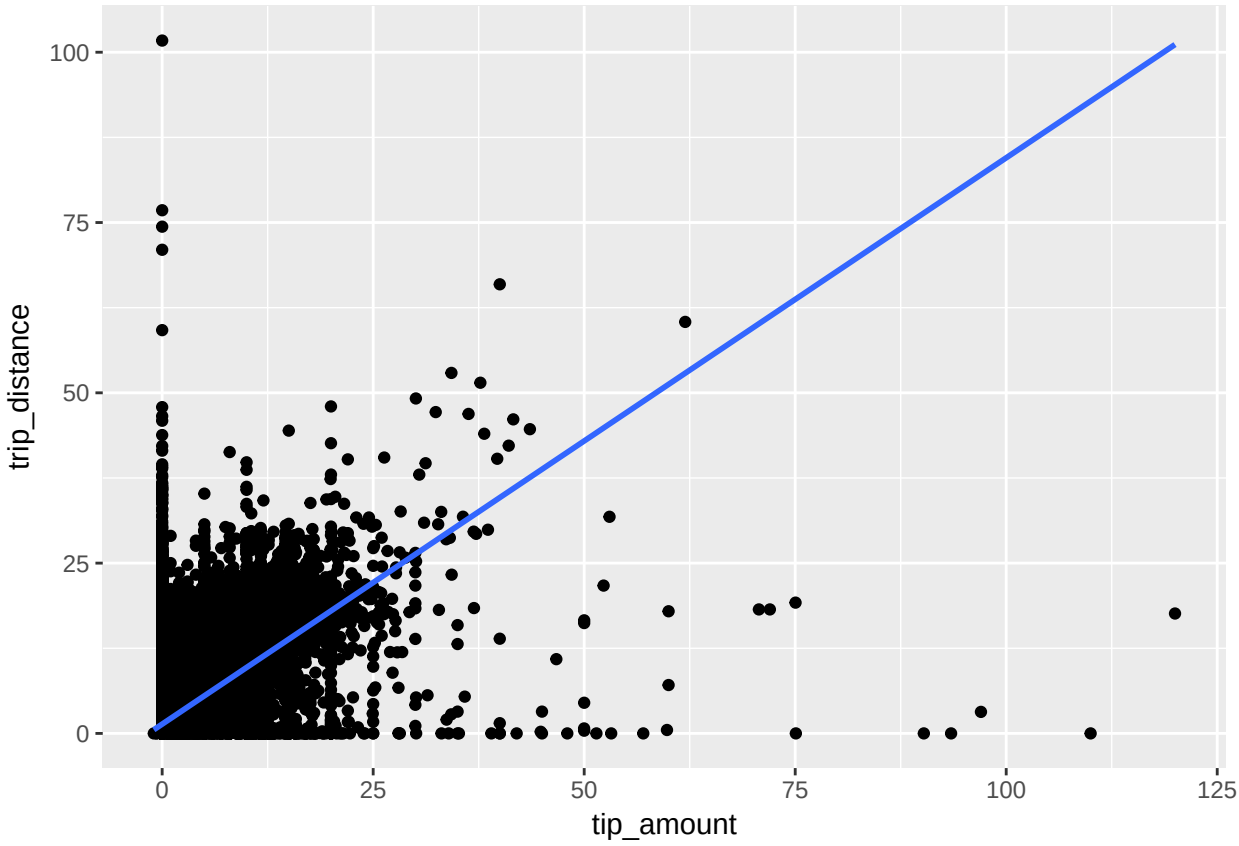


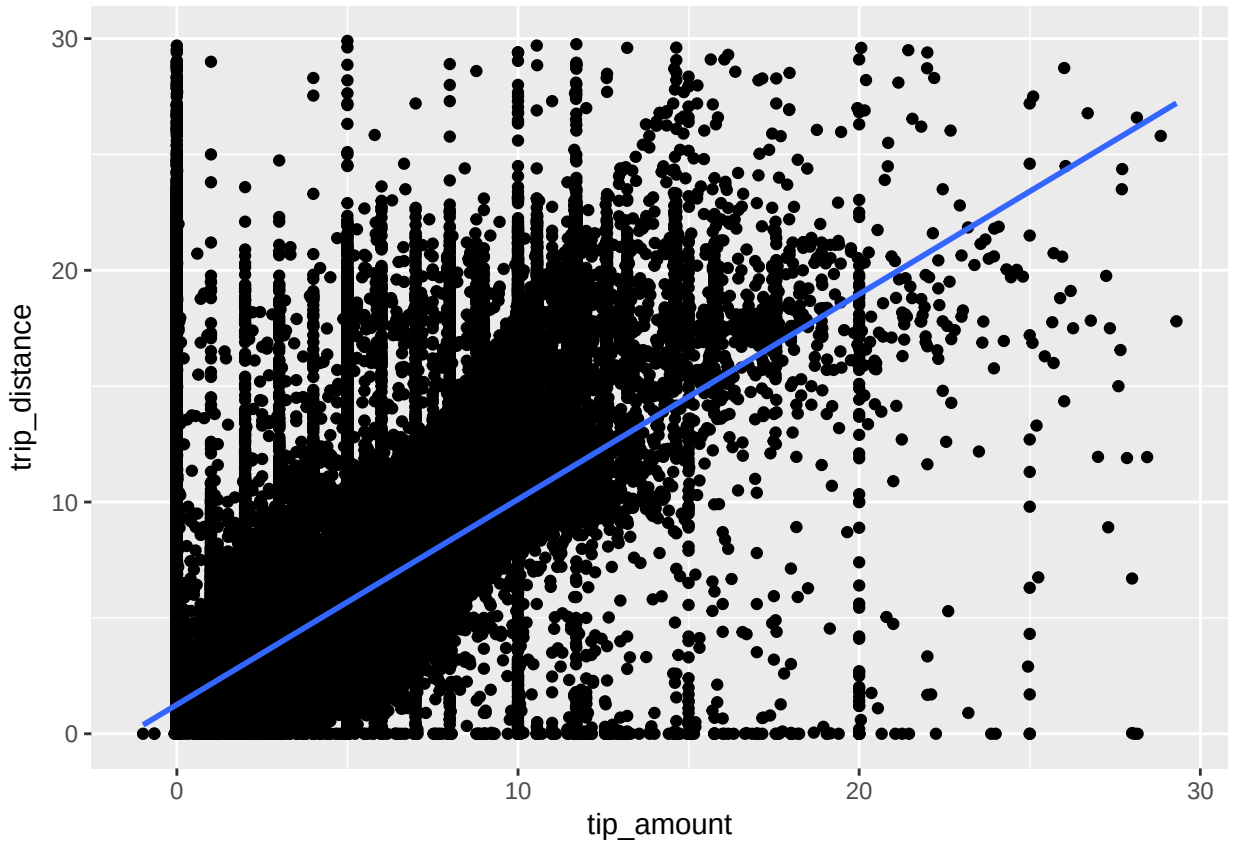
```
subset <- raw_data %>% mutate(monthDO = as.numeric(strftime(tpep_dropoff_datetime, format = "%m")), mon
subset %>% group_by(monthDO) %>% summarise(mean=mean(tip_amount))
```

```
## # A tibble: 6 x 2
##   monthDO  mean
##     <dbl> <dbl>
## 1       3  1.84
## 2       4  1.90
## 3       6  1.88
## 4       7  1.78
## 5      11  1.89
## 6      12  2.19
```

In the graph above, it seems tips grow in recent months despite a really bad period in November. The same analysis can be performed in hourly basis.

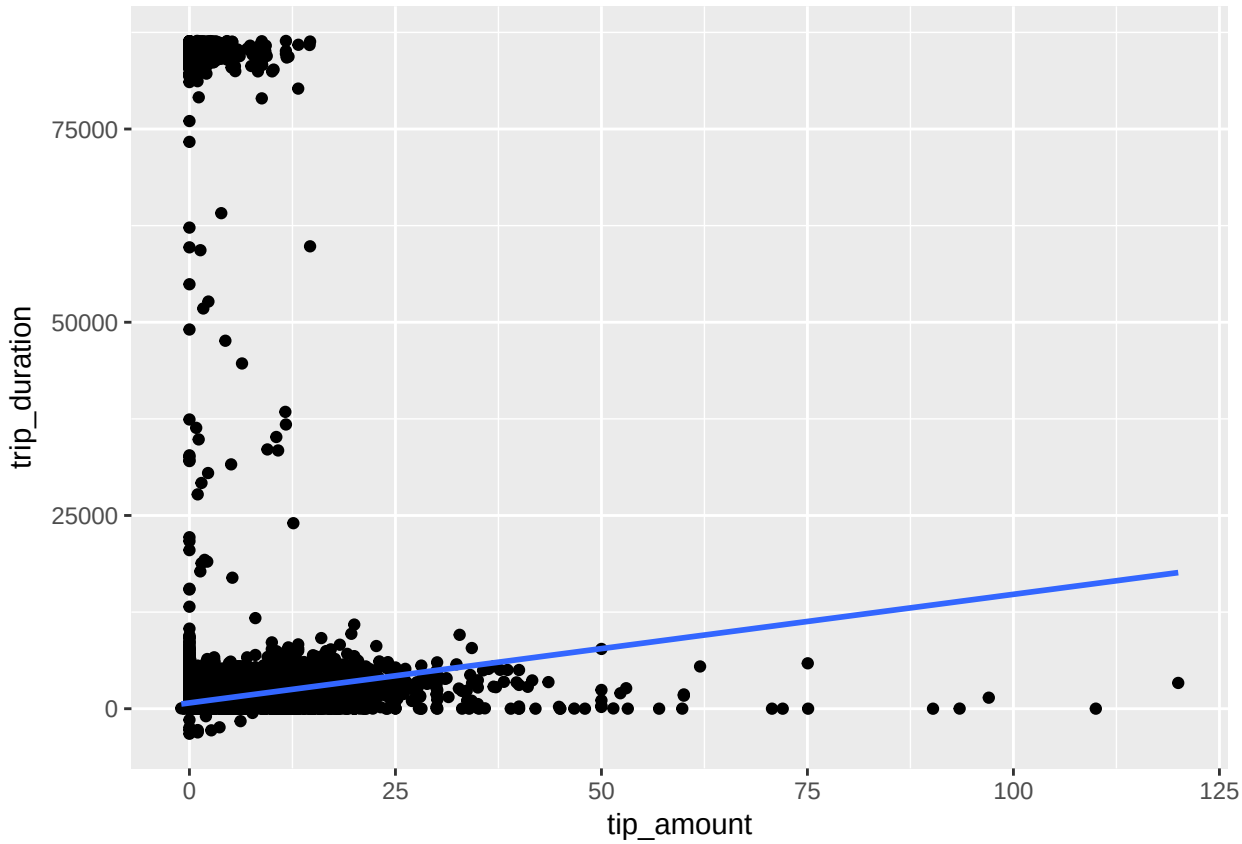
We can move now to analyze any trend among tip_amount and the most correlated features discovered before.

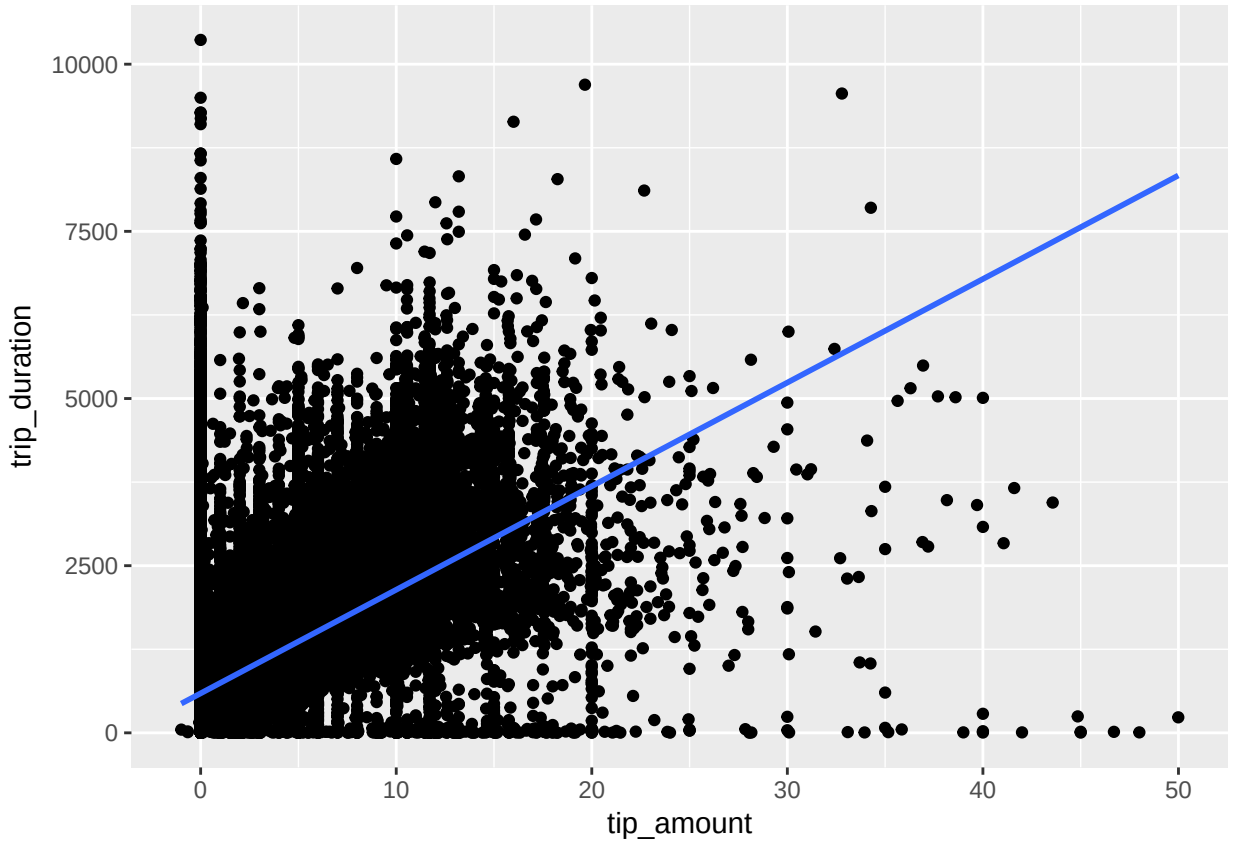




By removing outliers, we can observe a linear relationship among trip_distance and tip_amount, which means that normally long distances imply longer benefits. However, I expect a stronger relationship with trip_duration as in yellow cabs, fares are usually computed according to duration not distance.

In the same graph, we can notice other outliers like zero-distance trips.






```
##  [1] "VendorID"              "tpep_pickup_datetime"
```

```
##  [3] "tpep_dropoff_datetime" "passenger_count"
##  [5] "trip_distance"         "RatecodeID"
##  [7] "store_and_fwd_flag"    "PULocationID"
##  [9] "DOLocationID"          "payment_type"
## [11] "fare_amount"           "extra"
## [13] "mta_tax"               "tip_amount"
## [15] "tolls_amount"          "improvement_surcharge"
## [17] "total_amount"          "dayOfYearDO"
## [19] "dayOfYearPU"           "trip_duration"
```

Again, the relationship is linear trending and similar in strength to trip_distance.

With this plot we can conclude the exploratory analysis