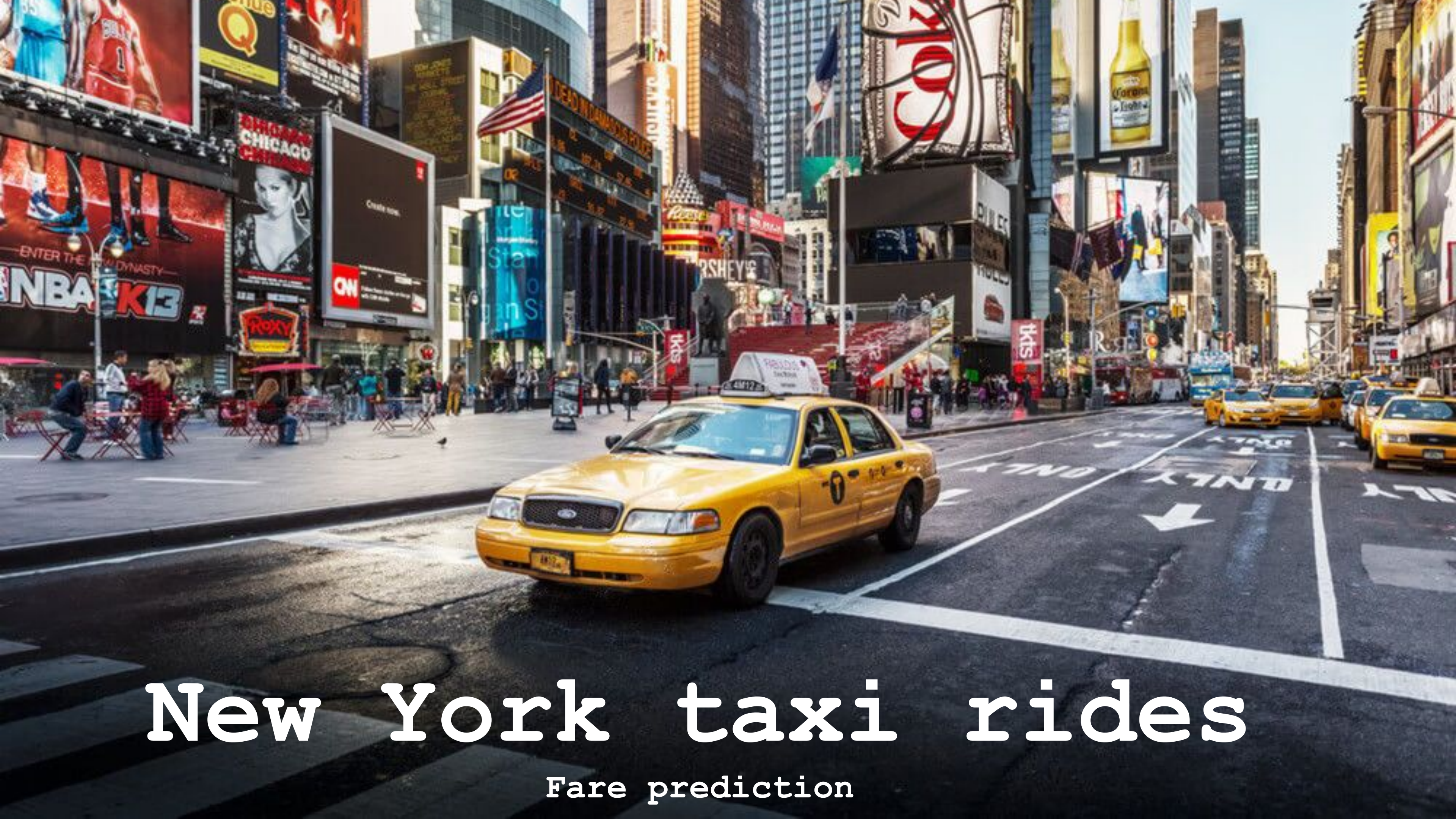



# New York taxi rides

Fare prediction

# Target  and Datasets description

*“Predict taxi ride duration in NYC”*

Datasets description:

=> Set : 55M records and 8 attributes

Records of  taxi rides in New York from 2009 to 2015

# Codebook :

id - a unique identifier for each trip
pickup_datetime - date and time when taxi ride started
passenger_count - the number of passengers in the vehicle
pickup_longitude - the longitude where the meter was engaged
pickup_latitude - the latitude where the meter was engaged
dropoff_longitude - the longitude where the meter was disengaged
dropoff_latitude - the latitude where the meter was disengaged
fare_amount - dollar amount of the cost of the taxi ride
source: https://www.kaggle.com/c/new-york-city-taxi-fare-prediction/data

# Steps

I - Data loading

I- Data Cleaning and Manipulation

III- EDA

IV- Modeling

- Feature selection and engineering
- model testing
- model selection

V - First conclusions

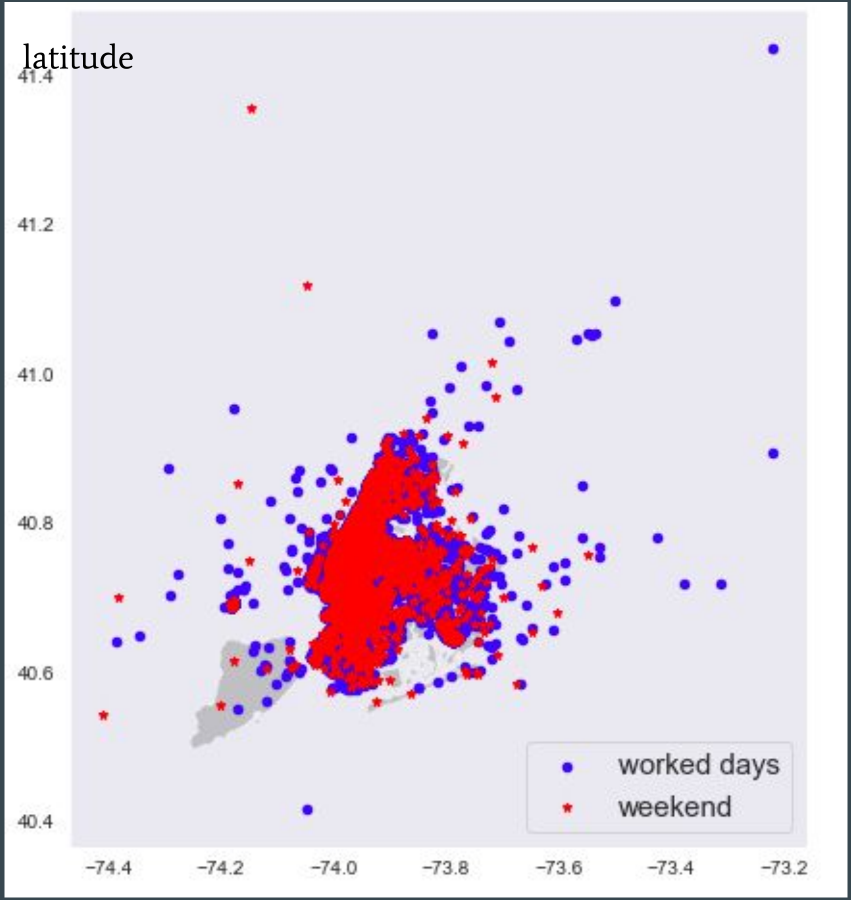
## Data Cleaning

- drop duplicates
- drop columns not significant
- drop null values
- drop inconsistent data such as : ride with not distance, no passenger, no fare_amount.

## Data Manipulation

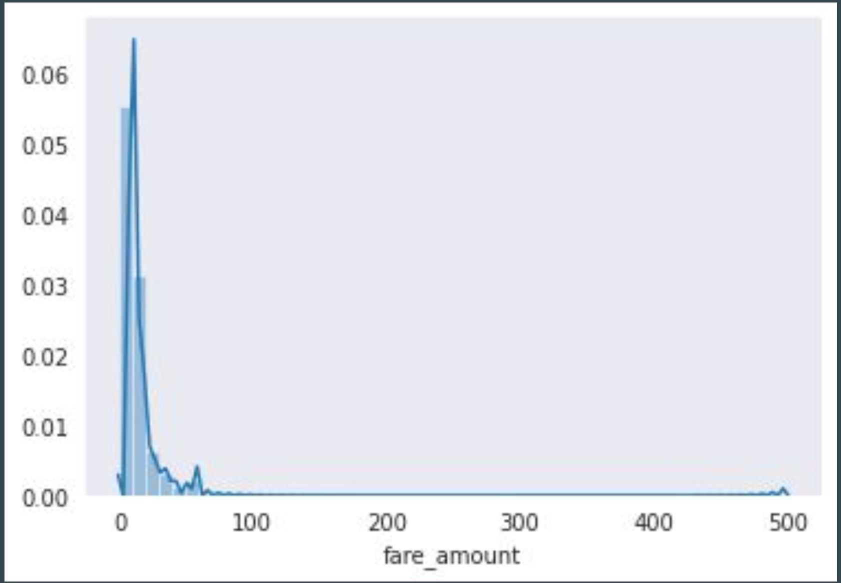
- Convert to datetime format
- Split date in Month, day, weekend, hour of the day
- calculate an approximation of the distance
- reduce the spatial scope to around Manhattan, Brooklyn, Queens, Staten Island

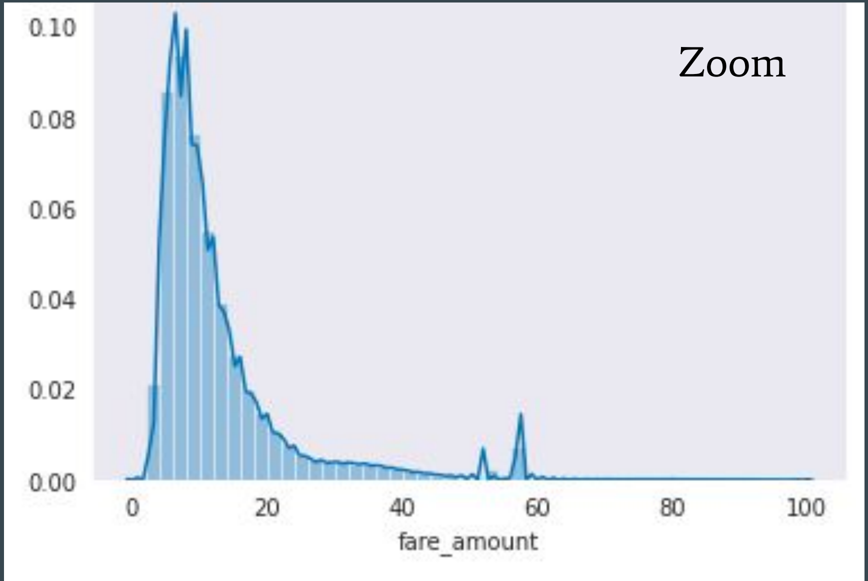
# Let’s have a look to the geographical distribution


longitude

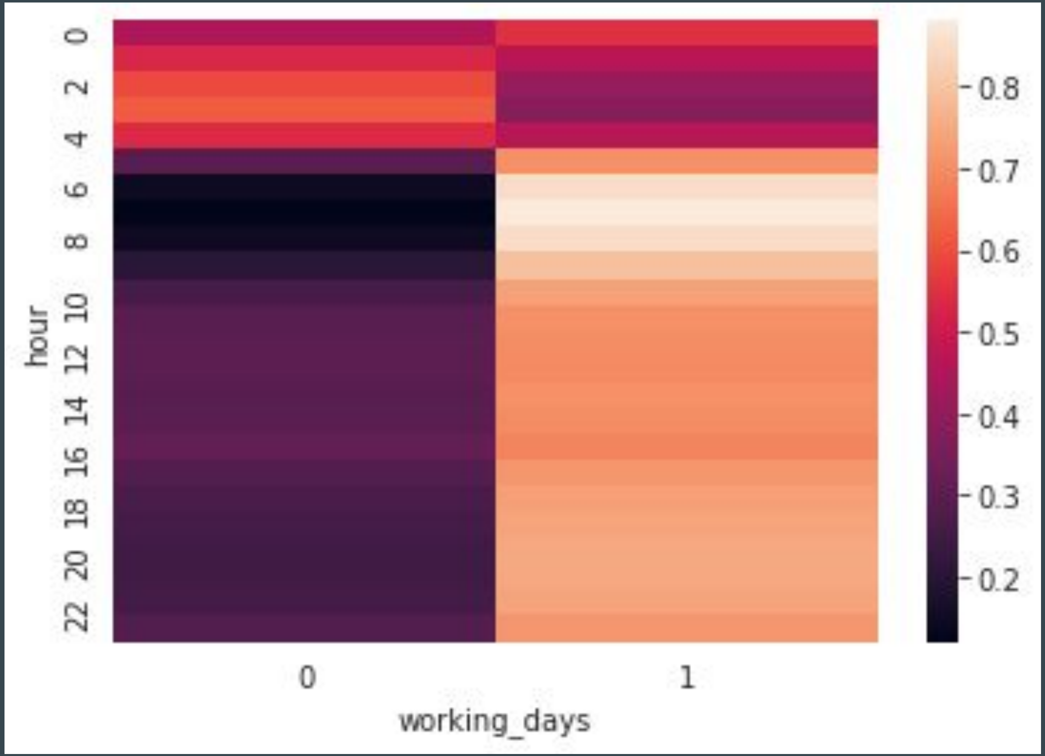
# Fare amount distribution



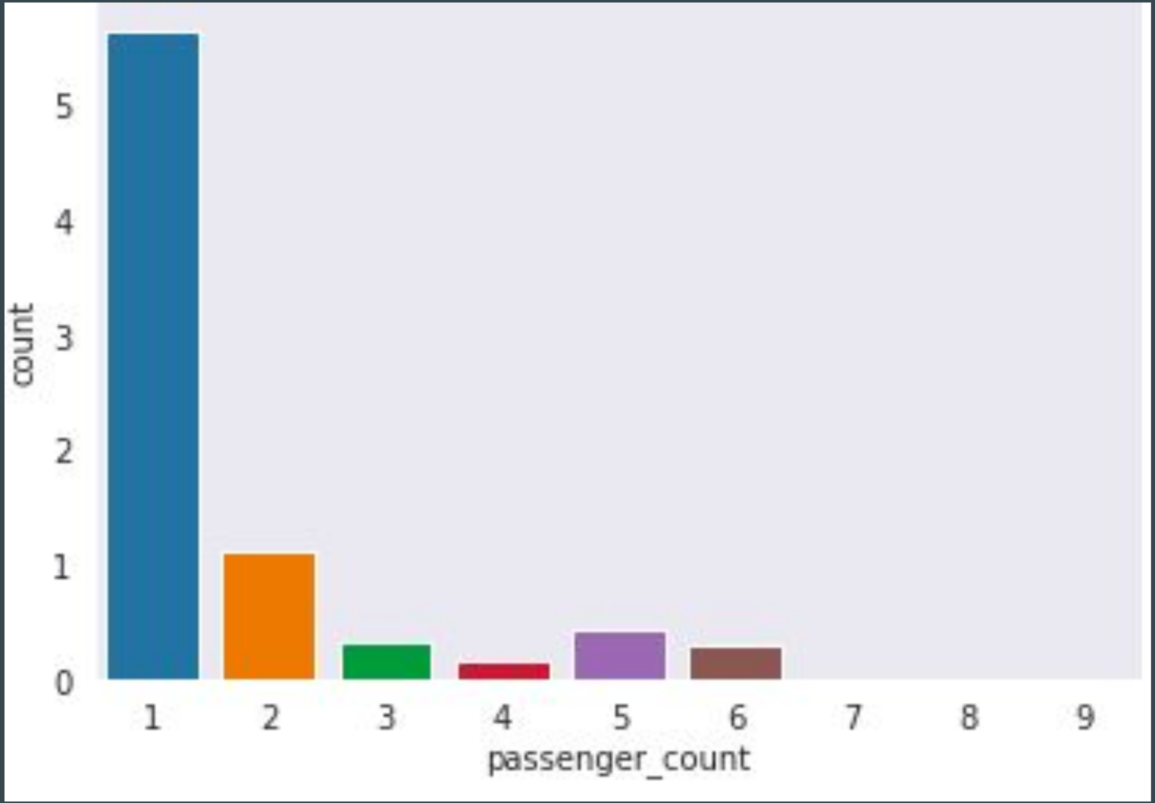

Mean fare: 12.9$

# How rides are split depending on the hour/day type




Worked day : 1
weekend : 0

# Mostly single passenger rides



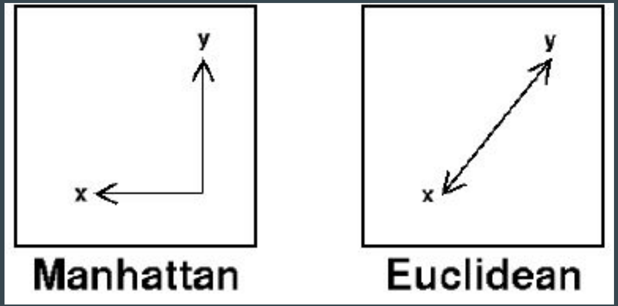
# 2 types of distance calculation tested

## With haversine formula

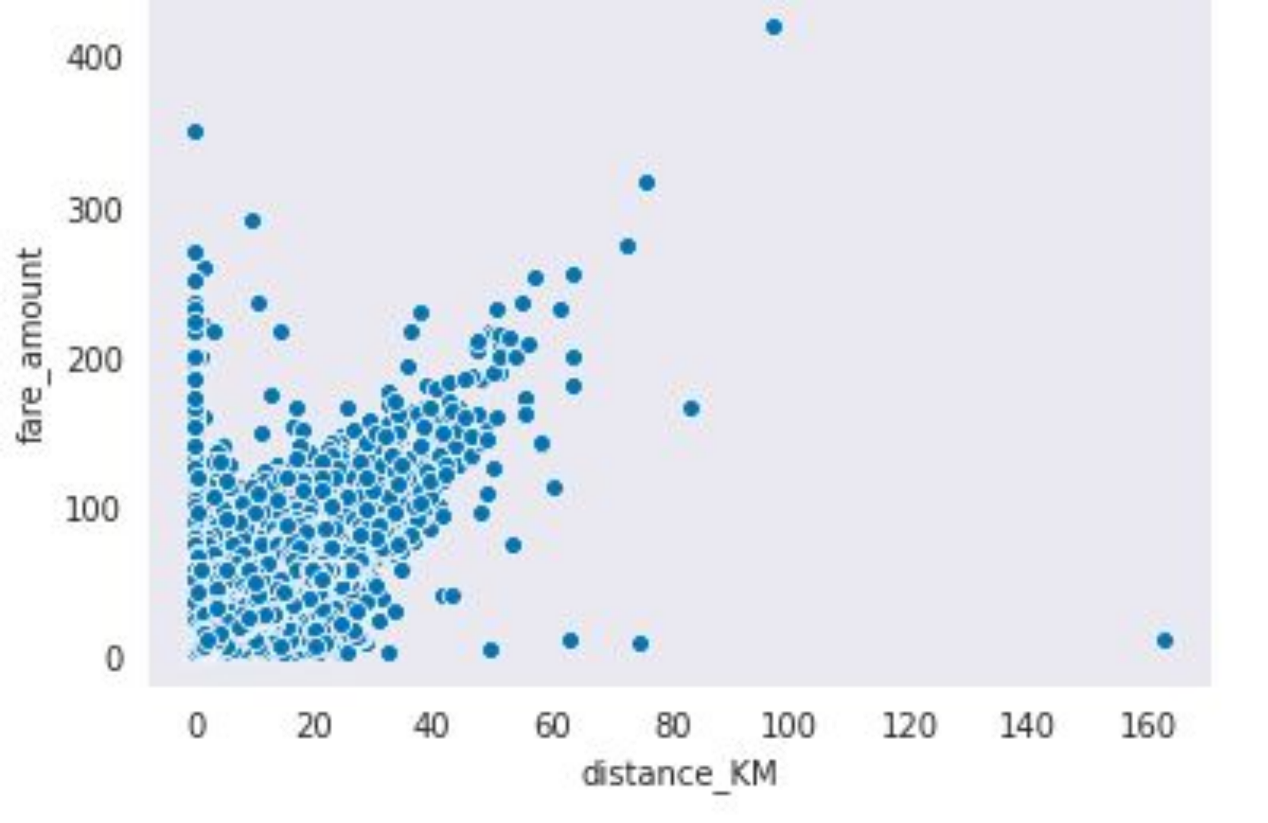
```
dlon = lon2 - lon1

dlat = lat2 - lat1

a = (sin(dlat/2))^2 + cos(lat1) *
cos(lat2) * (sin(dlon/2))^2

c = 2 * atan2( sqrt(a), sqrt(1-a) )

d = R * c (where R is the radius of
the Earth)

 R = 6367 km
```

## With Manhattan distance

The distance between two points is the sum of the absolute differences of their Cartesian coordinates



# Fare amount vs. Distance




Correlation : 0.89

# Modeling : Tested model

Target : *“To cluster the insignificant data in order to save information without too much attributes during the prediction”*

Step 1: Feature selection and engineering

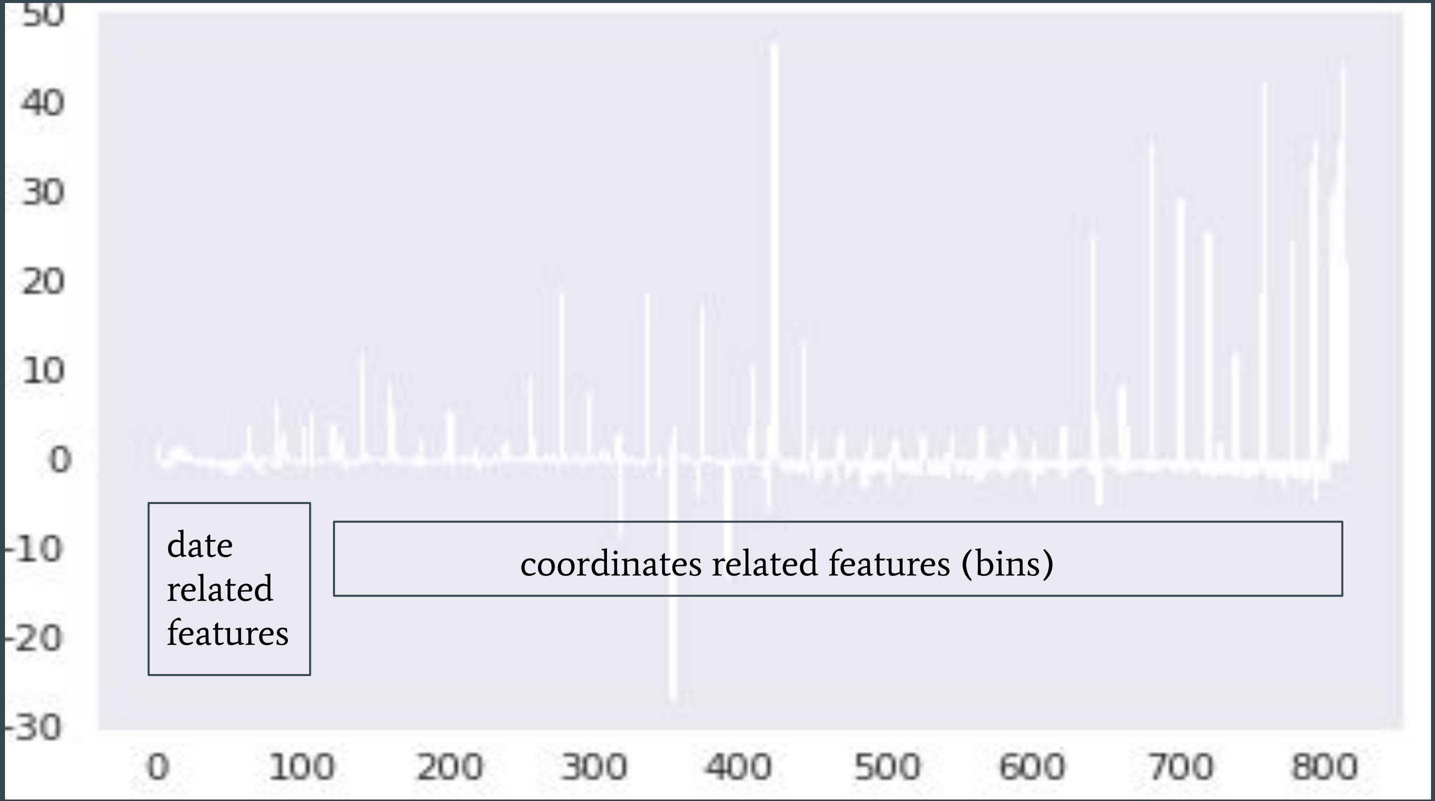
- coordinates engineering:
    - treat latitude and longitude as categories
    - bin them with qcut (20 bins for each) and create virtual points combining latitude and longitude

Step 2 : model testing :

- OLS without feature engineering: R-squared : 0.312
- OLS with feature engineering (Haversine) R-squared : 0.847
- OLS with feature engineering (Manhattan) R-squared : 0.832
- Ridge with built-in cross validation and feature engineering (Manhattan) R-squared : 0.841

# Selected model: Ridge

R-squared is good :R-squared : 0.841 and It tackles the issue of  coefficient being really high in OLS model



# First conclusions

Model is good

Possible improvement:

- test it with Haversine distance calculation.
- use more data