# A Network Tour to Flight Delay in the US

FENG Wentao, FU Yan, SUN Zhaodong, WANG Yunbei
*Science et Techniques de l'Ingenieur, EPFL, Switzerland*

***Abstract*—This is a report for Network Tour of Data Science. In this project we analyzed on flight route data set and flight delay data, using graph theory and machine learning method, tried to create a model to predict possible delay time for given features (e.g., departure city, date), and advise on best choice of flight.**

## I. Introduction

In this project, we explored our data set and set up a model predicting delay in minutes for given conditions, and based on this model we can advise on choosing flights on different seasons, airlines and departure hours. Since it is difficult to get the whole world delay data set, the delay analysis is mostly based on the US flight delay data in our project.

Generally, in this project we tried to solve these problems:

- What will affect mostly on the delay of a flight, could it be departure season, hours or airlines?
- And can we predict how long will a flight delay with given certain situations?
- Or, if someone wants to fly from city A to city B, can we give him or her some advice on choosing flights based on the delay rate?

So, to solve these problems, we created our graph using airports as nodes, the distance between airports after Gaussian kernel as weights and used the spectral graph to do clustering. Also, we used machine learning method to do the regression on features (departure date, hour, airport, destination airport, distance and air time) and successfully predicted delay time in small root mean square error.

## II. Data Exploration

### A. OpenFlight

*1) Overview of data set:* data set is obtained from: Open Flight Data Set (https://openflights.org/data.html). This OpenFlights/Airline Route Mapper Route Database contains 67,663 routes (EDGES) between 3,321 airports (NODES) on 548 airlines spanning the globe.

Below shows how we created our graph. In our first thought, the airports in the same region should have stronger connections and show same properties (e.g. average delay time in our project), because airports in the same region should have a similar background (e.g., flight policies, infrastructure level) and we can analyze different conditions on airports from a similar background.

We had latitude and longitude of departure and arriving airports so we can calculate their distance. Then we first normalized the distance feature and then converted distance to weights with a Gaussian kernel, airports with longer distance will have lower weights:

$$W = exp(-\frac{Distance^2}{\delta^2})$$

Then we created our graph with airports as nodes, Gaussian kernel as weights. Moreover, we plotted the eigenvalues of our graph, noticing that at 97th eigenvalue there was a jump. So we performed k-means clustering with k=97 and used the predicted k means result as the labels over our graph. We can see in figure 1 that approximately there are clusters in continents or countries.

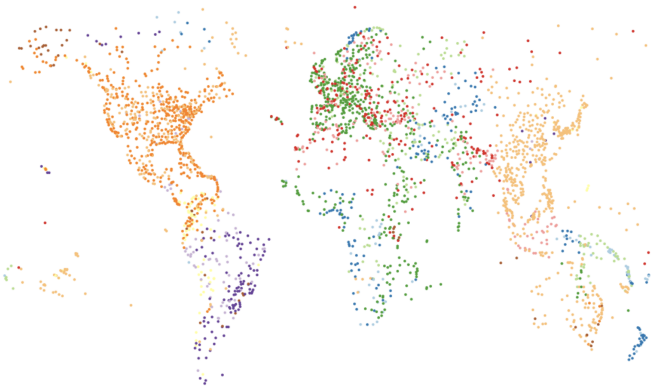

Figure 1: Clustering on Open Flight data set

We also analyzed on the passengers of airports. In routes, the column `Equipment` contains the aircraft types of that routes. We calculated the passengers of that route using the maximum of the aircraft capacity. For routes where multi-aircraft are used, we calculated the averaged capacity. The equipment not on our list are mainly helicopters and other small planes, which are not commercially used, so dropping these routes will not affect much. And we calculated the total in-going and out-going passengers based on routes flying through that airport and listed top 10 busiest airports. Comparing the list to the real top 10 busiest airports worldwide, we can see that two lists overlap a lot. In table I, we can see that the US is the country with largest flight passengers.

### B. Flight Delay

*1) Overview of data set:* As the US is the busiest country with largest air passengers, it is typical to look into details of the US and then apply a similar method to other regions. From Kaggle (Flight Delay(https://www.kaggle.com/usdot/flight-delays)) we down-

| Country | Total Flight Passengers |
|---|---|
| United States | 3.536720e+06 |
| China | 2.619732e+06 |
| United Kingdom | 9.498741e+05 |
| Spain | 8.445502e+05 |
| Germany | 7.737909e+05 |

Table I: Air passengers of countries

loaded a new data set containing delay information of US flight routes.

This data set is collected by the U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics, who tracks the on-time performance of more than 5,714,000 domestic flights operated by large air carriers from 01/01/2015 to 31/21/2015. Features include flight number, airline, departure and arrival time, delay time, air time, distance, origin and destination airport and delay reason( not for all delayed flights). Also, the documents including the relationship of IATA code and full information of airports and airlines are given.

Moreover, we also performed data pre-processing on flight delay data set. We selected flights not canceled and not diverted and converted data into the desired data type.

*2) Statistic Analysis:* Then we did some statistical analysis of different features, and here we showed some interesting findings. We can see that with different airlines, the day of the week, month, time of the day and departure city, the delay time differs. And generally, the arrival delay time would be shorter than departure delay time, which means the airline will speed up a little to counter back departure delay.

First we looked at the averaged delay time of different airline in figure 2. Generally Hawaiian Airlines and Alaska Airlines have lowest delay time.

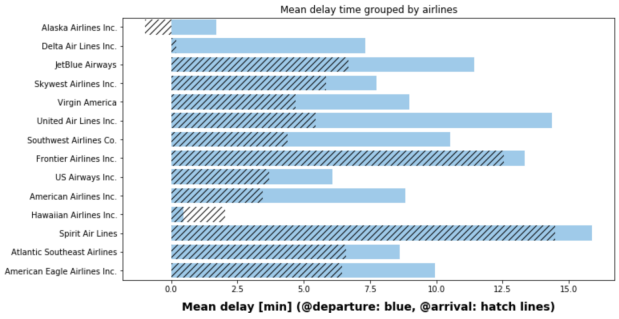


Figure 2: Mean delay time grouped by airlines

To see if the day of the week of a flight will influence the delay time. In figure 3, we can see that Monday has the longest averaged delay and Saturday the shortest.

Next, we plotted averaged delay time of different months in figure 4. Generally, there is a fluctuation in season. Spring and autumn(February, March, April and September, October, November) have shorter delay time while the rest have longer time. June and December have the longest delay time, which may be explained by the summer holiday and Christmas.

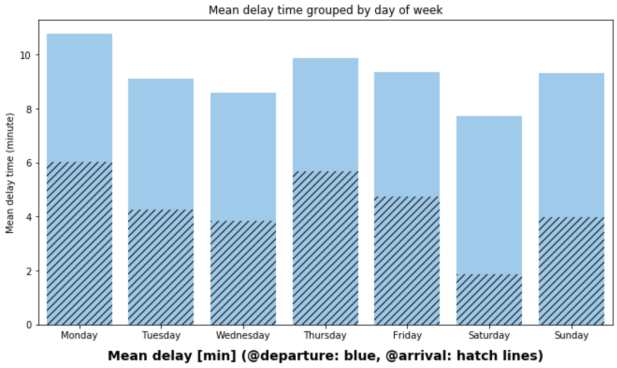


Figure 3: Mean delay time grouped by day of the week

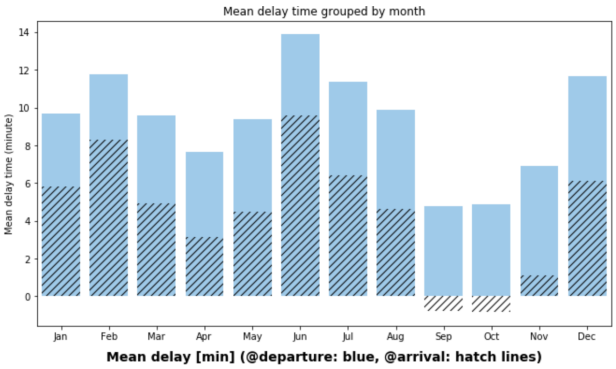


Figure 4: Mean delay time grouped by month

Finally, we plotted the delay time over time of a day in figure 5. It is interesting that the delay time grows gradually after 5 am and slowly drops after 20 pm. It reflects in reality, that morning flights are less likely to delay, while later flights, because of delay of former flights, will be more likely to delay.

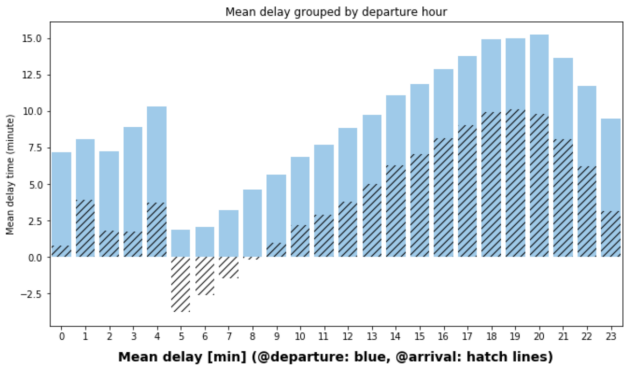


Figure 5: Mean delay time grouped by departure hour

## III. Data Exploitation

### A. Graph Clustering and Delay Analysis

We generated a new graph again using the same method mentioned above, but this time we only used data from the US. We first used spectral clustering to analyze delay patterns. Reversely, we used the delay patterns to partition the graph.

Finally, We used the transductive learning to recover the whole delay patterns.

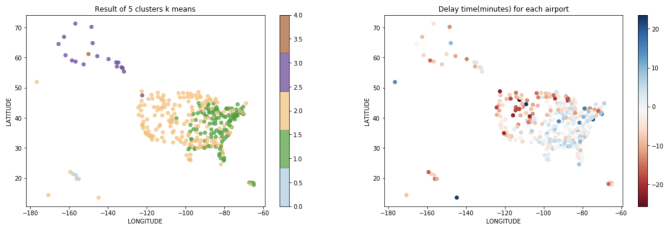


Figure 6: clusters for the geographical positions(left) and delay time distribution(right)

*1) From Spectral Clustering to Delay Analysis:* figure 6 (right) shows the mean arrival delay time of all airports. Airports in the west coast and east coast have different patterns. The west coast has a lower delay time. From this plot, we assumed that the number of clusters should be small so we computed the first ten eigenvalues to decide the number of clusters. In figure 7 the eigenvalues after index 5 kept almost stable, so we chose five as the number of clusters.

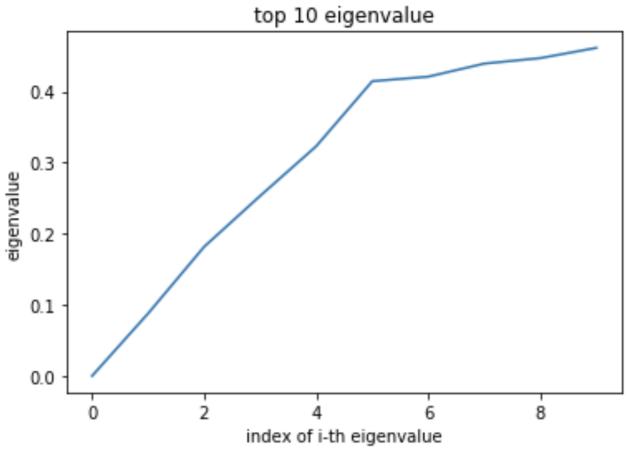


Figure 7: The first 10 smallest eigenvalues

Then we performed the k-means algorithm on the first five eigenvectors, and the results are shown in figure 8. In figure 8, all the nodes are embedded on the 2-dimensional plane, and the color represents the corresponding class. In figure 8 (bottom), we zoomed in the left part and observed this largest cluster could be divided into two classes, which might provide some important information. For better data visualization, The clustering result is shown in figure 6(Left), and all the nodes were positioned corresponding to their geographical positions. There is a close relationship between the result of clusters and the delay time for each airport as shown in figure 6.

To conclude, since the delay time is related to the result of clusters, we can perform spectral cluster to predict the punctuality of each airport especially in the mainland of the US. To quantify the performance of the clusters, we binarized the delay distribution (delay or not delay) and compute the accuracy between ground truth and the result of clusters(only

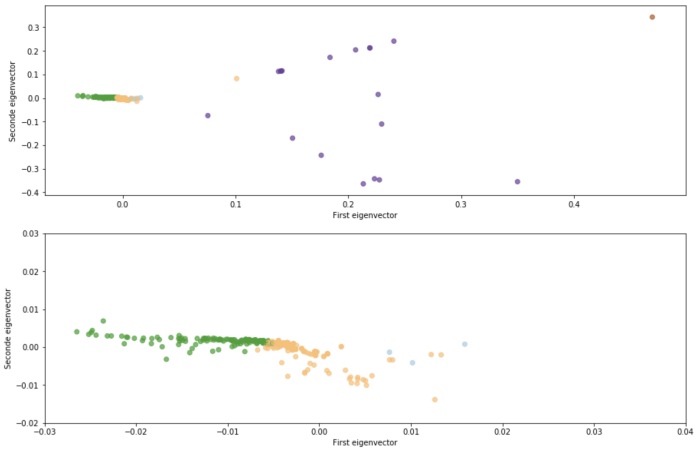


Figure 8: clustering result embedded in the 2-dimension

consider the node in the mainland). The accuracy is 64.3%, which proves we can use the spectral clustering to find the delay pattern. In Alaska and Hawaii outside the mainland, this method might not work. However, from the data analysis for airlines, flight routes in Alaska and Hawaii are more punctual, and there is no need to do prediction.

*2) From Delay Patterns to Graph Partition:* This time, we still embedded our nodes on the 2D plane using first and second eigenvectors as axes, but the label on the nodes are the delay of nodes rather than the result of k-means. For convenience, we binarized the delay, and there are only two conditions which are delay or not delay. We wanted to partition the network using the delay information and graph embedding as shown in figure 9. By zooming in the right part of the 2D embedding, we can find an obvious threshold, which is 0.004 in the first eigenvector axis.

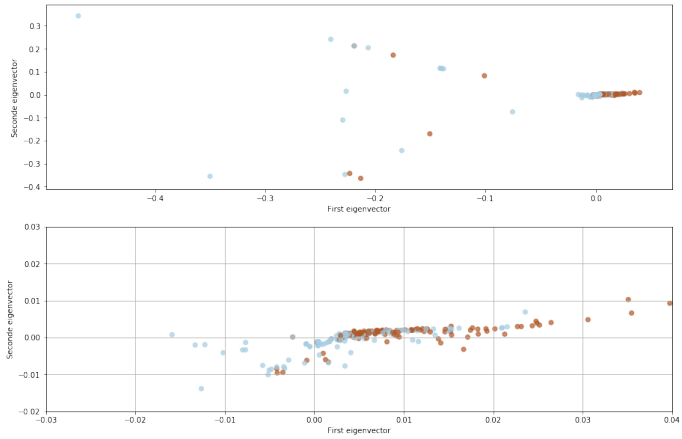


Figure 9: 2D graph embedding with delay label(red: delay, blue: not delay)

After using this threshold to divide the graph into two classes, the result is shown in figure 10 (left). We compared the partition result and the delay ground truth (figure 10 right).

The accuracy is 69%, which proves the delay information can also be used to partition the graph.

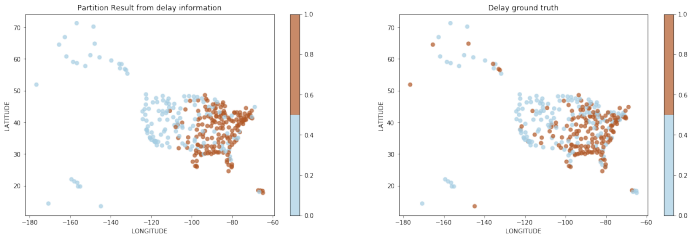


Figure 10: Partition result from delay information(left) and binarized delay ground truth(right)

*3) Transductive Learning:* In this part, we used Transductive Learning to do clustering and gave some sense in recovering the whole world delay signal. In figure 11 (left)we used 30% data to recover the rest part, and the result is shown with accuracy 72.3%. figure 11 (right) is the ground truth. We

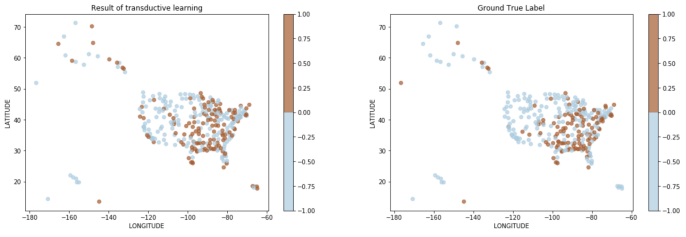


Figure 11: Result of Transductive Learning

can see that using 30% data can recover the US whole data quite well. Then we wanted to know if we can recover the whole world delay data using this method. However, the US airports only take a very small percentage (13.961%) of the whole world, and if we used 13.961% US data to recover, the accuracy was just 56%. So it is unwise to recover whole world data based on our graph (especially when our graph was created only based on airport distance).

### B. Machine Learning Regression

We want to go further on delay analysis, so we decide to build a model to predict the delay in minutes given more data.

*1) Data pre-processing:* There are many significant outliers in our data set. For example, there are 32 flights delay more than 24 hours, which is a very extreme situation. Moreover, these outliers will have side effects on our prediction. Considering we have a huge data set (with samples more than 5,700,000) and the dominant delay time is within 30 minutes, we used flights with the delay shorter than 30 minutes which takes account for about 90% of the whole. Since most of the data set can be seen as that original data are under the influence of Gaussian noise with zero mean and unit variance. It is preferable that the observations of all samples are symmetric around the origin point. So, we also remove the labels that are smaller than -30(negative label means the flight arrived in advance).

`DISTANCE` and `AIR_TIME` are two numeric features in our case(others are categorical), and their magnitudes are much larger than the total types of each categorical features. To avoid their dominance in $L2$-norm penalty, the standard normalization(see Eq.1)is applied.

$$\tilde{z} = \frac{z - \mu}{\sigma} \tag{1}$$

*2) Feature generation:* Flight delay prediction is highly dependant on the time of flight. In the original record, we have the information about the year, month, day, departure time. However, we can reveal more useful information by generating new features. Taking the `DAY` as an example, whether that day is weekday matters. The passenger flow on the weekdays is greatly higher than that on weekends, as there are many business trips. Similarly, the position of the week in a year also contains passenger flow information. People are likely to travel in summer vacation. As the result of increasing passenger flow, the probability of flight delay ascends. Using this analysis, we generated three additional features: `DAY_OF WEEK`, `DAY_OF_YEAR`, `WEEK_OF_YEAR`.

*3) Categorical feature:* Most of our features are categorical, like: `AIRLINE`, `ORIGIN_AIRPORT`. Categorical data can be number(month, day), text(the code of airport) or binary label(delay or not delay). There are two major classes of categorical data, nominal and ordinal. Our categorical features are all nominal. To be specific, there is no concept of algebra meaning among the values of those features. For example, the number 5 stands for Friday and the number 1 stands for Monday. Friday is not larger than Monday, and five Mondays are not equivalent to a Friday. To involve the semantics of categories, we have to implement encoding to them.

The most common encoding method is one-hot encoding. The hot state(value equals to 1) is assigned to the category that the sample belongs. Other categories remain cold state(key equals to 0). See the example in table II. Dummy variable has similar effects. One-hot encoding is easy to implement.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Mon | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Tue | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Wed | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Thu | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Fri | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Sat | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Sun | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Table II: One-hot encoding of weekdays

However, in our case, this method is pretty inefficient. There are 9 categorical features in our data set. Some of them has hundreds of categories, such as: `DAY_OF_YEAR`, `AIRPORT`. It is unwise to encode them to one-hot key, as the total number of dummy features increases to hundreds. For some tree-based algorithm, large amount of dummy features causes the structure of tree skewing to them and increases the layers of tree to achieve good accuracy. To escape this dilemma, we apply optimal splitting over categories that we will discuss more in the section of model selection.

*4) Model selection:* Gradient boosting decision tree(GBDT) is a powerful algorithm that is widely used in regression problem. As its splitting concept, it has reputation for efficiency, accuracy. And, it is an ensemble method, so it is not likely to encounter over-fitting. We chose a optimized gradient boosting decision tree algorithm as our model, which is called LightGBM. It has several additional advantages that are beneficial to our task.

1) **Faster speed** LightGBM splits the continuous values into bins and construct histograms. Compared to other GBDT algorithm(like GradientBoostingRegressor of scikit-learn), the complexity of histogram-based algorithm depends on the number of bins instead of the number of data.
2) **Less memory usage** Storing discrete bins consumes less memory than continuous value. Moreover, as it is based on the histogram, the memory consumption of the list for storing pre-sorted value does not exist.
3) **Optimization for category** Common optimal solution for categorical features is partitioning all categories into two subsets at certain leaves. LightGBM partitions categories based on sorted histograms. This method is friendly to the regression problem and much more efficient than one-hot encoding.

Our data set is large for a personal computer(98683 samples after sampling). At first, we tried to run Random Forest on it. However, it always crashed when using the whole data set, and it took more than 10 minutes on one simple training even though we downsampled the data set. So, LightGBM is a good choice for us. It costs little memory, which means the kernel will not die unexpectedly. Also, we can implement the hyper-parameter grid search on it because it only takes several seconds to fit the data.

*5) Results:* After hyper-parameter tuning, our model can get RMSE (Root Mean Square Error) around 2.12 on train set and RMSE around 2.31 on the test set. According to the experience, our model can work well and meet our expectation. Since the LightGBM can automatically evaluate the importance of each feature through times of splitting over them. We get the figure 12. We can conclude that the travel distance and travel time have a critical impact on the possibility of delay. Meanwhile, the importance of `DAY_OF_YEAR` also proves that our analysis on feature generation is reasonable.

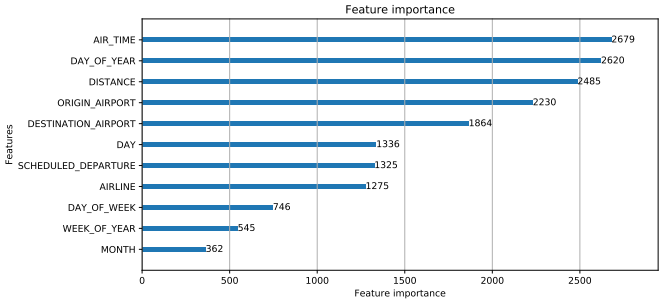


Figure 12: Feature importance

*6) Exploiting our model:* We propose two applications with our model. The first is prediction mode. Users can input details of their flights, and we can predict the delay for given flights. The second is discovery mode. Users provide the date, departure airport, destination airport as necessary information and airlines as optional information. We can suggest the airline and the departure time to prevent delay as much as possible. table III is an example output for discovery mode.

| |
|---|
| 2019-07-21 |
| From: LaGuardia Airport (Marine Air Terminal) |
| To: Chicago O'Hare International Airport |
| We recommend United Air Lines Inc. at 0 o'clock |
| Estimated delay is between -15.00 min and -10.00 min |

Table III: Discovery mode example

## IV. Conclusion

We created our graph using airports as nodes, the distance between airports after Gaussian kernel as weights.

We analyzed delay in two ways: first, we binarized the delay time into 1 (the departure airport tends to delay) or -1 (the departure airport tends to arrive earlier), then we used k means, spectral graph and transductive learning to perform clustering. The accuracy is 64.3%(5 clusters), 69.4%(threshold eigenvalue is 0.004) and 72.3%(recovering from 30% data) respectively.

Then we wanted to use more features (including departure date, hour, airport, destination airport, and distance) than the distance to predict the delay time. We used LightGBM, a tree-based algorithm. After model building and hyperparameter tuning, we got a model with prediction RMSE = 2.3, which worked quite well. Moreover, we also developed a recommendation system. Given departure date and airports, our recommendation system will return the best combination of airline and departure time, and also the predicted delay minutes.

## References


[1] Shuman D I, Narang S K, Frossard P, et al. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains[J]. IEEE Signal Processing Magazine, 2013, 30(3):83-98.

[2] Luxburg U V . A tutorial on spectral clustering[J]. Statistics and Computing, 2007, 17(4):395-416.

[3] LightGBM: A Highly Efficient Gradient Boosting Decision Tree Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu, Neural Information Processing Systems, 2017

[4] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.