饿了么

# Pickup and Delivery Time Prediction


Tim Wu Guangyu, Brittany Miu, Alec Dortkardesler, Xilin Chen, Chiharu Taniguchi


# AGENDA

## 01 Business Insights

A short introduction of our project aimed at predicting Eleme pick-up and delivery times.

## 02 Feature Engineering

New features found from business insights

## 03 EDA & Data Anomalies

A brief description and analysis of the data.

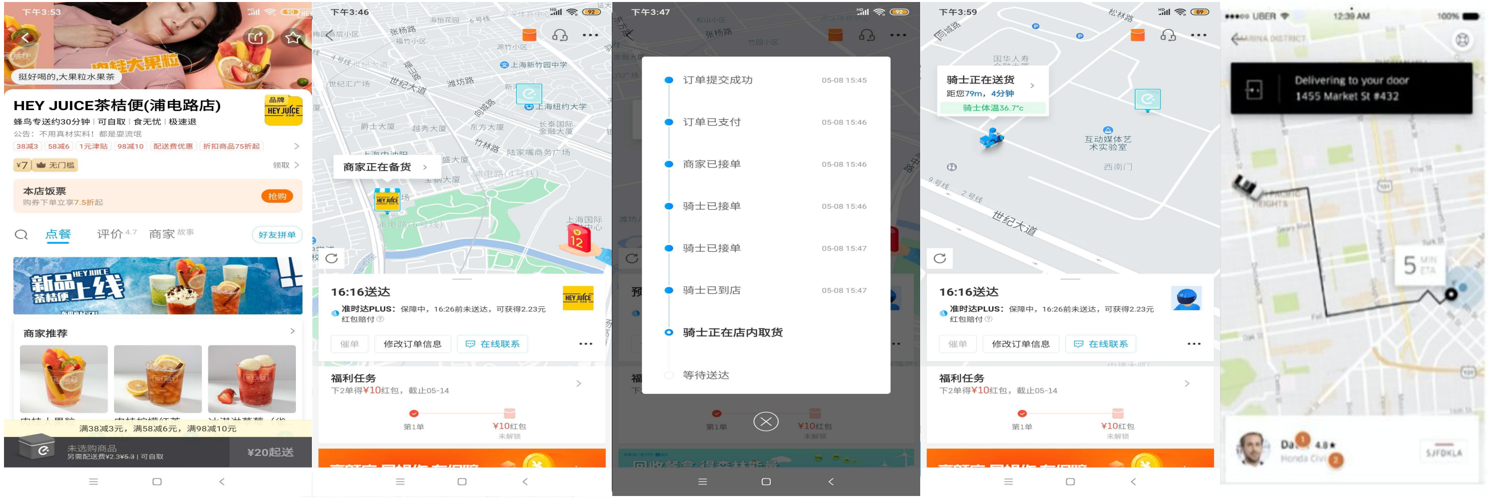
## 04 Route Prediction

Route prediction by using Machine Learning.

## 05 Prediction Models

What type of model did we use to achieve the most optimal results?

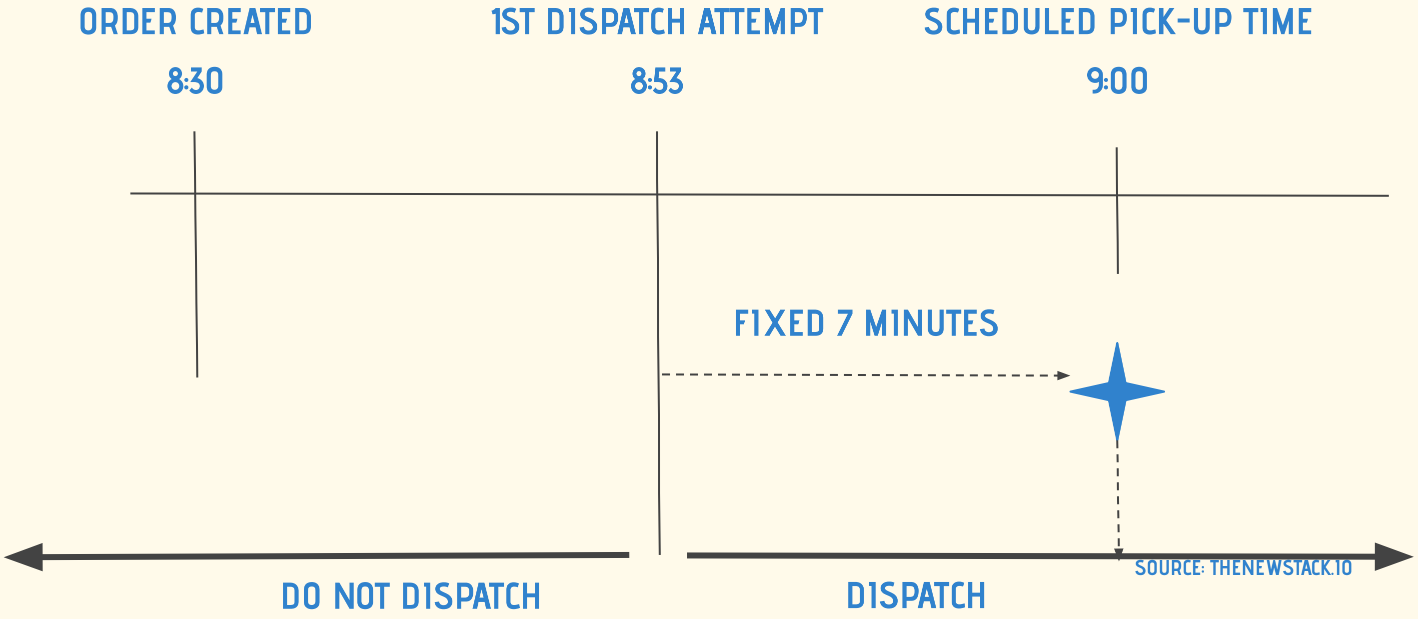
# Business Insights




Eyeball ETD prediction | ETD prediction | Time point tracking | Real-time route tracking | OR | ETA prediction (Uber Eats)

eater browsing → order created → dispatch → delivery-partner arrival → food ready → delivery-partner begins trip → delivery-partner arrival → food dropped-off

# THE PROCESS BEHIND FOOD DELIVERY APPS



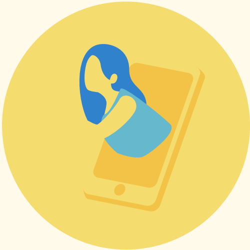
# THE MOST OPTIMAL STATE

## CUSTOMERS

Shortest delivery
time for the eater

## COURIER

Most accurate
arrival time for
pick-up



## VENDORS

Most accurate food
preparation estimate
for the restaurant

# FEATURE ENGINEERING

# New Features Based on *Historical Customer Information*

This is actually inspired by our personal experiences. When a courier arrives within the vicinity of a customer, they might need to walk an extra distance to deliver the food—including taking body temperature check, or finding an alternative entrance because there may only be one entrance open due to the pandemic.

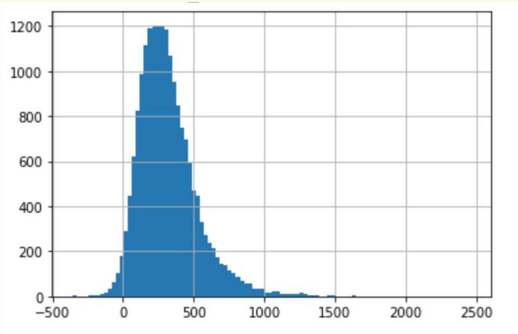
The process:

(1) Identify customers that have ordered at least 3 orders
(2) Subtract the predicted enroute time (distance/speed) from delivery duration
(3) Outcome is an estimated "climbing" time

The distribution of "climbing" time for customers with at least 3 orders is shown on the right.

The median estimate for *est_climbing_time* is **288 seconds**;

- This will be used as an estimate for customers that did not provide enough historical data



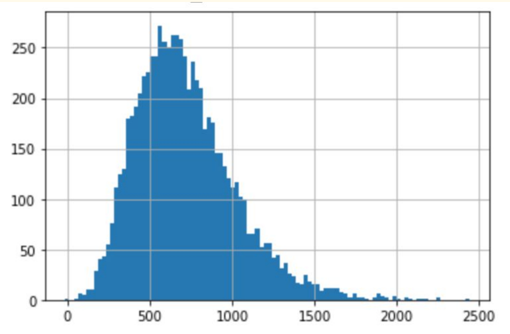
# New Features Based on *Historical Vendor Wait Time*

Many businesses reported that one of the hardest things to predict is **how long the vendor will need to prepare an order**. Given that information, we attempted to estimate the average amount of time that a courier will need to wait at the vendor until the food is ready for pickup.

This is a very rough estimate with a future-peaking problem—but it should be fine if given the real data, which is vast in quantity and longer in time. The process went as follows:

(1) Identify vendors that have prepared at least 5 orders (for robustness)
(2) Subtract the predicted enroute time (distance/speed) from the pickup duration
(3) Outcome is an estimated wait time for the vendor

The distribution of wait time for vendors with at least 5 orders is shown on the right.




The median estimate for *est_wait_time* is **676 seconds**;

- This will be used as a robust substitution for vendors that did not provide enough historical data

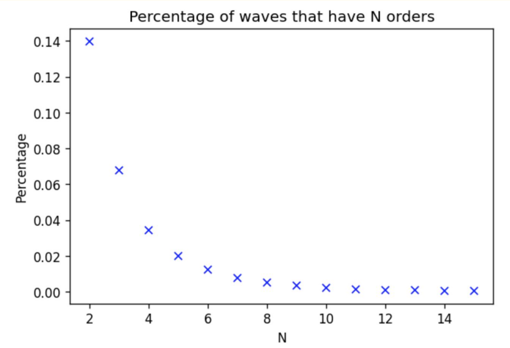
# New Features Based on *Courier Information*

Initially, we created a feature to measure the number of orders that the platform can handle in an hour. We can get, however, a more granular measure of how many orders that a courier is handling at a given time. (in an hour or in a wave)

This can potentially provide insights into how busy the courier is. **The less busy they are, the more timely their deliveries will be**. Furthermore, it leads us to do an additional prediction: **courier route in a wave.**

We can see that all couriers received at least 2 orders in a given wave, with many of them actually receiving more than that (some even had 15 orders in a wave).

If we can predict the route taken by a courier, we can get more accurate distance features that, in turn, contribute to the overall time prediction.



# EDA &
# Data Anomalies

# GENERAL DATA AND ORDERS DISTRIBUTION

### WEATHER GRADE

0 - Good
1 - Mildly Bad
2 - Bad
3 - Extremely Bad

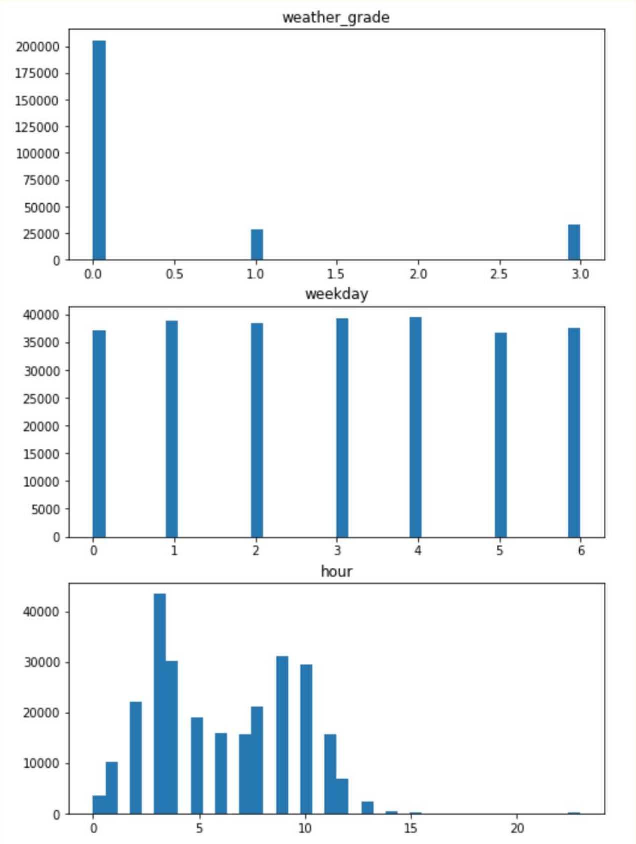
For the month of February, most days experienced good weather—though if not, it may potentially be extremely bad weather.

### WEEKDAY

We initially expected a discrepancy between the amount of orders placed on weekdays versus weekends. Given the data, however, the number of orders are more or less even across a week—potentially due to the pandemic lockdown,

### HOUR

The number of orders every hour shows a two-modal distribution, with both spikes taking place in the morning. This may be a sole result of the partial data provided for a given day, where most entries are happening in the morning.



# TRENDS BY DATE

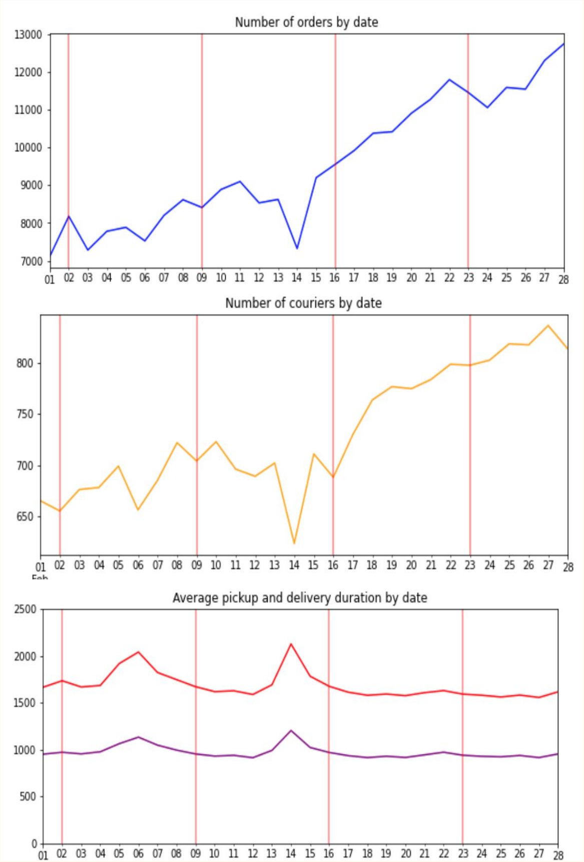
*What about the trends found across dates?*

By plotting the number of orders as a month-wise time series, we can clearly observe:

1. An upward trend (more people ordering!)
2. More couriers delivering orders toward the end of the month

The average pick-up and delivery times seem to be jointly decided by the number of orders and couriers working that day. An interesting note:

- On the 14th, the number of couriers dropped more drastically than orders, and the duration indicators spiked. We assume there is an non-repeat event factor *(e.g. the couriers are delivering flowers instead of food orders during the valentine's day)*



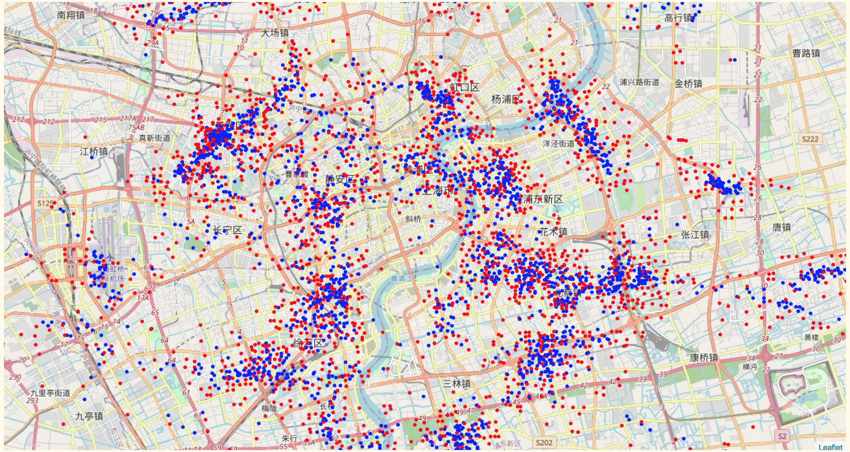
# GEOGRAPHICAL DATA

*Visualizing Pick-Up Delivery Locations*

Since talking about distance is abstract, we plotted the locations of order pick-up and delivery locations in an effort to get a better sense of the current situation.

The distribution shows clear clusters; an expected observation considering food delivery rarely extends beyond a 3 km radius of the vendor. The shape of the clusters are oval, which might be centered at malls or plazas located on main streets.

The original geolocation coordinates lead us to the sea near Dalian, which is impossible. After shifting the coordinates, we still struggled with determining which city the data captures—but it is likely a city with a similar urban size to Shanghai.



# DISTANCES

### PICKUP DIST

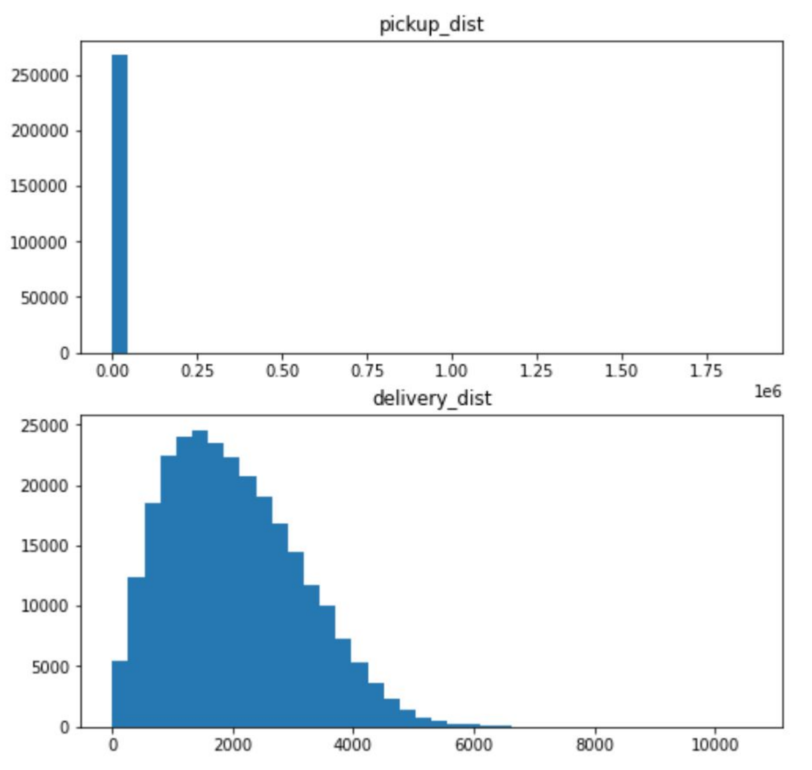
Distance from Point 1, where the courier is when they are assigned an order, to Point 2, where the order will be picked-up (vendor).

### DELIVERY DIST

Distance from Point 2, where the order will be picked-up (vendor), to Point 3, where the order will be delivered (customer).

These two features are extracted from the distance data, order by order. As a result, we can observe a very clear anomaly:

- *Pickup_Dist* has an extremely high value at 1e6 (see the graph of the diagram of pickup_dist) magnitude (1000 km)—this is unreasonable and needs to be cleaned.
- *Delivery_Dist* follows a skewed but reasonable distribution, with a max distance around 10 km.



# PICKUP & DELIVERY TIMES

## ESTIMATE PICKUP TIME

Time is displayed in terms of duration, the difference between the estimated timestamp and the assigned-time timestamp.

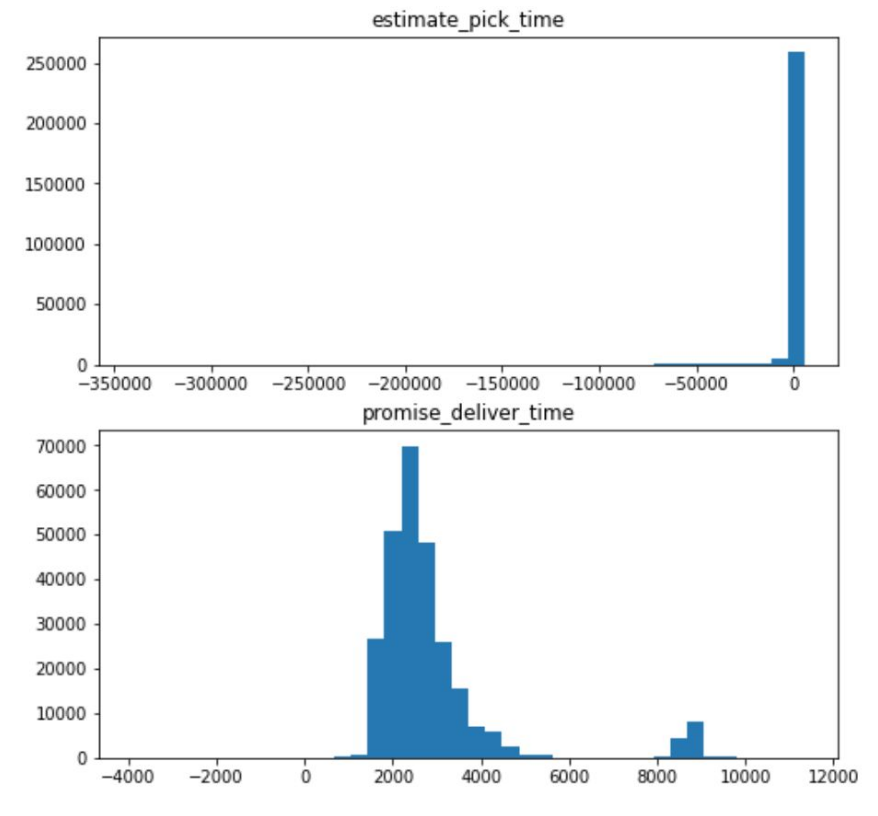
A negative value means the estimated pickup time is before the assigned time. How can an order be picked-up before it is assigned?

Further analysis shows that some orders are pre-placed, i.e., placing an order for breakfast the night before. The platform, however, still estimates a time close to the time when the order is created—thus, estimated pickup time is no longer valuable to our predictions.

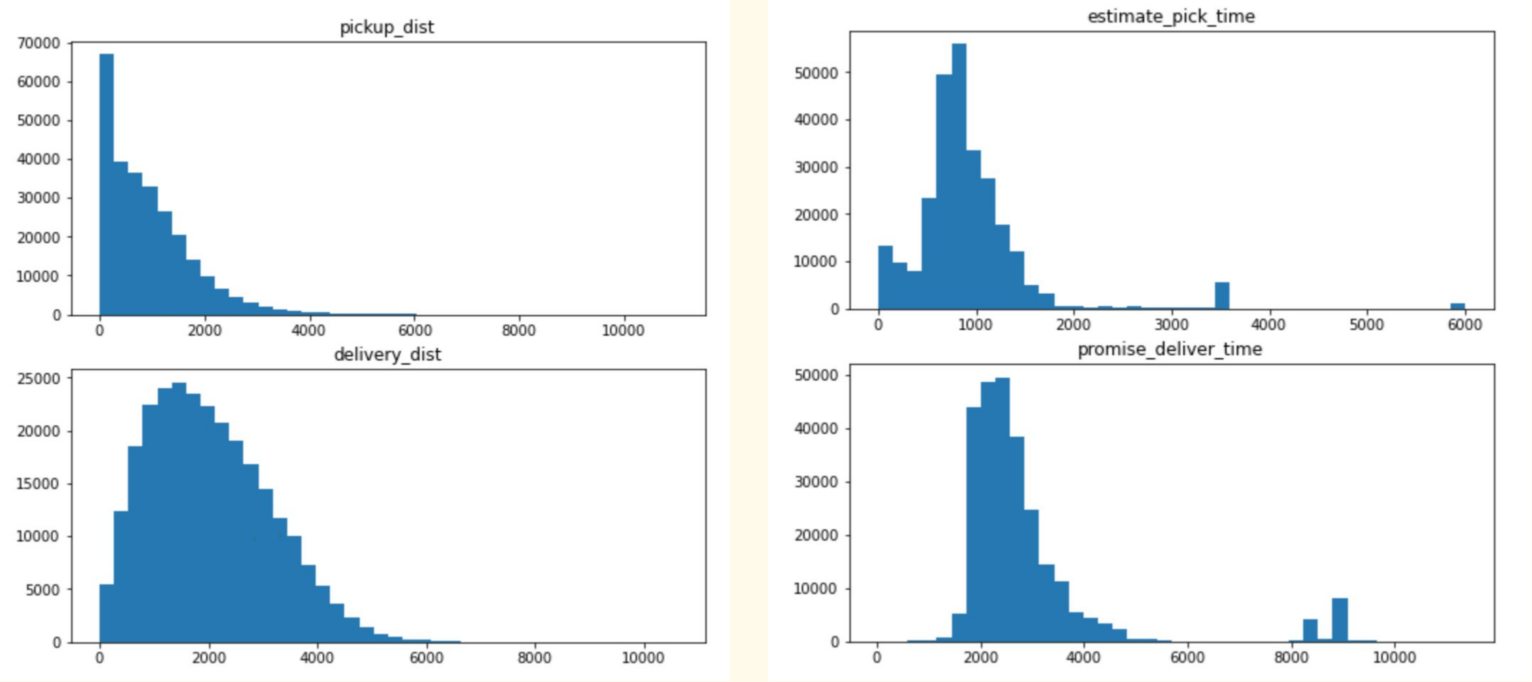
- This needs to be cleaned

## PROMISE DELIVER TIME

The duration format is similar to the one mentioned above. Most orders are promised to arrive within an hour, with a few exceptions arriving after two or three hours (presumably longer distances). We have a small number of negative duration problem cases as well.



# CLEANING THE DATA

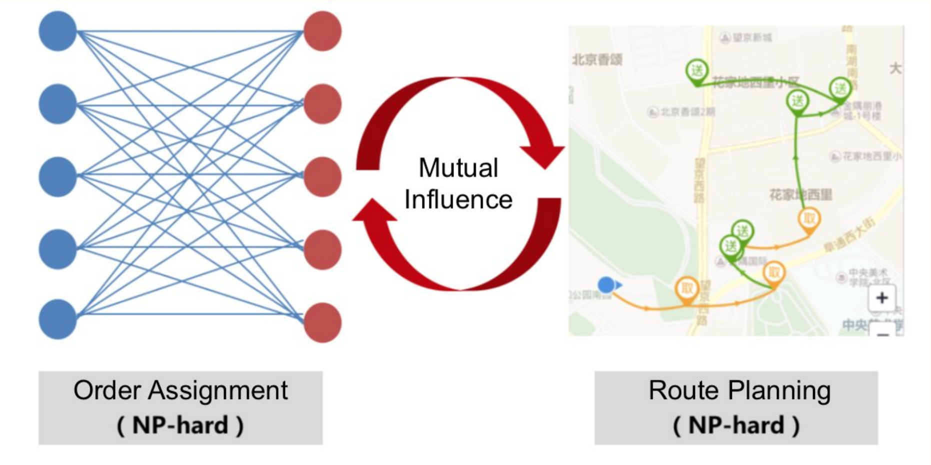


# Route Prediction

# Route Prediction Problem

For the platforms, order assignment and route planning are two of most difficult problems being faced.

These problems interfere directly with each others and are NP-hard by themselves. Luckily, in our case, we will not need to worry about the assignment process as order information is given. All we care about in regard to our time prediction problem is the discovery of the actual route taken by the courier.

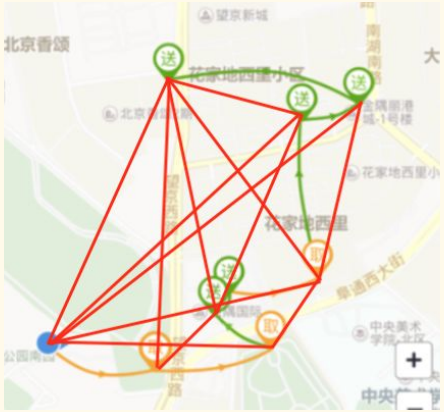



Q., Ru. “AI-application in Delivery Time Prediction.” *InfoQ*. (Translated)

# Route Prediction Problem

*X* is comprised of three components:  distance matrix, time features, and environmental features.

**Distance matrix** is the representation of distances between all sources and target locations. If we draw every possible link between every pair of locations given in a wave, the length of all these links can be represented orderly in a matrix, providing us with all of the distance information.



**Time features** include the estimated pickup times and promised delivery times that the platform generated. This is helpful in capturing the decision to prioritize an urgent order, being given an upcoming promised delivery time.

**Environmental features** include weather, weekday, order amount across the system and, for this courier, their speed and level. This is helpful for customizing the prediction for a specific courier in a specific scenario.

← Distance Matrix: Native Pairwise Distance

# Route Prediction Problem

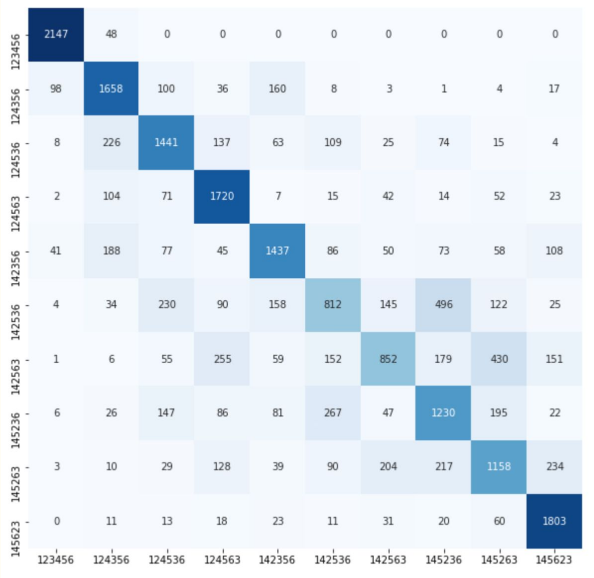
For starter, we consider the **two-order scenario**, where a courier is assigned two orders in a wave.

Given the inputs (*X*), there are ten possible routes. If we denote the assign-pickup-deliver locations of Order 1 as 1/2/3, and of Order 2 as 4/5/6, then we have the following potential routes:

- 123456; finish first order and *then* start second order
- 124536; courier is assigned to second order and picks it up before delivering first order
    - Note: Sequence 213654 is impossible since the assign-pickup-deliver sequence must stay true

This way, the output space has only ten classes, instead of the full permutation 6!=120. *X* and *y* are created from historical data and then trained with the **XGBoost classifier**.

The resulting confusion matrix is on the right, with the x-axis as prediction and the y-axis as actual. The model is predicting well.




**Confusion Matrix of Prediction**

# PREDICTION MODELS

# Note on Preprocessing

The features that we finally selected for the prediction models are:

- Order specific: 'create_duration', 'confirm_duration'
- Environmental: 'weather_grade', 'weekday', 'hour',
- System-wide pressure: 'order_count_this_hour', 'order_count_this_wave',
- Courier pressure: 'order_count_this_courier_this_hour', 'order_count_this_courier_this_wave',
- Courier characteristics: 'level', 'speed', 'max_load'
- Platform estimation: 'estimate_pick_time', 'promise_deliver_time'
- Vendor and customer historical info: 'est_wait_time', 'est_climbing_time'
- Distance features: 'naive_pickup_dist', 'naive_delivery_dist', 'seq_pickup_dist', 'seq_delivery_dist'

**Categorical features** are **one-hot encoded** (dropped one to avoid multicollinearity)

**Numerical features** are transformed through **MinMaxScaler** to normalize.

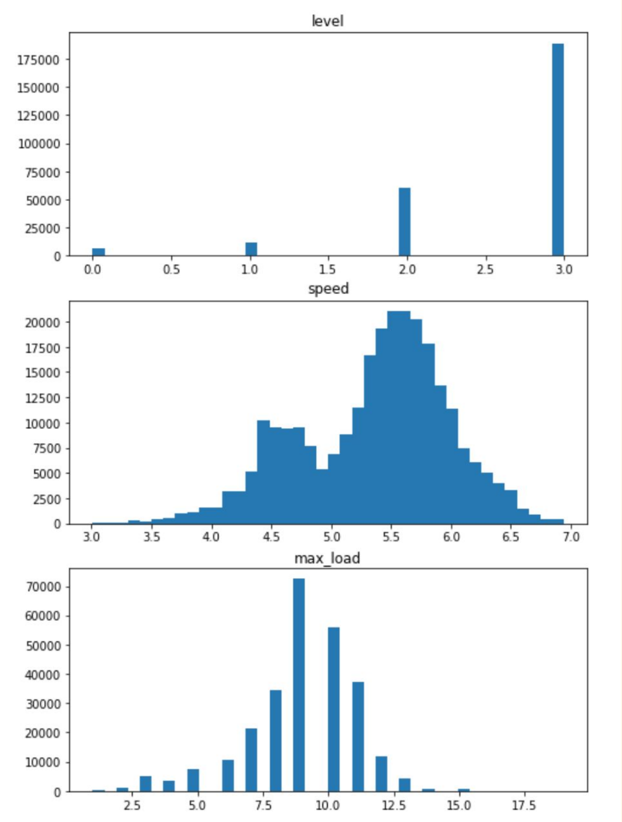
Note that the GBM and XGB models are immune to scale differences in variables — however, proper scaling will help the models train faster.

# COURIER RELATED DATA

### LEVEL
A majority of the couriers are rated three-stars.

### SPEED *(bimodal distribution)*
From the data, we assume this is a measure of the average moving speed of a courier on a given day. The data suggest some couriers did move faster than others on average. The distribution shows two modes around 4.5 and 5.5 respectively, does this imply use of different transportation vehicles? (Scooter v.s. Motorcycle)

### MAX LOAD
We assume that this is the maximum number of orders that a courier can handle at once in one way. The majority handle 8 orders, but some can handle as many as 17 orders. Kudos to them.

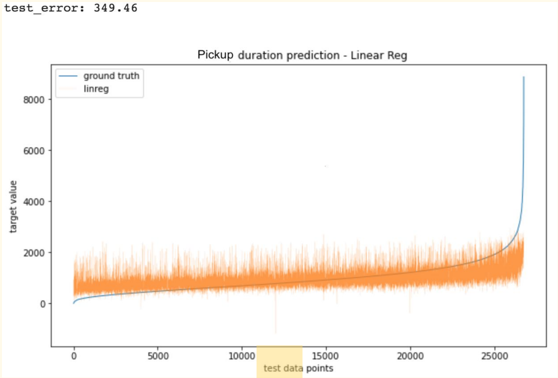


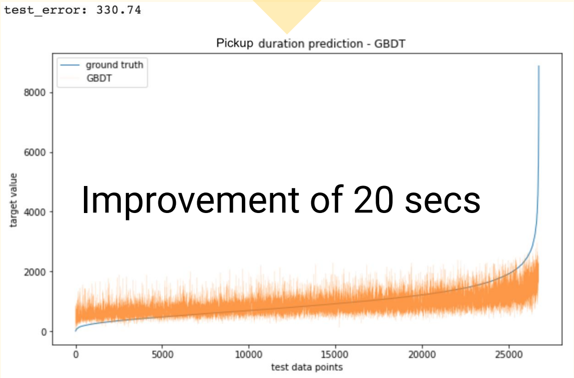
# Pickup Time Prediction: LinReg vs. GBM

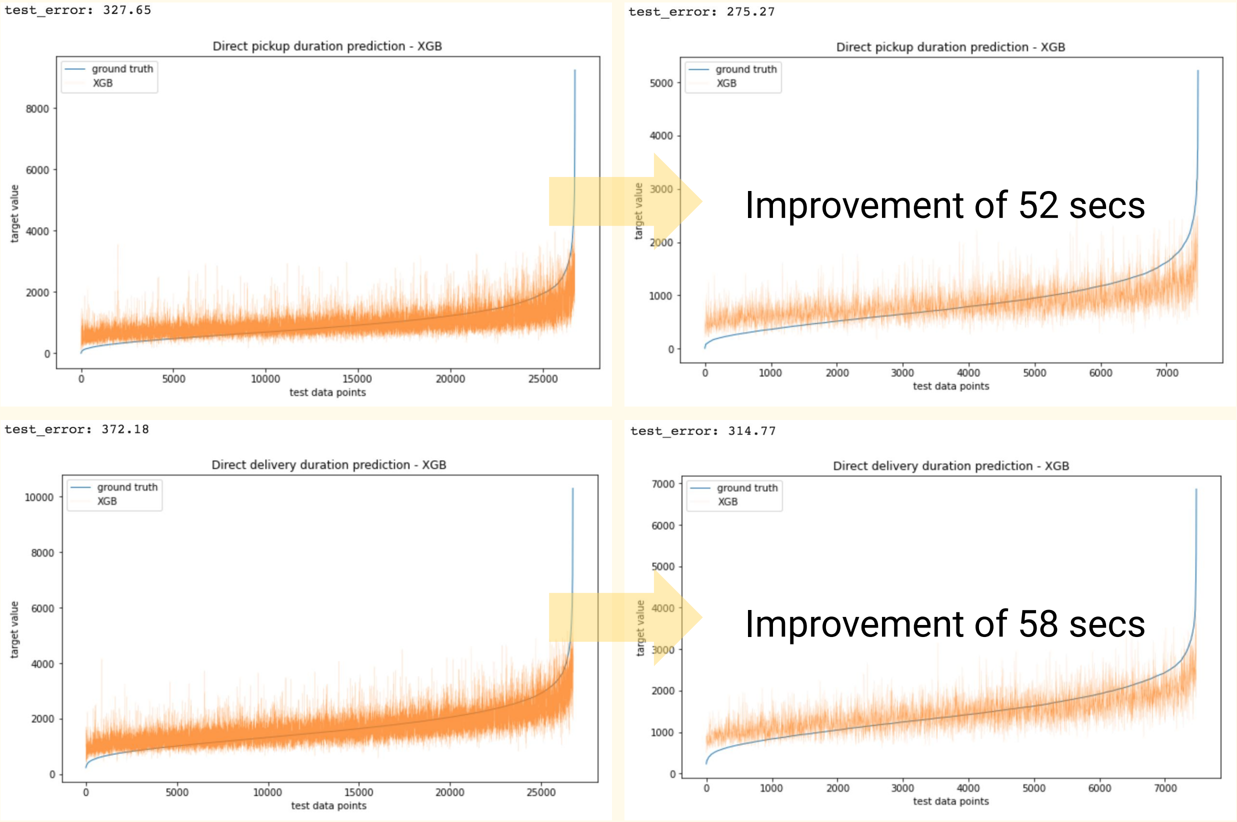
*Visualization of Prediction Results* — The test data is re-ordered by actual time duration, having the shortest actual pickup duration starting on the left, gradually going to the longest duration on the right.

Semi-opaque orange lines are the predicted durations.

We can see that GBDT achieves MAE = 330, which is 20 seconds better than the linear regression model.






# XGBoost vs. XGBoost + Route Prediction

Remember for the two-order scenarios, we can predict the route taken by the courier. If we only look at those samples, the MAE measure **improves by almost a minute**, down to 4 minutes error for pickup and 5 minutes for delivery.

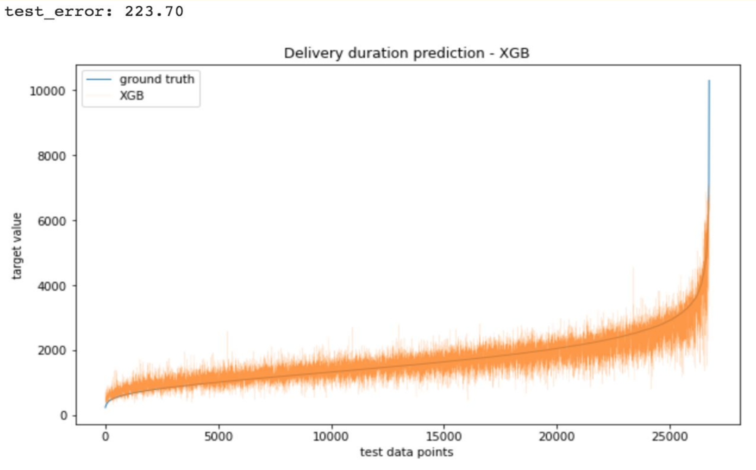
This shows **the value of route prediction**. If we are given more data and thus able to predict the route for three or more orders, the overall prediction accuracy will most likely improve a lot.

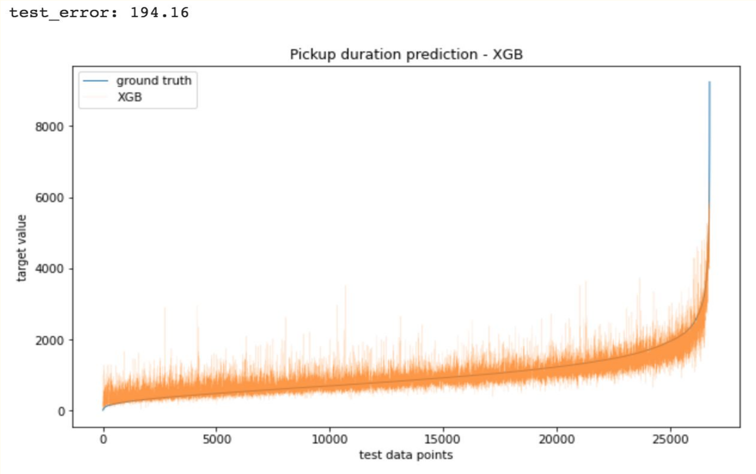
# Predict Delivery (Pickup) Given Pickup (Delivery)

If we are given at least one timestamp for an order, we can significantly improve the prediction.

Below are the results for delivery given pickup (left) and pickup given delivery (right).

- MAE errors around **3.5 minutes**



# Summary of the Solution Pipeline

## Data Cleaning

Clean abnormal data entries (super-long distances and significant time deviations), by either capping them or replacing them with a reasonable estimation.

## Feature Engineering

1. Order specific
2. Environmental
3. System pressure
4. Courier pressure
5. Courier characteristics
6. Platform estimation
7. Vendor wait time
8. Customer climb time

## Route Prediction

Use distance matrix and environmental features as *X*, route sequence as *y*. Train a ML classifier to predict route taken by a courier, and use predicted route to estimate travel distance better.

## XGBoost Training

Train XGBoost regressors with small learning rate and many shallow weak-learners. Tune the hyperparameters so that the regressor is both accurate and efficient.

## Customized Prediction

Based on the missing values in the test data, devise three predictive models to handle different scenarios. Minimize overall prediction errors by utilizing all given information.

# Thanks!

Do you have any questions?