Predict Delivery Time

Ketong Lin

1 Main results

The goal of our model is to predict the delivery time. First, we need to construct a new column "duration", which is the difference in time (seconds) between "created_at" and "actual_delivery_time", i.e.

$$\text{duration} = \text{actual\_delivery\_time} - \text{created\_at}$$

Throughout this report, we use "duration" as our output.

The final prediction for the test data is saved in the file "submission.csv". The two newly appended columns "predicted_duration" and "predicted_delivery_time" are the predicted results.

- "predicted_duration" has the same meaning with the variable "duration" described above which represents how many seconds it takes to deliver the order since it's created.
- "predicted_delivery_time" is the date and time when the order is predicted to be delivered.

1.1 Observation from the data

- **Observation 1**: *There are significant outliers in "duration", which may not fit for training purpose.*

  In some cases, the "duration" time can be as long as 2000 hours, which equals to several months. We take a closer look like at the top 3 cases in Table 1.1. In the top case, the order was created on 2014-10-19, but delivered on 2015-01-29. If this is not a system error, the delivery time must be required by the customer. Otherwise, no one would wait for that long for a delivery. This is not something that we want to predict using any model. So we need to exclude these extreme cases. We choose 4 hours as the cutoff. People usually won't wait for more than 4 hours without canceling the order unless that's the time they scheduled on purpose.

- **Observation 2**: *The distribution of "estimated_order_place_duration" is a bi-mode distribution (Figure 1.1).*

  This observation implies that the estimation of order_place_duration may be a little too coarse. It's more like a binary distribution which should not be the case in real life. And from the feature importance analysis in the next section we can see that "estimated_order_place_duration" doesn't play an important role as expected. We may need other information to further improve this part of duration. We leave the discussion in Section 2.

- **Observation 3**: *There are unexpected negative values in some of the numerical features.*

  Based on the description of data, all the features should have positive value. However, we found there are negative values in the following features:

created_at	actual_delivery_time	total_items	subtotal	duration(s)
2014-10-19 05:24:15	2015-01-25 19:11:54	1	1695	8516859.0
2015-01-28 08:34:06	2015-02-01 16:25:25	3	1520	373879.0
2015-02-16 02:24:09	2015-02-19 22:45:31	4	4980	332482.0

Table 1.1: Outliers of "duration"

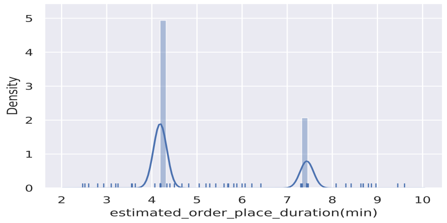


Figure 1.1: Distribution of estimated_order_place_duration

Model	Elastic Net	Random Forest	Gradient Boost	XGBoost
MAE(s)	687.31	652.23	620.37	615.02

Table 1.2: Model comparison in MAE

- “min_item_price” has 13 negative values in the historical data and 4 negative values in the test data.
- “total_onshift_dashers” has 21 negative values in the historical data and 2 negative values in the test data.
- “total_busy_dashers” has 21 negative values in the historical data and 10 negative values in the test data.
- “total_outstanding_orders” has 44 negative values in the historical data and 13 negative values in the test data.

Our first guess is that those negative values are errors. However, since the negative values also show up in the test data, we cannot simply remove those rows. So we choose to keep it the way it is. But those features may require a closer look in the future.

1.2 Results from the modeling

- **Observation 4**: *XGBoost performs the best amongst all the four model classes we considered, which are Elastic Net Linear Regression, Random Forest XGBoost and Gradient Boosting.*

  The metric of performance we choose is *mean absolute error(MAE),* which is the absolute difference between predicted “duration” and observed “duration”. The value of MAE is easier to interpret since it has the same unit as “duration”. The results are in Table 1.2.

  - The performances of Gradient Boost and XGBoost are very close. We choose XGBoost because of its time efficiency.
  - In the Elastic Net Linear Regression model, we exclude the categorical features, including “order_protocol”, “store_id”, “market id” and “store_primary_category”.

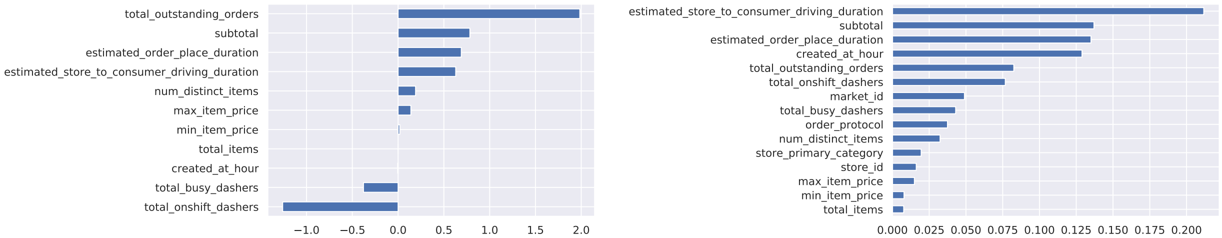
- **Observation 5**: *The feature importance in XGBoost model is shown in Figure 1.2b, the main important features are “estimated_store_to_consumer*

  The importance of “estimated_store_to_consumer_driving_duration” and “estimated_order_place_duration” is not surprising, since those two periods of time accounts for the majority of the “duration” of delivery. The second important feature is more interesting, which is “subtotal” that represents the “total value of the order submitted”. It’s more important than “total_items” which is the least important feature. This leads to the Observation 6

- **Observation 6**: *The “total value of the order” has more predictability on the delivery time than the the “total number of items”.*

- **Observation 7**: *The time of order “created_at” make a difference on the delivery time.*

  We extract the “hour” of the order created time from the feature “created_at”. The intuition is that the same order placed at midnight will likely take longer to be delivered than that at a regular lunchtime or dinner time. The feature importance proves that.



(a) Linear model coefficients

(b) Feature importances from XGBoost

Figure 1.2: Feature Importances

- **Observation 8**. *The Elastic Net Linear Regression model shows that the most significant features are "total_onshift_dashers", "total_busy_dashers" and " total_outstanding_orders".*
  - *The "duration" of delivery increases along with "total_outstanding_orders" and decreases along with "total_onshift_dashers" and "total_busy_dashers".*

Unlike XGBoost model, Linear regression doesn't emphasize "estimated_store_to_consumer_driving_duration" that much. We also notice that the three variables, *"total_onshift_dashers", "total_busy_dashers", and" total_outstanding_orders" are highly correlated with each other. The correlations are more than 0.9. So we use Elastic Net to deal with the multicollinearity issue.*

2 Potential relevant features

1. Back_log_orders: how many orders are waiting in line before this order. Inspired by Observation 2, we need a feature to improve the estimation of order place duration. The number of "back_log_orders" represents your position in the line, which should have an impact on the order processing time.
2. Dasher's id: the identification of the assigned dasher. The delivery time depends on the dasher's personal driving habits. Maybe some dashers always drive slower than the estimated driving time.
3. Scheduled_delivery: a boolean variable. *True* means the delivery time of the order is scheduled by the customer in advance. This could exclude the cases that are not for training purpose.

3 Model Assessment

If there is a model already in production. We can follow the steps below to assess the the new model's performance:

1. Data validation: the training data set should adequately represent the data distribution that currently appears in business.
2. Model quality validation: using cross-validation to compare the performance between the new and old model and test if the new model is overfitting. In this step, we could customize the loss function to better accommodate our purpose. Since the orders that are very early/late are also much worse than those that are only slightly early/late, we can use an asymmetric loss function to train the model.
3. Test for robustness: once the model validation is completed, we should perform tests to see if the new model can handle specific edge cases, for instance, null values or extreme values in features. In some special cases, a ridiculous prediction may have very bad influence on the business.
4. Test for response time: changing the model could impact the model's operational performance. Therefore, a stress test is needed to measure the operational efficiency, such as average response time.