Predicting Delivery Days for e-Commerce Purchases

Eshwar Nag Pilli, Aishwarya Sharma

December 15, 2021

Abstract

With the E-Commerce business growing more than ever and the increasing number of options available for the users, it has become increasingly crucial for the businesses to enhance the user experience and fast delivery of the shopped items, which requires effort. Thus, leveraging the power of machine learning can help businesses predict the accurate delivery dates, not only for user notification but also help manage and plan the delivery cycle. This project outlines the methodologies used for this purpose and the outcomes.

1 Introduction

1.1 Motivation

E-commerce businesses have seen a boom in internet traffic in the past few years. With the availability of all our favorite products at our fingertips, it is convenient and time-saving. Thus E-commerce websites invest a considerable amount in sales and marketing strategies to enhance the user experience, such as personalized product recommendations, faster checkouts, one-click payments, etc. An essential aspect of such businesses is delivering the ordered items from the warehouse to the buyer.

The accuracy of shipping estimates plays a significant role in providing a hassle-free and trusty customer experience. There is still research in this area, given the growing importance. Thus, we decided to leverage the power of machine learning methodologies to help businesses estimate accurate delivery times, which can also be used in decision-making.

1.2 Challenges

Delivering packages from seller to buyer is made up of two steps: 1) Handling time: taken by the seller to package the item until it is handed over to the carrier. 2) Transit time: time taken by the carrier to deliver the package.

Many factors affect the delivery time in this journey, such as the distance, mode of transportation, product handling, and other technical and logistic factors. It has become even more challenging during the COVID-19 period, as the changing shipping restrictions have affected the delivery period.

The main challenge in predicting accurate delivery dates is identifying factors that can influence the time to deliver the products and their relationships. Next is to identify the machine learning algorithm that accurately fits the data.

From the business point of view, it is critical. Predicting a delivery date later than the actual delivery date is worse than the product arriving before the estimated delivery date. Thus, the algorithm should be penalized more in cases where it predicts the delivery date to be less than the actual date.

1.3 Solutions and Results

We used Multivariate Linear Regression, Multi-Layer Perceptron, Random forest, and Ensemble learning methods to solve the defined problem. The algorithms were implemented using the SkLearn machine learning library.

After training the models using the algorithms above, we conclude that the ensemble methods were able to have a minimum value for loss and a higher R-Squared score, which makes it better than the other algorithms. The results are mentioned in a detailed way in the following sections.

2 Problem Setup

Given the dataset of 20 million randomly selected shipments from transactions on eBay, we want to predict the delivery date of the items or packages sold by businesses and individual customers. The dataset contains business to customer (B2C) and customer to customer (C2C) training examples. The delivery date is estimated as the total number of calendar days to reach the buyer after successful payment. It is calculated by

$$\text{delivery date} = \text{payment datetime} + \text{delivery days}$$

As discussed in the previous section, we penalize the model for estimating an early shipment date when the actual order arrives late. The loss function is given by:

$$L = \frac{1}{N} \cdot abs([P_E \cdot \sum_{\text{early shipments}} (\text{actual delivery days - predicted delivery days}) + P_L \cdot \sum_{\text{late shipments}} (\text{predicted delivery days - actual delivery days})]) \quad (1)$$

where $P_E = 0.4$ and $P_L = 0.6$

An ideal model would be the one that can accurately predict the delivery dates given a test example. However, such a model is unrealistic, and prediction errors are bound to exist. Thus, the best model should give lower values of L.

The dataset provided contains certain irregularities, such as missing values, invalid values, inconsistent formats for zip code, inconsistencies in date like a delivery date before the payment date. It is crucial to address the irregularities before training the machine learning model with examples. Such irregularities may lead to a decrease in the model performance.

3 Solution Approach

3.1 The problem

With many options available to buy the products, the user holds power. Thus E-commerce businesses invest a considerable portion of time and money in enhancing the user experience. After successful checkout, the user is given an estimate of when they can expect the product to reach their address. It is a bad user experience when the arrival is delayed, i.e., the product arrives later than what was earlier notified. Thus we need an accurate estimate of delivery time.

What if we can utilize the enormous data set of these businesses for the shipments and the power of machine learning algorithms to predict the accurate estimate of the delivery for a particular order. We also have to consider that there can be several factors affecting the time it takes for a product to travel from the warehouse to the user, so there will be some loss, and our goal will be to minimize this value.

3.2 Research and Survey carried out

Before approaching the problem, we **researched** how to tackle the prediction of delivery dates using machine learning.

1. "Predicting Shipping Time with Machine Learning," a project presented by MIT (2012) where they predicted the delivery time of freights for Maersk (A shipping company in Denmark). They found that Random Forest worked surprisingly better than neural networks regarding prediction time.

2. Purdue University had written an article on estimating delivery time for industrial pieces of equipment using machine learning approaches. They found that Random Forest and SVM Ensemble models performed better with low Root Mean Squared Errors (RMSE).

3.3 Data Cleaning and Feature Engineering

Before applying any machine learning model, we did some **cleaning** to remove null values, label encoding, and calculated the total estimated time by adding up two feature columns and subtracting them with the payment date and time.

We converted the features represented as strings (i.e. "*b2c_c2c*", *package_size*) to discrete numeric encodings.

Also, we did some feature engineering where we had the buyer and seller's zip codes and calculated the zip distance. We then mapped the latitude and longitude and calculated them using Haversine distance. The Haversine formula calculates the shortest distance between two points on a sphere using their latitudes and longitudes measured along the surface by considering the radius of the Earth.

Also, the time of order or shipping has to be considered because if the order is after 5:00 P.M., we should take the next business day for further processing. This information was also utilized while pre-processing the data.

3.4 The approach

We tried to approach this as a regression problem with more than one variable that helps in determining the delivery dates. So we applied Multi-variate Linear Regression and MLP Regressor. Multi-Layer Perceptron is a basic Artificial Neural Network model present in the sci-kit-learn library.

Some of the features of this algorithm are:

1. It does not have an activation function in the output layer.
2. Squared error is used as a loss function in the regressor models, whereas cross-entropy is used in classification models.

We have used this model to learn about the basic deep learning models implemented from the sci-kit-learn library. We observed that the model's accuracy is almost the same as the accuracy of the multivariate linear regression model. *concluded that could improve the accuracy further by choosing other algorithms.*

To further improve the prediction model, we tried **XGBoost Ensemble Algorithm**.
It stands for Extreme Gradient Boosting and is an ensemble algorithm that is used to boost the decision trees for higher speed and performance/accuracy.
We used the basic gradient boosting algorithm to provide better results. The main algorithmic features that made us select this model are:

1. Sparse Aware implementation with automatic handling of missing data values.
2. Block Structure to support the parallelization of tree construction.
3. Continued Training to further boost an already fitted model on new data.

The drawback of using this algorithm is that it is sensitive to outliers. The outliers affect the performance of the model. For this reason, we try to search for a new, better algorithm.

Finally, as the MIT research paper suggested, we implemented the random forest regressor to our dataset to reduce the loss.

1. This model solves the problem because it is less prone to overfitting than Decision Tree and other algorithms.
2. Random Forest algorithm outputs the practical importance of features.

4 Experiments and Results

4.1 Data Description

The dataset consists of 2 million randomly selected shipments from transactions on eBay with 19 features.

- **b2c_c2c** - It of type string with values b2c or c2c. This explains if the transaction is between business to customers or between customers and other customers.
- **seller_id** - This is a unique ID given to each seller for identification. It is of type Long Int.
- **declared_handling_days** - The number of days taken by the seller to ship the carrier from the day of acceptance.
- **acceptance_scan_timestamp** - The date and time when the carrier has accepted the package for the final shipment. The values in this are of type timestamp.
- **shipment_method_id** - The integer type attribute defines the type of shipping service declared by the seller.
- **shipping_fee** - Transportation and handling charges charged by the seller for shipping the items. All the values are in USD.
- **carrier_min_estimate** - The minimum estimate of the number of required days by the carrier for the specified service.
- **carrier_max_estimate** - The maximum estimate of the number of required days by the carrier for the specified service.
- **item_zip** - The US Postal zip code of the package origin/source.
- **buyer_zip** - The US Postal zip code of the package destination.
- **category_id** - An integer type data attribute that categorizes the package by its type.
- **item_price** - The price per item involved in the transaction.
- **quantity** - Number of items involved in the transaction.
- **payment_datetime** - A timestamp attribute that clocks in the time when the payment has been made for the particular transaction.
- **delivery_date** - The actual delivery date of the package. This is the attribute that we need to predict using the other attributes.
- **weight** - A scalar value that determines the sum of the weight of all quantities involved in the transaction.
- **weight_units** - It defines the weight scalar value by telling if the measurements are in kilograms or pounds. Pounds are represented as one, and Kilograms are denoted by 2.
- **package_size** - It is a categorical value which categorizes the packages based on its sizes.
- **record_number** - Unique integer number given to the transaction to identify them.

4.2 Performance metric

We have used *weighted average absolute error* as our evaluation metric to calculate the loss. We also use the loss function defined in section 2 to compute the loss for the applied algorithm. The process penalizes our model with 0.4 for every early delivery and 0.6 for late delivery— the lesser the loss, the better the model. We set a baseline score for L as 0.75 and aimed at having the value below it.

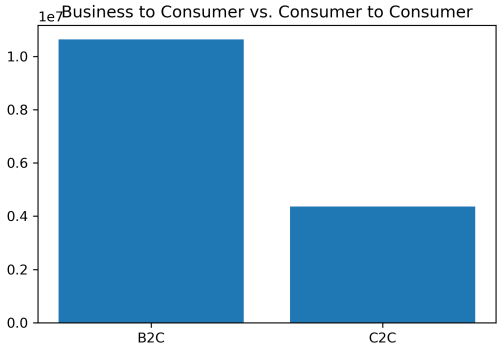

Figure 1: Number of B2C and C2C

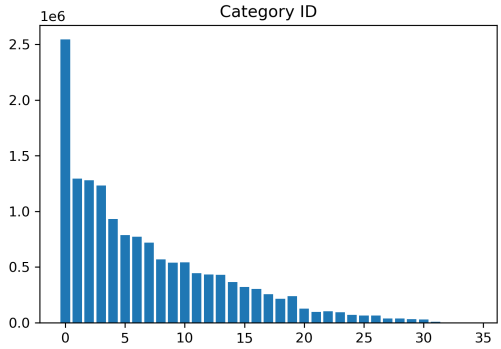

Figure 2: Count per Category Id

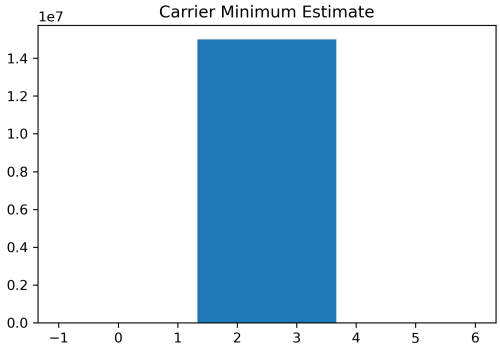

Figure 3: Carrier Minimum Estimate

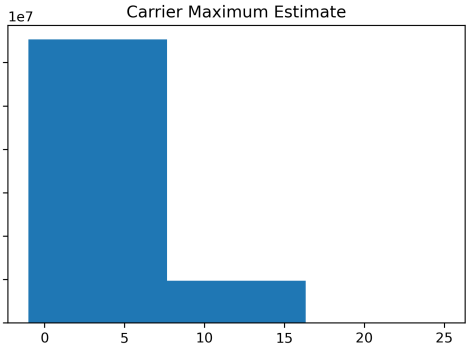

Figure 4: Carrier Maximum Estimate

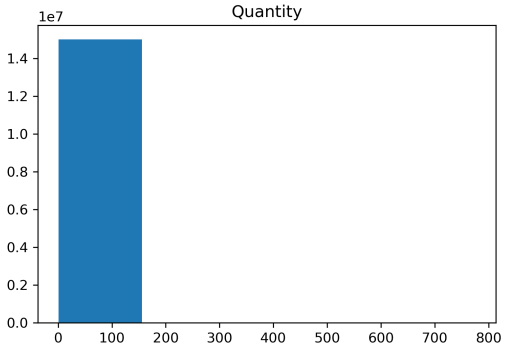

Figure 5: Quantity

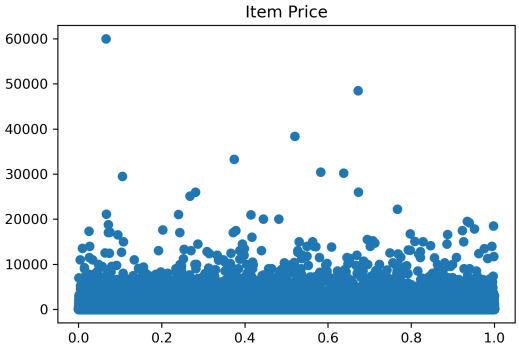

Figure 6: Item price of the packages

4.3 Hyper-Parameters

We have built an MLP model with five layers with an activation function 'relu' in each layer and *max_iteration* to 100. We did cross-validation to get the best results based on the hyperparameter selection.
We have used GridSearchCV, an exhaustive search over specified parameter values for an estimator. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid. We have used it in our Random Forest, where we have given n_estimators and defined the depth of the tree.

4.4 Results

After implementing Linear Regression and MLP Regressor to improve our prediction, we started implementing XGBoost Ensemble and Random Forest. And as the research paper from MIT mentioned Random Forest performs better as compared to Neural Networks, we found similar results where the loss function was less in the case of Random Forest.

As you can see the results in the table below, XGBoost Ensemble and Random Forest Regressor perform better than Multi-variate Linear Regression and MLP Regressor.

RESULTS		
ML MODELS	LOSS	Mean Absolute error(MAE)
Multi-variate Linear Regression	0.4270386788776306	0.9708902460699951
MLP (Multi-Layer Perceptron)	0.43909781761496497	0.9925409588139359
XGBoost Ensemble	0.3969708041049623	0.9226218924181868
Random Forest Regressor	0.393509592132839	0.9218217784096125

4.5 Learnings

1. Data cleaning is an important part and a necessary step. Removing unwanted data simplifies data analysis by keeping useful data only. It is easier to generate valuable insights and actions with properly cleansed data.

2. We studied and understood different regression models and their implementation.

3. We learned why ensembles are used to achieve better predictive performance on a predictive modeling problem than a single predictive model. Understanding this by the model reduces the variance component of the prediction error by adding bias.
   Advantage of using ensemble methods:

   - Performance : An ensemble can make better predictions and achieve better performance than any single contributing model.

   - Robustness : An ensemble reduces the spread or dispersion of the predictions and model performance.

5 Conclusions and Future Work

We have successfully achieved a good accuracy below the baseline value of the loss function.

Cleaning the data is crucial to ensure no faulty data points that might affect the classifier modeling. We have planned to do more data cleaning, excluding the weekends for the $number\_of\_handling\_days$ and $no\_of\_days\_after\_payment$ column. We conclude that the Random forest and XGBoost provide more accuracy than other learning algorithms we utilized for this project. As for the future scope, we realize more room for feature engineering. We achieved an accuracy score less than the baseline score we set up for the project. However, we are sure that we can improve this score further. We plan to dig deeper into more ways to solve this problem.

We plan to look into other machine learning algorithms such as *DeepETA*: A Spatial-Temporal Sequential Neural Network Model for Estimating Time of Arrival in Package Delivery System.

6 Acknowledgments

We want to thank Dr. Janardhan Rao Doppa for helping us with the project. Special kudos to Eval.ai for hosting the eBay Machine Learning challenge, which gave us a platform to work on a real-time dataset.

Contribution: No individual responsibilities. There were equal contribution.

References

[1] F. Wu and L. Wu, "DeepETA: A Spatial-Temporal Sequential Neural Network Model for Estimating Time of Arrival in Package Delivery System," AAAI, vol. 33, no. 01, pp. 774-781, Jul. 2019.

[2] Predicting Shipping Times with Machine Learning Jonquais, A.C. (2019). Predicting Shipping Time with Machine Learning. Diplome d'Ingenieur, Logistique, Universite le Harve Normandie, dspace.mit.edu/bitstream/handle/1721.1/121280/Jonquais_Krempl_2019.pdf?sequence=1&isAllowed=y.

[3] End-to-End Prediction of Parcel Delivery Time with Deep Learning for Smart-City Applications Araujo, A.C., & Etemad, A. (2020). End-to-End Prediction of Parcel Delivery Time with Deep Learning for Smart-City Applications. ArXiv, abs/2009.12197.