CAP5610: Machine Learning

Final Project Report:
Ebay’s Delivery Date Prediction System

Course Teacher:
Dr. Yanjie Fu

Submitted by:
Sanjay Shanbhag
Ashutosh Avadhani
Chethan Sringeswara
Md Rezwanur Rahman Jahin

University of Central Florida

December 2021

1. Introduction:

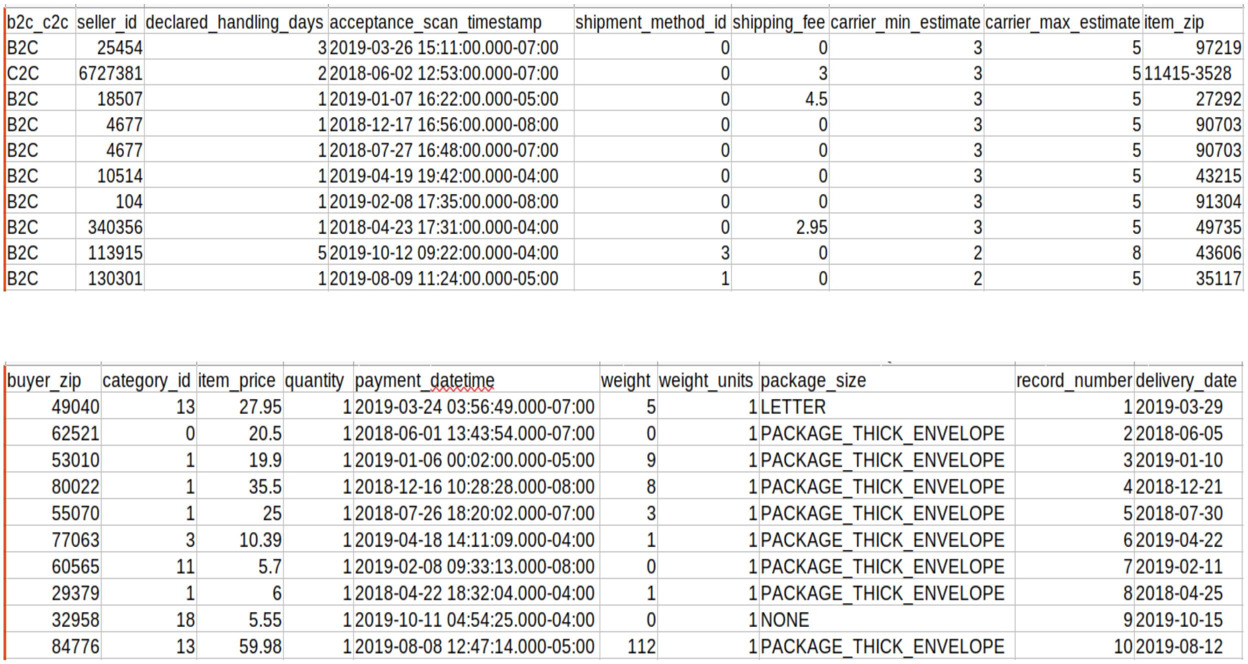
E-Commerce is replacing the mainstream shopping experience and the experience of shopping on an E-Commerce website is influenced by a lot of factors. One of the key factors is suggesting products and services to customers based on their likings. A lot of studies have been published and improvements on the previous model are being done for solving this problem. And most of the solutions to this problem are based on machine learning techniques and there are a lot of ranking algorithms.

The accuracy of shipping estimates is another crucial factor, and it plays a significant role in providing a hassle-free and trusty customer experience. However, this area has not received enough attention within the machine learning community despite its growing importance in the new online world. In this project, we are going to dive into this problem and propose viable solutions with proper justifications.

2. Description of The Dataset:

The dataset being used in this project is collected from eBay 2021 University Machine Learning Competition. The published dataset contains 15 million labeled records, the label being the actual delivery date. Each record represents individual orders placed by the customers of the e-commerce website. The data consists of 19 features including information like seller type (business to customer or customer to customer), seller zip code, buyer zip code, packet size, item price, shipping method, etc. Some of the features are anonymized to preserve customer and service providers' confidentiality. A snapshot of the dataset is presented in **Figure 1**.

The data provided had a few inconsistencies. To mention a few, the buyer or item zip code is given in four different formats, one of which is even invalid. The payment time and acceptance time can be from different time zones. Moreover, inconsistency can be in the labels as well. The given delivery date can be before the order date or months after the order date. To make our proposed models properly, the dataset has been preprocessed heavily.



b2c_c2c	seller_id	declared_handling_days	acceptance_scan_timestamp	shipment_method_id	shipping_fee	carrier_min_estimate	carrier_max_estimate	item_zip
B2C	25454	3	2019-03-26 15:11:00.000-07:00	0	0	3	5	97219
C2C	6727381	2	2018-06-02 12:53:00.000-07:00	0	3	3	5	11415-3528
B2C	18507	1	2019-01-07 16:22:00.000-05:00	0	4.5	3	5	27292
B2C	4677	1	2018-12-17 16:56:00.000-08:00	0	0	3	5	90703
B2C	4677	1	2018-07-27 16:48:00.000-07:00	0	0	3	5	90703
B2C	10514	1	2019-04-19 19:42:00.000-04:00	0	0	3	5	43215
B2C	104	1	2019-02-08 17:35:00.000-08:00	0	0	3	5	91304
B2C	340356	1	2018-04-23 17:31:00.000-04:00	0	2.95	3	5	49735
B2C	113915	5	2019-10-12 09:22:00.000-04:00	3	0	2	8	43606
B2C	130301	1	2019-08-09 11:24:00.000-05:00	1	0	2	5	35117

buyer_zip	category_id	item_price	quantity	payment_datetime	weight	weight_units	package_size	record_number	delivery_date
49040	13	27.95	1	2019-03-24 03:56:49.000-07:00	5	1	LETTER	1	2019-03-29
62521	0	20.5	1	2018-06-01 13:43:54.000-07:00	0	1	PACKAGE_THICK_ENVELOPE	2	2018-06-05
53010	1	19.9	1	2019-01-06 00:02:00.000-05:00	9	1	PACKAGE_THICK_ENVELOPE	3	2019-01-10
80022	1	35.5	1	2018-12-16 10:28:28.000-08:00	8	1	PACKAGE_THICK_ENVELOPE	4	2018-12-21
55070	1	25	1	2018-07-26 18:20:02.000-07:00	3	1	PACKAGE_THICK_ENVELOPE	5	2018-07-30
77063	3	10.39	1	2019-04-18 14:11:09.000-04:00	1	1	PACKAGE_THICK_ENVELOPE	6	2019-04-22
60565	11	5.7	1	2019-02-08 09:33:13.000-08:00	0	1	PACKAGE_THICK_ENVELOPE	7	2019-02-11
29379	1	6	1	2018-04-22 18:32:04.000-04:00	1	1	PACKAGE_THICK_ENVELOPE	8	2018-04-25
32958	18	5.55	1	2019-10-11 04:54:25.000-04:00	0	1	NONE	9	2019-10-15
84776	13	59.98	1	2019-08-08 12:47:14.000-05:00	112	1	PACKAGE_THICK_ENVELOPE	10	2019-08-12

Figure 1: Snapshot of Dataset

3. Project Objectives:

The goal of the project is to estimate the delivery date based on the information for each order. Based on the 18 features that each order has; we have developed machine learning models that can predict delivery dates with particularly good accuracy. This task of estimated dates can be done by estimating the number of days one order took to reach the customer from the day it has been ordered. If this can be done successfully, then adding the number of days with the order placement date gives the delivery date. Here in **Figure 2**, the entire process is demonstrated, from placing orders at checkout to the product reaching the customer.

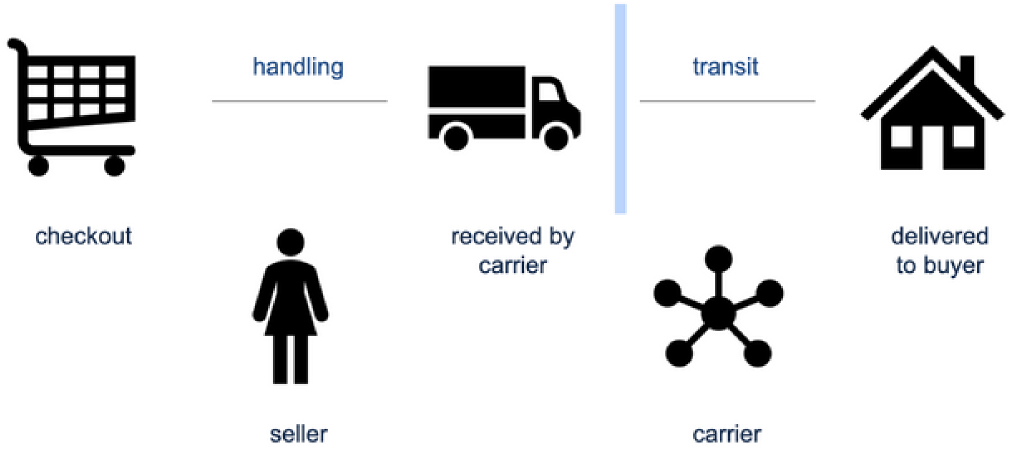


Figure 2: Full Picture of Order Placement to Item Delivery

4. Data Preprocessing:

One of the most important and challenging parts of the project was cleaning and preprocessing the data. And a significant effort was spared to do this job as perfectly as possible because the model is only as good as the data.

First, the buyer and the seller were localized to get an idea of the delivery distance. To do so the geographical coordinates, latitude, and longitude were extracted from the buyer and the item zip codes. This task is performed using “pgeocode,”[9] a Python library for high-performance off-line querying of GPS (Global Positioning System) coordinates, region name, and municipality name from postal codes. Then the geographical distance between the buyer and seller was calculated using the latitude and longitude coordinates. Some of the zip codes were only four digits and, in the US, there are no valid zip codes with only four digits. Thus, we have discarded the rows with invalid zip codes.

Second, time synchronization was another important part of the preprocessing process. The time of payment by the buyer and order acceptance by the career is given in their local time zone. So, all the timestamps were needed to be converted to some universal time zone. Moreover, the time zone information is not given for all the orders. In these scenarios where the time zone is missing, we have first calculated the time zone using latitude and longitude. And after that, this information was used to unify the times. All the payment time and acceptance time were converted to GMT.

The time needed to be considered along with the date because if an order is placed after 2 pm then it is processed the next day and so the dates were adjusted accordingly afterward.

Third, the weight of the packages is given in two different units, kg, and lbs. To train a model this anomaly had to be resolved and so all the weights were converted to kg. For some of the orders, the package weight was missing. As this can confuse the model, we have calculated the average weight for each package size and set this value for the orders with a missing weight according to their package size. Another important feature of the dataset is the type of package. The package type can be of seven distinct types. This can encode from 0 to 6 based on the size. For example, the smallest package type 'Letter' was encoded to '1', and the type 'Very Large Package' was encoded to '6', whereas the type 'None' then was encoded to '0'. Other feature columns such as 'b2c_c2c' that represent the seller type were also encoded.

Finally, the dates given for payment by the buyer, orders accepted by the career, and products reaching the customer were utilized to calculate the number of days required in each phase. The actual handling days were calculated by subtracting the payment date from the carrier acceptance date. Again, the number of days required for a product to reach the customer from the date it has been ordered (delivery time) was calculated from the adjusted payment date and delivery date. In fact, the delivery time has been used as the label for the machine learning models.

5. Data Visualization and Feature Selection:

The eBay data set contains 18 different features. But all these features do not necessarily influence the delivery time, finding out which features influence, is the objective of this project. Some of the features do not repeat and some of them are randomly set as identifiers. These features don't have significance and thus need to be discarded. So, there is a necessity of visualizing the given data to better understand the pattern.

To begin with, some of the features have been plotted to understand the distribution of data. After that, we have calculated the correlation matrix to know what are the best features that have a high influence on prediction.

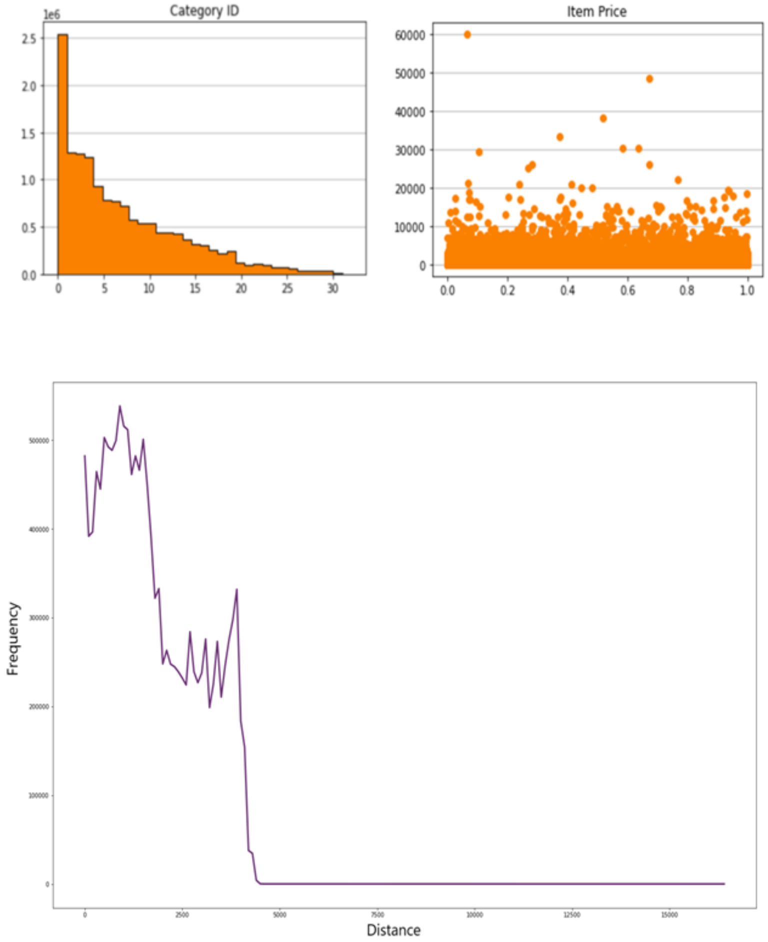
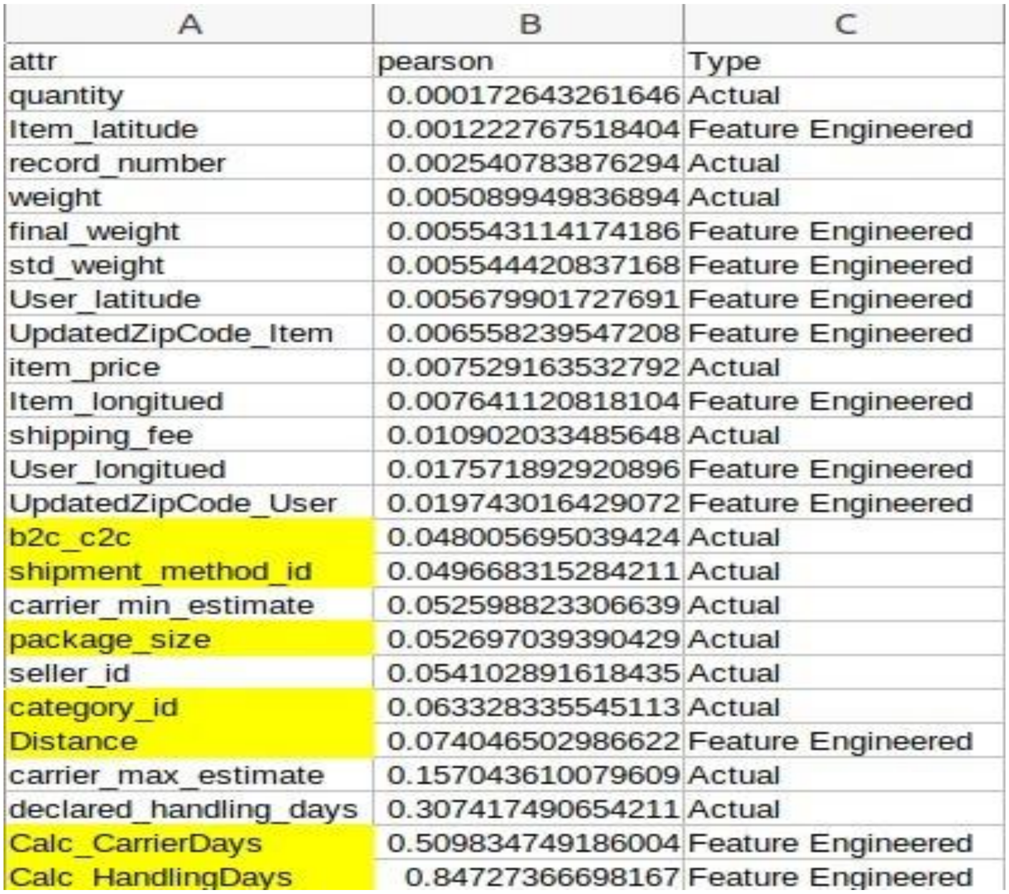


Figure 3: Variable's relationship

It is seen that items with smaller category id are more frequently ordered and most of the items are delivered within a threshold distance. Whereas price is equally distributed and has truly little significance.



A	B	C
attr	pearson	Type
quantity	0.000172643261646	Actual
Item_latitude	0.001222767518404	Feature Engineered
record_number	0.002540783876294	Actual
weight	0.005089949836894	Actual
final_weight	0.005543114174186	Feature Engineered
std_weight	0.005544420837168	Feature Engineered
User_latitude	0.005679901727691	Feature Engineered
UpdatedZipCode_Item	0.006558239547208	Feature Engineered
item_price	0.007529163532792	Actual
Item_longitued	0.007641120818104	Feature Engineered
shipping_fee	0.010902033485648	Actual
User_longitued	0.017571892920896	Feature Engineered
UpdatedZipCode_User	0.019743016429072	Feature Engineered
b2c_c2c	0.048005695039424	Actual
shipment_method_id	0.049668315284211	Actual
carrier_min_estimate	0.052598823306639	Actual
package_size	0.052697039390429	Actual
seller_id	0.054102891618435	Actual
category_id	0.063328335545113	Actual
Distance	0.074046502986622	Feature Engineered
carrier_max_estimate	0.157043610079609	Actual
declared_handling_days	0.307417490654211	Actual
Calc_CarrierDays	0.509834749186004	Feature Engineered
Calc_HandlingDays	0.84727366698167	Feature Engineered

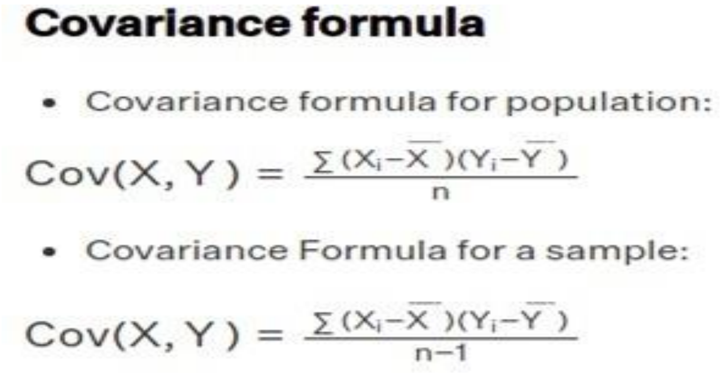
Figure 4: correlation Matrix

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[\, n\Sigma x^2 - (\Sigma x)^2 \,]\,[\, n\Sigma y^2 - (\Sigma y)^2 \,]}}$$

Figure 5: correlation formula

The features namely "carrier_min_estimate", "carrier_max_estimate", and "declared_handling_days" were skipped as these features were feature engineered to get namely "Calc_CarrierDays", "Calc_HandlingDays". Hence these features are used for training. Taking about seller_id, it was skipped even though it was relevant because, these seller ids will be different and some new values in the quiz set, hence we wanted to avoid confusion in the model.

Coming to feature selection, we had 2 relevant options to check the relevancy of the features in the dataset with the target variable namely Correlation and Covariance.



Covariance formula

- Covariance formula for population:

$$Cov(X, Y) = \frac{\sum (X_i - \overline{X})(Y_i - \overline{Y})}{n}$$

- Covariance Formula for a sample:

$$Cov(X, Y) = \frac{\sum (X_i - \overline{X})(Y_i - \overline{Y})}{n-1}$$

Figure 6: covariance formula

Covariance: In simple words, it tells us if the features considered are increasing or decreasing w.r.t each other. Say if both are increasing w.r.t each other, then the covariance would be positive else negative. But covariance has a drawback i.e., if the values in the features are scaled, then the covariance coefficients also change. Hence considering this fact, the correlation coefficient was chosen for the relevancy measure. When it comes to correlation, it gives both the magnitude and direction of relevancy. But here in the results of correlation posted below, modulus has been applied so as the sort and get the top relevant features.

For increasing the model predictability, we have also taken XG-Boost feature importance to get the most influencing attributes of the dataset.

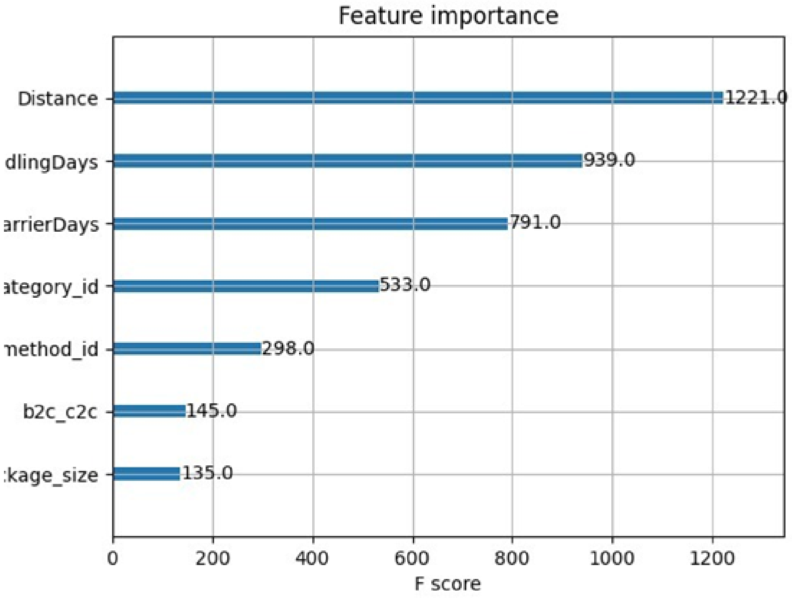


Figure 7: Feature Importance Regarding F-Score

6. Related Works

Today almost all the supply chain industries are using the advances of machine learning models for predicting their delivery date. There are specific research areas that are focusing on having multiple legs of the shipment related to product delivery problems in large companies such as Walmart, but also in companies that are smaller in scale and their delivery time is in minutes such as food delivery services [5]. This [6] article by Walmart explains the details on how to build modular structures while calculating the delivery date. For our project, we also explore the benefits of splitting the different legs of shipment into smaller segments, handling time, and shipment time. Another study done by Joseph Magiya [7], details how we can modify our input variables for different models and which in turn helps us in identifying some of the important features. In this paper, the use of XGBoost and regression were implemented to predict the delivery date. We also have implemented XGBoost because of its ability to determine feature importance.

7. Model Pipeline:

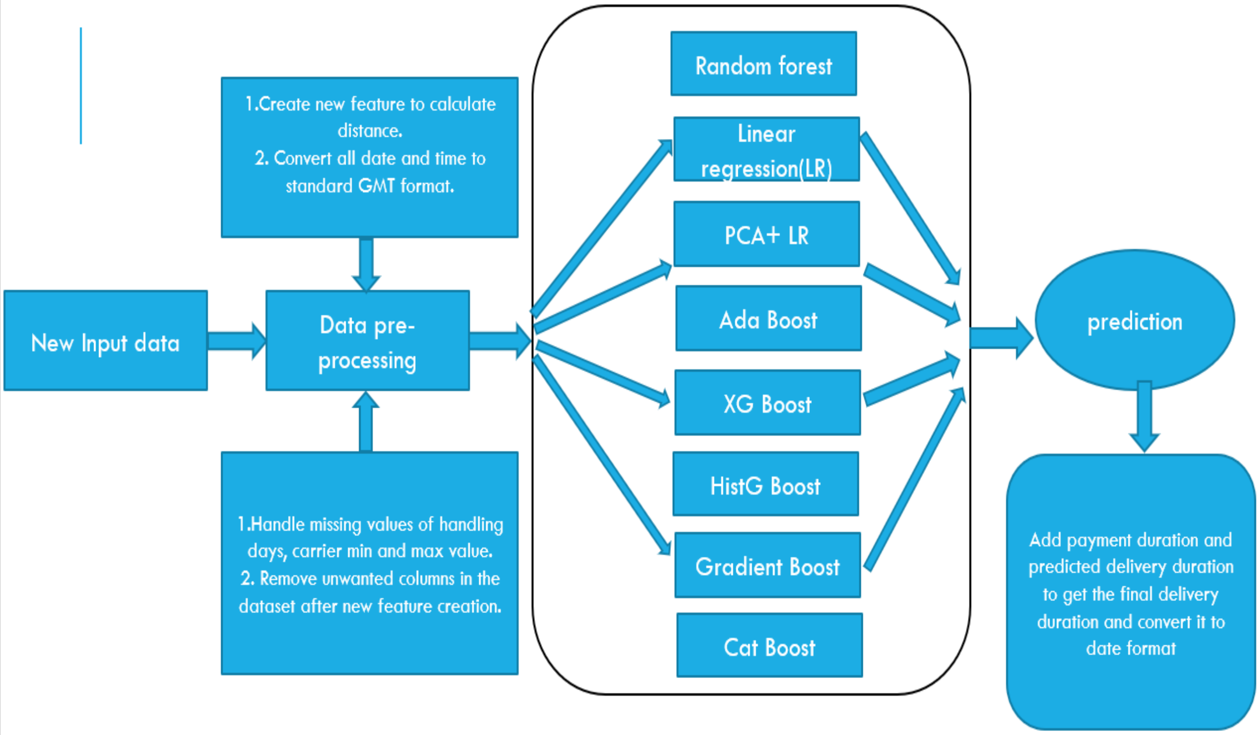

New Input data
1.Create new feature to calculate distance.
2. Convert all date and time to standard GMT format.
Data pre-processing
1.Handle missing values of handling days, carrier min and max value.
2. Remove unwanted columns in the dataset after new feature creation.
Random forest
Linear regression(LR)
PCA+ LR
Ada Boost
XG Boost
HistG Boost
Gradient Boost
Cat Boost
prediction
Add payment duration and predicted delivery duration to get the final delivery duration and convert it to date format

Figure 8: Model Pipeline

8. Proposed Models:

The eBay Contest dataset that has been considered for this project is a very diversified one and to achieve the best possible performance with machine learning models we have explored eight of them. The models are discussed in detail below:

Linear Regression:

Linear Regression is an approach for modeling the relationship between a scalar response and one or more explanatory variables. In the case of modeling the number of days required to complete the delivery is the response variable that has been calculated based on other feature variables.

Here linear regression model has been fit using the least square method. We are trying to reduce the error by observing the variation in the explanatory variables.

Random Forest:

It is one of the most popular ensemble models and a lot of research has been done into this. Its model tries to make predictions by fitting numerous decisions tree to the sub-set of the dataset and gives the average value as the result. It works well if the data is non-linear, and it reduces the variance that occurs in a single decision tree.

Ada-boost:

It is another heavily utilized ensemble model. It trains weak learners and results are given based on the vote of collective weak learners. It tries to improve from the previous iteration and decrease the mean square error in upcoming iterations. Initially, we have taken the default decision tree as the weak-learner but when it is compared to linear regression as a weak learner, we got better results.

Gradient Boost:

It is like the other boosting algorithms but here the weak learners are decision trees. It creates an ensemble of these weak learners to train the model and it builds these trees in a stage-wise manner and uses gradient decent to decrease the error. Here we use a specific configurations like maximum nodes, layers to maintain the weak learners weak but we still construct them in an additive manner.

Histogram Gradient Boost:

It is also one of the ensemble training models. And it is one of the popular methods in the tabular regression method. It tries to eliminate the problem with the simple gradient boosting algorithm. When a large amount of data is used for training purposes the gradient boosting becomes slow. It is discretizing a large input variable into a few hundred values. So, by tailoring input values it increases the speed of the algorithm.

XGBoost:

It is an open-source software library that provides support for variants of gradient boost trees like histogram gradient boost, regularized gradient boosting machines, etc., which follows the gradient descent approach i.e., when an ensemble is created using a weak learner, next tree appended with a reason to reduce the overall loss without any changes in the previous model. The training is stopped only after the loss crosses a certain threshold. The implementation of the algorithm was engineered for efficiency of compute time and memory resources. It was shown in one of the survey's that xgboost was the fastest and efficient in terms of execution speed and memory utilization. The graph in **Figure 9** mentions the same.

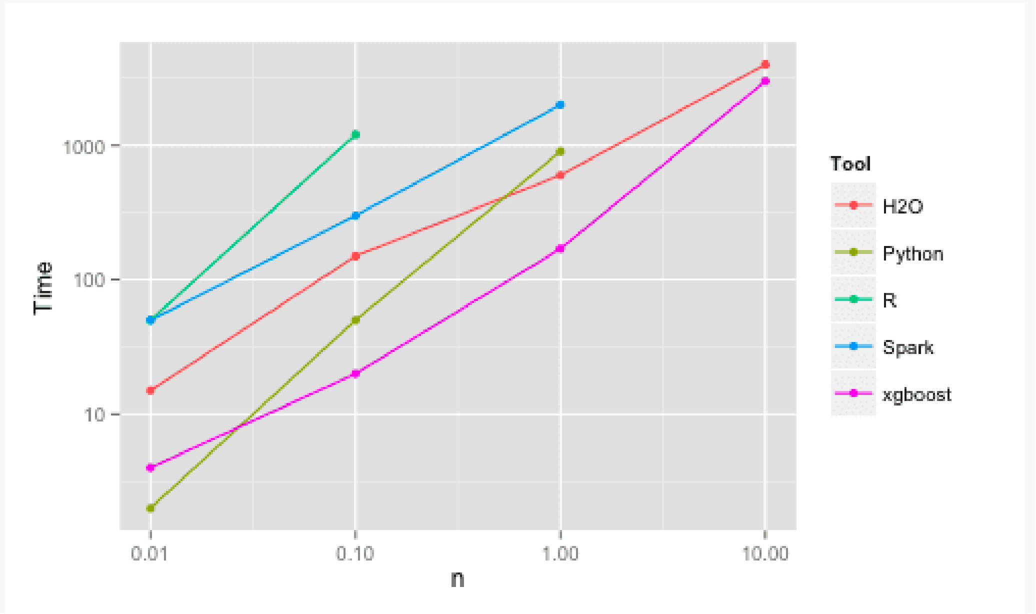


Figure 9: XG Boost Performance

When it comes to Kaggle and other machine learning competition, Xgboost is a go-to algorithm. Moreover, it can be called a scalable algorithm as it can be used against the larger scale datasets.

Important features and their rank can be plotted and calculated in xgboost and hence a model based on the prominent features can be trained.

Cat Boost:

Another machine learning model that is mostly used in various machine learning competitions is none other than Cat boost with its advanced machine learning pipeline features. The prominent features of Cat Boost include:

1. Model checkpointing feature
2. On the go training and validation accuracy/loss visualization
3. Faster than XGBoost
4. Provides feature importance maps.
5. verbose param provides better analysis as compared to the existing implementations.
6. model can be saved in various formats like .cbm(cat boost format), .onnx etc., making it ready for production environments.
7. It gives a better interpretation of the model and that way it is just an accuracy game.
8. Can handle various data issues like missing value and encoding of categorical variables.

Voting Regressor:

It is a combination of all the machine learning models trained above. Here the prediction has been made using all the above model and we take the vote for every iteration to pick the final output for that iteration. Initially we have taken the unweighted voting method and we also implemented the weighted voting regressor. We have taken the performance of our model as the weights.

9. Results:

We have trained eight different models for the data set. The data set consists of more than 500000 data records. We report the accuracy of RMSE for all the trained models. Here we have taken accuracy as two different scenarios, TOP1 and TOP2.

Here, TOP1 accuracy is calculated based on exact match of the predicted delivery date by the ML model with the actual value. Whereas the TOP2 accuracy is calculated based on a relaxed assessment criterion. If the predicted value falls within a three-day window, one day before and after the actual delivery date then the prediction in regards is accurate in TOP2 accuracy.

RMSE is a quadratic measure of the average magnitude of the error. It can also be defined as the square root of the average squared difference between the actual output and the predicted output. It will penalize large errors since it takes the root of the squared error.

Here, f is forecasted output and, o is observed output.

Models	**TOP1 Accuracy**	**TOP2 Accuracy**
Linear regression without PCA	70.21%	99.94%
Random forest	66.60%	98.78%
ADA Boost	70.53%	99.97%
XG Boost	60.09%	99.80%
Gradient Boost	67.43%	99.80%
Histogram Boost	66.48%	99.75%
Cat Boost	66.45%	99.75%

Table 1: Results of various models.

For linear regression model we have taken different number of components i.e., number of features to check whether the model performs differently.

Linear Regression with PCA component numbers	**TOP1 Accuracy**	**TOP2 Accuracy**
2 components	27.65%	68.62%
3 components	34.19%	78.56%
4 components	33.01%	80.13%
5 components	34.07%	84.67%
6 components	70.65%	99.97%

Table 2: Linear regression with multiple components.

Models	**RSME values**
Linear regression without PCA	0.3665
Linear regression with PCA (component 6)	0.3665
Random forest	0.4019
ADA Boost	0.3728
XG Boost	0.3548
Gradient Boost	0.3568
Histogram Boost	0.3894
Cat Boost	0.3553

Table 3: RSME of the various models.

Looking into the above results, it is pretty clear that Linear regression and Adaboost with Linear regression are outperforming various tree based boosting algorithms. The results seem shocking but there is a specific reason for that, when it comes to regression tree-based techniques cannot extrapolate the underlying data, hence when outside data that is not in the training is presented to the model, it predicted the highest value that was presented to it. This can be proved from a simple example shown below.

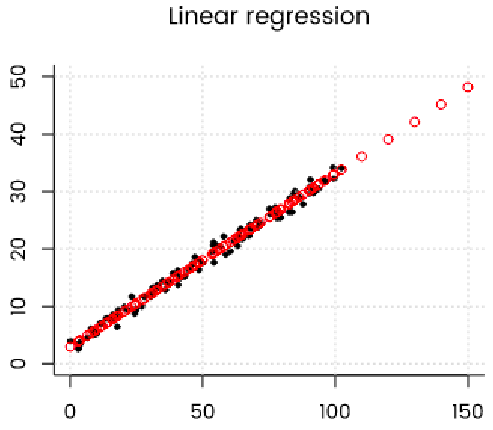


Figure 10: Linear Regression

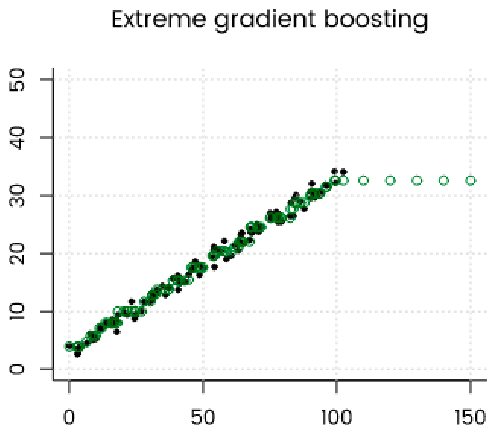


Figure 11: Extrema Gradient Boosting

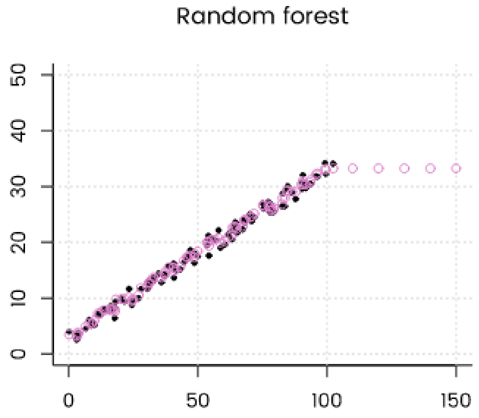


Figure 12: Random Forest

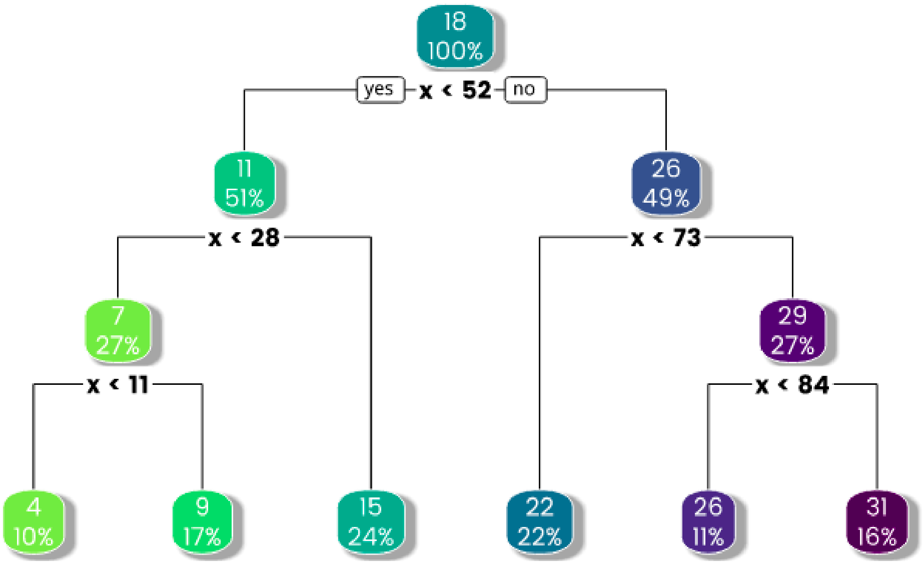


Figure 13: decision tree Example

Now if we consider the 3 models namely extreme gradient boosting , linear regression and random forest and fit it on a linear data , (the black dots represents the actual target value and the colored dots represents the predicted values of the trained models), now looking into **Figure 9, Figure 10, Figure 12** it is clear that , when the prediction is made on the test data (out-of-set data) , the predictions of the linear regressions is in the same direction but coming to random forest and extreme gradient boosting tree , the predicted values are stagnated to the highest target value in the training data, hence not able to predict the correct values. The reason behind this is shown in the decision tree shown in **Figure 13**, wherein it clearly shows that the tree always gives out either the lower bound or the higher bound, which in this case would be 4,9,15,22,26,31. Now considering our trained models, it is clear that the models that have performed worse have utilized a tree based base estimator (may be a decision tree) and hence they are producing lower accuracies as compared to Linear Regression and Adaboost with Linear regression.

10.Conclusion:

The focus of this project was to predict delivery dates with better accuracy to improve customer experience. From the experiment conducted for this project we have proved that the delivery date can be predicted almost perfectly if it is done in a window of three days. For orders that take more time, 4-5 days, declaring the delivery date in a window of three could be a good idea.

From the perspective of analyzing the effectiveness of different machine learning models, we found out that simpler models like linear regression outperformed several complex models. This is due to the nature of the feature set and the correlation of the response variable with the feature set. The best performer of the ML models was an ensemble method, ADA-Boost with linear regression. Another ensemble method Random Forest also performed adequately well.

Another major reason for the simpler models outperforming the complex models is due to the preprocessing. The features were modified heavily based on instinct and correlation between features. Thus, the feature extraction played a major role in this project.

Code Link

https://github.com/Ebay_Delivery_Date_Prediction_system

11. References

1. https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/
2. https://towardsdatascience.com/why-you-should-learn-catboost-now-390fb3895f76
3. https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/
4. Jihed Khiari ID, Cristina Olaverri Boosting Algorithms for Delivery Time Prediction in Transportation Logistics
5. http://freerangestats.info/blog/2016/12/10/extrapolation
6. https://thenewstack.io/how-uber-eats-uses-machine-learning-to-estimate-delivery-times/
7. Customer Delivery Time(CDT) Prediction using Machine Learning | by Nitish Gopu | Walmart Global Tech Blog | Medium
8. https://www.researchgate.net/publication/344871967_Predicting_Package_Delivery_Time_For_Motorcycles_In_Nairobi
9. https://pypi.org/project/pgeocode/