جامعة الملك فهد للبترول والمعادن
King Fahd University of Petroleum & Minerals

MATH405: Learning from Data

Term 201

Term Project

Prepared by

Mohammed Aljaloud

201639600

Table of contents

Introduction *4*

Data preparation and EDA *5*

Literature Review *6*

Mathematical description *8*

Gradient Boosting **8**

Random Forest **9**

Models Tuning *10*

XGBoost **10**

Random forest **12**

Graphics *13*

Train Data **13**

Test Data **14**

Results *15*

Conclusion *15*

References *16*

Table of figures

Figure 1: Example of services uses forecasting models 4
Figure 2: Data pair plot 6
Figure 3: XGBoost features 7
Figure 4: XGBoost mathmatical model 8
Figure 5:Random forest mathmatical model 9
Figure 6:Train data on XGBoost model 13
Figure 7:Train data on Random Forest model 13
Figure 8:Test data on XGBoost model 14
Figure 9:Train data on XGBoost model 14

Table 1 : XGBoosting parameters 10
Table 2: XGBoosting parameters tuning 11
Table 3 : Random Forest parameters 12
Table 4 : Random Forest parameters tuning 12
Table 5: Accuracy comparison 15

Introduction

Food and grocery delivery services has increased significantly last couple of years, especially during the pandemic. When a customer order from a delivery service; the application calculate the expected time for the delivery in matter of second and start count town until the driver arrive. Almost all the time it takes what the model predicted for the customer based on some attributes. The accuracy of the expected time comparing to the true value depends on the complexity of the model, for example when two pins inserted in google maps, the number of cellphones on the route contribute to the given expected time of the trip. Our project is a subproblem of the delivery service model, where is the distance is constant 3.19KM between two fixed points and we need to calculate the travel time given the day of the year, period of the year, day of the week and period of the day. The data provided are for years 2017, 2018 and 2019 and the goal is to find an appropriate model to forecast the time in 2020.

Figure 1: Example of services uses forecasting models

Data preparation and EDA

In dealing with our data, there were couple of inconsistent values and dropping them was the proper thing to do since they are small number of rows. The data is cleaned and now ready exploratory data analysis. In order to see the relation between different attributes and the time; the pair plot is the right choice. Shown below (FIGURE02) shows the pair plot data, the three years shows similar distribution behavior for the time. Although the visuals show behavior, we need come up with an inclusive conclusion. A KS test is conducted, and it ensured that we cannot reject that all three years do not follow normal distribution. As a result, the three years data were grouped, and the forecast journey start here.

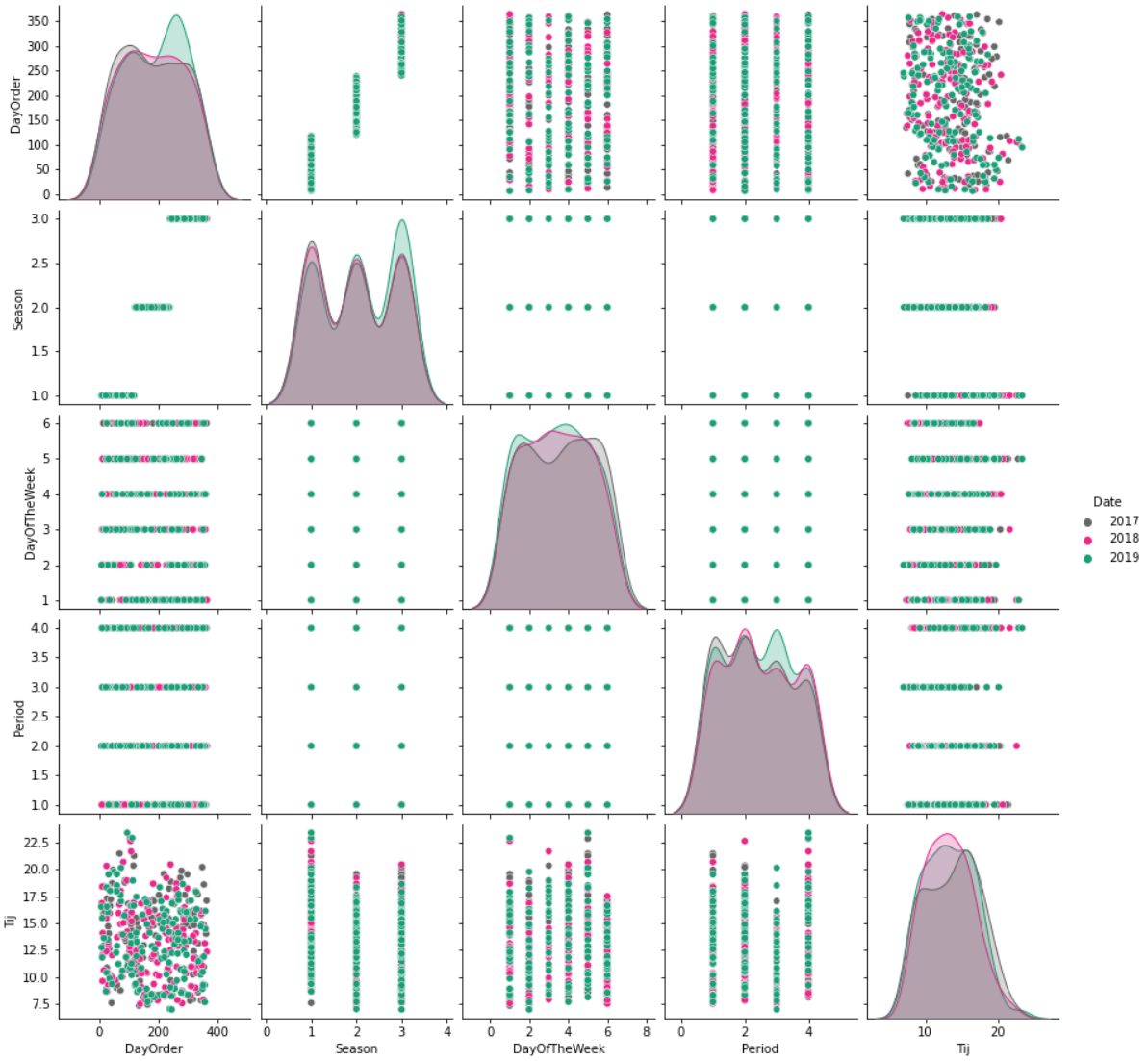

Figure 2: Data pair plot

Literature Review

There are thousands and thousands of forecasting methods and there is no systemic way to find the best model since each dataset tells a different story and require different ways to deal with it. In my project, the forecasting methods that showed good results are gradient boosting methods and random forest. The idea of boosting started in 2003 by two computer science professors Yoav Freund and Robert Schapire where they formulated AdaBoost algorithm which is short for adaptive algorithms. Although the two techniques share more differences than similarities, the main idea is the same which is multiple simple sub models create a bigger complicated model. Jerome H. Friedman, an American statistician saw the potential of

boosting on the gradient for regression application. There are many famous gradient boosting algorithms such as CatBoost, XGBoost and LightGBM. The choice of which one to choose depends on the data if categorical variables are present in the data CatBoost can deal with them without encoding. LightGBM is the fastest among them and XGBoost has the highest accuracy. Random forest also has the idea of building multiple small models that produce more complex model. First random forest model paper was established in 2001 by Leo Bierman, an American computer scientist. The main differences between Gradient Boosting and Random Forest are the relation between the sub-models and how they combine the result.

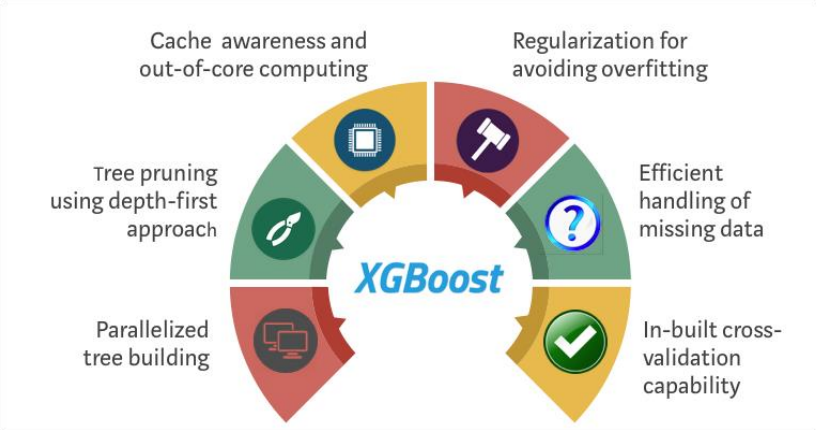

Figure 3: XGBoost features

Mathematical description

Gradient Boosting

The word XGBoost stands for eXtreme Gradient Boosting. It is decision-tree based, there are three features that makes it outperform any Gradient Boosting algorithms. It uses L2, L1 norm and number of lead nodes to penalize the model and avoid model overfitting. In addition to that it has built in cross-validation. The FIGURE below shows a simple mathematical representation of XGBoost algorithm, the first equation is the objective that the algorithm tries to minimize. The first component is the loss function which is included in all Gradient Boosting algorithms, what makes XGBoost so powerful is the second component. It is called the regularization and Gamma is the Lagrangian multiplier which controls the model complexity, the second equation is simply the objective at time t. From the figure below, the objective function is function of functions, and it can not be optimized using traditional methods. For that reason, the algorithm uses Taylor's Theorem to transform the objective function to Euclidean distance domain to be able to use traditional optimization methods.

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{t-1} + f_t(x_i)) + \Omega(f_t)$$

$$where\ \Omega(f_t) = \gamma T + \frac{1}{2}\lambda ||w||^2$$

Figure 4: XGBoost mathmatical model

Random Forest

Random forest is a supervised machine learning algorithm used for both classification and regression. The algorithm builds multiple decision tree to construct the forest. The output of the algorithm is an aggregation of it is independent decision trees. In the project, Random Forest Regression offered by Scikit-learn is used. The FIGURE below shows Random forest regression from Breiman paper in 2001, which is what Scikit-learn algorithm based on, the only modification is in the first equation where they used mean absolute error rather than mean squared. It uses bootstrap aggregation; bootstrap is another word for random sampling with replacement. This technique allows the model to better understand the bias and variance.

$$E_{\mathbf{X},Y}(Y - h(\mathbf{X}))^2$$

$$E_{\mathbf{X},Y}(Y - av_k h(\mathbf{X}, \Theta_k))^2 \rightarrow E_{\mathbf{X},Y}(Y - E_\Theta h(\mathbf{X}, \Theta))^2.$$

$$\begin{aligned} PE^*(forest) &= E_{\mathbf{X},Y}[E_\Theta(Y - h(\mathbf{X}, \Theta)]^2 \\ &= E_\Theta E_{\Theta'} E_{\mathbf{X},Y}(Y - h(\mathbf{X}, \Theta))(Y - h(\mathbf{X}, \Theta')) \end{aligned}$$

$$\bar{\rho} = E_\Theta E_{\Theta'}(\rho(\Theta, \Theta')sd(\Theta)sd(\Theta'))/(E_\Theta sd(\Theta))^2$$

Figure 5:Random forest mathmatical model

Models Tuning

XGBoost

The Table01 below shows the parameters used for XGBoost regression, there are so many more, but I found these to have the largest affect. In order to find the optimal parameter that reduce the error, there are two powerful technique called Randomized Search CV and Grind Search CV. The input of the techniques is the model and the parameters that has influence on the learning process and give back the optimal values for these parameters. The Table02 shows Randomized Search CV versus Default value. Grind Search CV is expensive and when the user does not have a known small range for each hyper parameter because the algorithm tries every possible combination.

XGBoost Parameters

Parameter	Description	Affect when increasing
n_estimators	Number of trees in the forest	Overfitting
Booster	‘gbtree’ and ‘dart’ are tree based ‘gblinear’ uses linear function	N/A
max_depth	Maximum depth of the tree	Overfitting
Learning_rate	Step size shrinkage in update	Overfitting
min_child_weight	Minimum sum of instance weight needed in a child	Underfitting
Base_score	The initial prediction score	N/A

Table 1 : XGBoosting parameters

Comparison between RandomizedSearchCV output and default parameters value

Parameter	RandomizedSearchCV	Default
n_estimators	250	100
Booster	gbtree	gbtree
max_depth	6	6
Learning_rate	0.2	0.3
min_child_weight	1	1
Base_score	1	0.5
Reg_lambda	375	1
MAE	2.9872158223990506	3.01872860699321

Table 2: XGBoosting parameters tuning

Random forest

The Table03 below shows some of the parameters of Random forest regression, these parameters that were tuned to improve the result of the algorithm. The Table04 shows comparison between different parameters value.

Random forest Parameters

Parameter	Description	Affect when increasing
n_estimators	Number of trees in the forest	Overfitting
max_features	Number of features to look for split	Overfitting
max_depth	Maximum depth of the tree	Overfitting
min_samples_split	Minimum number of samples required to split	Underfitting
min_samples_leaf	Minimum number of samples required to be count as leaf node	Underfitting
bootstrap	Random sampling with replacement	N/A

Table 3 : Random Forest parameters

Comparison between RandomizedSearchCV output and default parameters value

Parameter	RandomizedSearchCV	Default
n_estimators	1200	100
max_features	Sqrt(n_features)	All features
max_depth	50	None
min_samples_split	10	2
min_samples_leaf	4	1
bootstrap	True	TRUE
MAE	3.0993622844724413	3.2330357936710956

Table 4 : Random Forest parameters tuning

Graphics

Train Data

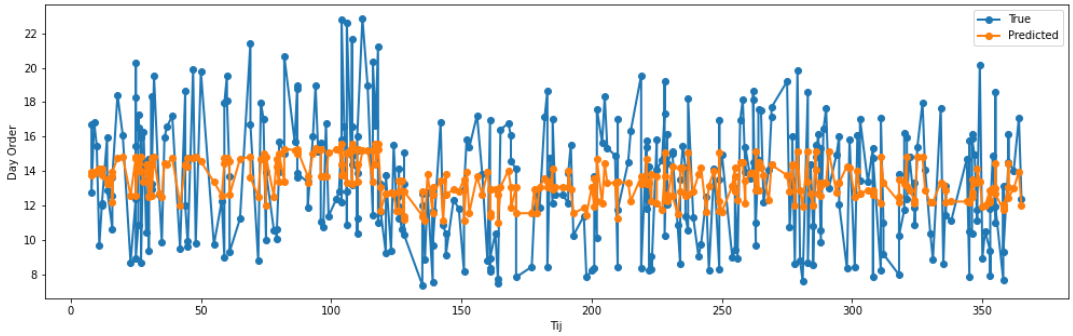

Figure 6:Train data on XGBoost model

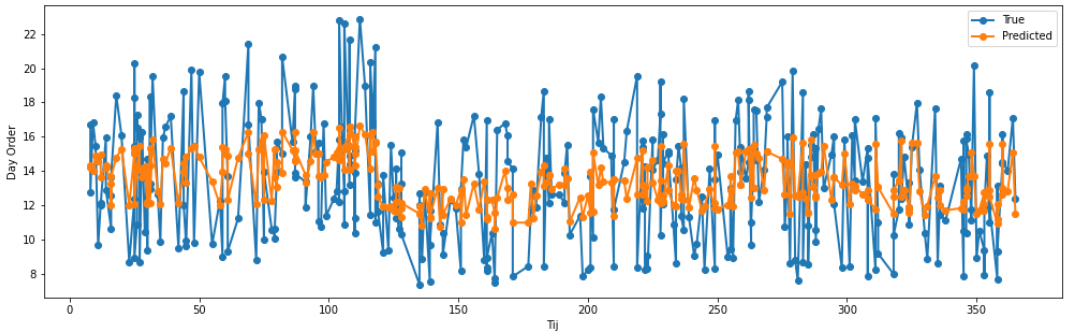

Figure 7:Train data on Random Forest model

Test Data

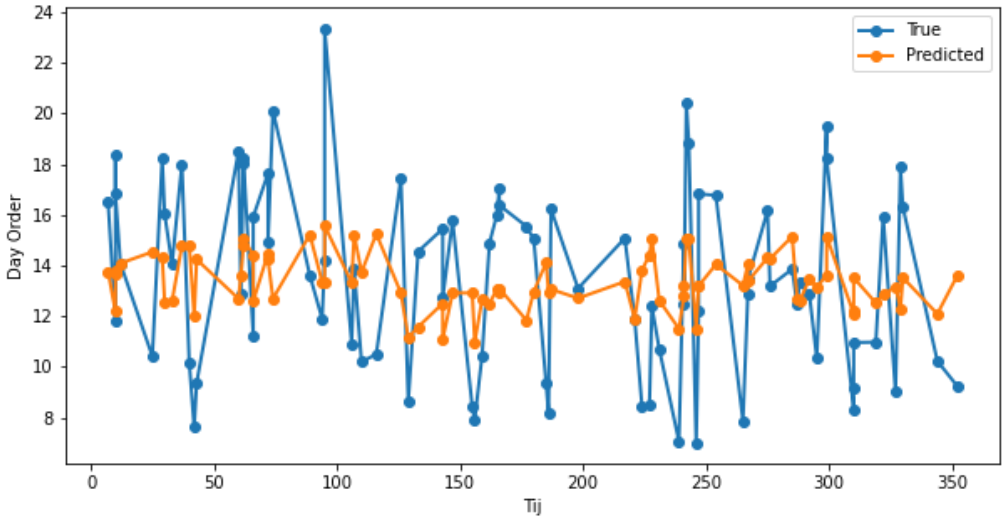

Figure 8:Test data on XGBoost model

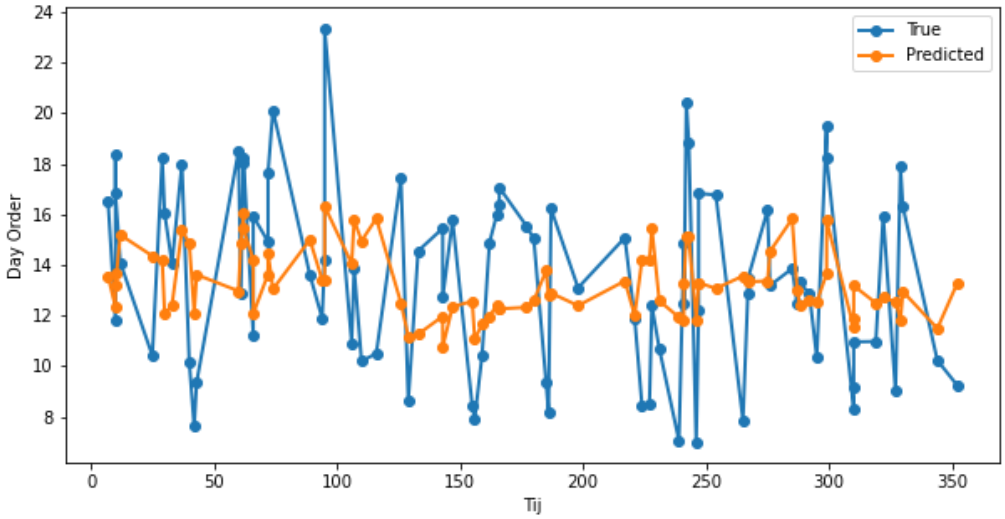

Figure 9:Train data on XGBoost model

Results

The results from both models were close when comparing MAE and MAPE. It is true that both algorithms based on decision trees but each work in different mechanisms. Tuning the hyper parameters improved the output of both models when using Randomized Search CV. From the above figures we can see that both models capture the pattern but with lower magnitudes. A higher magnitude can be achieved by increasing the learning rate in XGBoost but we would sacrifice the accuracy of the points near the mean line. On the other hand, forcing Random Forest to meet the magnitude will cause overfitting. The table below shows MAPE and MAE values for both.

Error measurement	Random Forest	XGBoost
MAE	3.0993622844724413	2.9872158223990506
MAPE	23.41123836125479	22.4018230190773

Table 5: Accuracy comparison

Conclusion

In conclusion, services are competing to have the best forecasting model. It contributes so much to customer satisfaction and loyalty. Our project is a smaller and less detailed model that tries to answer the same question “How long will it take?” based on couple of input variables. There were couple of poor trials using Principal Component Regression, polynomial regression, LASSO and ridge regression. Also, there were promising trial using Long short-term memory (LSTM) artificial recurrent neural network since it deals with cyclical behaviors well, but it was a bit advance for me. XGBoost scored better accuracy than Random forest, tuning parameters is an important step when dealing with detailed models and can improve the prediction significantly.

References

1. https://medium.com/
2. https://scikit-learn.org/
3. https://xgboost.readthedocs.io/
4. Breiman, Leo. " RANDOM FORESTS." University of California, Jan. 2001. URL: https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf
5. Chen, Tianqi, and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." University of Washington, Aug. 2016. URL : https://arxiv.org/pdf/1603.02754.pdf