Predicting Order Lead Time for Just in Time production system using various Machine Learning Algorithms: A Case Study

Saurabh Singh*
Department of MPAE
Netaji Subhas University of Technology
New Delhi, India
saurabhbazzad@gmail.com

Umang Soni*
Department of MPAE
Netaji Subhas University of Technology
New Delhi, India
umangsoni.iitd@gmail.com

Abstract—**Predicting lead time can have some serious consequences on Supply Chain Management (SCM) system including reduction in inventory and associated costs, increase in throughput and yield. The aim of this research paper is to predict the order lead time in a Just in Time (JIT) manufacturing environment. An attempt has been made to predict the order lead time or delivery time for a restaurant using several features. Different supervised state-of-the-art regression machine learning algorithms were used and their accuracies were compared on the given dataset.**

Keywords—Just in time (JIT), Machine Learning (ML), Lead Time (LT), Supply Chain Management (SCM), Prediction

I. Introduction

Machine Learning (ML) is a field of Artificial Intelligence through which computer systems can have the ability to "learn" from data using statistical techniques without being explicitly programmed. In particular, ML can be defined as an automated process that extracts pattern from (historical) data [1]. Machine Learning can be classified into Supervised: data is labeled, Unsupervised: data is unlabeled, Semi-Supervised: mostly unlabeled data with some labeled data, and Reinforcement: agent learns a policy to maximize its rewards over time. We will be dealing with only Supervised Machine Learning Algorithms. Supervised Learning can further be divided into Classification: when the output variable is categorical and Regression: when output variable is real or continuous valued. Since order lead time is a continuous valued function, we will be using regression techniques only. There are various regression algorithms available, including Linear Regression, Polynomial Regression, Decision Tress, Random Forest, Lasso, SVM (Support Vector Machine), Ridge etc. We have used some of these algorithms in this paper.

Applications of Machine Learning are ubiquitous, from using it to process text, speech and images to training deep neural networks for application like self driving cars. It has been adopted by many industries for applications ranging from predicting if a transaction is fraud to predict if a particular user will buy a product or not. It is also one of the most researched topics in academia, given its possible applications in fields of science, mathematics, engineering, medicine and others.

A Just in time (JIT) production is an idealized concept where we supply the required material at the desired place at the required time. Just-in-time (JIT) production methods have been proven to be an efficient way of manufacturing [3] [4] [5]. It reduces the various costs involved in a supply chain, such as cost of warehousing, security and insurance, increasing throughput and yield [2]. Levitt and Abraham further highlighted the supremacy of JIT systems over other manufacturing processes [6].

Restaurants and other food delivery businesses are perhaps one of the prime examples of just-in-time production environment, wherein the food items are produced only after an order is received and delivered at the required place in very little time.

Michele Banko and Eric Brill showed that, when the training data is sufficiently large, the various Machine Learning Algorithms, including very simple ones performed almost equally well even on a very complex task such as Natural Language Disambiguation [7]. They also noted that we should direct efforts towards increasing data size and emphasize our focus less on comparing different learning techniques on small data sets. This idea was further supported by P. Norvig et al. [8]. However, it should be noted that till today, most of the data sets used are either small or medium sized, as the extra data may not always be cheap to collect, and in some cases might even be unfeasible. Therefore in this paper, we will use various supervised regression machine learning algorithms and use them to predict the order lead times, and compare the results generated from the same.

Order lead time is defined as the time difference between placing an order for a product and receiving the product, in this case, for example- it will be the time difference between placing the order with the restaurant and getting the order delivered to the required destination.

The rest of the paper is organized as follows. Section II discusses about various Machine Learning Algorithms used in obtaining results for this paper. Section III gives a brief overview of the problem. Section IV discusses about the research methodologies used in the paper and compares the results obtained from various algorithms. Future research agenda is provided in Section V. Section VI concludes the paper with Acknowledgements.

II. Lead Time Prediction With Machine Learning

In this section, we will very briefly talk about the various Machine Learning algorithms we use in this paper before moving on to describe the problem, get results, and compare them using various means. In this paper we are predicting lead time, which is a real valued function, hence we use only regression algorithms to predict the lead time. The algorithms we are going to use in this paper are- Linear Regression, Ridge Regression, Lasso Regression, Decision Tree Regression, Random Forest Regression, and k-Nearest Neighbors Regression.

In multivariate Linear Regression, where the output variable is a function of more than one input variable, we try to fit a hyper-plane through these data points such that a cost function such as Mean Absolute Error (MEA), Mean Squared Error (MSE) or Root Mean Squared Error (RMSE) is minimized over all the data points. The equation of the hyper-plane can be found either using the normal equation, which finds the equation of hyper-plane at once but is highly sensitive to number of dimensions in training set as its time complexity varies from $O(n^{2.4})$ to $O(n^3)$, where n is the number of dimensions in the dataset, depending on the implementation. Or the equation of hyper-plane can also be found by using gradient descent algorithm, which after random initialization of model parameters, iteratively changes the values of model parameter such that at each steps, the reduction of cost or error function is maximum, and in the end the equation of hyper-plane is such that it best fits the data-sets. Alternatively other modified forms of gradient descent algorithm can be chosen to provide out-of-core support for large datasets, these include Stochastic Gradient Descent and Mini-Batch Gradient Descent Algorithms.

Other modifications of Linear Regressions are also available to prevent the model from over-fitting the dataset. These models penalize higher values of model parameters along with the error in predicted values, which prevents the model from having larger model parameters.

Ridge Regression is an instance of Linear Regression with regularization. It is used to overcome the problem of selecting wrong variable for the Linear Machine Learning model with multiple variables. It penalizes the size of regression coefficients to find the value of tuning parameters. It uses l_2 norm of weight vector, i.e, it adds a penalty of $\frac{1}{2}\ \alpha\ (\theta_i)^2$ for each training instance i in the training-set and α is the regularization parameter ranging from 0 to 1.

Lasso (Least absolute shrinkage and selection operator) is also a Linear Regression Technique that performs regularization on selected variables [9]. Its statistical analysis is similar to that of Ridge Regression, except, it considers the absolute values of sum of regression coefficients, i.e it uses l_1 norm of weight vector, and adds a penalty of $\alpha\ |\theta_i|$ for each training instance I in the training-set where α is the regularization parameter. Lasso Regression automatically performs the feature selection and is preferred when there are only a few useful features in the training set as it sets the model parameters for the less useful features to 0. When the value of regularization parameter α is set to 0, both Lasso and Ridge Regression models become Linear Regression models.

Decision Tree can be simply defined as a disjunction of conjunctions of constraints on the attribute value of instances [10]. Decision Trees were originally designed for Classification tasks only, however many modifications of original algorithm are now available, which can be used for Regression tasks also [11]. There are various methods of constructing a regression tree, the most common being recursive partitioning. The implementation of Decision Trees used is CART (Classification and Regression Trees). If the Decision tree to be constructed is a regression tree as required, then at each node the tree is split into leaf nodes such that the error, as defined by error function, is reduced over the leaf nodes. This process is repeated iteratively until the desired accuracy is achieved.

Random Forest Regression is a meta estimator that fits a number of classification or regression decision trees based on various subsamples of the given dataset and it improves predictive accuracy and controls over-fitting by using averaging [12]. Random Forest Regression models like other Ensemble methods, work on the principle of Wisdom of Crowd, and if each individual tree or classifier is only slightly better than pure guessing, then a sufficiently large number of such classifiers will be collectively better than any single one of them. Further, to increase the efficiency even more, Soft-Voting can be used in place of Hard-Voting, which takes into consideration the "confidence" of each individual classifier when taking the average of the of the individual classifiers.

K-Nearest Neighbors Regression is a regression based on k-nearest neighbors algorithm, the value is found by taking an average or a weighted average of k- nearest neighbors, where neighbors are found by calculating the Euclidean distance between samples over all the features. In some modifications of this algorithm, we take average or weighted average of all samples within a given distance, i.e. average of all samples in a given radius from a sample.

III. Background

Just-in-time (JIT) is an idealized concept is Supply Chain Management (SCM) where no inventory needs to be managed and required inventory is supplied at the place at the required time. This has may benefits which include but are not limited to reduction in size of inventory, reduction in warehousing or storage cost, increased throughput and more efficiency. This system however places some requirements on the vendor, such as having sufficient capacity to supply anytime without passing cost of overcapacity to buyer. It also requires that the vendor takes into account the total lead time for the deliveries, which in itself is a function of various times, such as processing time, breakdown time, fixing times among others. Machine Learning and Artificial Intelligence can be applied to almost every field of Supply Chain, and has the capacity to drastically improve the supply chain, and help in creating new models of the Supply Chain which take into account uncertainty, and hence can be applied to real life, non deterministic environment. J. Mula et al. reviewed the various applications of Artificial Intelligence in Supply Chain Management, such as in aggregate planning, material requirement planning, manufacturing resource planning, inventory management and supply chain planning [13]. In this paper, we study a particular case of the JIT system, which are the restaurants. Restaurants supply the required order to the customer when and where needed by them. Delivering order to their customers on time is perhaps one of the foremost objectives of restaurants. In order to achieve this, they must take into account the order lead times too, and should be able to predict it accurately. They must also identify the features on which this order lead time depends, and use them to train a model for predicting it accurately.

IV. Research Methodology

Before fitting the data into a Machine Learning model, we need to gather more information about the data. This includes the information related to the description of raw data, and also the statistical information about the data. We need to fill null or empty values with suitable values such as mean or median of the data. Visualization of the data needs to be done if required, by making different scatter plots, histograms etc, to see a pattern in the data. To avoid curse of dimensionality, we select only features that have high correlation with the target variable. To find the correlation between the target value, i.e., the order lead time and features by comparing values of Pearson, Kendall and Spearman coefficient of correlation. We select the features that have highest correlation to target variable and fit them to various Machine Learning model. Then we use this model to make predictions on test set, calculate various errors on the results obtained and compare the results obtained by various models.

A. Data Description

TABLE I. Raw Data Information

Feature	Data-type
Customer ID	Number (Integer)
First Time	Time-stamp
Recent Time	Time-stamp
No. of Orders	Number (Integer)
No. of Orders in last 7 days	Number (Integer)
No. of Orders in last 4 weeks	Number (Integer)
Amount	Number (Integer)
Amount in last 7 days	Number (Integer)
Amount in last 4 weeks	Number (Integer)
Average Distance from Restaurant	Number (Float)
Average Delivery Time	Number (Integer)

The Table shows various features of raw data and their data-types.

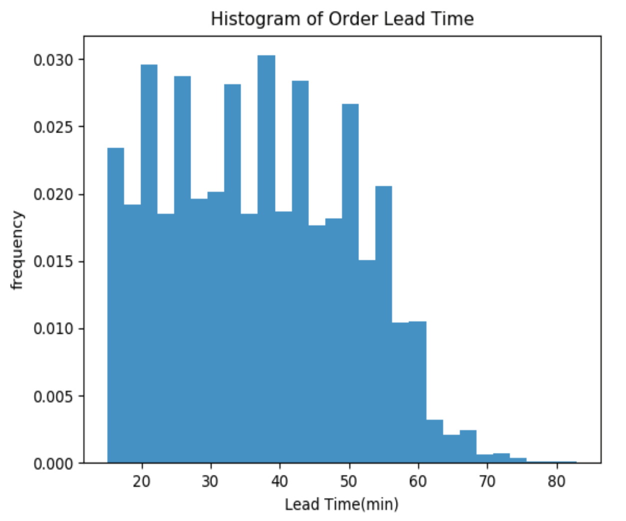

Fig. 1. Histogram plot of frequency vs Lead Time (in minutes).

TABLE II. Correlation Measure

Feature	Pearson's Coefficient of Correlation
Customer ID	-0.0293
No. of Orders	0.0040
No. of Orders in last 7 days	-0.0044
No. of Orders in last 4 weeks	0.0185
Amount	0.0027
Amount in last 7 days	-0.0010
Amount in last 4 weeks	-0.0156
Average Distance from Restaurant	0.0886
Average Delivery Time	1.0000

The table shows Pearson's coefficient of correlation of different attributes relative to the target attribute- average delivery time.

B. Finding Correlation between Target attribute and other features

Correlation measures how much a specific attribute depends on other attribute. Often we use Pearson's coefficient of correlation to measure the linear correlation between different attributes. The value of correlation coefficient varies from -1 to 1. A value of 1 indicates a perfect positive correlation, i.e., as one attribute increases, the second attribute increases. Conversely, a value of -1 indicates a perfect negative correlation, i.e., as value of one attribute increases, the value of other attribute decreases. A value of 0 indicates no linear correlation between the two attributes. However, there may still be a non- linear correlation, which can be measured with the help of Kendall's coefficient of correlation and Spearson's coefficient of correlation.

Only those attributes with maximum correlation with the target feature are selected for fitting into the model.

C. Prediction and Evaluation of ML Models

We fit the data to various machine learning algorithms, and the predictions are compared with true value, and errors are measured with various techniques. The various techniques used for measuring errors are Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).

Values of various parameters such as selection of value of k in k-NN and selection of maximum no. of leaves in decision tree were chosen by selecting the most optimal value of the parameter that minimizes the errors. The following results were obtained after evaluating different models on the test set and measuring errors from different techniques.

TABLE III. ML ALGORITHMS EFFICIENCY

ML Algorithms	**Method of Error Measurement**			
	MEA	***MSE***	***RMSE***	***MAPE***
Linear Regression	11.292	175.259	13.238	38.126
Ridge Regression	11.293	175.259	13.238	38.126
Lasso Regression	11.322	176.569	13.287	38.211
Decision Tree Regression	11.311	175.961	13.265	38.162
Random Forest Regression	11.293	175.374	13.242	38.062
k-Nearest Neighbors Regression	11.341	176.986	13.303	38.310

The table compares efficiency of different Machine Learning Algorithms on various error measures.

V. CONCLUDING REMARKS AND FUTURE RESEARCH

In this research paper, the authors did a case study on predicting the lead time for a Just-in-time manufacturing environment. The results obtained by various machine learning model were presented and compared using several methods. Machine Learning models are able to predict order lead time up to a great extent. However, this paper does not take into account various conditions such as traffic, and other factors that change in real time and can have a great impact on order lead times in a Just-in-time environment. Future research shall take into account these factors that change in real time, and use other state of the art technologies such as IoT (Internet of Things) to make predictions even better. Machine Learning predictions depend on the size of the data-set, hence a higher dataset will also be used to improve predictions give by the models.

VI. ACKNOWLEDGMENT

The authors of this paper would like to thank the Institute for their continued support, Netaji Subhas University of Technology (NSUT), New Delhi. This work has been done under the department of MPAE, NSUT. The authors would also like to thank the Department of MPAE, NSUT for their continued support.

REFERENCES

[1] Kelleher, J., Mac Namee, B., D'Arcy, A.. Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies. MIT Press: 2015.

[2] L. Cory, "Just-In-Time approach to IC fabrication," Solid State Technology, pp. 177-179, May 1986.

[3] E.S Buffa and R.K Sarin, Modern Production/Operations Management. New York, NY: John Wiley & Sons, 1987.

[4] D.J. Lu, Kanban: Just-In-Time at Toyota. Stamford, CT: Productivity Press, 1985.

[5] R. J. Schonberger, Japanese Manufacturing Techniques: Nine Hidden Lessons in Simplicity. New York, NY: The Free Press, 1982.

[6] M. E. Levitt and J. A. Abraham, "Just-In-Time Methods for Semiconductor Manufacturing", SEMI Advanced Semiconductor Manufacturing Conference, 1990.

[7] M. Banko and E. Brill, Scaling to very very Large Corpora for Natural Language Disambiguation. ACL '01 Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, pp. 26-33, July 2001.

[8] Peter Norvig et al. , The Unreasonable effectiveness of data, IEEE Intelligent Systems, Vol. 24 Issue 2, March 2009, Pages 8-12.

[9] R. Tibshirani, Regression Shrinkage and Selection via the Lasso, Journal of the Royal Statistical Society, pp. 267-288, 1996.

[10] Tom M. Mitchell, Machine Learning, McGraw-Hill Series in Computer Science, pp. 53.

[11] L. Breiman, J.Friedman, R.Olshen, and C.Stone, "Classification and Regression Trees", Wadsworth, Belmont, CA, 1984.

[12] L. Brieman, "Random Forests", Machine Learning, 45(1),5-32,2001.

[13] J.Mula et al. , Models for production planning under uncertainity: A review, International Journal of production economics, 103 (2006) 271-285.

[14] G. van Rossum and F.L.Drake (eds), Python Reference Manual, PythonLabs, Virginia, USA, 2001. Available at http://www.python.org

[15] D. Ascher, P.F. Dubois, K.Hinsen, J. Hugunin and T. Oliphant, Numerical Python, Lawrence Livermore National Laboratory, Livermore, California, USA, 2001. Available at http://www.pfdubois.com/numpy

[16] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[17] API design for machine learning software: experiences from the scikitlearn project, Buitinck et al., 2013.