Robust Supply Chains with Gradient Boosted Trees

Pradeep K. Mahato
Department of Computer Science
The University of British Columbia
BC, Canada
pradeep.mahato@alumni.ubc.ca

Apurva Narayan
Department of Computer Science
The University of British Columbia
BC, Canada
apurva.narayan@ubc.ca

***Abstract*—Supply chain networks often experience various internal and external events that lead to shipment failures. Despite advancements in various machine learning models, the problem of avoiding service level failures remains intricate and hard to solve. While multiple attempts have been made by various researchers to make supply chains resilient, this is still an open problem. Moreover, explainability in the field of machine learning is a challenging task that assists in decision formation along with transparency.**

We develop a machine learning pipeline with gradient boosted decision trees to mitigate service level failures in supply chains. Our framework is simple, easy to implement, and provides a promising result. It provides explainability to prevent service level failure in time sensitive supply chains such as food manufacturing. Our model can be used for rapid deployment with state-of-the-art prediction accuracy while establishing trust within the decision-makers.

***Index Terms*—Supply chain management, machine learning, explainable AI, gradient boosting decision tree (GBDT), light gradient boosting machine (LightGBM), extreme gradient boosting (XGBoost)**

I. Introduction

Forecasting supply and demand is a common problem in supply chain optimization. In 2001, Nike ousted its $400 million demand forecasting software due to various prediction issues [1] while in 2014, Walgreen, a famous drugstore, replaced their existing Chief Financial Officer due to forecasting error to the tune of $ 1 billion [2]. Good forecasting of demand or sales amount plays a crucial role in the supply chain network. It reduces excess inventory while streamlining the manufacturing processes. While any future prediction is never perfect, a good forecast helps avoid the overestimation of inventory and also checks underestimation leading to sales cut. Forecasting errors occur primarily due to two main reasons. Firstly, drastic deviation in demand, transportation time or production problems, and secondly due to poor integration, lack of understanding between individual features and weak coordination.

Supply chain is a complex network involving various independent modules [3], [4]. Breaking the entire commerce problem into multiple components helps scale production but it also introduces the problem arising due to lack of coordination. Even a minor misalignment results in fatal sales error. Perishable commodities are affected the most due to such slips.

Particularly, forecasting in a supply chain network is a very challenging problem and multiple attempts were made in the recent past. Especially, in the food industry overestimation leads to resource wastage primarily for products with low shelf-lives while underestimation leads to a shortage in supply leading to an increase in the illegal high sales price. In [5], Papadopoulos et al. stressed the importance of a sustainable model for operations management which balances the consequences of economic and environmental factors. With the recent advances in the field of Machine Learning (ML) and the growing availability of data, the supply chain problem has become a leading problem to solve in operations research. [6], [7] studied on the effect and mitigation of circumstances leading to disruption in the supply chain network and [8], [9] examined multiple demand forecasting models and its associated risk analysis.

In this work, we present a model that uses decision trees to predict future order completion while considering major factors and their interactions. We attempt to provide prediction with model explainability and magnitude of failure. Our model takes sales data upon which we perform multiple ML techniques to forecast sales value and provide detailed insight into the role of each feature for such a prediction. This insight helps industries to act on actionable issues immediately to improve their sales in the future.

We begin by describing the vital algorithms used in our model followed by the sequence of their application. The major contribution of our work can be summarized as:

- Develop a decision tree based efficient and lightweight sales forecasting model
- Provide actionable insights associated with the decisions
- Evaluate and validate our results on the industry dataset

The remaining of the paper is organized as follow: Section II provides an overview of relevant research, Section III provides necessary preliminaries for understanding our model, Section IV describes our methodology. Section V presents experiments and evaluation of our approach and finally, we provide our conclusion and future work in Section VI.

II. Literature Review

Supply chain network has myriad of problems [4] and most of these difficulties are interlinked thereby turning even

978-1-7281-2547-3/20/$31.00 ©2020 IEEE

the simplest and small-scaled implementation can be quite challenging in the long run. Various industries and research organizations perform numerous studies to overcome such challenges. Kaggle, an online competition platform for machine learning applications has seen a significant rise in contribution to similar problems and their solutions. While forecasting is quite an established research area in itself, effective planning and mitigation of existing and expected problems are also gaining popularity. Zage et al. in [10] investigated the potential cause of risk in such networks. Using spectral analysis on big data traces from the e-commerce domain, they attempt towards identifying fraudulent vendors. Initially, using a fraudulent-accomplice strategy, they establish a graph-based network that is fed into a semi-supervised clustering algorithm eventually labeling an anomalous behavior as that of a fraudulent vendor. Similarly, Fan et al. in [11] describe a Supply Chain Risk Management (SCRM) framework which continuously monitors system performance and reports any emerging risk. Such a risk detection framework helps to initiate immediate and speedy planning actions. However the above-mentioned work is mostly theoretical and provides system understanding for developing a similar setup. George Baryannis et al. in [12] describes the importance prediction based on model interpretability. They define a generic model for a selective predictive framework which emphasizes on individual feature importance. Using their framework they draw a comparison between performance and model explainability and its trade-off. Bruzzone and Orsoni in [13] were among the very few to incorporate Artificial Neutral Network for understanding the trade-off involved in different areas of network design. They demonstrated the correlation between logistic variables and supply chain costs as well as showed the effectiveness and flexibility of their model.

Eventually, a number of studies were performed in understanding the risk factors involved in the decision process. Papadopoulos.et al. [5] underlines the role of an abundance of data towards increasing system resilience and the significance of quality information, swift trust, and public-private partnership for enhancing toughness. Paul et al. [14] and Schmitt [15] provided multiple solutions visualizing and mitigating such risky situations.

The aforementioned works attempts towards addressing different problems occurring in SCN, whereas our work provides a completely different alternative in solving sales attainability prediction problem and understanding the factors involved in a poor outcome. We provide a framework which is an assimilation of various popular ML techniques in tandem offering reliable and actionable decision. Our framework is modular and researchers can exploit numerous ML frameworks using a similar architecture.

III. Background

In this section, we primarily discuss few basic ML methods used in our model.

A. Gradient Boosted Decision Trees

Gradient Boosted Decision Trees [16] (GBDT) is a decision support tool that uses an ensemble of decision trees built and optimized using boosting. Moreover utilizing gradient descent, GBDT converges toward the best possible ensemble of decision trees that are used to predict for a possible consequence. Decision trees are tree-like data structures with non-leaf nodes comprising of threshold values while leaf nodes contain a predicted outcome. Decision trees can be used for both regressions as well as classification examples. In the case of classification, GBDT uses log-likelihood to predict the class label.

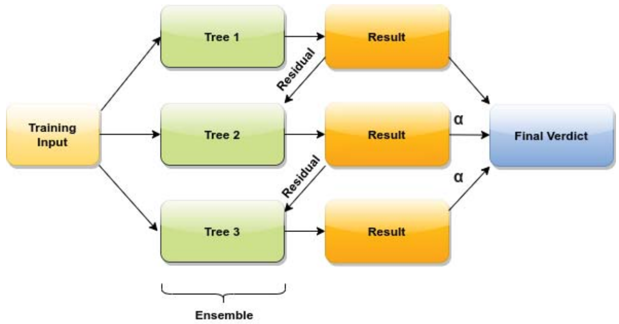


Fig. 1. Sample illustration of GBDT algorithm with 3 trees

Figure 1 provides an abstract understanding of the GBDT algorithm. Each iteration churns out an optimized decision tree and the residual error caused by this tree is improved upon while building the next decision tree. These set of decision trees together forms the core for GBDT. Each iteration walks closer toward finding a local minimum. Local minima (or minimum) is a point in the iteration chain that produces the least residual error. The learning rate (alpha) is used to scope the effect of each decision tree‘s output. Smaller the learning rate, the slower the objective function converges. Iteration steps depend on either the number of decision trees required or until no optimal set of trees could be formed henceforth. The final prediction uses these ensembles of trees to form a prediction for a given test input. Due to the non-continuous nature of the objective function, GBDT‘s are resilient towards an unbalanced dataset. This was the prime reason for selecting GBDT as the core of our ML model.

B. Correlation

Correlation (or covariance) function determines the similarity between two entities [17]. Correlation function helps in determining how do features interact among themselves. In practice, correlation helps determine interaction among the features and is often recommended to filter off trivial features thus helping reduce dataset dimensionality. We used correlation among features to help reduce our dataset dimension thereby improving the training duration.

Autocorrelation (ACF) [18] and Partial-autocorrelation (PACF) [18] are two predominant techniques often used in time series analysis. ACF is a mathematical tool for finding recurrent similarity which is often hidden due to the presence of noise or other factors. ACF helps reveal periodic similarities or correlations over the entire time-domain thus assisting in selecting a suitable time lag window. Moreover, randomness in data could be identified using ACF and if the time-series is found to be periodic then it helps in selecting the best time-series model.

While PACF is also an auto-correlation function but its not complete since it removes the existing correlation before comparing for next lag. PACF differs from ACF as it finds the residuals, which survives even after removing earlier correlations before comparing it with the subsequent lags. PACF is important as it helps discover hidden information in the time-domain which ACF fails to detect.

In general, performing correlation analysis among the features followed by making ACF and PACF analysis helps reveal much vital information hidden in a simple time-series dataset. It is often better to reduce features that are highly correlated among themselves as it helps avoid multicollinearity issues. Since our use-case involves time-series prediction thus employing these techniques in primary phases benefits our model as discussed in the later section.

C. Model Explainability

In recent days, ML models have advanced quite impressively due to the high availability of data and huge computing power. Nevertheless, if the model cannot be explained well then even a fancy AI model could cause some serious problems. In critical times if researchers and developers cannot troubleshoot their model‘s performance then it would lead to critical failures. This is where the importance of model explainability comes into play.

In 1953, Shapley [19] values were introduced to solve such a problem and they are largely inspired by the question “Can payoffs be justified and fairly distributed in a coalition ? ”. Shapley values, a method from coalition game theory, helps to determine the importance of each feature towards the model. For example, consider a team of 3 members namely X, Y, and Z are involved in the completion of a project. Furthermore, assume all three members have not joined the team at the same time. In such a case, one simple division of payoffs could be to consider the duration of work done by each individual. But, most of the time, such calculation is not justified.

In game-theory involving coalition, such complications could be resolved reasonably well [20]–[22]. While the maths involved in doing so is quite complex but in simple words, it could be understood as taking the average contribution of all possible sequences of developers joining to project. For example, if Y joins the project after X then the contribution of Y could be calculated by considering the amount of work left at the time of joining along with the overlapping skills with Z, Y could have brought in before Z joined. Ideally, such calculation is quite time-consuming as all combinations must be considered before concluding. Moreover, using the concept of determining justified payoff for each contribution, their importance towards the coalition could also be determined [20].

Shapley value of a feature value towards its contribution could be expressed as (adapted from [21]) :

$$\varphi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\,(p-|S|-1)!}{p!}(v(S \cup \{i\}) - v(S)) \quad (1)$$

where, S is the subset of features used, N is the set of feature vector, p is the number of features and v is the prediction of feature value in set S.

To the best of our knowledge, to date Shapley value is the only mathematical expression which defines the three axioms of interpretability:

- **Dummy player**: If the feature does not have any role towards the model‘s prediction then its contribution should be 0
- **Substitutability**: If the over two features add the same marginal value then they possess substitutability.
- **Additivity**: The sum of all contributions in the individual subset should match the overall contribution towards the entire set.

Overall, the better the model could be explained the stronger the confidence arises in the model. In [22], Lundberg et al. had recently proposed a model that uses an approximation of Shapley values for a model. In Section 4, we show the use of his framework to improve our model‘s explainability.

IV. Methodology

Figure 2 provides a high-level view for our prediction model. In this section, we explain each phase involved in our model along with examples.

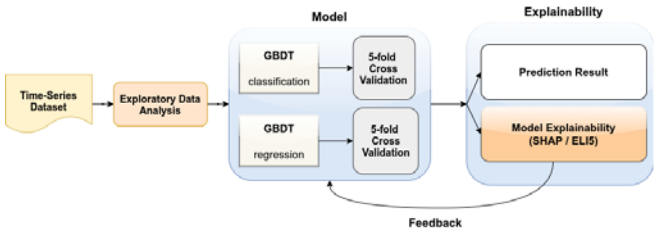


Fig. 2. Model-flow

As in all machine learning practices, the first step starts with cleaning, pruning, and modifying our dataset. We aggregated daily entries into a weekly summary for each category. In general, the supply chain network involves various subcomponents. Figure 3 shows the common blocks involved in the entire chain. The network begins with order forecasts based on historical requests. Since request forecast is a common problem and multiple research attempts have been made, we majorly focus on fulfillment prediction. Along with forecasted value, the current requests must also be considered for future time-frame. It must be noted that these requests could be canceled a few days before the production phase thus forecast model should accommodate such behavior as best as possible.

Moreover, we restrict the time-horizon to 2 weeks from present-day since the accuracy of prediction decreases the further one moves in the future. We prepare a consolidated record with primary columns such as material, plant, calendar week, forecast, order-request, order-shipped, and production.

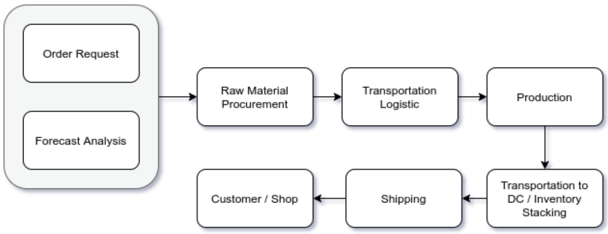


Fig. 3. Supply Chain Network

Since our dataset primarily involved time-series distribution, we proceeded with basic exploratory analysis. For all the features, we introduced a time lag of 5 weeks. We then discarded the entries which contained *NA* values and performed a cross-correlation analysis as well as ACF and PACF to determine individual feature lags. The primary objective of such exercise was to uncover hidden relationships within and among the features. This also provides an overview of the rough time-drag one must consider before applying advanced ML techniques.

Once the adequate time-drag is determined, we re-framed our dataset with individual features containing its relevant drag as well as dropping a few features which displayed high correlation among themselves. We also added columns for request fulfillment on a per week basis. This allowed us to apply binary classification as we had two classes. In actuality, such datasets are generally not well balanced. Synthetic Minority Oversampling Technique or SMOTE [23] is a popular data augmentation technique often used for addressing imbalanced classification. We used SMOTE to augment minority classes leading to an (almost) balanced dataset.

GBDT are among few models which handle imbalanced dataset quite well and also provide good explainability since it uses decision trees at its core. Apart from providing a prediction on the order fulfillment, we also wanted to predict the magnitude by which shipment could fall short off. We used 2 variants of GBDT with first for classification while the later meant for regression. As shown in figure 2, each model was treated with a k-fold cross-validation step. The parameter *k* along with all other hyper-parameters were determined using Grid-Search CV. We used both models to reach a decision. Moreover, most GBDT model provides prediction probability signifying the strength of the decision. Since both, the models perform independently, their outcome would certainly not be in sync with each other.

The problem at hand is basically to predict the order fulfillment in a given time horizon and in case of failure to determine the quantity with which would the industry be falling short off. Algorithm 1 shows our model selection procedure. We begin with our prediction dataframe, $X_{\text{prediction}}$ and the trained models for classification ($\text{model}_{\text{classification}}$) as well as for regression ($\text{model}_{\text{regression}}$). The algorithm iterates over each record in $X_{\text{prediction}}$ to predict the reachability of the sales order. A predetermined threshold (*threshold*) and tolerance margin (*margin*) is fixed before the prediction stage. Please note, these variables could be changed after model training thus providing flexibility to the analyst over their decision-making process. The viability of the order, (denoted as *class*) is determined using $\text{model}_{\text{classification}}$ along with its probability, *prob* while $\text{model}_{\text{regression}}$ provides the *magnitude* of the predicted shipment. If *prob* is less than the provided *threshold* or the shortfall of shipment (denoted as *magnitude*) is less than *margin* but *class* is labelled as *fullfill*, then we adjust the *prob* and flip the label. This is done for the entire $X_{\text{prediction}}$. As a result, with a reasonably good estimate of shipping prediction and magnitude, our framework helps analysts and sales representatives to understand the gravity of a problem and take timely actions.

Algorithm 1: Predict order fulfillment for future time-horizon

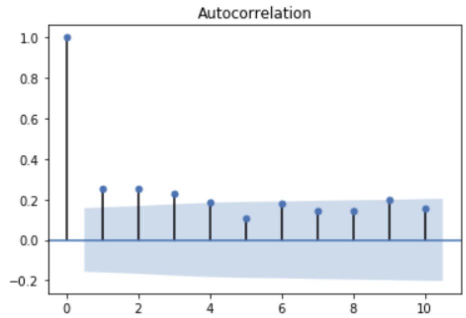
Input: $X_{prediction}, model_{classification}, model_{regression}$
Result: Prediction frame with magnitude and probability

foreach *record* $\in$ *dataframe* **do**
  magnitude $\leftarrow$ $\text{model}_{regression}$
  prob $\leftarrow$ $\text{model}_{classification}$
  class $\leftarrow$ $\text{model}_{classification}$
  if *prob* $\leq$ *threshold* & *class* = fullfill **then**
    prob $\leftarrow$ *100 - prob*
    class $\leftarrow$ not-fullfill
  else if *class* = fullfill & *magnitude* $\leq margin$ **then**
    prob $\leftarrow$ *model*$_{regression}$
    class $\leftarrow$ not-fullfill
  end

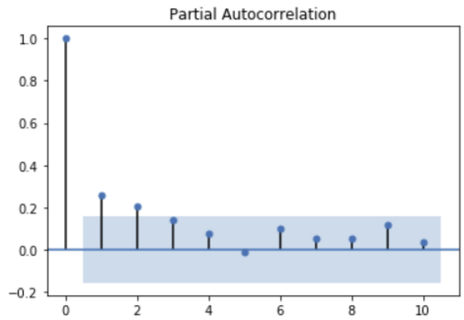
Frameworks such as SHAP [22], ELI5 [24], Plotly [25] are used to understand the black-box model. SHAP [22] framework provides a summary plot and dependency graph for each feature thus describing their contribution towards model prediction. For each specific decision being made, the SHAP framework provides individual feature contributions along with their impact, either positively or negatively towards the final verdict. ELIF5 takes a step further by magnifying granular details as well as returning a dataframe. Plotly renders this dataframe into much simpler waterfall representation. With the insight of such frameworks, one could perform a better and more reliable feature engineering across the entire structure. Finally, we exported our results in various formats for easier and flexible dashboard integration.

V. Evaluation

In this section, we demonstrate the application of our model in a real-world industry dataset. We applied our model on the historical record for sales and delivery for RICH's food products[1]. The implementation was done using Jupyter Notebook. Due to privacy issues, we would not provide the dataset but rather will present the experimental results and conclusion on the same. However, our approach is generic and can be applied in different circumstances.



(a) Order time series



(b) Forecast time series

Fig. 4. Auto Correlation and Partial Auto Correlation

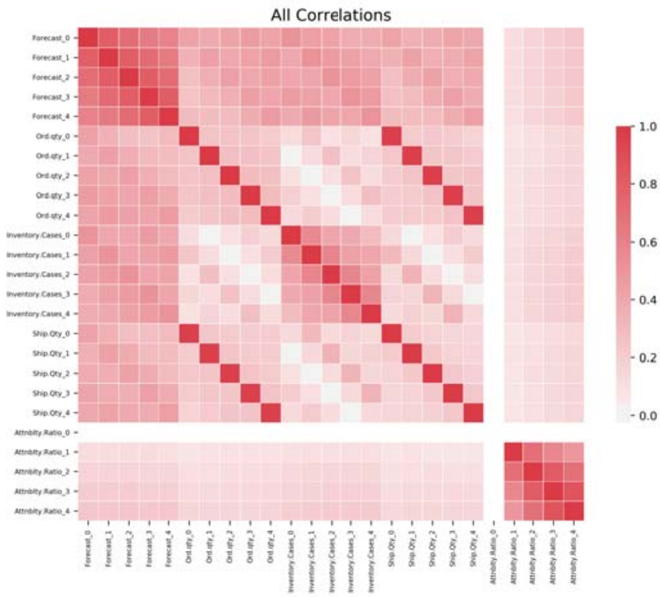


Fig. 5. Cross Correlation among features

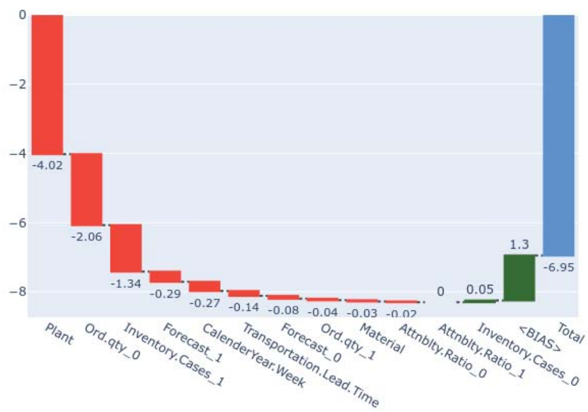


Fig. 6. Overall contribution towards attainability prediction for a product in future

As mentioned in section IV, data cleansing and pruning were followed by exploratory analysis. Our dataset is associated with the supply chain network for food products but a similar approach and understanding could be applied to varied use cases as well. Figure 4 and Figure 4 shows the interaction within the features. It provides an overview of ACF and PACF performed on order and forecast quantity respectively. Moreover, it was evident that a time drag of 2 weeks carried most of the hidden information. This was further apparent when cross-correlation was performed across all features with a time lag of 4 weeks which is presented in Figure 5.

We used ELI5[2] along with Plotly[3] to explain for any possible sales failure. Figure 6 explains our prediction for the shipment viability of a product in the future. We predicted the possibility to meet the order demand and explained the negative factors in the red bar with their possible magnitude of impact. Ignoring the significant contribution done by the production plant, we found the Ord.qty_0 (Order quantity for the said week) had the second most detrimental effect along with the low Inventory.Cases_1 (Inventory cases for the previous week) and Forecast_1 (Forecast projection for the previous week). While Inventory.Cases_0 (Inventory stock for the said week) has a weak favorable role towards the aggregated negative projection. With further analysis, it was found that the order quantity was significantly high while the inventory stock and order forecast, estimated the previous week, were pretty low. This was the prime reason for such low prediction due to possibly a risky future order-shipment target. With such insight, analysts and sales managers could immediately ramp up their production, therefore, easing out

[1]https://www.richs.com/
[2]https://eli5.readthedocs.io/
[3]https://plotly.com/

the negative factors in play. However it must be noted, in the supply chain industry, such production ramp-up must be given with reasonable heads-up to account for the time taken in raw material procurement and transportation among various production and distribution centers.

		PREDICTED LABEL					
		LightGBM		XGBoost		CatBoost	
		Fail	Pass	Fail	Pass	Fail	Pass
TRUE	Fail	**4460**	428	4209	679	4345	543
LABEL	Pass	467	**4319**	1007	3779	737	4049

TABLE I
CONFUSION MATRIX

We evaluated our model using LightGBM [26], XGBoost [27], and CatBoost [28]. 5-fold cross-validation (CV) with grid-search was used for parameter tuning in both classifications as well as regression activities. With the intention to understand the splitting ratio as well as to tune the hyperparameter, we performed a basic grid search cross-validation exercise with the folds set to 5. It must be noted that we performed these exercises to obtain the optimal parameter, however, given the fact that these exercises are both time and compute-intensive, such parameter tuning was performed during the weekends which are comparatively less occupied. We split our dataset into a 60:20:20 ratio for training, validation, and testing respectively. Since the dataset was severely imbalanced with a positive-negative sample proportion of 99:1 thereby we employed SMOTE. Minority balancing was performed leading to a 67:33 positive-negative sample distribution.

	F_1	ROC	Precision	Recall
LightGBM	**0.91**	**0.91**	**0.71**	**0.95**
XGBoost	0.82	0.83	0.70	0.94
CatBoost	0.86	0.87	0.69	0.92

TABLE II
F1 AND AUC SCORE

Table I shows the confusion matrix for GBDT, XGBoost, and CatBoost during the classification phase. LightGBM outperformed and it was followed by CatBoost while XGBoost performed the worst. XGBoost‘s classification contained a significant amount of false positives. LightGBM held the best F_1 score as well as the AUC value. CatBoost came pretty close while XGBoost performed the worst. Such behavior often is due to the missing categorical feature evaluation in the XGBoost framework which is present in both LightGBM as well as CatBoost. While LabelEncoder and one-hot encoding does provide a workaround to the problem but LightGBM handles categorical features much better [29].

We experimented using various boosting types provided by LightGBM. The results in the table III demonstrates the strength of Gradient Boosted Decision Trees (GBDT). Dropouts Meet Additive Trees (DART) came pretty close to GBDT in terms of F1 and AUC score while Random Forest (RF) performed the worst. Please note, we avoided

	F_1	ROC	Precision	Recall
GBDT	**0.91**	**0.91**	**0.89**	**0.91**
RF	0.68	0.76	0.99	0.51
DART	0.89	0.89	0.88	0.89

TABLE III
F1 AND AUC SCORE FOR VARIOUS OBJECTIVES IN LIGHTGBM

using Gradient-based One-Side Sampling (GOSS) since it lacks a stable convergence during optimization for every execution. GBDT convergences pretty fast with significantly higher evaluation scores while model training with DART takes considerably longer time which made us choose GBDT as the default booting type.

Precision and recall are among the most important parameters for the benchmark. While precision provides the accuracy of the result, recall reveals the percentage coverage for the identified classes over the entire dataset. Figure 7 and table II draws a comparison for both these metrics on all the three models.

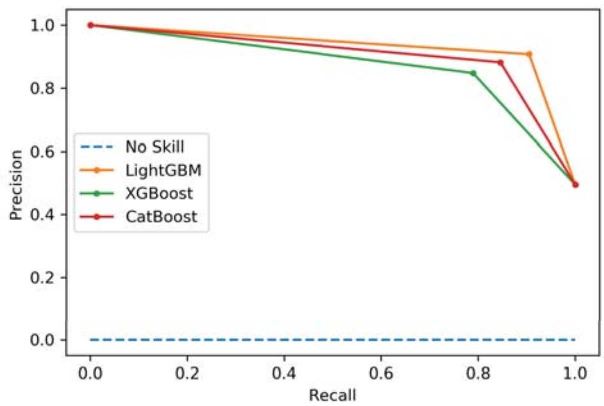


Fig. 7. Precision and Recall

Interestingly, XGBoost came pretty close to LightGBM while CatBoost performed the worst among the three in terms of precision. But it must be noted that fine-tuning XGBoost takes a considerable amount of time and memory while CatBoost is fairly simple in terms of using an optimal hyperparameter selection. LightGBM provides a balance between these two tools and often provides the best return with the least amount of time invested.

While these metrics and evaluations do provide developers ease of implementing decision tree models, it does require a careful understanding of hyperparameter tuning. We found LightGBM with GBDT boosting type to be the simplest model to implement. With the benefit of categorical feature integration baked right into its framework, tuning and testing with LightGBM is significantly easy for beginners along with being efficient. Nevertheless, with significant time invested in parameter optimization, XGBoost often leads the race but also comes at a steep cost of the compute usage. Technology grows exponentially faster. Even the most well-established companies must adapt to a high return-on-investment (ROI) model to survive such cut-throat race and LightGBM provides the best bang for bucks with best results in relatively less time.

VI. Conclusion

This paper presents an approach towards strategically solving a critical problem in the supply chain network. We explained the most common problem faced by data analysts involved in the decision-making process for any supply chain network and provided a possible solution to help ease the process. We presented our prediction and estimation system that builds confidence for easier adoption. Using a famous food industry dataset and various model explainability framework, we provided possible explanations that could help the user.

These experiments confirmed that Gradient Boosted Decision Trees using the LightGBM framework along with Shapley values provides explainability towards a decision being made using fairly basic model architecture. Such models, being relatively simple and easy to implement, saves time, and bolster vital decision-making processes. We believe our framework is broad, performs within acceptable error margin, and assists various other decision-making problems. However, continued efforts are needed to explainable decision-making models more robust and complete.

We further aim to expand the explainability of our model down to its granular detail along with continuous learning. The dynamic adapting capability of a model would increase its resilience and could be a fruitful area for further work. Such a resilient model can perform better towards variable product trends, seasonality shifts, and unforeseen disruptions.

Acknowledgment

This research was supported by MITACS grant (Project Number IT16503) and was performed jointly in collaboration with Rich Products Corporation[4]. The author would like to thank Curtis Boyle and Benjamin Ott for their continuous support during the tenure of the project.

References

[1] C. Koch, "Nike rebounds: How (and why) nike recovered from its supply chain disaster," 2004.

[2] S. S. Joyjeet Das, "Walgreen cfo's departure due to $1 bln forecasting error - wsj," 2014.

[3] V. Mani, C. Delgado, B. Hazen, and P. Patel, "Mitigating supply chain risk via sustainability using big data analytics: Evidence from the manufacturing supply chain," *Sustainability (Basel, Switzerland)*, vol. 9, no. 4, p. 608, 2017.

[4] S. A. Melnyk, R. Narasimhan, and H. A. DeCampos, "Supply chain design: issues, challenges, frameworks and solutions," *International Journal of Production Research*, vol. 52, no. 7, pp. 1887–1896, 2014. [Online]. Available: https://doi.org/10.1080/00207543.2013.787175

[5] A. Gunasekaran, Z. Irani, and T. Papadopoulos, "Modelling and analysis of sustainable operations management: certain investigations for research and applications," *Journal of the Operational Research Society: Special Issue: Sustainable Operations Management: Design Modelling and Analysis*, vol. 65, no. 6, pp. 806–823, 2014.

[6] D. Simchi-Levi, W. Schmidt, Y. Wei, P. Y. Zhang, K. Combs, Y. Ge, O. Gusikhin, M. Sanders, and D. Zhang, "Identifying risks and mitigating disruptions in the automotive supply chain," *Interfaces (Providence)*, vol. 45, no. 5, pp. 375–390, 2015.

[7] J. He, F. Alavifard, D. Ivanov, and H. Jahani, "A real-option approach to mitigate disruption risk in the supply chain," *Omega (Oxford)*, vol. 88, pp. 133–149, 2019.

[8] S. Datta, D. P. Graham, N. Sagar, P. Doody, R. Slone, and O.-P. Hilmola, *Forecasting and Risk Analysis in Supply Chain Management: GARCH Proof of Concept*. London: Springer London, 2009, pp. 187–203. [Online]. Available: https://doi.org/10.1007/978-1-84882-634-2_10

[9] K. Button and Z. Chen, "Demand forecasting errors and the ownership of infrastructure," *Applied Economics Letters*, vol. 21, no. 7, pp. 494–496, 2014.

[10] D. Zage, K. Glass, and R. Colbaugh, "Improving supply chain security using big data." IEEE, 2013, pp. 254–259.

[11] Y. Fan, L. Heilig, and S. Voß, *Supply Chain Risk Management in the Era of Big Data*. Springer International Publishing, 2015.

[12] G. Baryannis, S. Dani, and G. Antoniou, "Predicting supply chain risks using machine learning: The trade-off between performance and interpretability," *Future Generation Computer Systems*, vol. 101, pp. 993 – 1004, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X19308003

[13] A. Bruzzone and A. Orsoni, "Ai and simulation-based techniques for the assessment of supply chain logistic performance." IEEE Computer Society, 2003, pp. 154–164.

[14] S. R. E. D. Paul, Sanjoy K., "A quantitative model for disruption mitigation in a supply chain," *European journal of operational research*, vol. 257, no. 3, pp. 881–895, 2017.

[15] A. J. Schmitt, "Strategies for customer service level protection under multi-echelon supply chain disruption risk," *Transportation Research Part B: Methodological*, vol. 45, no. 8, pp. 1266 – 1283, 2011, supply chain disruption and risk management. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0191261511000221

[16] C. Maklin, "Gradient boosting decision tree algorithm explained," 2019.

[17] A. Kumar, "Covariance and correlation," 2018. [Online]. Available: https://acadgild.com/blog/covariance-and-correlation

[18] J. Brownlee, "A gentle introduction to autocorrelation and partial autocorrelation," 2020. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-autocorrelation-partial-autocorrelation/

[19] L. S. Shapley, *A VALUE FOR n-PERSON GAMES*. Princeton University Press, 2016, p. 307.

[20] C. M. Wild, "One feature attribution method to (supposedly) rule them all: Shapley values," 2018.

[21] "Shapley value," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Shapley_value

[22] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[23] "Smote for imbalanced classification with python," 2020. [Online]. Available: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

[24] "Welcome to eli5's documentation!" 2020. [Online]. Available: https://eli5.readthedocs.io/en/latest/

[25] "Dash overview," 2020. [Online]. Available: https://plotly.com/dash/

[26] J. Zhang, D. Mucs, U. Norinder, and F. Svensson, "Lightgbm: An effective and scalable algorithm for prediction of chemical toxicity–application to the tox21 and mutagenicity data sets," p. 4150, 2019.

[27] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system." ACM, 2016, pp. 785–794.

[28] Y. Zhang, Z. Zhao, and J. Zheng, "Catboost: A new approach for estimating daily reference crop evapotranspiration in arid and semi-arid regions of northern china," *Journal of hydrology (Amsterdam)*, vol. 588, p. 125087, 2020.

[29] A. Swalin, "Catboost vs. light gbm vs. xgboost," 2018. [Online]. Available: https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db

[4] https://www.richs.com/