Predict Product Backorders

January 9, 2018

1 Introduction

1.1 Description of the Problem

Product backorders is a typical supply chain problem. Backordering happens when a customer places an order for a product that is temporarily out of stock with the vendor and order cannot be fulfilled. It is a dream for any business, but it is also a massive problem if we do not know how to handle it. In this project, the goal is to identify the cause of backorder and use the past data around the backorders to develop a model to predict the probability of a product backorder. With the help of data analysis, a reasonable prediction on all products can go on backorders is expected. Such a prediction could immensely help the client to plan for a more efficient stocking and backorder handling. Goals of this project are:

- Provide an overall insight from data using exploratory data analysis.
- Identifying what the main features are caused backorders the most?
- Predict the probability of a product backorder.

1.2 Preview of Data Analysis

The client is looking for ways to improve backorders handling. With the help of data analysis, a reasonable prediction on the products that can go on backorder is expected. Such a prediction could immensely help the client to plan for a more efficient stocking and backorder handling. Using the dataset, I want to answer to main questions:

1. How common is backorder?
2. What is the relationship of features with backorders?
3. Based on backorder risks what would be biggest risks?

Answers to these questions will enable me to identify the main causes of backorders and predict the probability of backorders.

2 Data Wrangling

In this project, I used the dataset available on the Kaggle website. The training data file contains the historical data for the eight weeks before the week we are trying to predict product backorders. The data took as weekly snapshots at the start of each week.

Dataset was acquired from https://www.kaggle.com/tiredgeek/predict-bo-trial. Dataset composed of one file named "Training_Dataset."The training file was opened and stored in a data frame using python. The dataset contains the historical data, and it has 23 columns and 1687860 entries, and entirely has 100894 missing data also some entries of two columns include -99 values.The missing data is an example of Missing at Random data mechanism where missing data is related to observed data.

Dataset columns are defined in Table 1:

SKU	Random ID for the product
national_inv	Current inventory level for the part
lead_time	Transit time for product
in_transit_qty	Amount of product in transit from source
forecast_3_month	Forecast sales for the next three months
forecast_6_month	Forecast sales for the next six months
forecast_9_month	Forecast sales for the next nine months
sales_1_month	Sales quantity for the prior one month period
sales_3_month	Sales quantity for the prior three month period
sales_6_month	Sales quantity for the prior six month period
sales_9_month	Sales quantity for the prior nine month period
min_bank	Minimum recommended amount of stock
potential_issue	Source issue for part identified
pieces_past_due	Parts overdue from source
perf_6_month_avg	Source performance for prior six month period
perf_9_month_avg	Source performance for prior nine month period
local_bo_qty	Amount of stock orders overdue
deck_risk	Part risk flag
oe_constraint	Part risk flag
ppap_risk	Part risk flag
stop_auto_buy	Part risk flag
rev_stop	Part risk flag
went_on_backorder	Product went on backorder (target value)

Table 1. Dataset columns

Data wrangling goals: Prepare the backorder dataset for EDA and Modeling
Tasks performed:

- Handling missing Data
- Convert to binary
- Handling the outliers

2.1 Convert to binary

In this dataset, every categorical feature includes only two values: 'Yes' and 'No' for reducing memory usage binaries converted from strings ('Yes' and 'No') to 1 and 0.

2.2 Missing Data

In this part I try to find which columns in the dataset contain missing values and drop or replace those missing values so I'll have tidy data. All columns in dataset footer had missing values and represented as NaN, so I dropped that row. Columns "perf_12_month_avg" and "perf_6_month_avg" have missing value as -99.There is a strong correlation between "perf_6_month_avg" and "perf_12_month_avg". So, linear regression would use to filing missing values. However another interesting point to note here is that many observations have both "perf_12_month_avg" and "perf_6_month_avg" as null, so linear regression cannot fill such values, and we need to see another approach there. Probably we would like to check for the central tendency of the data and replace the null accordingly. It is visible from the seaborne plot(Figure 1) that data was not distribute normally. Therefore picking median to fill remaining values is a good choice.

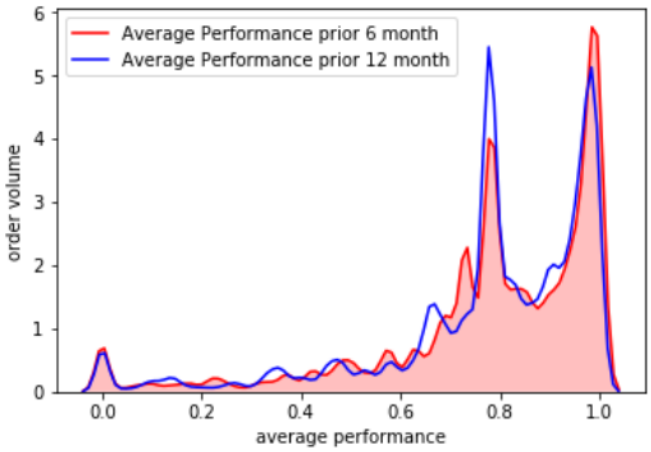

Figure 1. Source performance for prior six vs. twelve months

"Lead_time" column had 100893 missing values, and it was not clear if it was missing or not. It is quite likely that when "lead time" is missing, it is missing for a reason and not at random, which means a mean/median imputation strategy may not be appropriate. I preferred to decide by looking at data with calculating the proportion of backordered products vs. without a missing value in "lead time."

The calculation below shows how to handle missing data in "lead time":

1.Proportion of orders that "went_on_backorder" for missing "lead_time" records.

2.Proportion of orders that "went_on_backorder" for non-null "lead_time" records.

Went on backorder percentage for all orders that they "went on backorder" is 0.66%.

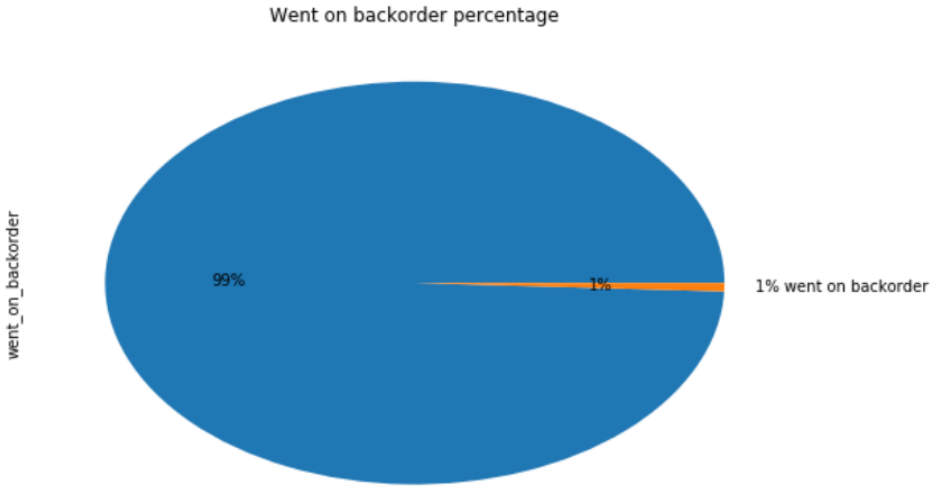

Figure 2. Went on backorder percentage

Based on the above proportion calculations the proportion of backordered products with missing "lead time" is 50% less than those without missing "lead time." The proportion of backordered products with missing "lead time" is half of the products with no missing values. The amount is significant enough that I decided not to replace the missing data in "lead time" and drop them.

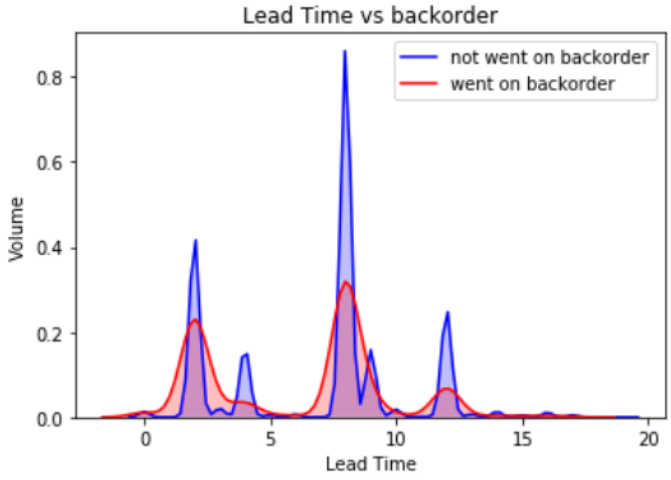

Figure 3. Lead time vs. went on backorder

2.3 Handling the outliers

The Next step is looking at the relationship between "lead time" and a fraction of products that went on backorder. Let's look at the "lead time" and how it changes the probability of went to backorder. Products that "went on backorder" with "lead time" eight weeks and then two weeks have the highest order volumes. The plot below shows the relation between "lead time" and the fraction of backorder.

The Figure 4 plot shows with longer "lead time" backorder proportion goes down. In the following Figure, two outliers noticed. One is at "lead time"=11 and one at "lead time" 52. For the point on 52, I believe there were not enough records to show the rest of point between 17 to 52. The point at "lead time" 11 should be given particular attention to its cause is known. For this reason, I am going to calculate the binomial probability distribution. As you see from the above calculations, the standard deviation of the binomial distribution is 3.23 standard deviation from the mean, so I am going to ignore this point for now.

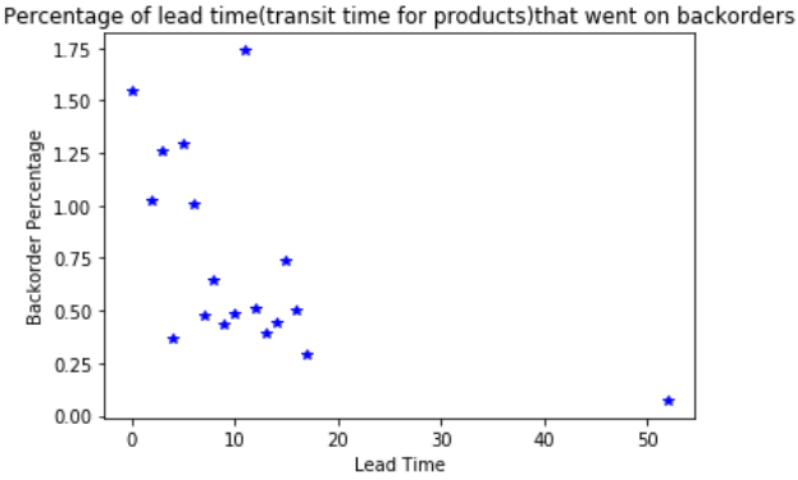

Figure 4. Lead time ratio

3 Explorotary Data Analysis

A list of questions about the data set were answered. There are 8 questions answered that aims to help achieve the goals of this project.

1. How common are backorders?
2. Given that, how likeliest are "backorders" based on the "part risk flags"? And how prevalent are they?
3. What's the relationship between product sales and forecast?
4. What's the relationship between "part risk flags" or are that unrelated concepts?
5. What's the relationship between "lead time" and "went on backorders"?
6. What aspects of supply chain represent the biggest risks?
7. Based on backorder risks what would be recommended improving first?

3.1 Relationship between total sales and total sales forecast

Figure 5 shows the strong relationship between total prior sales and total prior sales forecast. Product backorder can Happen because of the wrong forecast. For example, when for any reason sales forecast is less than actual sales of the month. In this data set from 11293 backordered

products, 4274 orders sales were more than sales forecast, so more than half of the backordered products happened because of the other reasons.

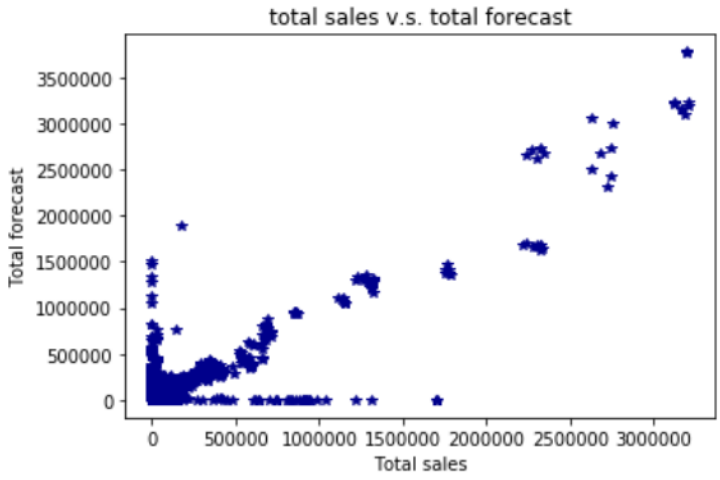

Figure 5. Total sales vs. Total forecast

3.2 Relationship between "lead time" and "went on backorders"

Figure 6 shows that "lead time" of backordered orders looks exactly like the plot of lead time with not backorder data. It means that most of the products not "went on backorder" and if we choose the random sample of data, it is the same distribution. Therefore we are going to see if "lead time" and "went on backorder" are dependent or independent to/from each other. Products that went on bacorder with lead time 8 weeks and then 2 weeks have the highest order volumes.

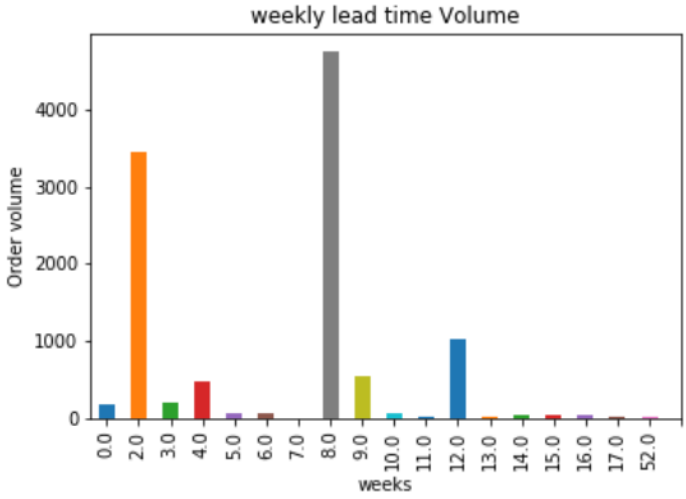

Figure 6. Lead time vs. went on backorder

3.3 "Total Sales" relationship with "Went on backorder"

Figure 7 shows with higher sales percentage of sales that went on backorder increased.

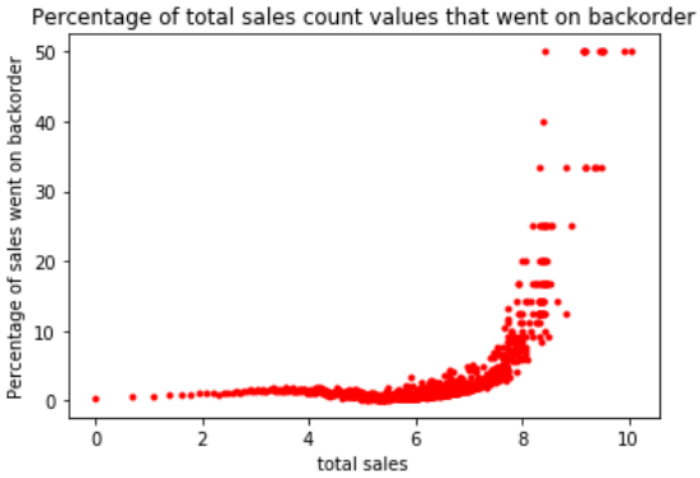

Figure 7. Total sales percentage that went on backorder

3.4 Relationship between categorical columns and "went on backorder"

The categorical columns in the dataset are:

- "potential_issue" - Source issue for part identified
- "pieces_past_due" - Parts overdue from source
- "local_bo_qty" - Amount of stock orders overdue
- "deck_risk" - Part risk flag
- "oe_constraint" - Part risk flag
- "ppap_risk" - Part risk flag
- "rev_stop" - Part risk flag
- "stop_auto_by" - Part risk flag

The Figure below shows that "lead_time" has the same relationship with "went on backorder" and "potential issues." It means when products with specific "lead time" did not have the "potential issue," the products did not go on backorder as well.

Same thing with parts overdue; there are no parts overdue from the source the products do not go backorder.The probability of products without any of risk that did not go on backorder is almost 98%. If the product did not have parts overdue, it is doubtful it went on backorder.On the other hand The probability of product had any of risks and "went on backorder" is very low but the intersting part is probability of product had "pieces past due","local_bo_qty","potential_issue" and "went on backorder" is 96%. It means the combination of these flags affects the going on backorder.

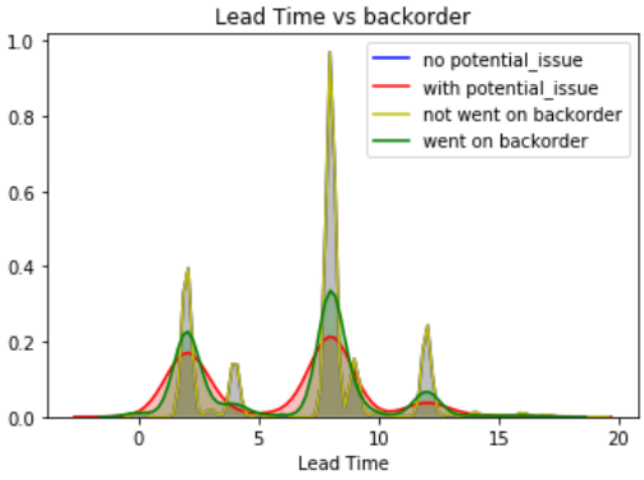

Figure 8. potential issue vs backorder

Relation of Went_on_backorder with	**P-Value**	**Chie2 Square**	**Chi2-Critical**
potential_issue	3.07548465734e-73	327.68081823	3.84145882069
pieces_past_due	0.0	8216.34330009	892.931831302
local_bo_qty	0.0	16767.3212128	713.557870509
deck_risk	4.98138077036e-52	230.357629836	3.84145882069
ppap_risk	2.74046384479e-30	130.798600861	3.84145882069
stop_auto_buy	0.00178555963213	9.75787726327	3.84145882069
oe_constraint	4.36293370618e-06	21.098395131	3.84145882069
rev_stop	0.0463094038953	3.9703042634	3.84145882069

Table 2. chi square and p-value of categorical features

I used crosstabulation and chi-square to find the relation between target variable with other categorical variables. Below are the calculated values.(Table 2)

All the relations have p-values less than 0.05 and we also have chi-square calculated value greater than the chi-square critical value. Based on these two evidences, I rejected the null hypothesis that variables are independent and went ahead with the alternate hypothesis. Here we can say that went_on_backorder is related to "potential_issue","pieces_past_due" , "local_bo_qty" ,"deck_risk" ,"oe_constraint","ppap_risk","rev_stop" and "stop_auto_buy", so I will keep all these features for modeling.

3.5 "Minimum recommended amount of stock" and "went on backorder"

With more "minimum recommended amount of stock" the order volumes decrease.The proportion of orders with the minimum recommended an amount to stock that "went_on_backorder": 0.66%. Another observation around the recommended stock where we can see that total perior sales were zero, however, the minimum recommended stock is kept at a high value. This could be a bad data or potential outliers(Figure 9).

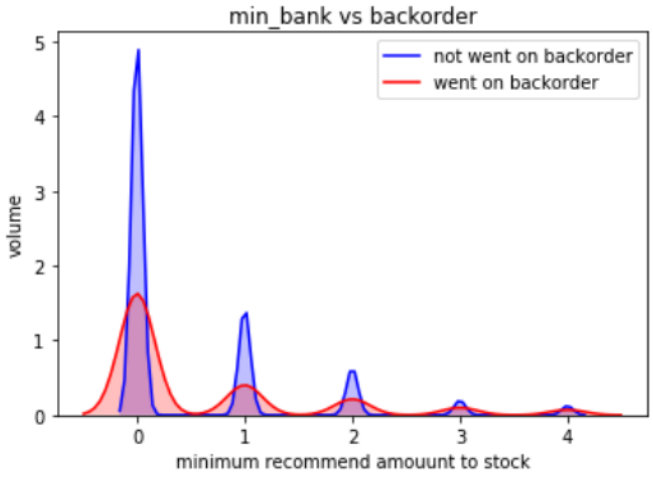

Figure 9. minimum recommanded amount of stock vs went on backordere

3.6 Data reduction

Since the dataset was massive, I decided to reduce data by capturing data from the total sales volume which is a significant reduction in data for not much loss of fidelity.

How I captured the total sales values is I used the cumulative sum of total sales volume.For data reduction, I captured 60% total sales volume, which is data was reduced to 7397 rows.

Using data reduction may save some computing time and also presenting a cleaner dataset for the predictive model. Backorder percentage is higher when we drop the NaN values in "lead time" that's because most of the orders were not backordered. There were no significant differences in the result of data reduction when I dropped missing values of "lead time". In my opinion, missing values in lead time do not affect the result of volume reductions.

Therefore other reasons might affect backorders.

In the following exploratory I answer these questions:

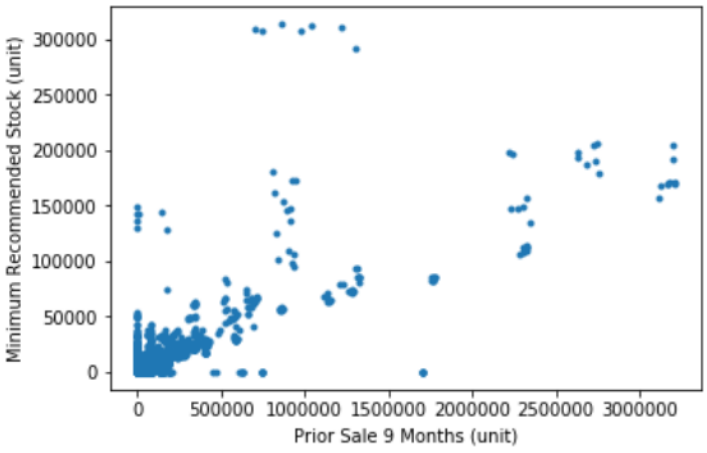

Figure 10. minimum recommanded amount of stock vs prior total sales

- Given that, how likely are backorders based on the "part risk flags"?
- How prevalent are they?
- What is the relationship between "potential_issue" and "pieces_past_due" are each
- What is the relationship between "potential_issue" and "pieces_past_due" are each represented by part risk flags or are they unrelated concepts?
- Based on the answers to these questions you could recommend: What aspects of the supply chain present the most prominent risks?
- Based on the risks, what would be recommended improving first?

3.7 Normalization

The attributes related to quantities were normalized (std dev equal to 1) per row. Therefore, parts with different order of magnitudes are approximated. For example: 1 unit of a expensive machine may be different from 1 unit of a screw, but if we standard deviate all the quantities we have, we can get a better proportion of equivalence between those items. The Figure 9 shows sales column's distributions after normlization.

Figure 11. Normalization histogram

4 Machine Learning

Imbalanced classification is a supervised learning problem where one class outnumbers other class by a large proportion.This problem is faced more frequently in binary classification problems than multi-level classification problems.The reasons which leads to reduction in accuracy of ML algorithms on imbalanced data sets:

- ML algorithms struggle with accuracy because of the unequal distribution in dependent variable.
- This causes the performance of existing classifiers to get biased towards majority class.
- The algorithms are accuracy driven i.e. they aim to minimize the overall error to which the minority class contributes very little.
- ML algorithms assume that the data set has balanced class distributions.
- They also assume that errors obtained from different classes have same cost.

Because the data set is very imbalanced so I add up the data that went on backorder to this sample.

In this data set target value is "went_on_backorder".

The features that I used to train the machine learning model are presented in the list below:

national_inv	lead_time	perf_6_month_avg
perf_12_month_avg	in_transit_qty	forecast_3_month
forecast_6_month	forecast_9_month	sales_1_month
sales_3_month	sales_6_month	sales_9_month
min_bank	potential_issue	pieces_past_due
local_bo_qty	deck_risk	stop_auto_buy
ppap_risk	oe_constraint	rev_stop

Model Features

4.1 Model selection

The training data set the amount of backoredered products are less than 1% of the whole products,so the data is very imbalanced. Therefore I add up the data that "went on backorder" to this sample. I used supervised learning to predict “went on backorder” product according to what they have reordered. Result shows the accuracy of 99%.

Data trained in two supervised models logistic regression, and random forest(Bagging-based ensemble).Comparing these two models in logestic regression model first because data was imbalance it showed high accuracy. That reason is, Logistic regression produces an estimated probability that a particular instance is from the positive class. It caused the classifier to over-predict positive instances. For some classifiers, it is not a significant problem, but I expect that logistic regression might be more sensitive to this mismatch between training distribution and test distribution.

After balancing the data set, I used regularization with my logistic regression model and used cross-validation to select the regularization hyper-paramete to find a suitable threshold that maximizes the F1 score (or some other metric).

A logistic regression model is searching for a single linear decision boundary in the feature space, whereas a decision tree is essentially partitioning the feature space into half-spaces using axis-aligned linear decision boundaries. The net effect is that it is a non-linear decision boundary, possibly more than one. This is nice when a single hyperplane does not readily separate the data points, but on the other hand, decisions trees are so flexible that they can be prone to overfitting. To combat this, I used the random forest. Logistic regression tends to be less susceptible (but not immune!) to overfitting.

I used ROC AUC score since it gives the probability of an estimator ranking a positive example higher than a negative example. This way it can evaluate the models before selecting a threshold for the decision function. Also I looked at 'Precision' as validation criteria because it is important that as many of the records predicted are correct as possible so that time is not wasted working on false positives.

	precision	recall	f1-score	support
0	0.85	1.00	0.92	2916
1	1.00	0.75	0.86	2016
avg / total	0.91	0.90	0.89	4932

RandomForestClassifier had good Precession score compare to all other classifiers.Model Validation: Model is trained on 80% of the data and and 32% is the test data below is the precision score on 10 folds.The features importance in random forest model are: sales_3_month and lead_time.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1500
1	1.00	0.99	1.00	966
avg / total	1.00	1.00	1.00	2466

Table 4. Random Forest precision and recall score

Precision-recall curves show how Precision and Recall metrics compete depending on the threshold defined for the decision function of the model.

Following is the ROC curve for the case in hand.

5 Conclusion

Knowing the cause of backordering help company's inventory management to plan for a more efficient stocking and backorder handling. A company can manage its inventory more efficiently using a prediction on the backorder risk for the products.In this project, the goal was using the past data and metadata around the product backorders, and provide a prediction of the potential products for backorders.

Data analysis showed that lead time has a relationship with product backorders, longer lead time, backorder proportion goes down. Also, sales have a strong relationship with back orders.

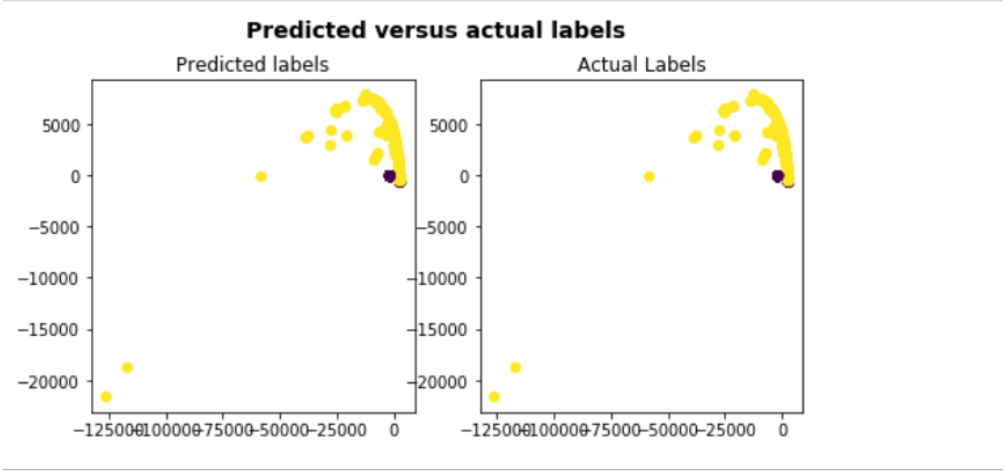

Figure 12. Predictable vs actual label

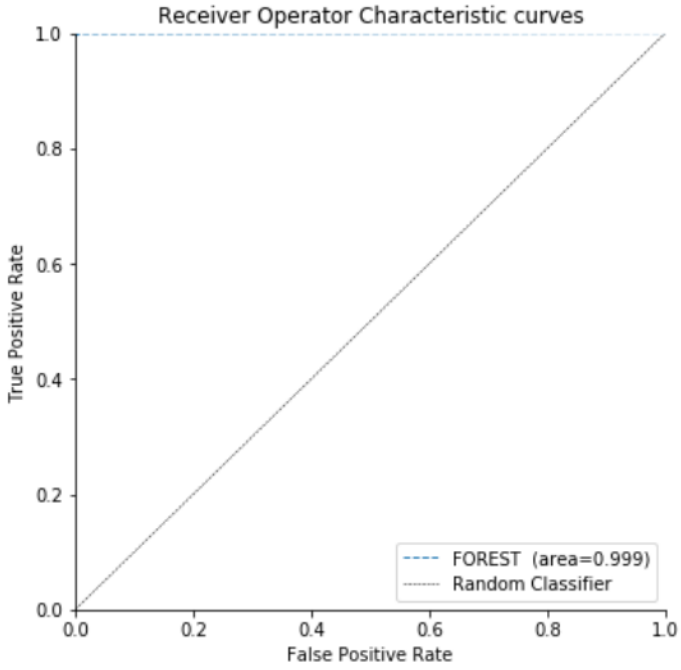

Figure 13. ROC curve

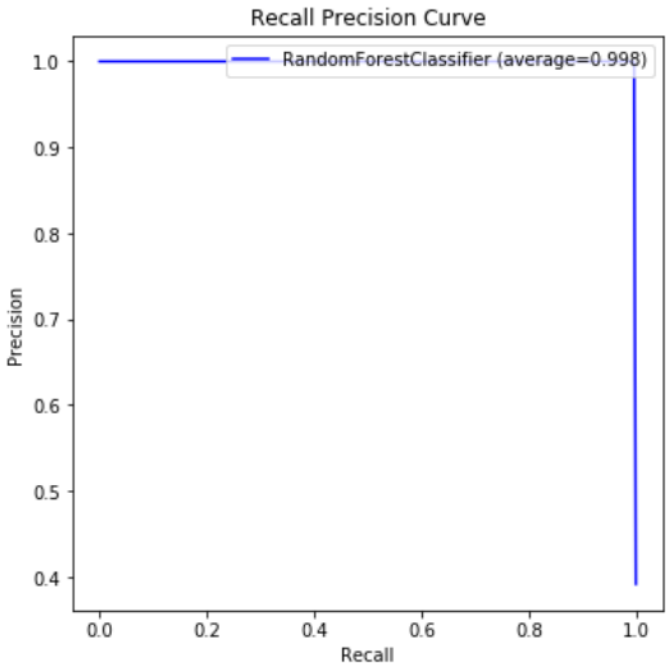

Figure 14. recall precision curve

All other categorical variables have relationships with backorders, but some of them have more. For example probability of a product that went on backorders and had three risk flags: parts overdue, potential issue, and local stock overdue is 97%. All the categorical variables relation with backorder have p-values less than 0.05. Also chi-square calculated values are higher than the chi-square critical value. Based on these two pieces of evidence, the null hypothesis was rejected that variables are independent and went ahead with the alternate hypothesis.

I used supervised learning to predict product backorders accurately. Dataset is highly imbalanced. We have only 0.66% data as 'Yes' to back order, so accuracy cannot be validation criteria here. Because of imbalanced data, I added more "yes" to backorder data to the dataset.I used 'AUC' score and 'Precision' as validation criteria.

After calculating AUC and precision-recall scores, RandomForestClassifier has better AUC and precision value. I decided to tune and validate these models.For tuning the parameters of the model, I used a mix of cross-validation and randomized search.

Based on the performances of the predictive models, I found tuned RandomForestClassifier as the most suitable predictive model to choose in this project. I recommend this model to the client as it has Auc 99% and a precision score of 1.

All solutions can be viewed in IPython Notebook in my github below.

https://github.com/hedib/DataScienceProjects/blob/master/project_Inventory/backorder.ipynb