SUPPLY CHAIN MANAGEMENT: A PREDICTIVE MODEL FOR INDUSTRY 4.0

Andreose Eugenio, Comotto Federico, Follari Lida, Rosito Juan Carlos.

Abstract – Nowadays with the opportunities given by Data Science with its fundamentals, bunch of methods and technologies, it is possible to solve problems paying attention to special needs, carving solutions that suits as much as possible the final user expectations and peculiarities. With this in mind our goal was to build a system fitting as much as possible the needs of a supply-chain, unknown in this R&D phase and therefore able to be twicked at time of deployment. Results obtained in our project enables industries to intervene immediately when an alarm of possible backorder is detected by a real time control system applying our classifier. First of all, a deep exploration analysis has been provided, in order to fully understand both real meaning and distribution of the attributes. In order to apply the classification models, data has to be fittable to manipulate it and to do statistical analysis, so the preprocessing carries out the treatment of the data, through different techniques from missing replacement to feature selection based on information gain and PCA filtering. Then, different type of classifiers have been trained using cross validation method, in order to select which model performs well, especially in case of data imbalance. In conclusion, a cost sensitive analysis, with parameters to be set at time of deployment, has been introduced to give a reinterpretation of algorithms and corresponding results, based on real industry costs and peculiarities proper of the particular physical supply-chain system.

Keywords – Data Science, PCA, statistical analysis, cross valu imbalanced data, backorder, classification model, cost sensitive, machine learning, supply chain

I. INTRODUCTION

In the "*Fourth Revolution"* age, where e-commerce has completely changed shopping habits, especially in terms of speed and availability, the industry world has to promptly satisfy customer needs, add value and maximize profit.

Shipping speed and reliability have become essential issues in customer satisfaction (new kind of needs), which is why, one of the main challenges that companies have to face every day, is to maintain a smart and flowing link between customer's orders and logistics.

Not too long ago, companies worked as single independent units, but, nowadays, skills, workforce and resources must be connected in order to deliver the best possible service. To fulfill this necessity a trend has become more and more a necessity: building an efficient Supply Chain.

A Supply Chain is an integrated manufacturing process wherein raw material are converted into final products, then delivered to customers (Beamon, 1998).

The critical issue, that a supply chain management context must prevent, because it could compromise the system, occurs when a customer orders an item that for whatever reason is not available.

This phenomenon goes under the name of "Backorder". More correctly, a backorder is an order for a good or service that cannot be filled at the current time due to a lack of available supply (www.investopedia.com/terms/b/backorder.asp).

Given the fact that Supply Chain is designed to maximize the efficiency, generally the numbers of backorders is a minority portion of the totality of recorded operations.

This statement introduces a typical scenario, transversal in the classification field, named Imbalanced Data, occurring when, in a dataset, the number of observations belonging to one class (rare class) is significantly smaller than the number of observations belonging to other classes.

Our project goal, objective of this paper, is to build a classification model able to predict, given some features regarding product's supply chain state, whether a product is going to go in a "went on backorder" issue.

This paper tries to provide a complete framework to solve the "backorder" problem and indirectly the imbalanced data one.

The roadmap implemented follows classical steps of Data Mining: beginning with exploration [II.a.], passing through preprocessing [II.b.] and feature selection [II.b.], getting to classification and evaluation [II.c.].

A fundamental final step has been the cost sensitive learning, [II.d.] a particular technique proposed by the literature (Ling C.X., Sheng V.S. 2008), able to manage "rare classes".

The relevance of our model lies in enabling industries to intervene immediately when an alarm of possible backorder is detected by a real time control system applying our classifier.

a. Dataset

Data used by our analysis was located on Kaggle platform.

We downloaded a training dataset containing 1687860 observations and a test set with 242075 observations (almost 2 million in total).

Each row represents a product crossing the Supply Chain, and the attributes define logistical aspect of the item in issue. More precisely, data are referred to weekly stock of products existing in each part of the whole chain, in a classical inventory format, for a total time frame of 8 weeks.

In order to better understand the meaning, a table, providing a brief description of the involved attributes, has been reported in APPENDIX A.

II. METHODOLOGY

a. Exploration

In this section statistical descriptive analysis techniques applied on data have been used to improve domain knowledge.
Attributes types have been defined. Distributions of all variables have been plotted and analyzed, through main indexes (such as mean, median, quartiles, variance) and boxplots.
Subsequently, linear correlation between attributes has been inspected thanks to plots and correlation matrices. A special focus has been placed on Pearson correlation between target variable and the remaining.

b. Preprocessing

Preprocessing phase involved different methods applied to prepare data for classifying procedures in virtue of results gained in the exploration phase.
First of all, a missing treatment has been performed, comprising methods such as: attributes binarization, missing pattern, matching evaluation, records elimination and most probable replacement technique.
Then, we have decided to remove a priori irrelevant and redundant attributes. Logistic Regression and Linear Correlation Matrix helped in supporting this decision.
A normalization of attributes has been computed.
Consequently, we have introduced a feature selection in order to select those variables that results more significative for predicting class attributes, and to reduce the dimensionality due to minimize computational cost of the final classifier.
In this case, filtering methods like information gain, and PCA (only for quantitative attributes) have been applied.
Principal Component Analysis (Song, Guo and Mei, 2010) is a statistical method with the goal of reducing dimensionality.
With this technique indeed, a dataset composed by *p* attributes, is resumed in *h* variables called "principal component". The first variable created as output of this process, is obtained finding the linear combinations of original dataset' s variables that maximize the total variance [2]. This variable represents the first principal component. The second principal component is the linear combination that again maximize total variance but is uncorrelated with the first principal component, and so on.
Usually should be selected as many principal components as enough to explain 70% - 80% of the total variance.
PCA method has been selected as quantitative attributes filtering because it gave also the possibility of a better interpretation of variables involved in each principal component. For this purpose, we'll take into account only those variables that best contributes to build the principal components that explain at least 70% of total variance.
It's also important to point out that, although we have autocorrelation analysis

c. Classification

1. Classifier

A classification technique (Classifier) is a systematic approach to building Classification Model from a Dataset: according to some relevant features, identified in the preprocessing phase. A Classifier tries to predict the class label of an unknown record. Classifiers can be grouped into specific categories depending on the criteria used for the classification task.
In this paper different type of Classifiers have been treated:

- Heuristic Classifier: Decision Tree with Gini Index as quality measure to compute the splitting, J48 (C4.5 decision tree) and Random Forest with 10 trees.
- Regression Classifier: Binomial Logistic Regression;
- Separation Classifier: Support Vector Machine with Log Loss function (Shalev-Shwartz, Singer, Srebro and Cotter, 2000) and Multilayer Perceptron with 1 hidden layer and 10 neurons;
- Probabilistic Classifier: Naive Bayes and Tree-Augmented Naive Bayes.

2. Measure

The most recognized starting method in order to look for the optimal classifier is to compute the Confusion Matrix. From the confusion matrix different measures can be derived.
For example, accuracy, the most popular classifier result measure. It keeps track of errors in the globality, doesn't matter if a class is totally mislabeled (misclassification errors and costs below). Depending on the case, this can be tolerated or definitely wrong.
Applying classifiers on Imbalanced Data, often leads to the fully mislabeled result; in cases where the minority class (rare class) is considered interesting, you don't want to erase it, this represents a problem.
In this project different metrics have been used, in particular:

- $\mathrm{P} = \frac{TP}{TP + FP}$ (Precision)
- $\mathrm{R} = \frac{TP}{TP + FN}$ (Recall)
- $\mathrm{F1}_{\mathrm{measure}} = \frac{2 * r * p}{r * p}$ (F1-Measure)
- Area Under the Curve (AUC)

The AUC has an important statistical property: the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance (Fawcett, 2005).

3. K-Fold Cross Validation

The real goal of each classification task is to select the classifier which likely provide the "best prediction

performance". A good approach is to split dataset into training and validation set: training set is used to train the classifier (learner), while validation set is used to evaluate its performance [1].
The K-Fold Cross Validation algorithm allows to repeat this trick K times and ensures that each record of the dataset is included into the training sets the same number of times and exactly one time in the validation set.
This method guarantees a better performance estimation of the classifier on new data.
During our analysis it has been adopted:

- K-fold Cross Validation algorithm with K = 10 partitions, stratified sampling[2] related to attribute "went on backorder", and seed = 1234 to set randomicity.

Cost Sensitive Learning

Cost-Sensitive Learning is a type of learning in data mining that takes the misclassification costs (and possibly other types of cost [3]) into consideration. The goal of this type of learning is to minimize the total cost. The key difference between cost-sensitive learning and cost-insensitive learning is that cost-sensitive learning treats the different misclassifications differently (Ling and Sheng 2008). Cost insensitive learning does not take the misclassification costs into consideration. In the real world the differences between different misclassification errors can be quite large depending on the application field (ex: Medical Diagnosis or Credit Fraud Detection).
So, depending on the case, Imbalance Data and/or different classes misclassification implications, a proper Cost Matrix must be set (the higher the cost corresponding to TP, TN, FP, FN occurrences, the fewer the number of occurrences will be).
In binary classification, Cost-Insensitive Learning works with this type of cost matrix associated with a confusion matrix:

	Predicted Negative	**Predicted Positive**
Actual Negative	TN, cost: 0	FP, cost: 1
Actual Positive	FN, cost: 1	TP, cost: 0

Table 1, ***Cost Insensitive Matrix***

However, when we deal with Imbalanced Data, where the minority class (positive class) is considered more interesting than the majority one, the cost (weight) associated with False Negative cases should be higher than the False Positive cost in order
In this paper has been computed a Cost-Sensitive Learning using three different type of cost matrix:

	Predicted Negative	**Predicted Positive**
Actual Negative	TN, cost: 0	FP, cost: 1
Actual Positive	FN, cost: 100	TP, cost: 0

Table 2, ***Cost Matrix 1***

	Predicted Negative	**Predicted Positive**
Actual Negative	TN, cost: 0	FP, cost: 10
Actual Positive	FN, cost: 100	TP, cost: 0

Table 3, ***Cost Matrix 2***

	Predicted Negative	**Predicted Positive**
Actual Negative	TN, cost: 0	FP, cost: 20
Actual Positive	FN, cost: 100	TP, cost: 0

Table 4, ***Cost Matrix 3***

III. RESULTS AND DISCUSSION

a. Exploration

The main results obtained through explanatory analysis are the following.
The dataset consists in 8 qualitative and 15 quantitative attributes. The qualitative attributes are binary, apart from "SKU" that is categorical and identifies product's ID randomly assigned.
Analyzing the target variable, "went_on_backorder", we have confirmed what we expected from backorder phenomenon. The percentage of products that actually went on backorder represents only 0,7% of the products involved in our training dataset (**Figure 1).**

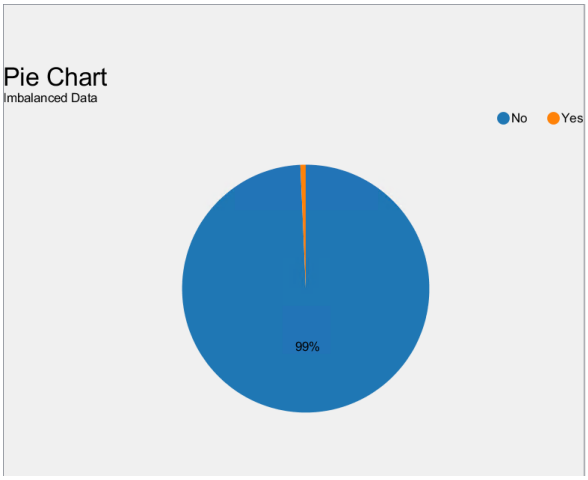

Figure 1, ***Imbalanced Data***

In order to better visualize and check for possible scale heterogeneity issues, a table containing main distribution indexes of each variables is detailed on APPENDIX A.

[1] Holdout method.
[2] Ensuring equal proportion of rare class in each K_i partition.
[3] P. Turney has been presented an exhaustive classification of types of costs (Turney, 2002)

So quantitative variables have different scale measures. This fact can represent a problem in classifiers model application, because attributes having bigger range (such as "national_inv") could be unfairly considered more important than others by some classification algorithms. In order to prevent this, behave, attributes have been normalized.
Plotting the distributions, we have detected outliers that affect each variable. In a statistical analysis context, removing records associated with outlier values could somehow improve the goodness of model fitting. In this case, instead, machine learning context, we have decided to keep all outliers, in order to train the classifiers to efficiently recognize and classify even those extreme values (Tallon-Ballesteros and Riquelme 2014).
From linear correlation analysis, emerged the presence of some extremely correlated attributes. In particular, variables representing forecast at different time ranges expressed in months ("Forecast_3_month", "Forecast_6_month", "Forecast_9_month") show an obvious high value of Pearson correlation between each other. This is also the case of variables representing sales ("sales_1_month", "sales_3_month", "sales_6_month", "sales_9_month") and the average performance of the source ("perf_6_months", "perf_12_months"). In [**Figure 2**] you can notice which attributes are correlated the most.

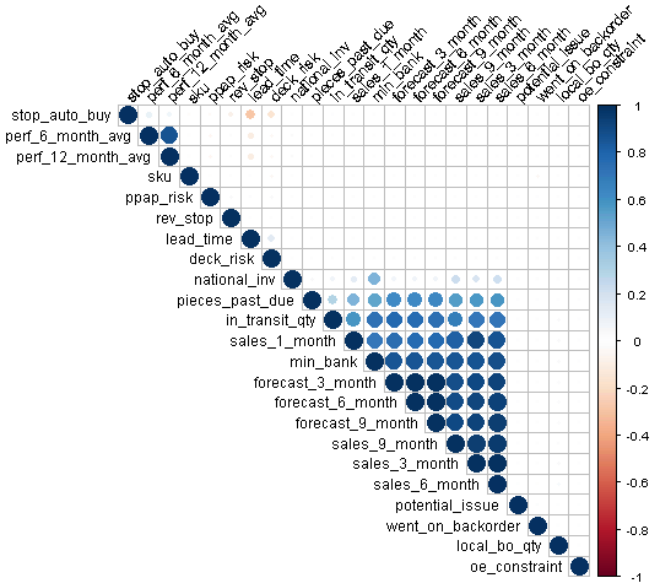

Figure 2, ***Correlation Matrix***

Moreover, as evident from [**Figure 2**] **,** the last five binary attributes present a low correlation with the target variable.
Another important discovered feature is the data sparsity. This issue is in accordance with the domain, because many attributes of dataset describe possible delay or fault of the Supply Chain. Again, every industry try to maximize the efficiency of this logistic process, and this is translates into the presence of many "0" values.
Finally, within dataset exploration, we have discovered a consistent amount of missing values. In particular, attribute "lead_time" contains 100893 missing, encoding with "NA". Attributes "perf_6_month_avg" and "perf_12_month_avg" contains respectively 129478 and 122050 missing values, both encoding with "-99". A large part of those missing values (100893) are present in the same records contemporaneously for each of the three variables (**Figure 3**).

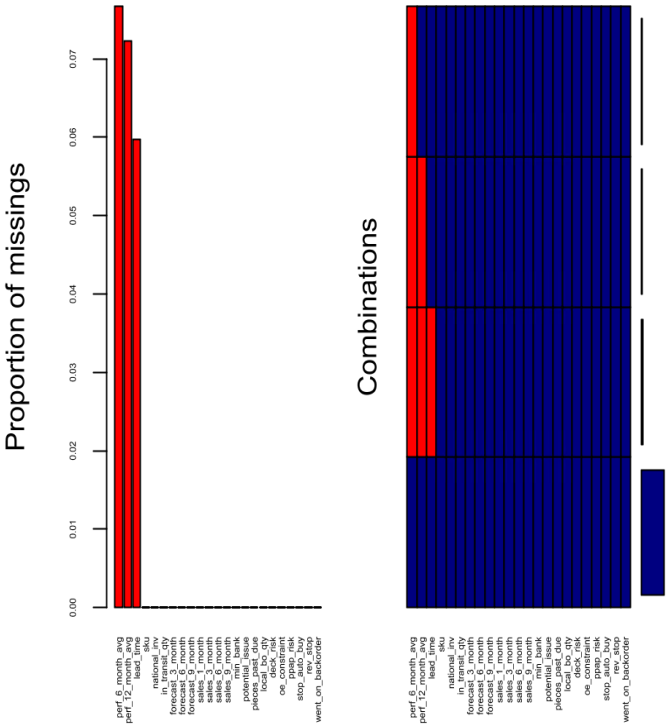

Figure 3, ***Missing Pattern***

b. Preprocessing

Herein statements, resulted from preprocessing phase, are exposed.
For practical reasons, all missing values have been encoded with "NA". Then we have focused on records presenting missing values in "lead_time", "perf_6_months_avg" and "perf_12_moths_avg" at the same time, trying to interpret the reason. For this purpose, we have compared the distribution of variables describing this pattern of records, with the variables distribution of a numerosity equal random sample of records with no missing. This analysis has been computed using "Student t-test" to control if exist difference in mean for quantitative attributes and using "X^2 squared test" to evaluate the influence for qualitative attributes. The outputs asses that patterns with missing values is significantly different from patterns representing other data. Unfortunately, as confirmed by the owner of dataset in Kaggle platform, those values of attributes regarding the performance of the source, are missing because they haven't been recorded. Consequently, those records have also "lead_time" values missing. Those values are thus of difficult interpretation and replacement. Our choice is to remove from the dataset all records associated with "perf_12_month_avg" missing values, because with a wrong replacement we'll probably decrease the accuracy of classifiers.

After the elimination, the only missing values that remains are in "perf_6_month_avg". As mentioned above (Image of perf correlation), exist an extremely high correlation between the two performances variable [**Figure 4]**, reason why we have decided to replace "perf_6_month_avg" values with a linear regression method.

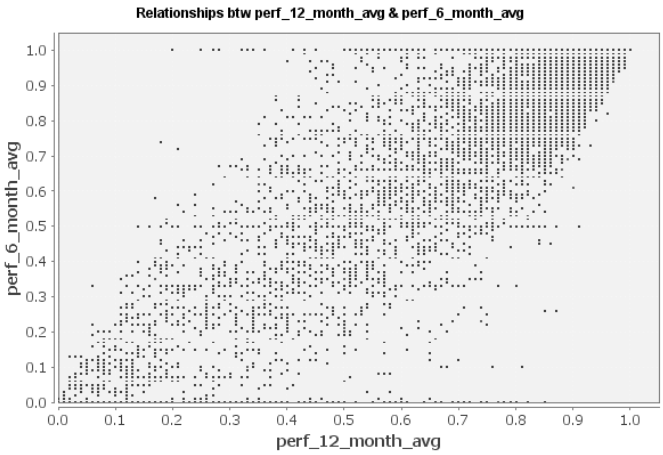

Figure 4, ***Correlation Perf 6 & Perf 12***

Again, due to a very high Pearson correlation, we have removed a priori redundant attributes, such as "forecast_3_months", "forecast_6_months" and "sales_1_months", "sales_3_months", "sales_6_months" because respectively "forecast_9_months" and "sales_9_months" already contains almost all the information. This subjective filtering found its ground in the fact that "forecast_9_months" and "sales_9_months" are only a cumulative total of the other related attributes.
As obvious, we have also removed attribute "SKU", because it only gives an ID to the product, and so results irrelevant to model application.
Later on, we have performed a logistic regression, in order to evaluate the relevance of the last five binary attributes on target variable. The output was that variable "REV_STOP" is not significant in predicting "went_on_backorder" at each fixed alpha significance level. For this reason, "REV_STOP" has been removed.
Then normalization has been applied.
Information gain filtering method suggested the removal of all the qualitative variables.
For the application of PCA method, we have chosen to use the variance/covariance matrix instead of the correlation matrix, because data have been already normalized. The output shows that all the quantitative attributes contribute to form the first four principal components that explain 73,72% of the total variance.
At the end of this phase, the dataset is "clean", reduced and ready for classifier application.

Variable	Count of NULL	% of Dataset
lead_time	100893	5.977569
perf_6_month_avg	129478	7.671134
perf_12_month_avg	122050	7.23105

Table 5, ***Missing Value***

Attribute	Data Type
national_inv	Quantity integer variable.
lead_time	Quantity integer variable.
in_transit_qty	Quantity integer variable.
forecast_9_months	Quantity integer variable.
sales_9_months	Quantity integer variable.
min_bank	Quantity integer variable.
pieces_pas_due	Quantity integer variable.
perf_12_months	Quantitative continuous variable.
local_bo_qty	Quantity integer variable.
went_on_backorder	Qualitative binary variable.

Table 6, ***Data Type***

c. Classification

Evaluation phase
After the preprocessing phase, the final Training Set (9 explanatory attributes and the target variable "went on backorder") has been processed by each classifier to select "the best one".
10-fold Cross Validation algorithm has been preferred to other validation techniques for the reasons pointed out in methodology. K equal 10 looked like a good trade-off between computational cost and performance quality:

- a number lower than 10, such as 3-fold Cross Validation, could restrict the performance evaluation, but improve the computational speed specially for our big dataset;
- a number higher than 10, such as Leave One Out Cross Validation (LOOC), could improve the

performance, but too expensive in term of computational cost for our case.

As previously mentioned, considering our imbalanced data, the accuracy measure doesn't actually give information about the quality of prediction. This asseveration is confirmed by the table n.

Classifier	Accuracy
Zero Rule	99%
Naive Bayes	99%
TANB	99%
Binomial Logistic Regression	99%
SVM	99%
Neural Network	99%
Decision Tree	99%
J48	99%
Random Forest	99%

Table 7, ***Accuracy as trivial measure.***

The baseline classifier named ZERO-RULES, which predicts for the "went on backorder" target class in every record a "No" value, performs an accuracy value equal to 99%, such as every other classifier. This is clearly the "went on backorder = NO" percentage in our dataset, ergo in our project accuracy is a trivial measure.
Each classifier implemented in the project has been evaluated through a cross-analysis of Recall, Precision, F1measure and AUC. Classifiers don't achieve excellent results, among other reasons, because the heavy imbalance in the backorder dataset doesn't help. Later on, a cost sensitive analysis will be presented to deal with this problem.
Classifiers' performances are hereby reported:

Classifier	Recall	Precision	F1-measure	AUC
Zero_Rule_Yes	0	/	/	0.5
Naive_Bayes_Yes	0.004	0.095	0.008	0.61
TANB_Yes	0.087	0.18	0.117	0.924
Logistic_Regression_Yes	0	0.095	0	0.679
SVM_Yes	0	/	/	0.5
Neural_Net_Yes	0.001	0.239	0.002	0.831
Decision_Tree_Yes	0.321	0.441	0.372	0.803
J48_Yes	0.079	0.579	0.139	0.739
Random_Forest_Yes	0.279	0.719	0.402	0.879

Table 8, ***Cross Validation Results***

Worst models detected by the evaluation phase are:

- SVM, which performance could be compared to ZERO-R performance. Both Loss Function and Hinge options achieved the same result. Probably SVM hyperplane is not been able to separate records into classes.
- Binomial Logistic Regression (LOG), which performance could be probably affected by the high number of outliers that have been deliberately left in the dataset;
- Naive Bayes, which values of AUC, Recall, Precision and F1measure are very low. Maybe the concept of independence assumption could be too strong.

Multilayer Perceptron obtains good values in terms of AUC and Precision but Recall and consequently F1measure are too low compared to other classifiers. All the four Heuristic classifiers, first of all Random Forest then Decision Tree provide good results. Finally, also the Tree-Augmented Naive Bayes (hybrid classifier between a Tree and the Naive Bayes) achieves good performance.

Selection phase
In particular these three different classifiers: Random Forest, TANB and Decision Tree, are chosen to be the candidate for the "best model". Every classifier has been selected for different logic reasons:

- Random Forest because is the model that achieves the best F1measure in absolute (0.402);
- Decision Tree because is the classifier that gets the best Recall value (0.321);
- TANB is the classification learner that realizes the best AUC level (0.9243).

It should be stressed that J48 achieves a precision value very high (0.579), but lower than Random Forest one (0.719) and it

has been decided to not include J48 in the "best models" group.
After a first selection, chosen classifiers are trained on the whole training set and validated with the new test set made available by the dataset owner: this strategy has been used to find the best model that will be recommended to the supply chain. **Figure 5** resumes the Random Forest, Decision Tree and TANB performance achieved on the test set [after] and in previous cross validation phase [before]:

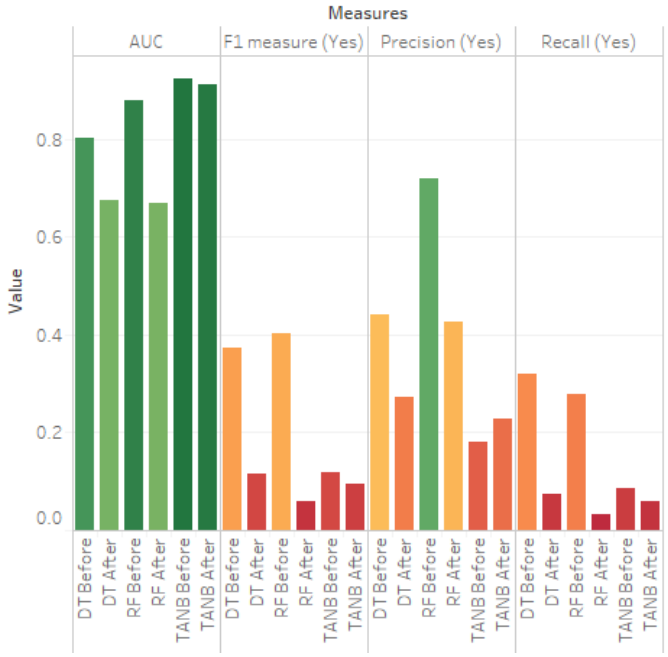

Figure 5, ***Performance After & Before***

As you can see, Decision Tree and Random Forest suffer from overfitting: Recall, Precision, F1measure and AUC drastically decrease. Although TANB gets the best performance only in terms of AUC, it performs as well as in evaluation phase.
This robustness in terms of generalization is appreciated and positively evaluated in a future concrete - business perspective. For these reasons, in our project, the final recommended model for supply chain willing to prevent "went_on_backorder" of products is Tree-Augmented Naive Bayes. Aware of the limits found in the evaluation part, in the following section we explain how to improve performances of the selected model using cost sensitive analysis.

Cost - Sensitive Learning
Important notes for this final part:

- Recall and Precision are used as measures of performance.
- As our "best model" only the TANB model will be applied with the cost sensitive approach;
- Pointing out that the ratio between NO and YES for went on backorder is about 140:1, we have adopted a cost matrix with values on the main diagonal equal to zero and False Negative cost greater than False Positive one to rebalance the weight of the target class.
- Three different False Positive costs are used to limit the Recall-Precision trade-off. It's cleared that lower is the FP's cost, higher is the Recall and lower is the Precision: classifiers are so induced to classify more records as positive one more than it would otherwise.

The implemented costs are merely subjective[4], but could represent internal costs to the Supply Chain, such as control costs. Under this perspective every cost matrix will reflect different needs in the supply chain. The first one with a FN cost equal to 100 and a FP cost equal to 1 could be recommended to well-integrated Supply Chain, where any possible control costs does not represent a problem (Recall is preferred). The second one (FN cost = 100 and FP cost = 10) could be recommended to a not as well integrated Supply Chain, where control costs could be a problem (Recall is preferred to Precision, but they have similar weights). The last one (FN cost = 100 and FP = 20) could be recommended to Supply Chain where control costs represent a full-fledged problem (Precision is preferred to Recall, but the weight is similar).

The Final Results related to Cost Sensitive Learning & TANB model are:

	TANB & COST MATRIX 1	TANB &COST MATRIX 2	TANB & COST MATRIX 3
Recall	0.787	0.294	0.181
Precision	0.049	0.162	0.216

Table 9, ***Cost Sensitive Performance.***

We can confirm that the first Cost Matrix benefits the Recall [0.787], while the second and the last one could be a good agreement between Precision and Recall.

IV. CONCLUSIONS

Our system strength is summarized by the metaphor of a dress by a dressmaker: able to be twicked following the needs of the supply-chain to be deployed. Dealing with economic and system peculiarities and enabling a prompt intervention when a backorder is going to occur.
So, one of our focuses was on preserving data as much as possible, we didn't pursue the downsampling way, and tried to give an interpretation, with a long preprocessing phase, enabling the final user to understand where to twick along the entire process, starting from cost-sensitive matrix parameters, ending in acting in the physical chain of supply in order to improve values in the data set from the source, for example

[4] It would be better to ask advice of a domain expert.

working on the minimization of the LEAD_TIME variable improving the logistic. Our research conducted us to choose a model, that we tested not being too prone to overfitting, Tree-Augmented Naive Bayes was the one, and able to be improved with a cost sensitive analysis in terms of data imbalance handling, typical in this domain, and in terms of economic aspects peculiar to many different and specific situations.

V. TECHNICAL DESCRIPTIONS

The computational effort is commonly necessary when it comes to the management of the data. On this study has been used a computational power of a machine with 16 of RAM and 4 physical cores, on this specific case the computer has been a Microsoft Azure Virtual Machine.
Through the process several software has been used. For the domain and statistics exploration and for the data management the software selected was R, for the application of classifiers and to manage the data, for does classifiers, was used KNIME; finally to represent the data graphicly the software used was Tableau.

BIBLIOGRAPHY

BEAMON B. (1998). SUPPLY CHAIN DESIGN AND ANALYSIS: MODELS AND METHOD. INTERNATIONAL JOURNAL OF PRODUCTION ECONOMICS. VOL. 55, NO. 3, PP. 281-294

FAWCETT TOM (2005). AN INTRODUCTION TO ROC ANALYSIS. INSTITUTE FOR THE STUDY OF LEARNING AND EXPERTISE,

FENGXI S., ZHONGWEI G., DAYONG M (2010). FEATURE SELECTION WITH PRINCIPAL COMPONENT ANALYSIS. INTERNATIONAL CONFERENCE ON SYSTEM SCIENCE, ENGINEERING DESIGN AND MANUFACTURING INFORMATIZATION.

LING C.X., SHENG V.S. (2008). COST-SENSITIVE LEARNING AND THE CLASS IMBALANCE PROBLEM. ENCYCLOPEDIA OF MACHINE LEARNING.

SHALEV-SHWARTZ S., SINGER Y., SREBRO N., COTTER A. (2000). PEGASOS: PRIMAL ESTIMATED SUB-GRADIENT SOLVER FOR SVM. MATHEMATICS SUBJECT CLASSIFICATION.

TALLON-BALLESTEROS A.J. AND RIQUELME J.C.(2014). DELETING OR KEEPING OUTLIERS FOR CLASSIFIER TRAINING?. NATURE AND BIOLOGICALLY INSPIRED COMPUTING.

TURNEY P. (2002). TYPE OF COST IN INDUCTIVE CONCEPT LEARNING. WORKSHOP ON COST-SENSITIVE LEARNING AT THE SEVENTEENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING, PP. 15-21

VI. SITOGRAPHY

Investopedia,“Backorder”,www.investopedia.com/terms/b/backorder.asp

VII. APPENDIX A

Attribute	Description	Data type	Statistics
SKU	Random product ID	Nominal digit string.	
NATIONAL_INV	National stock level of the ID product.	Quantity integer variable.	Q1:4, Mean:496, Q3:80
LEAD_TIME	Product supply-chain traversal time expressed in week.	Quantity integer variable.	Q1:4, Mean:8, Q3:7.87 NA'S:100894
IN_TRANSIT_QTY	Product quantity in transit from source.	Quantity integer variable.	Q1:0, Mean:44.1, Q3:0
FORECAST_3_MONTH	Selling forecasts for the expressed months.	Quantity integer variables.	Q1:0, Mean:178.1, Q3:4
FORECAST_6_MONTH	Selling forecasts for the expressed months.	Quantity integer variables.	Q1:0, Mean:345, Q3:12
FORECAST_9_MONTH	Selling forecasts for the expressed months.	Quantity integer variables.	Q1:0, Mean:506, Q3:20
SALES_1_MONTH	Sold product amount in the latest reported months.	Quantity integer variables.	Q1:0, Mean:55.9, Q3:0
SALES_3_MONTH	Sold product amount in the latest reported months.	Quantity integer variables.	Q1:0, Mean:175, Q3:15
SALES_6_MONTH	Sold product amount in the latest reported months.	Quantity integer variables.	Q1:0, Mean:341.7, Q3:31
SALES_9_MONTH	Sold product amount in the latest reported months.	Quantity integer variables.	Q1:0, Mean:525, Q3:47
MIN_BANK	Suggested minimum product amount to be kept stock.	Quantity integer variable.	Q1:0, Mean:52.77, Q3:3
POTENTIAL_ISSUE	Possible problem with product supplier.	Binary qualitative variable.	Yes = 907; No = 1686953;
PIECES_PAST_DUE	Product amount delayed by supplier.	Quantity integer variable.	Q1:0, Mean:2.04, Q3:0
PERF_6_MONTH_AVG	Average supplier performance in the past 6 months.	Quantitative continuous variable.	Q1:0.70, Mean:0.78, Q3:0.97
			NA'S:129478
PERF_12_MONTH_AVG	Average supplier performance in the past 12 months.	Quantitative continuous variable.	Q1:0.68, Mean:0.78, Q3:0.96 NA'S:122051
LOCAL_BO_QTY	Product amount delayed by supplier.	Quantitative integer variable.	Q1:0, Mean:0.626, Q3:0
deck_risk	This variable expresses potential problems in the product retrivial.	Qualitative binary variable.	Yes = 387483 No = 1300377
oe_constraint	This variable expresses potential problems in the product retrivial.	Qualitative binary variable.	Yes = 245; No = 1687615
ppap_risk	This variable expresses potential problems in the product retrivial.	Qualitative binary variable.	Yes = 203834; No = 1484026
stop_auto_buy	This variable expresses potential problems in the product retrivial.	Qualitative binary variable.	Yes = 61086; No = 1626774
rev_stop	This variable expresses potential problems in the product retrivial.	Qualitative binary variable.	Yes = 731, No = 1687129
WENT_ON_BACK ORDER	States if correspondent product went on backorder.	Qualitative binary variable.	Yes = 11,293; No = 1,686,953;

Table 10, ***Summary of Attributes.***