Predicting vendors' service level by meta learning and feature selection modeling

Clara Ramos Teixeira Puga

Master's Dissertation

Supervisor FEUP: Prof. Samuel Moniz

Integrated Masters in Industrial Engineering and Management

2018-07-02

To my parents and sister,

Previsão do nível de serviço de fornecedores através de modelação por meta aprendizagem e seleção de atributos

Resumo

A área de analítica empresarial tem sido alvo de destaque nos últimos anos, atendendo às suas virtualidades para potenciar o sucesso em várias fases do negócio. Por exemplo, na cadeia de abastecimento. Assim, e sendo os fornecedores uma das entidades críticas neste tipo de ecossistema, este projeto surgiu associado à implementação de ferramentas analíticas, focadas no papel que aqueles desempenham na performance das subsequentes entidades da cadeia. O objetivo central da presente dissertação foi descrever e posteriormente implementar um conjunto de procedimentos que visam permitir ações preventivas, face ao nível de serviço prestado pelos fornecedores. Em termos analíticos, o objetivo foi desenvolver um modelo preditivo para os indicadores de performance do fornecedor, previamente definidos.

A partir do caso de estudo trabalhado numa empresa de logística que opera essencialmente na distribuição de produtos na área da moda, foram definidos quatro indicadores: (i) discrepância entre o número de items acordado com fornecedor e o real aquando da entrega da mercadoria, (ii) discrepância entre a data acordada com o fornecedor e a real para entrega da mercadoria, (iii) taxa de devoluções por culpa do fornecedor e, por fim, (iv) número de falhas de etiquetagem de produtos (ausência ou defeito). Apenas os primeiros dois indicadores foram considerados como variável de previsão, devido à limitação inerente à insuficiente quantidade de dados disponível. Apesar do volume de dados não ser suficiente para previsão, estes foram incluídos para a elaboração de um documento no qual são apresentados os índices de performance do(s) fornecedore(s), tendo como propósito aumentar a visibilidade da marca para com o desempenho dos seus fornecedores. Após esta análise, os atributos para futura modelação foram selecionados e manipulados, resultando isto numa representação passível de ser usada para previsão.

Seguidamente, dois cenários foram considerados: um, no qual os atributos foram sujeitos a uma tranformação (redução dimensional), usando o método PCA e outro, no qual os atributos originais foram usados. Para ambos os casos, estas variáveis independentes foram ordenadas segundo o método mRMR. O ponto de partida para a procura de uma solução foi adicionar, iterativamente, cada atributo, pela sua ordem no ranking. Assim, na primeira iteração, apenas o primeiro atributo da lista é usado, acabando o ciclo por incluir todos os presentes na lista. A metodologia-base para a seleção dos algoritmos testados teve como princípio adicionar sequencialmente complexidade aos modelos usados. Os modelos de aprendizagem testados foram os seguintes regressores: *LASSO*, Árvore de Decisão, *Random Forest*, *Support Vector Regressor* e, por fim, *Stacking*.

O método que provou oferecer os melhores resultados foi aquele que consistiu em usar vetores gerados pela transformação de atributos por PCA. Dois modelos, *Stacking* e *Support Vector Regressor*, cujos erros se revelaram estatisticamente equivalentes, foram os que apresentaram melhor performance, para a variável dependente "discrepância de quantidades". Para a segunda variável definida, nenhuma das diferenças de performance entre os modelos testados se revelou estatisticamente significativa, com um nível de confiança de 95%. Uma das causas identificadas para estes resultados foi a quantidade insuficiente de dados para a aplicação deste tipo de metodologia.

Por fim, foi feita a integração das previsões geradas com um algoritmo de otimização, já existente na empresa, o qual gera a alocação dos recursos do armazém. Foi percetível uma redução de 20% do tempo médio de processamento de receções, num espaço de 3 semanas, em relação aos 3 meses anteriores à implementação.

Abstract

Supply chain analytics has been emerging as a powerful tool for business success in the area of logistics. Hence, and being vendors one of the critical entities in this type of ecosystems, this project appeared linked to the implementation of analytics, focused on vendors' performance and their impact for the consequent entities of the chain. The main goal of the current dissertation was to enable preventive actions from stakeholders concerning vendors service level, leading to a more efficient warehouse planning and vendor performance reports for brands. Thus, the analytical approach that allowed the accomplishment of this objective was based on a machine learning perspective, having as a goal the prediction of vendors' service level.

Being the case study developed in a logistics company that operates essentially in the fashion industry, the first step was to define the metrics for these entities' service level, which resulted in four indicators: (i) quantity discrepancy between real number of items delivered and the agreed with vendors, (ii) days discrepancy between real goods' delivery date and the agreed date with the vendor, (iii) returns rate by vendor's fault and, finally, (iv) labeling faults rate caused by vendors (no label or faulty label). However, only the first two indicators above-mentioned were subject of prediction task, due to data availability limitations regarding the last two indicators described. Notwithstanding, all data available from the four indicators was organized to report historical events and translate them into vendor scores. This report aim is to enhance stakeholders visibility towards current vendor performance. Consequent to this analysis, attributes for prediction were selected and manipulated to a representation ready for modeling.

Afterwards, two scenarios were considered: one, in which attributes were subject of a dimensionality reduction through PCA encoding and another with the original features. For both cases, a ranking was computed using the mRMR method. The starting point for the solution searh was to iteratively add one feature/vector from the ranking, by their order. Thus, on the first iteration, the most relevant feature from the list was used and the cycle ends with all features from the ranking being used for model training. The purpose of this cycle was to internally test which subset of features/vectors offered the lowest prediction error, for each learning model.

The supra mentioned learning models were chosen by starting with the simpler ones and consistently increasing the complexity. This resulted on the following sequence of regressors: LASSO, Decision Tree, Random Forest, Support Vector Regressor and Stacking.

The method that proved to offer the best results was the one which input were variables encoded by PCA. These vectors were afterwards modeled by two learning models that were statistically equivalent in the prediction error (stacking and support vector regressor), for the target variable quantity discrepancy. For days discrepancy prediction, none of the differences between models' performance revealed to be statistically significant (at a confidence level of 95%). The conclusion was that the available amount of data was insufficient for a conclusive statement concerning models' performance.

Finally, a service-oriented implementation through APIs was performed and integrated with the already existent optimization algorithm for warehouse resources planning. Hence, resources are allocated taking into account the service level of vendors, which resulted in a decrese of 20% of the time spent handling receptions in the warehouse. This value was computed considering the 3 weeks after implementation and compared with the last 3 months before implemenation.

Acknowledgments

To begin, the current work would have not been possible without the support of several people from HUUB (in special to Jorge Ferreira), for their constant motivation and incredible ideas. Also, I would like to show my gratitude for the encouragement and team spirit with my colleagues at the company.

In the second place, I would like to thank my supervisor at FEUP, Professor Samuel Moniz and Hugo Ferreira from INESC, for their expert inputs and valuable guidance.

To my father André, the smartest and kindest person I have ever known, for having taught me his passion for learning and that working hard is the key to success. To my mother Fátima, for showing me daily the importance of resilience and focus in life. To my sister Matilde, for her peculiar sense of humor and for being my companion in life. To my godparents Clara and Fernando, for all the support in this journey. I am eternally grateful to all of you.

I would also like to express my gratitude to all of my friends, that I have met in High School (Susana, Ana, Alexandra, João, Marta), FEUP (Raquel, Beatriz, Inês, Vanessa, Maria Inês, Maria João, José Edgar, Mariana, Nuno, Ricardo, André, José Miguel, Ana Catarina, Leonor), Germany (Cynthia, Alba, Ana, Antonio, Mario) and many others that were part of this adventure. For all the mornings, evenings and nights spent studying together. For the happy moments, funny jokes shared and for the support in the bad times. It would have not been the same without you all.

Last but not the least, I would like to thank the friends that saw me growing. To my oldest friend Diana, for all the motivation, caring and strength. To Sara, for the endless talks and laughs shared.

Table of Contents

Nomenclature

AM – Account Manager

API – Application Programming Interface

CV – Cross Validation

DT – Decision Tree

LR – Linear Regression

MAE – Mean Absolute Error

mRMR – minimum Redundancy and Maximum Relevance

MSE – Mean Square Error

PFS – Pick From Stock

PTS – Pick To Stock

PTSP – Pick To SPlit

SVR – Support Vector Regressor

P&L – Profit & Loss Statement

RF – Random Forest

RMSE – Root Mean Square Error

RSS – Residual Sum of Squares

VSM – Value Stream Map

Table of Figures

Table of Tables

1 Introduction

The current thesis was developed in an industrial environment, being focused on a set of problems that arise, mainly caused by vendors, in a logistics company operating on the fashion industry. These vendors represent the supply chain parties that are responsible for goods manufacturing and which their service may be B2B or B2C.

A vendor, as a crucial entity of a supply chain, is able to impact consequent players of the chain, since a value is created by their input and thus the final product/service may be dependent on that contribution. Thus, certain events caused by these entities may harm another intervening parties, such as the logistics company and the brand. However, identifying these event's patterns is challenging when the number of vendors is high and many factors are linked. Adding to this, data dimension may be a relevant issue too: data availability and quality may be critical points to enable knowledge extraction.

On the other hand, analyzing these patterns and understanding the variables that can trigger those behaviors can offer beneficial inputs. With this knowledge, predicting the likelihood of some of these events happening in future orders may also be useful. These could enhance warehouse operations planning and inform brands about whether the vendor is trustworthy or not, in certain types of categories identified.

Addressing mainly the fashion industry, this research work tackles the predictability problem, considering both brand's perspective and logistics company planning, with the ultimate goal of assuring final customer satisfaction. Despite focusing on vendors, this thesis also provides a broad view for understanding the drivers to the appearance of issues that, at first sight, seem to be dependent on vendors, but in a more detailed analysis the causes may be, for instance, from the type of material produced. By way of explanation, the factors that may lead vendors to incur on those type of behaviors.

Having this, the aim of this project is to provide a set of tools that can enhance decision making in terms of warehouse operations, financial and strategic management, with the overall purpose of decreasing costs and gaining report business insights. The detailed goals can be found in the next subsection.

1.1 Thesis Objectives

Following a top-down approach, the most relevant topics for this work are described.

Essentially, outputs from the current project aim to provide a set of tools that allow preventive actions (warehouse, financial and strategic planning) and increase visibility of current vendors performance. The benefit of a successful accomplishment of the first goal highlighted is precisely the avoidance of reactive actions, enhancing thus the efficient activity planning for the logistics company. The second goal stated, targeting mainly the brand, involves the development of reports that use historical data to score vendors, which may provide valuable insights for vendor selection.

For this to be possible, understanding business strategy and align it with vendors' role and influence on that represented a crucial step. For instance, defining types of bad impacts (in terms of products quality and service) of a vendor in the supply chain, to both logistics company and its client (the brand). This includes issues that are possible to detect on the sphere of a logistics company. After identifying and prioritizing issues that emerge due to these entities, a more analytical analysis of this data was the following objective.

Being vital to a chain of this type the supply of the goods, analyzing drivers that lead to the occurrence of a low vendor service level may represent a step forward to the mitigation of those. Thus, variable identification and posterior influence analysis on the target studied was also one of the goals specified. This aimed to offer business insights that can be helpful for warehouse planning improvement but also, by transmitting this information to the brand, trigger better decision making by the brand owner. And it is straightforward the compatibility of what has been described with the strategy of a logistics company: the more successful the brand is, the more likely are sales orders to increase and thus, the more likely are the revenues for the logistics service provider.

Once previously mentioned goals are achieved, the final objective was the development of a tool that uses those inputs to predict vendors' service level was defined as the last one to be accomplished. In analytical terms, this involved data understanding and preparation. For the business, it means establishing a baseline for logistics' company activities planning and a report of vendors performance for the brand. Finally, for this to be available for the stakeholders, a service-oriented implementation that allows prediction and reports access by users was then comprised on the last goal.

1.2 Project Methodology

Following the order of the goals before mentioned, the first concern was to deeply understand the business, including not only the internal processes, but also company's vision, values and strategy. After this, an analysis guided to the topic of the current dissertation was developed by identifying the main issues related with vendors, from different departments at the company. Matching this information with the data extracted from the database, it was possible to identify the main improvement opportunities.

The next step represented already the data preprocessing phase, where data from different sources of information were organized, aggregated and standardized. This preliminary process dealt with inconsistent, noisy and missing data that was found. Still in this step, the representation of the data was the subject of analysis and modification. Afterwards, having the data prepared, a descriptive analysis was performed aiming as an output a support for the qualitative arguments withdrawn in the primary states.

With a clear view of the data available, the performance indicators were defined in four: (i) quantity discrepancy between real number of items delivered and the agreed with the vendor, (ii) days discrepancy between real goods delivery and the agreed date with the vendor, (iii) returns rate by vendor's fault and, finally (iv) labeling rate (no label or faulty label). However, only the first two were subject of a prediction phase, since the amount of returns and labeling data was considered insufficient to be modeled. For the report of vendors performance, stated as one of the objectives of this project, all of the four indicators were included.

Once the data was ready for processing, the most relevant and less redundant features with the target variable were determined. This was performed using the mRMR algorithm. Parallel to this, dimensionality reduction using PCA generated vectors that were also ordered by the mRMR criterion. Hence, two different datasets were created for each target variable.

Having a list of the ordered features and vectors as an output, the prediction task was the next step. The methodology applied here was to iteratively add complexity to the models used.

I.e., starting with the simpler learning models and then continually testing more complex models and test the variance on their performance. On top of the predictions from each of those models, a meta learner was applied, aiming to learn from the predictions of the base learners.

Lastly, implementing all prediction results as a service made available to the stakeholders, represented the final stage. Two APIs were developed: the "service level predictor API" that generates predictions depending on the input given (model, target variable and type of encoding) and the "Request Generator API" that periodically requests predictions to the previous API mentioned, if a new reception is inserted on the system.

This encompassed not only a service-oriented application for the predictions generated, but also the use of the historical data to compute the KPIs defined and, afterwards, provide visibility concerning vendors performance in a format of a report for brands.

1.3 Dissertation Structure

This dissertation starts with the current chapter as an introduction to the problem, where the purpose of this topic is briefly explained.

The second chapter, named as theoretical background, is a compilation of the information found in the literature that support the decisions of the methods implemented to solve the proposed problem.

The third chapter introduces the case study for this project: company organizational structure, current approach, business understanding and specific goals that were expected to achieve.

The fourth chapter describes the methodology followed in the industrial context where this work was applied. Data preprocessing and engineering, variable selection, prediction models and hyperparameter tuning are explained.

The fifth chapter is based in all previous chapters, presenting a low-level explanation of the methodology implemented. Preprocessing and modeling results, implementation architecture and vendors performance dashboard are the focal points discussed on this section.

The sixth chapter includes the conclusions derived from the development of the current thesis and also possible valuable approaches and methods to be applied in the future.

2 Theoretical Background

2.1 Supply Chain Management

Being this project oriented to companies focused on providing a set of services that are part of the supply chain, it seems relevant to analyze this topic. Hence, supply chain can be described as a group of entities oriented to provide all services or goods necessary to fulfill a customer request. Chopra and Meindl (2004) divide these entities in five: supplier, manufacturer, distributor, retailer and, finally, the customer; highlighting the linkage between them that allows the share of different type of assets. Thus, to the management of this flow was assigned the term of supply chain management (Lambert, Cooper, and Pagh 1998).

Another perspective is offered by Christopher (2011) where the author starts to explain that while logistics focus on the processes within the business, supply chain management reaches a wider scope by dealing and integrating with downstream and upstream entities. Nevertheless, both concepts seek an ultimate goal of achieving customer satisfaction at the lower cost possible.

In addition, and linked to the data mining concept, Supply Chain Analytics (SCA) allows useful findings from the data extracted by many sources present in this type of system (Sahay and Ranjan 2008) and appears as a possible solution to enhance integration along the supply chain, taking into account the inherent variability and risk (Wang et al. 2016).

2.1.1 Supply Chain Analytics

SCA can be described as the application of business analytics methodologies on big data collected along the supply chain (Wang et al. 2016). This concept of "big data" represents a high volume of data (Demirkan and Delen 2013), and is becoming popular due to its applicability nowadays, caused by cheaper storage costs and multiple sources of information.

Supporting that the use of the knowledge extracted by data analytics methods is one of the main reasons for supply chain success, Wang et al. (2016) joins three global types of data analysis usually performed with big data: (i) descriptive, (ii) predictive and (iii) prescriptive analytics. The first category is mainly used to detect improvement opportunities by the analysis of the current situation (Tiwari, Wee, and Daryanto 2018) with applicability in diversified areas. At the second place, events forecast based on mathematical algorithms, with supply chain gathered data inputs, attempt to predict and classify future events, providing meaningful insights for decision making. These methods (data and text mining, for instance) also offer the ability to capture data patterns and cluster them, identifying the roots behind the predicted behavior (Wang et al. 2016). Lastly, prescriptive business analytics aims to answer what should be done, in a certain situation, to improve business performance, and this can be supported by simulations, optimization, decision modeling and expert systems (Delen and Demirkan 2013).

2.1.2 Suppliers Efficiency Evaluation

Forker (1997) stated that "a firm's output can be only as good as the quality of its inputs". The author uses this as an explanation for the fact that both process management and management tools practiced by the supplier play crucial roles for a company's success.

Koprulu and Albayrakoglu (2007) submitted in their research an approach to vendor selection, that assigns six variables to the ranking criteria: cost, quality, delivery, flexibility, innovation and trust.

Another method for supplier evaluation, but already targeting fashion industry, is suggested by Jia et al. (2015), where for the evaluation computation, three different type of group factors are considered: economic (cost, quality, on-time delivery, rejection rate), environmental (toxic chemical usage, water consumption, energy usage, pollution) and social (under age labor, working hour, human rights care and workers' heath monitoring). This methodology's goal is to provide a tool that allows the decision maker to select the most sustainable supplier, according to the criterion and weights variables chosen.

2.2 Data Preparation

The current subchapter denotes some literature findings concerning the phase before modeling, from the problem formulation until the preprocessing of the data. Thus, this chapter starts reviewing frameworks that support the methodologies afterwards mentioned, following a top-down approach.

2.2.1 Overview of Data Analysis Methodology

Data mining is not only a technique that aims to provide knowledge discovery but also the ability to use it (Ian H. Witten 2006). In order to standardize the data mining process, Wirth and Hipp (2000) introduces a framework that can be applied in different fields but where the purpose is the same: achievement of relevant insights from the data.

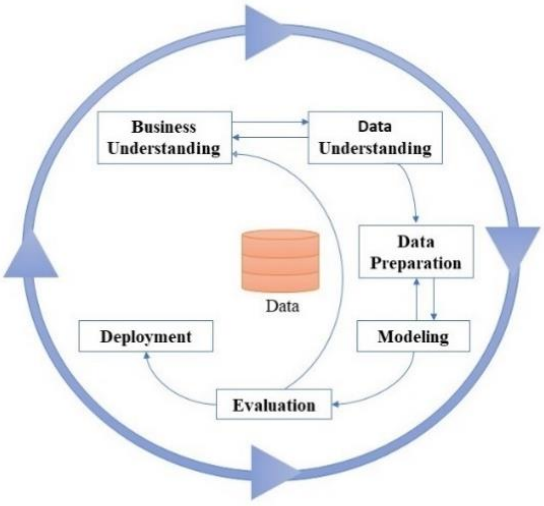


Figure 1: Data mining procedure adapted from Wirth and Hipp (2000)

The starting point illustrated in Figure 1 is the understanding of the business, where the goals from the analysis are defined in a way that assures the alignment with the business scope, intending to guide the subsequent steps.

After this, we have the data understanding stage that includes procedures like data gathering and a preliminary analysis. This analysis is used to identify valuable/worth of exploring data and check its quality. Finishing this, all the requirements to properly define the project are set.

The next step is data preparation which is the process where to apply the manipulations needed to ensure that a dataset is ready to the next phase: modeling. The models to apply to the data that provide the knowledge wanted are chosen and their parameters are tuned. With the output of that phase, it is necessary to analyze its feasibility and performance through certain metrics (evaluation). Finally, deployment in this context comprises the operationalization of the use of the models created.

Data mining is then presented as a continuous, iterative and incremental process, where steps may be repeated.

Already in an oriented approach of data mining to the process industry, a perspective by Ge et al. (2017) encompasses a set of sequential steps: data preparation, data preprocessing, modeling and performance assessment and, the last step, data mining and analytics applied to the trained model that then may lead to an output of knowledge discovery.

In fact, both perspectives seem to match in the sequence of the approach, even though the concepts used seem to diverge. While in the first mentioned the includes processes such as missing and noisy data handling in the data preparation phase, the latter includes it on the data preprocessing phase. However, it seems to exist a similarity between the order and type of processes performed.

2.2.2 Data Preprocessing

Quality of data in databases is an important problem since it affects the performance of mining procedure (Han and Kamber 2006). Thus, the preprocessing phase is highly relevant to step in, even though it is time consuming, since it occupies the majority of the time along the process (Pyle 1999).

Different types of solutions exist to improve data quality, depending on the quality issue to be studied. Han and Kamber (2006) divides preprocessing in five categories: (i) descriptive data analysis, (ii) data cleaning, (iii) data integration and transformation, (iv) data reduction and (v) data discretization.

Descriptive Data Analysis

Before starting to modify the data it is important to have an overall picture of what information the dataset contains and type of problems that can be solved with this data (Ian H. Witten 2006). For that, statistical measures such as central tendency and dispersion are used (Han and Kamber 2006). A particular type of data analysis that can be conducted is time series analysis, where trend and seasonality can be detected (Brockwell and Davi 2002).

Data Cleaning

This process includes the handling of noisy, inconsistent and missing data. Actually, this can be applied not only in the preprocessing but also in the data analysis phase (Hui Xiong).

The issue considered as the most challenging by Ian H. Witten (2006) is noisy data. This can be defined as values that do not fit in the distribution of that attribute (García et al. 2016). Detecting outliers can be performed by distance measures (removing data points that are far from a certain threshold) or even by analyzing the probability density function of the variable. Binning, regression and clustering can also be used to detect outliers.

Data Integration and Transformation

Joining data from different sources of information can be stated as the goal of data integration. Transformation includes manipulation of the data representation or value. Some examples may be: normalization, feature construction based on other attributes or even smoothing techniques for reducing noisy data (Han and Kamber 2006).

Data Reduction

Decreasing the data volume with certain techniques also represents an interesting tool to allow the integrity maintenance of the information within the data and, at the same time, enhance the performance of the application of prediction algorithm. Han and Kamber (2006) addresses 5 different methods: (i) data aggregation; (ii) feature selection, (iii) dimensionality reduction, (iv) numerosity reduction and (v) discretization.

Concerning dimensionality reduction, Principal Components Analysis is one of the methods and consists in the ranking of the dimensional subspace/line *u* accordingly to the variance projection. I.e., being *D* a set of points of interest to study, the linear direction of *u* that best explains its variance will be the first principal component and consequently the second best one the second principal component and so on. To capture the components that best explain a non-linear direction in the output, Kernel Principal Component Analysis is used, where non-linear transformations are applied to the inputs leading to a set of linear combinations in the feature space (Zaki and Jr. 2014).

2.2.3 Feature Selection

Being one of the methods described in section 2.2.2 for data reduction, feature selection may be an enabler for enhancing model's performance. Silva and Leong (2015) distinguishes three approaches: (i) filters (selection of features from the set without any type evaluation of this extraction), (ii) wrappers (learning algorithms to evaluate if the features filtered are relevant), (iii) embedded (feature selection is performed during the learning process) and (iv) hybrid approaches (that use both filtering and wrapping techniques).

Methods such as Pearson's correlation coefficient, information gain, mRMR, relief score and consistency-based filters belong to the filter category (Silva and Leong 2015). The wrapper approach includes metaheuristics (genetic algorithms) and heuristics search algorithms (sequential search). Embedded types can be, for example, using support Vector Machine, and, using this classification algorithm, excluding the features taking into account their importance on the prediction task (Tsikriktsis 2005). Due to computational effort reasons and simplicity in outputs interpretation, filters are the category more in-depth analyzed in this work.

Correlation is a dependency measure and applicable in univariate analysis. However, this method does not detect "spurious correlations": when two variables are considered by this coefficient as highly correlated but, in fact, there is no cause-effect relation between them, they are actually irrelevant to each other behavior.

Information gain starts by choosing a feature depending on its gain to the prediction: selects the features that are more relevant to the target variable. The type of analysis here can be classified as an univariate analysis, which does not consider the redundancy within features (Tang, Alelyani, and Liu 2014).

Relief score method is also an univariate type of method (Tang, Alelyani, and Liu 2014) based on the capability that a feature has of separating instances between classes. One of the drawbacks of this methods is that it is likely not to detect redundancy (Silva and Leong 2015). Finally, consistency-based types relies on the consistency of the attribute values versus the class label. However, often admits that a certain variable is relevant, when it is not actually important to classify the class label.

The importance of a multivariate analysis and mRMR

In the context of univariate analysis, Cover (1974) exemplifies, with two different experiments, their relevance to the classification of a certain item. Considering the experiments $X = \{X_1, X_2\}$ and $P_e(X_j)$ as the error probability of X_j, where $j = \{1,2\}$, the author used an example where $P_e(X_1) < P_e(X_2)$. Thus, the experiment X_1, to classify a certain variable *V*, seems to incur in a lower error than X_2. However, when repeating the experiment measures, noted as X_j', the conclusion was that $P_e(X_2, X_2') < P_e(X_1, X_2) < P_e(X_1, X_1')$. Therefore, if just one experiment is tested, the best one is X_1. On the other case where two experiments are allowed, the repetition X_2, X_2' provides a lower error probability. The analogy with feature selection appears in the literature, stating that features selected individually may not integrate the best group of features that more accurately classify or predict a certain variable (Peng, Long, and Ding 2005).

Maximum-Relevance-Minimum-Redundancy (mRMR) may be a possible solution for this, since it includes redundancy as an important measure for feature selection. To begin, mutual information is explained: being S a group of j features, such as $S = \{s_1, \dots, s_j\}$. The mutual information between both can be defined as expression 2.1 shows:

$$I(s_1, s_2) = \iint p(s_1, s_2) \log \frac{p(s_1, s_2)}{p(s_1)p(s_2)} ds_1 ds_2 \quad (2.1)$$

, where $p(s_1)$ and $p(s_2)$ represents the probability density function of s_1 and s_2 and $p(s_1, s_2)$ represents the joint density of the pair of features.

From this, maximum relevance is defined by, first, computing mutual information between all items of the variables. After, these values are summed, and a rank is computed by the mean value of the mutual information between all values of the feature and the class label c.

$$Relevance = \frac{1}{|S|} \sum_{s_j} I(s_j, c) \quad (2.2)$$

, where s_j with the higher value is considered the most relevant, being $|S|$ the number of features of S. In order not to choose relevant variables that are redundant between them, the concept of minimum redundancy is explained by Ding and Peng (2003). It is based on the mean of the relevance between two features. Thus,

$$Redundancy = \frac{1}{|S|} \times \frac{1}{|S|} \sum_{s_i, s_k} I(s_i, s_k) = \frac{1}{|S|^2} \sum_{s_i, s_k} I(s_i, s_k) \quad (2.3)$$

, where s_i, s_k are two random features of S. As lower as this value is, the lower mutual information the variables have between them. Hence, less redundant are. The mRMR will choose the group of features that maximize $\varphi = (Relevance - Redundancy)$. The results achieved in this study stated that, when compared with maximum relevance methods, mRMR offered, to the majority of cases analyzed, lower error rate for classification.

As a great number of datasets include mixed types of data (categorical and continuous), it is relevant to analyze if the aforementioned methods are accurate for these cases (Doquire and Verleysen 2011). Actually, φ of categorical and continuous data cannot be directly comparable. Hence, the author states that by: (i) separating rank lists for both types of data and then (ii) evaluating their accuracy in the prediction step is a more reliable way of selecting features. This procedure can be considered a wrapper technique, since it uses prediction in order to compare and rank both lists. However, the author states that, while exhaustive models (embedded exhaustive search) generate $2^{|S|}$ models, this method will create a maximum of $|S| - 1$ models, requiring less computational effort.

2.3 Methods for Vendor Service Level Prediction

This chapter introduces some practices found in the literature that are compatible with the resolution of part of the current project: prediction. For that, it is firstly useful to categorize the type of problem tackling: supervised learning. The purpose is to identify patterns within the data and use this knowledge to predict unknown data (Rokach 2009).

2.3.1 Linear Regression and the Shrinkage Concept

Linear regressors are simple models that have the advantage of their interpretability, being proven to offer better performances when compared with more complex models, in some cases. Defining $X = \{x_1, \dots, x_F\}$ as a vector with F elements, the liner model would be built in the form shown in equation 2.4:

$$f(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_F x_F \quad (2.4)$$

These $\beta_j, j \in [0, F]$ are the coefficients of the variables *X*. The analogy with prediction task can be explained by taking *X* as the value of a set of *F* features of a certain dataset, *f(x)* the prediction from the linear model and, finally, β_j the weight of the *j* feature on the model.

Since the purpose is to minimize the prediction error, the solution for the coefficient values is computed, most commonly, using least squares, which aim is to minimize the residual sum of squares. Being training data characterized by $\{x_{11}, \ldots, x_{NF}\}$, where x_{NF} is the value of feature *F* for the *N*-th instance and $y = \{y_1, \ldots, y_N\}$ the target variable, the minimization of the residual sum of squares is given by expression 2.5, which represents the goal of the model fitting (Hastie, Tibshirani, and Friedman 2001).

$$Minimize\ RSS = Minimize \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{F} x_{ij}\beta_j)^2 \quad (2.5)$$

As noticeable from the previous explanation of linear models, these do not include any penalization for the weight given to each variable: there is no limit for β_j. Plus, rarely are the weights assigned null values, which makes the interpretability of the model complex, especially when the number of estimates/features is large (Hastie, Tibshirani, and Wainwright 2015). Hence, in order to solve the current problem, introducing a limit to the feature importance on the model seems to be an interesting analysis to perform.

For that, Lasso regression provides this by multiplying λ (penalty) with the sum of β_j norms. These coefficients can be null and hence Lasso is able to perform feature selection while minimizing the prediction error by the expression 2.6.

$$\hat{\beta} = \min_{\beta_0, \beta_j} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{F} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{F} |\beta_j| \right\} \quad (2.6)$$

If λ null, the expression is equivalent to a linear model. In fact, a difference between both is the constraint 2.7 relative to the equation 2.6.

$$\sum_{j=1}^{F} \beta_j \leq t \quad (2.7)$$

Another shrinkage methods exists, such as the Ridge regression, in which in the constraint 2.7 β_j is replaced by β_j^2. Because this technique is not able to assign null values to the coefficients, it includes all variables in the model, not discarding features (Hastie, Tibshirani, and Friedman 2001).

2.3.2 Support Vector Regressor

The concept behind this model first appeared in classification problems, and then was adapted for regression (Djouama et al. 2016). This learning model starts by defining a function (a convex ε-insensitive loss function) that penalizes under and overestimates that surpass a distance ε from the real output. If this penalization is equal for both under and overestimates, the loss function is symmetrical. This function is "insensitive" since it does not penalize estimates that are positioned along the width of the tube: ε. This region is then built around the estimated function that will estimate the output *f(x)* in which the data points that belong to this tube are not considered as error estimates, but the ones outside are penalized. This area must be as small as possible, while containing the highest possible number of data points. On the other

hand, the loss function (that measures the misestimates) must be minimized, forming a multi-objective problem.

Analytically specifying (the case where *f(x)* is linear): being $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ the training data, *M* the order of the polynomial that is chosen to estimate the function, $\langle w, x \rangle$ the dot product between *w* and *x* and $\|w\|$ the Euclidean norm of *w*:

$$f(x) = \langle w, x \rangle + b, x, w \in \mathbb{R}^M \tag{2.8}$$

Since *w* defines the flatness of *f(x)*, the first goal can be represented as:

$$\min \frac{1}{2} \|w\|^2 \tag{2.9}$$

Plus, the other goal is the minimization of the estimates error, which is characterized by expression 2.10:

$$C \sum_{i=1}^{N} \xi_i + \xi_i^* \tag{2.10}$$

Figure 2 represents a one-dimensional linear SVR in which ξ_i, ξ_i^* are slack variables that specify how many points are acceptable outside the ε-tube. The illustrated potential support vectors refers to the training data points that can modify the hyperplane position if removed.

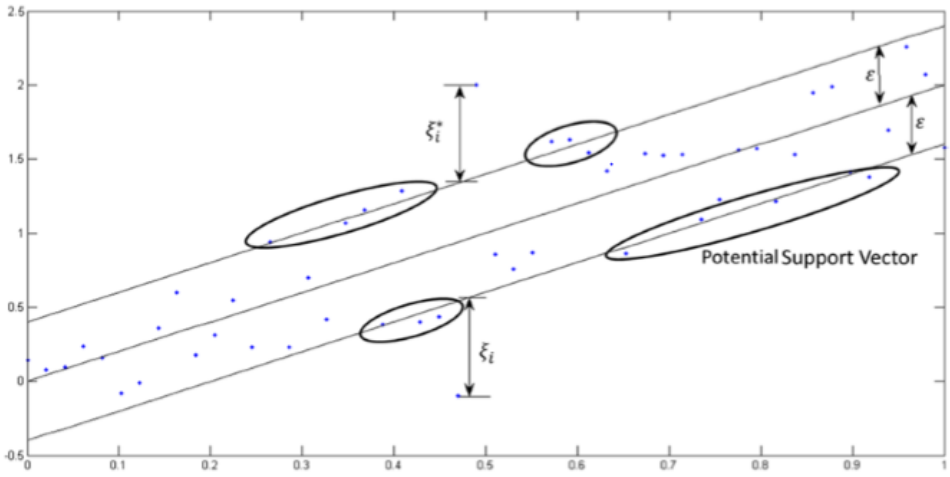


Figure 2 One dimension Linear SVR

(Awad and Khanna 2015)

Hence, the final optimization problem can be defined as:

$$\text{Minimize } \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i + \xi_i^* \tag{2.11}$$

$$\text{s.t. } y_i - \langle w, x \rangle - b \leq \varepsilon + \xi_i^* \tag{2.12}$$

$$\langle w, x \rangle + b - y_i \geq \varepsilon + \xi_i \tag{2.13}$$

$$\xi_i^*, \xi_i \geq 0 \tag{2.14}$$

And the loss function:

$$|\xi|_\varepsilon = \begin{cases} 0, & |\xi| < \varepsilon \\ |\xi| - \varepsilon, & otherwise \end{cases} \tag{2.15}$$

The trade-off between the flatness of the function and the error can be tuned by the hyperparameter *C*. Increasing *C* means giving more importance to the error. The scope of this method also covers non-linear functions, where kernels are used to map the data, allowing their representation in a higher dimensional space (Awad and Khanna 2015). By constructing this optimization problem, SVR method offers a generalized error minimization, by focusing on the area described before instead the overall training error (Basak et al. 2007).

2.3.3 Ensemble-based Models

One of the main advantages of ensemble-based models is their capacity for improving prediction performance (Rokach 2009). These types of models consider various predictors/classifiers, which "vote" on the final output. There are four main components in this type of methodology: training dataset, inducer, diversity generator and a combiner. The former represents a part or all of the original dataset that is already labeled and will be used to train the model. An inducer can be defined as the algorithm that will receive as an input the training dataset and will analyze the relationship between the various attributes and the target variable. The diversity generator aims the diversification within the learning models, in order to enhance ensemble efficiency. Finally, combining methods will, from the different learning model outputs, retrieve only one output (Maimon and Rokach 2005; Rokach 2009).

The use of these models' outputs to decide the final output can be conducted in a dependent way (when the prediction process is sequential and is guided from the previous outputs from predictors) or independently (the prediction output from all predictors is considered in one single step).

Boosting-based models

AdaBoost is a dependent and model-guided instance selection model since it uses the prior classifier output to focus on the misclassified instances in the next step, by changing the train dataset giving more weight to these. The first stage includes running a weak learner on the training dataset in which each pattern is assigned the same weight. Then, after analyzing the misclassified classes, these weights will increase on the misclassified ones, directing the next iteration to the focus on those. This is applied to binary classification, but in order to multiclass classification, the version AdaBoost.M1 or AdaBoost.M2 can be used. These models are recognized by enhancing the performance in comparison to simple weak learner classification, since these weak learners are combined and result in a strong learner. This is achieved by iteratively improving the classification accuracy.

However, if many iterations are computed, then the model is prone to overfit, leading to inaccurate results. This can be rounded by maintaining the iteration number parameter low. An additional improvement for the weights given for multiclass classification model AdaBoost.M1 is the BoostMA and AdaBoost.M1 W (Rokach 2009).

Another type of boosting model is the stochastic gradient boosting in which it is iteratively performed an improvement towards the previous iterations, using the mean square error as a cost function (Friedman 2002).

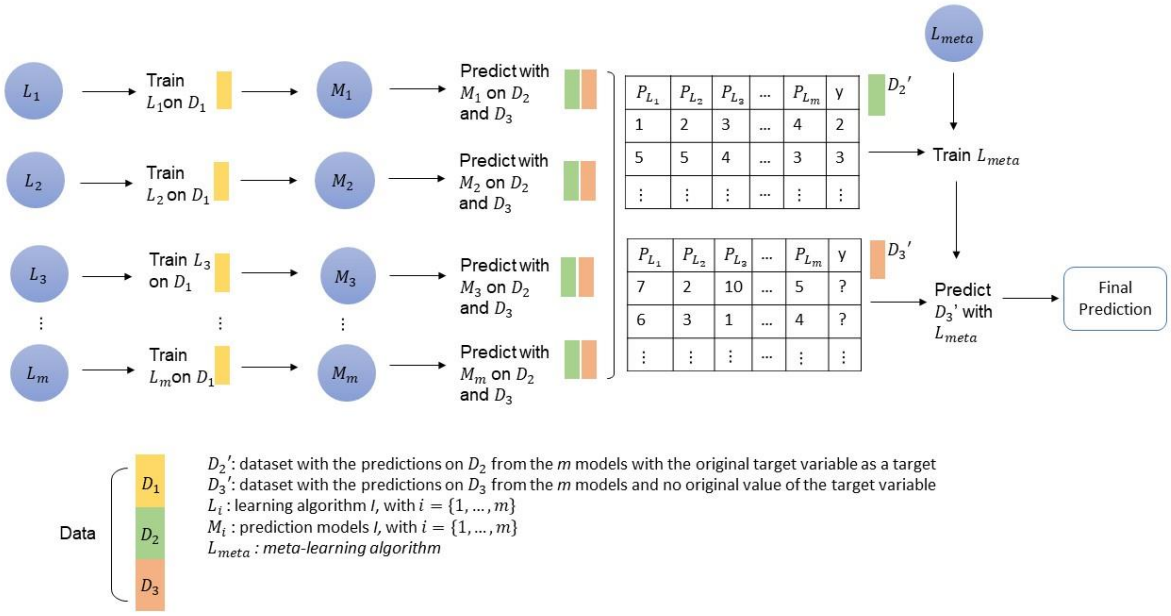
Bagging-based models

Also named as bootstrap aggregating technique, bagging starts by selecting instances from the original dataset, with replacement. The group of these instances will form the training dataset, from which the predictor will be built, and its output stored. The next iteration will choose the same size training dataset, using sampling with replacement. This will result in different training datasets with the same size, where duplicated instances can appear or sometimes, an instance may not appear in any of the datasets. This will be performed until the stopping criteria is met (maximum number of iterations). The output will be given by the

composite bagger classifier as the most voted prediction. Just like boosting, bagging also requires a weak learner as inducer (Rokach 2009). The similarity with the Random Forest algorithm can be perceived when explaining that in this case the inducer is a decision tree and the attributes that will be analyzed to decide at a tree node will be the same as the ones from the original dataset.

Stacking

The main idea behind this technique is to introduce an algorithm that will learn from the predictions/classifications performed by a set of learning algorithms. I.e., a set of classifiers or regressors are built from several learning models leading to an output: a dataset with their predictions, to which the original target variable value is added. A meta-learner (a standard learning algorithm) is then trained on this last dataset. When an instance without a correspondent target value is subject of a prediction task, the same logic is applied but without training: the predictions from the first level models on this data are stored and then the meta-learner will give the final output. With this, the purpose is to detect in which target variable spectrum the models perform best and worst and managing the weights of their predictions on the final output from there (Ren, Zhang, and Suganthan). Figure 3 identifies the necessary steps to apply this method.



P_{L_1}	P_{L_2}	P_{L_3}	...	P_{L_m}	y
1	2	3	...	4	2
5	5	4	...	3	3
⋮	⋮	⋮	...	⋮	⋮

P_{L_1}	P_{L_2}	P_{L_3}	...	P_{L_m}	y
7	2	10	...	5	?
6	3	1	...	4	?
⋮	⋮	⋮	...	⋮	⋮

Figure 3 Stacking method adapted from GitHub

Choosing base learners that will be included in this ensemble may also be relevant for the current problem. In fact, they should be diverse to enhance their generalization ability, which can be measured by the error correlation between models (Aldave and Dussault 2014). There is a final goal of achieving a trade-off between bias and variance in this type of models. This is explained in section 2.3.4.

2.3.4 Model Validation and Selection

Measuring model performance simply by the error rate of the prediction model once in all training data is not reliable, since it may be biased and hence give an optimistic value (Ian H. Witten 2006).

A possible way of avoiding that is the hold out method, which splits the dataset into a training and a testing sets. A stratified holdout chooses this *k* datasets guaranteeing that they are approximately representative of the original dataset. The most common is the 10 *k*-fold

cross-validation, where the dataset is divided, with a stratified holdout method, into 10 different datasets (Ian H. Witten 2006). In each of the total 10 iterations, 9 of the datasets are used for training the model while the one left out is used as test dataset to evaluate the model's performance. On the next iteration, another set is left out, instead of the previous one, which will make part of the 9 sets for training. This method was then elected for the current thesis: the validation step is performed using *k*-fold cross-validation.

However, if just computed once, this method outputs may be biased. Krstajic et al. (2014) shows that performing a repeated *k*-fold cross-validation provides more reliable results, not only for measuring prediction error, but also searching and selecting the models' hyperparameters. If the partition is random, this procedure repetition may generate different folds in each repetition, being the final error estimate the mean of those.

Another topic worth highlighting is how to compare models' results and how to choose the best. For that, and because prediction error is a common variable that is being included in all models described, it is relevant to understand its constructing parts.

Estimate error can be divided in three parts: bias, variance and irreducible error. Considering a function $Y = f(x) + \mathcal{E}$, the error can be characterized by expression 2.16, where $f(x_0)$ is the target mean.

$$error(x_0) = E\left[\left(Y - \hat{f}(x_0)\right)^2 \middle| X = x_0\right] \Leftrightarrow$$
$$error(x_0) = \sigma_\varepsilon^2 + \left[E\hat{f}(x_0) - f(x_0)\right]^2 + E\left[\hat{f}(x_0) - E\hat{f}(x_0)\right]^2 \Leftrightarrow$$
$$error(x_0) = irreducible\ error + bias^2 + variance \qquad (2.16)$$

where the irreducible error represents the discrepancy between the target variable and its mean. The following term, bias, is the difference between the expected value of the estimation and the real target mean. Lastly, variance is the squared expected discrepancy of the estimated value and its mean (Hastie, Tibshirani, and Friedman 2001).

And the interest derived from this analysis arises by the inference concerning model complexity: the higher the complexity, the higher the variance and the lower the bias. In an optimal scenario, a balance between bias and variance should be found and the reasonable values for these terms depend on the data that is being used for modeling. A more complex model may detect more accurately the limits of the solution. On the other hand, it is more likely to overfit, capturing the nuances on the data and thus, tend to model those, leading to a higher variance (Zaki and Jr. 2014). The optimal predictor is able to provide a low bias and variance and can be estimated by early stopping, using the validation set and stopping training when the generalization error on the validation set is the smallest (Hansen 2000).

To conclude this topic, model comparison should also include the error of the error estimate. I.e., compare if the means of the prediction error from two different models are significantly different. For that, Ian H. Witten (2006) assumes a Student's distribution for the mean of the error estimates generated by a determined learning model. Firstly, $d = x_k - y_k$ is computed, where x_k is the prediction error of a certain learning model on the fold k and y_k is the prediction error using another learning model. Then, a t-student score is computed using the mean of the difference $\bar{d} = \bar{x} - \bar{y}$ between means of the prediction errors on *k*-fold cross validation from two different models. Equation 2.17 shows the formula to compute

$$t = \frac{\bar{d}}{\sqrt{\frac{\sigma_{\bar{d}}^2}{k}}} \qquad (2.17)$$

, where σ_d^2 refers to the variance of the variable *d*. Afterwards, a two-tailed hypothesis test is formulated, being the null hypothesis "there are statistical evidence that $\bar{x}$ is not significantly different from $\bar{y}$" and the alternative hypothesis being "there are statistical evidences that $\bar{x}$ is significantly different from $\bar{y}$". Based on the degrees of freedom *k*-1, z-score is computed. If *t* is higher than *z* or lower than -*z*, null hypothesis is rejected. Thus, model's performance can be considered as statistically different.

3 Problem Context

This work was developed in an industrial environment, in a company named HUUB. The company was founded in 2015 and is a startup focused on guaranteeing logistics service to brand owners, while providing also supply chain management insights. The customer segment of the company is dominated by the kids' fashion industry. Its strategic plan is to continue targeting this specific market. The company is the link between different entities along the supply chain: vendors, carriers, brand owners and final customers. Figure 4 shows this interaction in more detail.

The company's central source of information is *Spoke*: a platform that is connected to the database in which stakeholders such as employees and brands (HUUBs' clients) have access and contribute to.

3.1 Organizational Structure

Before getting into detail about the information flow, it may be relevant to describe the internal organization of the company. This is divided in six departments: Account Management, Business Intelligence & Artificial Intelligence, Financial & Human Resources, Information & Technology, Marketing & Sales and lastly, Operations.

Account Management department is the main responsible for the communication with the client when it concerns to the assurance of a good onboarding process, support of the brands' use of *Spoke* platform, report of the operations status and, finally, business insights that may be helpful for their performance as a brand.

Business Intelligence & Artificial Intelligence section is mainly focused on the knowledge extraction from the data collected. This department develops tasks like descriptive, predictive and also prescriptive analytics. The main purpose is to improve the performance of HUUB, through business insights extracted from the data.

Financial & Human Resources main activities include both the financial control and key performance indicators control of the company. Plus, this department is also responsible for the management of the human resources, meaning all bureaucracy inherent.

Information & Technology is responsible for the maintenance and the development of the *Spoke* platform. This includes the continuous support for the current version of the platform and the improvement and development of new features and functionalities.

Marketing & Sales represents the prevailing point of contact with new possible clients, by contacting with potential clients, communicating the company's service and promote it in fairs or direct contact to the client.

Operations oversees the warehouse management, meaning this the control of the inbounds, outbounds, packaging & packing and labeling in order to ensure the fulfillments of customer orders. A warehouse planning is performed, in each season, concerning the estimated quantities and dates for the receptions from the suppliers. This information is communicated to HUUB by the brand, which contracted these terms directly with the vendor. Thus, this information is not directly communicated from the vendor to HUUB.

3.2 Business Understanding

Finding its value proposition pillars on the offer of a set of activities related with logistics services, the company provides the link between distant points of the supply chain: from the vendor to the final customers. From now on, in this dissertation, the term client adopts the

meaning of HUUBs' clients (brand owners) and customers are the clients of HUUB clients (final customer).

Hence, aligned with this, clients' satisfaction towards what is provided depends on the fulfilment of their customers' sales orders, meaning the whole distribution management of its goods in the time interval agreed upon, with some inherent quality standards.

3.2.1 Operations Planning Current Approach

Warehouse activities are planned, at the beginning of each season, based on the information received by the brands packing lists and estimated dates for receptions. Several warehouse workers are temporary, and their schedule is planned considering that information. This planning is performed by an optimization algorithm, based on linear programming, that allocates warehouse resources and which input is the information provided by the vendor and the brand owners.

Labeling is a process that can be performed by HUUB or by the vendor, depending on the type of agreement performed. Labeling that is of HUUBs' responsibility is also settled and added to the service provided and thus charged. An *Excel* sheet is prepared with these dates and respective quantities, being the management of the activities based on that information.

To understand the interaction between vendors and HUUB, Figure 4 maps the most relevant processes. This can be interpreted as a cycle which trigger is the purchase order done by the brand and ends with the outbound of the goods to the final customer. Some of the processes are noted below:

Purchase order is submitted by the brand. Here, the terms are agreed upon (delivery date, quantity and type of products) with the vendor and afterwards communicated to HUUB. Information about products (quantity and type) is organized into a packing list (PL), even though it is not always transmitted to HUUB. Sometimes, the company is not aware of the products it will receive nor their quantities.

Concerning the purchase order: this can be divided into a single reception or multiple ones, depending on the terms. Thus, estimated dates for goods delivery can be different for distinct receptions of the same purchase order.

Sales order is an order made by a customer to the brand. The list of products and final customer data such as address and contact are known to HUUB. Thus, the products are prepared (packaged and packed) and then shipment is outsourced by a company that will pick these products and distribute it to the final customer.

Pick to Stock, Pick from Stock and Pick to Split can be nominated as processes in which the goods are handled. Picking to stock is when a product is placed on a certain warehouse location and this information is saved on *Spoke*. A product picked from stock is one moved from their location of the warehouse to prepare to the shipment outbound. Lastly, picking to split occurs when product(s) go directly from the reception process to pack & packaging.

Pack and Packaging include the specific preparation depending on the brand. I.e., packs, flyers, stickers and packaging are determined by the brand, being this a personalized process per brand.

Reception represents the handling of the items received. This includes the registration of the dates and quantity of items received. After this, if products are already labeled and there is no fault on those labels, they will be then picked to stock (PTS) or picked to split (PTSP). However, if there is a need for labeling, that is the following step after reception. Then, products are ready to the next phase: PTS or PTSP. PTS happens if the products received do not need to be shipped right after, so they are placed on warehouse shelves as stock. In cases where there

is a sales order from a customer, then the goods are directed to the packing and packaging process. To this it is called *Split*. Part or all items of the reception can go directly to the process of packing and packaging (P&P) and this is described as PTSP. Besides above-mentioned processes, a reception may need to be returned to the vendor in case there are faulty items or no packing (polybag, for instance) for the products when it is mandatory.

Labeling is a process that just occurs in two scenarios: the brand agreed with HUUB that this is a process done in the warehouse; or bad labeling/lack of label due to vendors' fault. In the second case, this process involves the identification of the product, label printing and their application.

Shipment handling requires sales order details (customer address, quantity, products type and date for shipment). With this information, outbound is managed to fulfill customer sales order. This is a process dependent of all others before mentioned: if there is a delay on a reception, shipment outbound can be affected.

Having now a more detailed view of warehouse activities, subsection 3.2.2 explains vendors' impact on the business.

3.2.2 Vendors Impact on the Business

A set of issues related with vendors' activities affect both processes and profits in the company. For instance, delays of goods delivery are common and have a significant impact on HUUBs' warehouse planning and can even put at risk products availability at the final customer in the planned time. This can compromise one of the ultimate goals of the supply chain management: customer satisfaction. In addition, products conformity was detected as one of the causes for product returns, meaning this incurs additional expense, time and material. Plus, final customer satisfaction can also be influenced by these types of incidents. Again, the dots are all connected and can impact company's profitability, being therefore an area of interest to analyze. These non-planned events drive to the loss of resources time without any type of reward, emerging here another improvement opportunity.

From figure 4, it is noticeable that the issues addressed above can occur in multiple processes along the supply chain. A non-on-time delivery can destabilize the normal warehouse activity, since the planed tasks have to be rescheduled in order to process the reception when it is delivered. In the same logic, processing more/less than the expected quantity of goods can also lead to extra time spent reorganizing the activities. Plus, a product return from the final customer due to vendors' fault, for instance, must pass through many processes, incrementing the non-valuable time spent.

A similar analysis can be described concerning bad labeling problems from the vendor, which can imply time lost to detect the problem in the reception phase and the surplus of planned time to decide whether to relabel or to return to the supplier. This last situation is analogous to the non-conform items detected at the warehouse. Handling these events with the best strategic plan is difficult in terms of planning and management (Sahay and Ranjan 2008).

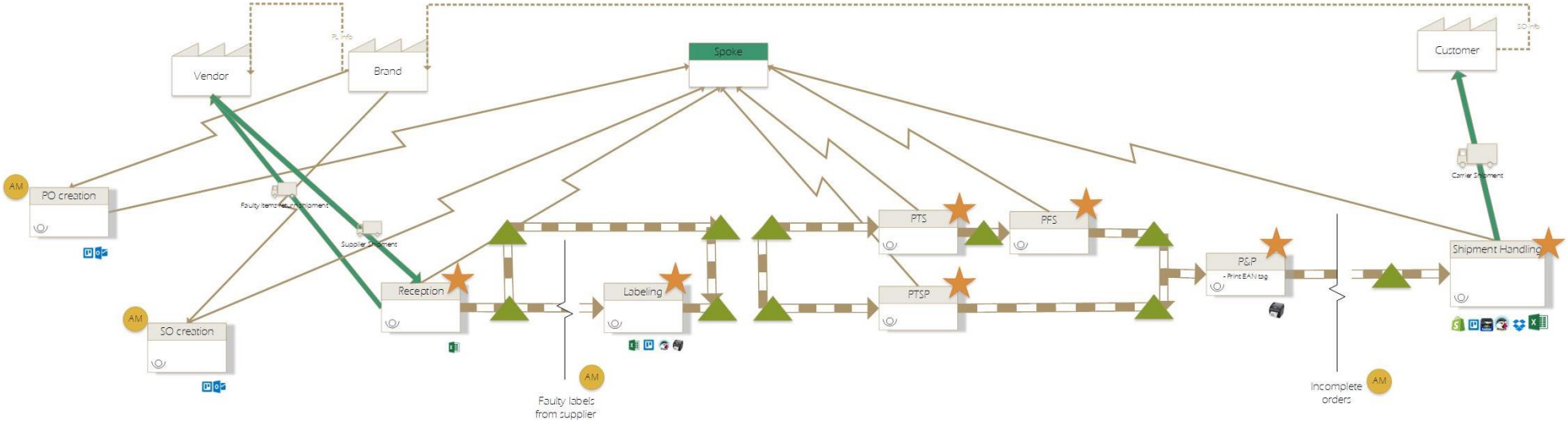


Figure 4 Warehouse value stream map

Therefore, HUUBs' interest in implementing procedures that help the brand to deal with these problems triggered by the vendor is to join an additional competence to the catalogue: provide a powerful group of advices that can be valuable for HUUB clients' performance. The detection of these issues is done as they appear during daily activities, but they are usually undocumented or stored in an unorganized fashion making retrieval difficult. More concretely, part of the information is stored in a database that was not designed for this purpose. Additionally, formatting is not standardized, making it more difficult to extract the data.

Finally, data duplication exists. The same or related information can be found in several sources and expressed both qualitatively and quantitatively. All this leads to the need for an intense data preprocessing phase.

The methodology followed in this step was highly related with the understanding of the business. Since the aim of the current project is not only to reduce HUUBs' costs but also to provide the band owners with a clear vision of what is happening concerning vendors and its impact on their performance. Accordingly, in this analysis, the information from different departments was collected, organized and categorized into four classes of vendor related issues (see Table 1).

Table 1 Identification of vendors faults

Class	Sub Class	Fault Level	HUUB cost risks	Brands' risks	Relation with returns
Labeling	No label	1	Time to label Return to supplier	Unavailable stock	
	Wrong label	2	Time to relabel Return to supplier	Unavailable stock Decrease customer satisfaction	Wrong item received
Packaging	No polybag	1	Return to supplier	Unavailable stock of the goods to fulfill customers' orders	
	Faulty polybag	2	Return to supplier	Unavailable stock Decrease customer satisfaction	Faulty item received
Variance reception date		2	Re-planning warehouse activities Unavailable resources	Unavailable stock Decrease customer satisfaction	Canceled sales order due to delay of delivery to the final customer
Variance quantity delivered		2	Re-planning warehouse activities Time to notify the brand Unavailable resources	Unavailable stock Decrease customer satisfaction	Canceled sales order due to delay of delivery to the final customer

Fault levels were defined depending on their likelihood to impact customer satisfaction. A fault that is highly likely to result in a return is considered to be the worst type of fault (level 2). On the other hand, faults that are less likely to be perceived by the customer are assigned a level 1.

Returns causes identification

As returns due to vendors fault are a cause of the classes described in Table 2, these are studied not as a primary fault, but as a consequence. VSM position means where in the Value Stream Map of the Figure 4, the return reasons may be detected.

Table 2 Identification of returns' reasons

Return reason	Possible Cause	VSM position
Faulty item received	Faulty item delivered by vendor	Customer&Logistics' company
	Carrier or logistics' company fault	
Wrong item received	Wrong label by vendor	Customer&Logistics' company
	Wrong label by logistics' company	
	Carrier fault	
No packaging	No packaging from vendor	Logistics' company
Faulty packaging	Faulty packaging delivered by vendor	Logistics' company
		Customer
	Damaged by carrier or logistics' company	
No pack	No pack from vendor	Logistics' company
Faulty pack	Faulty pack delivered by vendor	Logistics' company
	Damaged by carrier or logistics' company	
Delay	Vendor Delay on the delivery	Logistics' company
		Customer
	Carrier fault	
	Logistics' company fault	
No label	Vendor fault	Logistics' company
	Logistics' company fault	

It is useful also to highlight that it may be possible an interrelation between return causes. For instance, a faulty packaging can lead to a faulty item received. Plus, it can happen that the return reason is not only one of the mentioned-on table 2, but many. Thus, the possibilities are $\sum_{k=1}^{8}\binom{n}{k}$, $n = 8$. An improvement opportunity in terms of data collection was found here, concerning the standardization of return cause.

3.2.3 Data Storage

The data gathering step led to the detection of some incoherencies and missing data. Even though there is a flow of information between processes and *Spoke*, this data flow is frequently not carried out and when it is, one may doubt its reliability. Some cases are detailed below.

Receptions' properties include estimated delivery date agreed with the vendor, quantity and real delivery date. Estimated reception date and quantity are inserted on *Spoke* by Account

Managers and are fields that can be updated after the first insertion. Real delivery date field is sometimes inserted by Account Managers, other times by Warehouse Operators, in an *Excel* file or can be found in another platform named *Trello* that is used to communicate. Despite the various sources of data implemented in the company, the real delivery date is rarely stored. Only 180 receptions of a total of 1460 have their information available concerning the delivery real dates. To summarize, there are three main sources from which this data can be extracted: warehouse *Excel* sheet, *Spoke* and *Trello*.

Products' properties represent product family, subfamily, type, season and material. From this set of attributes, material is the only field that is not standardized. I.e., there is no standard for how to insert a certain type of material. Figure 5 shows some examples of this field.

6	Main fabric : 65%PES,33%CV,2%EL
7	43% WO 31% CO 15% PA 5%PC 45% WO 33% CO 15% PA 5% PC
8	100% Cotton Jersey
9	75% Cotton, 22% Polyamide, 3% Elasthanne

Figure 5 Model material database slice

Since it was relevant for this project to extract this type of data, a need for a standardization of this data seemed to be a relevant procedure to enable future data manipulation.

Labeling data (date, number of labels, vendor to which the reception is linked to) is stored by warehouse operators, in an *Excel* file (see Table 3).

Table 3 Warehouse Excel Labeling Data

Client	Vendor	Labeling Date	# Labels	Type
5	13	15/05/2017	155	Wrong label
8	2	18/05/2017	200	Wrong label
7	25	23/06/2017	2	No label
6	34	01/08/2017	30	Wrong label

However, there are other situations in which Account Managers, while inserting notes on *Spoke* platform concerning the purchase order, insert label faults (bad labeling or no label) and this is saved automatically on a database. Similarly, as returns storage data, there is one field of this type for each purchase order. Figure 6 illustrates this field, where the need for preprocessing phase is evident to organize this information. However, the return can be caused by diversified reasons. Some causes are present on *Dropbox* where an image of the customer card sent is stored.

Returns are handled by both Warehouse Operators and Account Managers. There are two situations that can occur: (i) brand notifies HUUB that a product from a return will be received; (ii) there is no communication and the item is received without warning. The second case involves a higher amount of time spent, since it is necessary to inform the brand and confirm that the item(s) are valid to be exchanged.

After collecting return reason and customer information, a picture of the item as well as a picture of a document sent by the customer indicating return reason are saved in a *Dropbox* file. An open field on the database (which purpose is to save a reference ID), is sometimes used to write notes concerning returns. This information is saved and linked to a purchase order.

Therefore, there is no standardization of the return reasons neither any storage of this data on the company's database.

1	RETURN NAP UK PART 1
2	return staff allow/PO Shawana
3	return #145000310
4	return #54307
5	LBL + RE - peça mal etiquetada
6	Wrong Label
7	re label scarfs
8	#Peças mal etiquetadas 2

Figure 6 Returns and labeling data

As previously described, we can see that the information is stored in multiple platforms. Plus, many fields have to restriction on the type of data inserted, leading to the appearance of apparently different data, but in fact contain the same information.

3.3 Implementation Objectives

By being a company that has access to multiple type of data from the brand, vendor and final customer, it appeared the opportunity to make use of them and offer business insights to the brand and to support warehouse activities planning. These insights can cover a variety of areas and entities present in the supply chain. However, in this project, the focus is on the vendor and inherent quality dependent of that entity.

Assurance of quality standards such as products quality, customer satisfaction and the meeting of planned delivery dates represent relevant challenges that are worth of tackling.

Firstly, the detection of the types of issues that occur due to vendors' faults can be considered as the first goal, including analyzing patterns among the available data.

The second objective of this project was to improve decision making of warehouse, financial and resources planning by implementing a tool that uses historical events to score each vendor, in certain categories faults (discriminated in section 4.4) and using it as an input to plan resources allocation in terms of time and the management of materials stock (in the case of the labels). The mitigation of unavailable resources at critical moments represented the main improvement opportunity here.

Predicting vendors' service level represented the third goal. In other words, the likelihood that the vendor will, for example, deliver the products with delay, quantifying it. Due to lack of data concerning returns and labeling, building prediction models was only an objective for days discrepancy and quantity discrepancy indicators. Aligned with this, these predictions main purpose is to use them as inputs for the warehouse resources allocation optimization algorithm. With this, resources can be allocated more efficiently and thus reduce the average time of reception handling.

A parallel goal, but aligned with the purpose of the project, was data standardization concerning any field that could be a valuable input for the followed approach.

Lastly, developing a service-oriented application to deliver the outputs of this work with the stakeholders.

4 Methodological Approach

This chapter provides an in-depth description of all the steps taken since the beginning of this project. This includes the approach followed for both prediction and to vendor scoring.

4.1 Data Preparation

Before targeting the prediction task, preparing data for its manipulation was the first step. This comprised the definition of which variables would be included for modeling and a data cleaning step for not only these variables, but also for the data that was used to score the vendors.

4.1.1 Variables for Prediction

Two target variables were defined. To measure the deviation between dates of reception and the agreed date for delivery, days variance was the elected variable. Secondly, to study the deviation between quantity delivered and the agreed with the vendor, quantity variance was considered a possible measure as following described.

Days variance represents the delta between the date in which the reception was delivered to the logistics' company warehouse and the estimated date for this event.

$$\Delta days = reception\ date - estimated\ date \tag{4.1}$$

Quantity variance refers to the discrepancy between the actual quantity delivered and the estimated one. Similarly, delays variance was computed by:

$$\Delta quantity = delivered\ quantity - estimated\ quantity \tag{4.2}$$

Thus, two models were built. However, being related with the reception, the type of features used was similar. Identifying these possible drivers (features) that impact targets variables values was an essential step to the construction of a data frame that would be used to generate the analytics models. From the available data, the following set of attributes were selected:

Vendor represents the entity that delivers the products.

Vendor properties refer to vendors' country, city, currency, number of updates to the estimated quantity delivered and to the estimated delivery date.

Product details may also have an impact on the issues already identified. These properties include products' material, type, family and subfamily.

Season is a property of the product.

Reception volume is an indicator of the total volume of products per reception.

Brand is linked with the reception level. The brand performs a purchase order to the vendor and communicates it to the logistics' company. To this order is associated a reception, with the goods agreed in the packing list of the purchase order. Since a vendor can serve more than one brand, it may also be interesting to identify to which brand the reception is linked to.

Price of the purchase order is the sum of the price of all products belonging to the purchase order linked to the reception.

Lastly, this information is joint in two data frames: one for each target variable, but both with the same set of features above mentioned.

4.1.2 Data Preprocessing

Data that was spread into different types of sources was integrated when necessary. For the prediction problem, database and warehouse spreadsheet provided valuable insights for the development of this project (shown in Figure 7). For the scoring problem, labeling faults and return data was extracted from *Dropbox*, warehouse spreadsheet and database.

Prediction data pre-processing

This required not only the extraction of interesting information, but also the validation of data and the removal of duplicates. Afterwards, a data cleaning process was performed to extract the data to a representation that allows the manipulation in the next steps.

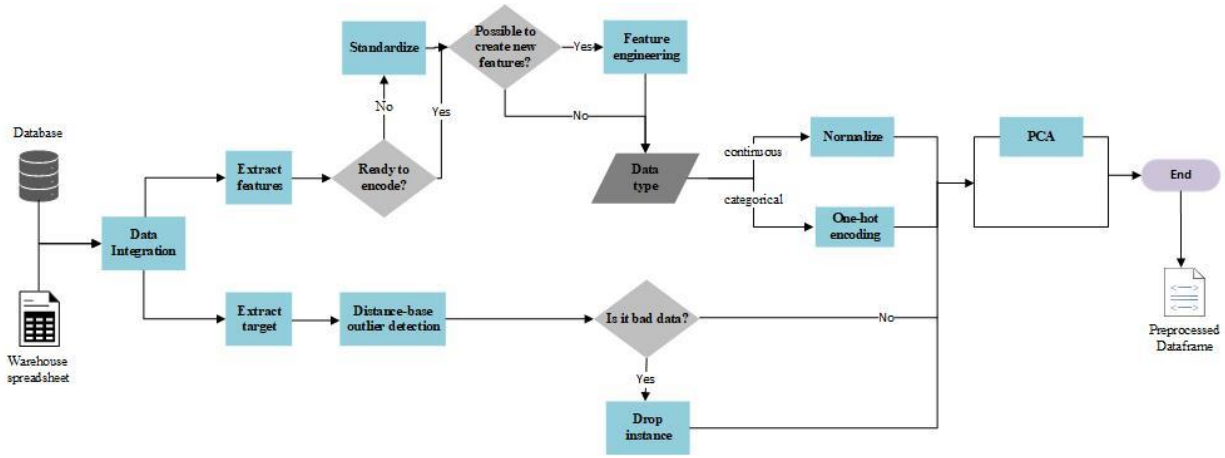


Figure 7 Data preprocessing approach for prediction

Subsequently, continuous features were normalized and since the dataset to be manipulated includes categorical data, encoding these categories is subject of analysis in this study. Figure 7 represents the flow of this process.

One final remark should be done concerning standardization step. This refers to features that were not ready to be normalized (if continuous) or to perform one-hot encoding (if categorical). In the context of this project, this was related with fields on the platform that were open to insert any type of data (numerical, categorical and characters).

If already ready to encode, it may be interesting to create new features from the ones available. Feature engineering, in the context of this work, represented the extraction of material composition of each product, that was achieved by matching a certain bag-of-words to the material field. This bag-of-words included an already standardized products' materials list. The match between this list and the words extracted from the cells was performed and the percentage associated with that material was also stored. The final representation must be a feature representing the material and the cell of each product item will store the percentage of that material.

Concerning target variables, an analysis to the discrepancy of the values from the average of the variable led to a set of data points considered, in this case, as outliers when far 3σ from the average (where σ is the standard deviation). This method was selected, for the ΔDays dataset, after analyzing the average of the target variable. Thus, only values distant $\pm 3\sigma$ from the mean were considered as not common. To this outliers' set, only points that were, on the business context, highly dubious to be real values, were removed from the data frame. For the ΔQuantity dataset, all data was considered as reliable.

Labeling and returns data pre-processing

Integration and standardization processes for labeling fault caused by vendors' and returns were very similar to the ones describe above. Since this information on the database was not standardized, return reasons were filtered and transformed into the representation where vendor, date of fault detection and identification of the sales order is stored. Labeling faults were discriminated into the two types previously mentioned.

4.2 Modeling

After data preparation, modeling represented the following step. Firstly, dataset partition was performed. Secondly, modeling. It was in the training instances that the hyperparameters were selected, using a Grid Search (brute force testing over all defined hyperparameters) with Cross Validation, and where the elected procedure for feature selection was applied. Plus, model validation using cross-validation was performed also on the training set and, finally, the model is built using all training data and its performance tested using the test dataset.

4.2.1 Partition in Train and Test Data

Having the features defined and the preprocessing executed, the dataset is ready for modeling. Nonetheless, learning models were not trained using all instances from the dataset. Instead, dataset was divided in two: train and test set. The former represents the one where the algorithm will learn and the latter, where it will be tested. The fraction used was 70% for the training set and 30% for the test set. A summary of this division is showed in Table 4.

Table 4 Test and train data

	Test	**Train**	**Total**
Δ Quantity Dataset	14540	33926	48466
Δ Days Dataset	52	124	176

4.2.2 Feature Candidates Ranking

Due to high computational costs that make it inviable to run an exhaustive search within the set of features that best predict the target variable, a sequential search was applied. The starting point of this search was the previously computed features rank list by their relevance and redundancy towards the target (maximum-relevance-minimum-redundancy criterion). The approach was applied on the training dataset, having as an output a ranking of the features more relevant and less redundant with the dependent variable. Figure 8 represents the algorithm that encompasses the steps for ranking of the features.

Since high dimensionality of the datasets was a property that was detected to be a cause for the high running time of the learning phase, PCA was also applied as an alternative approach. Vectors with length *n* were generated by this method and afterwards ordered by the mRMR criterion.

Thus, hyperparameter *n* was chosen considering the trade-off between the RMSE and running time of the algorithm trained with these vectors. For the days discrepancy dataset, 10, 40 and 70 vectors were generated, ordered, and tested. Similarly, for the quantity discrepancy dataset, 40 and 70 vectors were tested. Since training times were found to be infeasible if features are higher than 70 vectors, but this dataset has a higher dimensionality than days

discrepancy, 40 was tested but not 10. This was the followed approach, since ΔQuantity dataset has also a considerable number of instances, which makes the learning model computationally more expensive than ΔDays. Thus, between testing 10 or 40 vectors, 40 vectors may be able to capture more variance and therefore this was the elected *n*.

Algorithm 1: Feature rank by mRMR criterion

Input: dataset with all features $D = \{X_i^j\}$; target variable t_i with $i = \{1, \dots, k\}$ and $j = \{1, \dots, n\}$,
with *k* number of instances and *n* features
Output: rank of features ordered by mRMR criterion *S*
$S \leftarrow \phi$
for each X^j **do**
  $relevance(X^j, \mathrm{t}) \leftarrow I(X^j, \mathrm{t})$
  for each X^g **do** //with $g = \{1, \dots, n\},\ g \neq j$
    $redundancy(X^j, X^g) \mathrel{+}= I(X^j, X^g)$
  $score_{X^j} = relevance(X^j, t) - redundancy(X^j, X^g)$
S $\leftarrow sort(score_{X^j})$

Figure 8 Feature ranking by mRMR criterion pseudo-code (Peng, Long, and Ding 2005)

There was no theoretical evidence found that ensures this, and for an optimal solution search, all *n* should be tested. Although, searching the optimal solution would result in computational efforts that would make this work non-valuable at the short-term (4 months). The decision behind this methodology was supported by the objective goals defined at the beginning: achieve service level predictions, which led to a hybrid approach between computational effort and expected deliverables at the end of the thesis work.

4.2.3 Experimental Setup

Having a list of the features ranked, the combinations of features were performed starting with the first one from the rank. The performance of the learning models with this feature was tested, saved and the following feature on the rank was added to the set of candidates $F_{candidates}$. Hence, learning models were trained, tested, and the performance saved considering each set of features. The stopping criteria was when all features were included on the $F_{candidates}$ set. The purpose was to find the features that offer the best prediction. The error measure used for the test set was the root mean square error (RMSE) and the saved model was the one which solution was the one with lowest RMSE on this test set.

Notwithstanding, this was not the only measure considered for model evaluation and selection, it was only used for saving in memory the model. Section 4.3 explains how model selection and evaluation was carried out.

Furthermore, other parameters such as learning models and their hyperparameters were iteratively chosen. Besides feature selection, learning models were tested in a range of a list of different models and then, in each of those, the hyperparameters were also subject of a grid search with 10-fold cross validation. Thus, hyperparameters and learning models were evaluated, for each set of $F_{candidates}$, with cross-validation and its metrics saved for evaluation.

Having already a general framework of the prediction approach, the selection of the learning models and their hyperparameters are discussed below in this section. The starting point was to first train simple algorithms and then continuously adding some complexity.

Linear Regression was the first and the simpler model explained in section 2.3.1. This model allows the interpretability of each β_j as the weight given to each feature having as a goal the least squares minimization. However, this interpretability may be limited when the number of features is high and there is no limit for the weights assigned. Due to that fact, Lasso regression was the first model tested.

Lasso Regression aims to decrease model variance by introducing a parameter λ that will determine the amount of regularization applied and ideally offer a more interpretable model. As explained in Section 2.3.1, this algorithm is able to assign null values to the weights of the variables included on the model. It is then able to perform feature selection, which is an additional analysis for feature importance for this predictor.

Because one of the goals is precisely to understand the main drivers for the target variable behavior, relevant insights may be extracted by this analysis. The hyperparameter tuned is then λ, which is tested in an interval for the values 0.5, 1, 2 and 10. As lower this value, the closest to a linear regression the model is, since this parameter is null on a linear regression model.

Random Forest can be defined as an ensemble model that generates several decision trees (weak learners) in order to build a model that combines all of these in one strong learner. The hyperparameters tuned were the criterion to measure the quality of the split of a node, the number of estimators (trees) and the maximum depth of each tree.

The criterions searched for node splitting were MAE (Mean Absolute Error) and the MSE (Mean Squared Error). The number of estimators (trees) tested were 5, 10, 15 and 20. Finally, the maximum depth of the tree tested were 15, 30, 50, 60, 70, 80. Decision trees were also tested separately, in order to analyze which hyperparameters resulted in a best error estimate.

Ensembles provide generalization, while a single decision tree is not able to provide that. However, for low amount of data, which is the case on the Δ Days dataset, this generalization may not be achieved.

Thus, both models are tested. A final remark should be done concerning maximum depth: even though this was the interval tested, for Δ Days dataset, the maximum depth allowed was of 60, aiming to avoid overfitting. This concern cause for overfitting was due to the lower number of features of this dataset in comparison with the Δ Quantity dataset.

Support Vector Regression is able to map the data in a linear space but also in a non-linear space. Thus, this type of mapping tuned by the hyperparameter Kernel. The kernels tested were the linear and the nonlinear RBF (Radial Basis Function). Plus, as mentioned in section 2.3.2, adjusting *C* means defining the weight of the prediction error on the objective function and therefore the allowed margin of the hyperplane. The values tested are 0.5, 1, 5 and 10.

Stacking was the last model tested and the first level predictors tested were the best performing weak learners tested.

This work proposes an approach to deal with stacking prediction. The inputs for the algorithm are: a dataset *D* which was already preprocessed; the learning models that are going to be the regressors of first level for this model, L_m; the features already ranked by the criterion previously explained *S* (section 2.3.4); the number of features available on the dataset and, lastly, the model that will learn from the predictions of all L_m: meta model M_m.

In a first step, a grid search of the hyperparameters of the models L_m is performed. The parameters that offer the best prediction are then saved and only these models L_m^* are going to be used in the ensemble model. Notice that this is done for each $F_{candidates}$ (set of features). Thus, this is tested n_f times.

Figure 9 illustrates the pseudo-code for this experimental setup above described.

Algorithm 2: Experimental approach

Input: dataset *D*; regressors L_m; hyperparameters H_{L_m}; rank of features *S*; number of features n_f; meta model M_m
Output: best prediction s_{best}
$n \leftarrow 1$; *temp* $\leftarrow 1$
while $n < n_f$
  for *S*, $S \in \{S_1, \ldots, S_n\}$ **do**
    $F_{candidates} \leftarrow S$
    for each L_m **do**
      Step 1: Grid Search with 10-fold Cross Validation
      Randomly split *D* into 10 equal-size subsets: $D = \{D'_1, \ldots, D'_{10}\}$
      for each H_{L_m} **do**
        for $k \leftarrow 1$ *to* k **do**
          $s_k \leftarrow model(D \backslash D_k, L_m, H_{L_m}, F_{candidates})$
          $s_{incumbent} \leftarrow mean(s_k)$
      Save H_{L_m} with the minimum $s_{incumbent}$ as $H_{L_m}^{best}$
    Step 2: Compute stacking with L_m *and respective* $H_{L_m}^{best}$
    $s_{incumbent} \leftarrow stacking(D, L_m, H_{L_m}^{best}, F_{candidates}, M_m)$
    Step 3: Save the best solution
    if *temp* $\leftarrow 1$ **do**
      $s_{best} \leftarrow s_{incumbent}$
    else if $s_{incumbent} < s_{best}$ **do**
      $s_{best} \leftarrow s_{incumbent}$
    temp+=1
  n+=1

Figure 9 Experimental approach pseudo-code

It may be relevant to make a critical observation concerning L_m grid search. In fact, hyperparameters are being tuned separately in each L_m. This was the elected method due to the lack of computational resources to test more combinations or even a random search with more than one hyperparameter set per model. Thus, this may not be the best way of performing a stacking model and may even cause overfitting.

Getting in more detail on the stacking algorithm and with the aim of avoiding optimistic results, 10-fold cross validation was performed, as algorithm 3 (Figure 10) shows. The main steps are: (i) train the first-level regressors and transform predictions into a dataset, (ii) train the meta-regressor, (iii) save a dataset with the predictions of the base-learners in a test dataset and, finally, (iv) compute the RMSE. Other evaluation metrics are also computed, but since the model saved in memory is the one that offered the lower RMSE, this is the metric represented in the algorithm.

Algorithm 3: Stacking with CV pseudo-code

Input: dataset *D*; regressors L_m; hyperparameters $H_{L_m}^{best}$; meta regressor M_m
Output: solution *s*
Step 1: CV approach to prepare a training set for the second-level regressor
Randomly split *D* into 10 equal-size subsets: $D = \{D'_1, \ldots, D'_{10}\}$
for $k \leftarrow 1$ *to* k **do**
  Step 1.1: learn first-level regressor
  for $t \leftarrow 1$ *to* t **do**
    Learn regressor $L_m^{(tk)}$ with $H_{L_m}^{best}$ from $D \backslash D_k$
  Step 1.2: construct a training set for the second-level regressor
  for $x_i \in D_k$ **do**
    Get a record $\{x'_i, y_i\}$ where $x'_i = \left\{L_m^{(t1)}(x_i), \ldots, L_m^{(t10)}(x_i)\right\}$
Step 2: learn a second-level regressor
Learn the meta regressor M_m from $\{x'_i, y_i\}$
Step 3: *relearn first-level regressors*
for $t \leftarrow 1$ *to* t **do**
  Predict with regressor $L_m^{(t)}$ and $H_{L_m}^{best}$ from *D*
Step 4: *Compute RMSE*
$s = \text{RMSE}(M_m(L_m^{(1)}(\boldsymbol{x}), \ldots, L_m^{(T)}(\boldsymbol{x})), \boldsymbol{x})$

Figure 10 Stacking with CV pseudo-code adapted from Aggarwal (2014)

After the previous remark, learning models and the meta model itself must be specified to construct this stacking model. Two meta-learners were tested: LASSO and the SVR.

4.3 Model Selection and Evaluation

Despite one of the goals of these predictions is to find the most accurate possible results, it is also the interest of the company to have robust and scalable algorithms that, with new data are able to be adapt and provide a reasonable error. For this to happen, one of the concepts found on the literature that support this is the model selection taking into account bias and variance trade-off. A model that offers low RMSE between the actual target value and the predicted one, but has a high variance, can be a candidate for overfitting. Thus, regarding an analysis to the RMSE of each model, both bias and variance weighted on the decision of which model to implement.

Also, the repeated cross validation average and standard deviation computed aim to provide a realistic expected RMSE and allow the comparison between models' performances. In the case of the data frame with the low amount of data, this was still performed since this is a model to be scalable. In addition, explained variance, MAE and R^2 were also computed.

Despite this, model selection is performed by choosing the one that offers a statistically significant better error prediction, at a significance level of 5%. This is executed by computing *t* statistic (see section 2.3.4) and comparing it with the *z-score*, that, for a two-tailed test is 2.262. If *t* is lower than -2.262 or higher than 2.262, it is considered that there is statistical evidence that the means of the error prediction between two different models are different, with a confidence of 95%.

4.4 Vendors' Service Level Indicators

Aligned with what was defined on Table 1 and 2, four indicators were identified as representative of a vendor's service level: on-time delivery, quantity delivered, number of returns and number of label faults.

On-time delivery

$$\overline{\Delta days_v} = \sum_{i=1}^{n_r} \Delta days \,, \forall\, v \in \{1, \dots, n_v\} \tag{4.3}$$

$$\sigma_{\Delta days_v} = \sqrt{\sum_{i=1}^{n_r} (\Delta days - \mu(\Delta days))^2}\,, \;\forall\, v \in \{1, \dots, n_v\} \tag{4.4}$$

Quantity delivered

$$\overline{\Delta quantity_v} = \sum_{i=1}^{n_i} \Delta quantity \,, \forall\, v \in \{1, \dots, n_v\} \tag{4.5}$$

$$\sigma_{\Delta quantity_v} = \sqrt{\sum_{i=1}^{n_i} (\Delta quantity - \mu(\Delta quanity))^2}\,, \;\forall\, v \in \{1, \dots, n_v\} \tag{4.6}$$

Returns

$$returns\ score = 1 - \frac{\sum returns}{\sum total\ items\ received} \,, \forall\, v \in \{1, \dots, n_v\} \tag{4.7}$$

Labeling

$$labeling\ score = 1 - \frac{\sum labeling\ faults}{\sum total\ items\ received} \,, \forall\, v \in \{1, \dots, n_v\} \tag{4.8}$$

, where n_i is the number of items of each reception, n_r is the number of receptions and n_v is the number of vendors *v*.

Both returns and labeling were computed using the total of items' returns by vendor and the total of items that were delivered with no label or with the wrong label. This may be redundant if a label fault caused a wrong item sent to the customer, who returned the item. Thus, it is important to discriminate return types by their cause (represented in table 2). Although this information is missing at the moment of this project execution, these KPIs' can then be adapted to those situations easily, adding drilldown criterions to the return KPI.

Packaging and packing faults were not used as a metric since there is no available data in any source about this type of vendor fault. However, measuring this type of incidents were identified and there is an awareness for its storage of this data on the future.

5 Methodology Assessment

5.1 Data Preparation

This subchapter starts by describing one of the major challenges found during data preparation: the need for the maintenance of consistency between data when integrating different sources of information (see section 5.1.1). Data gathering section details the followed approach for data extraction. Data transformation also required an intense phase mainly for the material field as described in section 5.1.2. Finally, a brief descriptive analysis of both target variables and the features was performed (see 5.1.3 and 5.1.4, respectively).

5.1.1 Data gathering

Reception delivery date: reception dates were often not registered in the database, which led to the need of extracting this data through another available source. Again, by information collected from warehouse workers, it was perceived that the registered data on an external spreadsheet was more reliable. The causes behind this are from 2 types: (i) reception dates are first registered in the excel spreadsheet and just sometimes in the database, (ii) planned delivery dates, for a certain reception, can be agreed with the vendor several times and is updated by the Account Managers in this spreadsheet. However, not always this information is inserted in the database. Plus, the planned date, in the database, is inserted by default if not inserted by the Account Manager. I.e., if not registered, then automatically this date is equal to the reception date creation. To tackle this, it is deemed that the most feasible values are the ones from the spreadsheet and then, if no data is found, database source is used, with the restrictions showed in Figure 11.

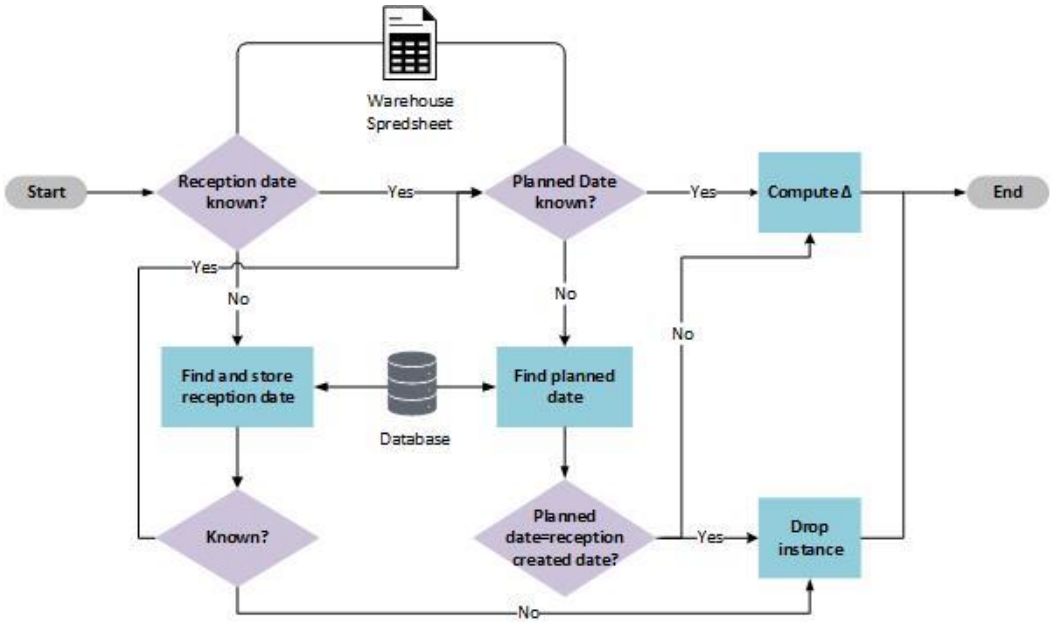


Figure 11 Delivery delays exception

Quantity variance: estimated quantity of products for a certain reception is a field that can be updated, since the vendor may not be able to communicate the exact quantity that will produce. Thus, the first estimated quantity inserted is considered as the value to compute Δ quantity. Actual received goods quantity are stored automatically when PTS or PTSP occurs.

Labeling: a spreadsheet from the warehouse and data from the database were the sources used. As noticeable from Table 3 in section 3.2.3, purchase order identification was

never registered. Thus, this was treated as an exception. Knowing by warehouse workers that the labeling process is executed with a maximum of 10 days after the reception of that purchase order, this case was handled by filtering the receptions which client and vendor matched the *Excel* instances and which reception date was the closest to the labeling date. This method does not seem effective in a case where it is likely to have multiple receptions of that vendor and brand in a period, say, 1 month. Assuming this happens, the accuracy of this method does not seem reasonable. However, and relying on business understanding step, it was concluded that the likelihood that this occurs is considerably low. This was proved by the search in the database, where receptions were filtered with a window of 10 days and just one reception was found.

Returns faults: a match between the data collected from the database and in the *Dropbox* was performed. Information had to be extracted from a picture, thus this had to be done manually one by one.

5.1.2 Data Transformation

After having the data organized, the consequent step represented the transformation of this information to a proper representation to be the input for the learning models. Feature engineering and standardization were included in this stage.

Products' material was one attribute from which new features were created. Firstly, a list containing all items' materials was created. Then, a match between this list and substrings of the material field was performed. Composition percentages were detected by the presence of "%" and the material(s) present after a string that contained this symbol were associated with the percentage value. This field before applying data cleaning is shown in Figure 5 (section 3.2.3). The result after applying the filter developed is shown in Table 5.

Table 5 Model material after data cleaning

Product_model	polyester	viscose	elastane	cotton	woven	polyamide
1	0.65	0.33	0.02	0	0	0
2	0	0	0	1	1	0
3	0	0	0.05	0.95	0	0
4	0	0	0.03	0.75	0	0.22

Labeling dates and types were extracted from warehouse spreadsheet. However, there was no identification concerning to which reception those items belonged to. After matching this information with the data showed in Table 6, this enabled the constraint of the items list that were delivered by the vendor with that type of fault. I.e., even though there was no record detailing which item was delivered with a label fault, knowing the purchase order in which the incident occurred, limits the options. This can be valuable for future implementation to find patterns in this behavior.

Returns were filtered both in the database and from the information in the *Dropbox* to the ones caused by the vendor. Each photograph of customer card sent by the customer was checked and the information was written in a spreadsheet. The final representation is shown in Table 6.

Table 6 Returns due to vendors' fault

SO	Vendor	Brand	Date	…	Type
2015	1	4	15-04-2018	…	Faulty item
2016	2	5	19-05-2018	…	Wrong item
2070	1	4	22-05-2018	…	No label
3015	5	6	05-06-2018	…	Delay

5.1.3 Target Variables

The two variables of interest to predict in this project were the "days discrepancy" and the "quantity discrepancy", as above-mentioned. The dataset that includes the target days discrepancy is named as Δ Days. Similarly, for the target variable "quantity discrepancy", Δ Quantity refers to the dataset that contains features and the dependent variable.

Δ Days

Data points farther $\pm 3\sigma$ from the mean (points considered as bad data) were removed and the distribution of this variable before and after removing bad data is shown in figure 12.

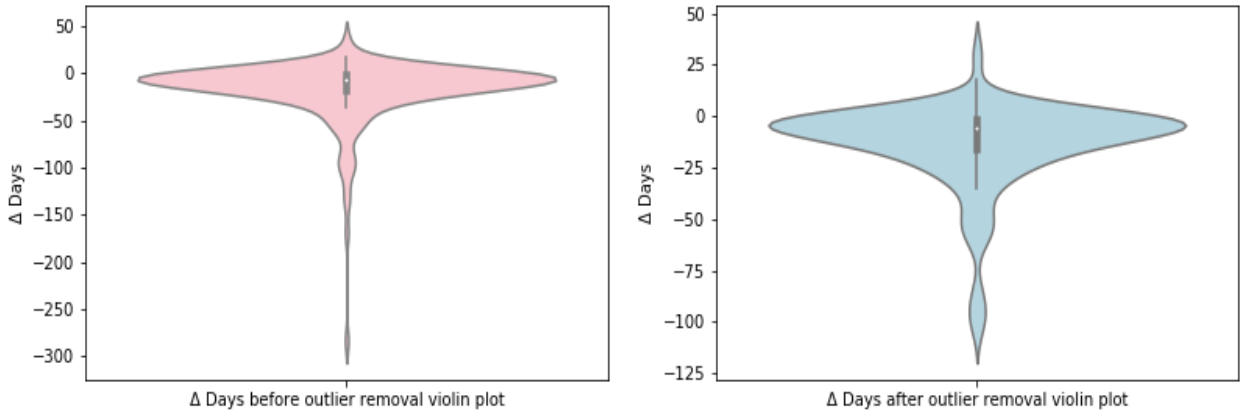


Figure 12 Δ Days target variable before and after outlier removal violin plots

The interval [-16.0, -1.8] from figure relative to the variable after outlier removal stands out by its clear concentration of points, representing this half of the data points of the entire dataset. On the other hand, an interesting set of data points [-105.0, -75.0] is noticeable. This may be relevant due to the considerable delay that this represents. Predicting this type of delays would be highly valuable for the management of the logistics' processes.

Δ Quantity

Despite 1172 data points are far $\pm 3\sigma$ from the mean, in this dataset, these points were not considered to be bad data. In fact, these values are possible and are important to include in the model. The range of these points belong to [-904.0, -44.7[and]33.9, 100]. A reception that arrives with less than -904 items is possible and it was checked to be reliable data. Hence, applying the rule of $\pm 3\sigma$ seems not beneficial when these points are important occurrences in this business context. Therefore, they were maintained in the dataset since they are considered as important outliers to include in the model. Figure 13 shows the distribution of Δ Quantity target variable.

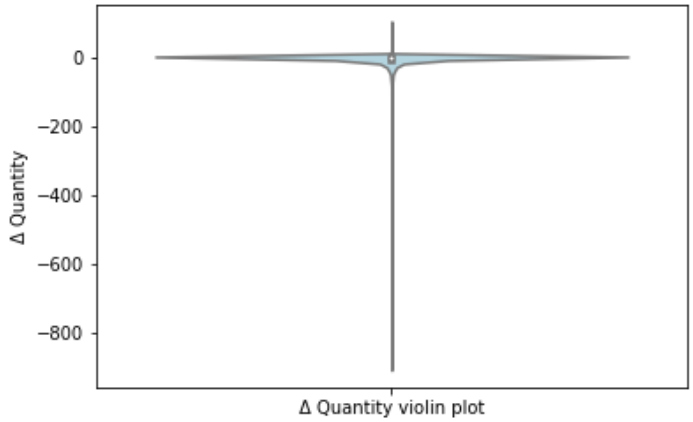


Figure 13 Δ Quantity target variable violin plot

A summary of both target variables is shown in Table 7.

Table 7 Target variables' descriptive statistics

	Count	Mean	σ	Min	25%	50%	75%	Max	Kurtosis	Skewness
Δ **Quantity**	48466	-5.7	14.5	-904.0	-6.0	-6.0	-1.0	100.0	945.998	-19.564
Δ **Days**	176	-13.4	21.1	-105.0	-16.0	-6.0	-1.8	31.0	6.013	-2.242

As expected, kurtosis is closer to 3 (normal distribution value) for the target variable of Δ Days, since the tail is less heavy when comparing with the Δ Quantity target variable. Also, as noticeable from Figures 12 and 13, the asymmetry of the probability density distribution is more intense for the Δ Quantity. This analysis may be relevant to evaluate the error measures of the model's performance.

5.1.4 Features Descriptive Statistics and Ranking

Having target variables ready to be part of the dataset to be modeled, features already standardized and feature engineering applied, features descriptive analysis, ranking and encoding was the following step.

Figure 14 shows the mutual information between features, for both datasets. This was performed before one-hot-encoding data transformation.

Since the features are the same, these values were aggregated in the same table. The upper triangle refers to the Δ Quantity dataset and the lower triangle shows Δ Days dataset features' mutual information. This table does not include all the features, due to the dimensionality that would represent. However, the relevant feature interactions for the posterior analysis are included.

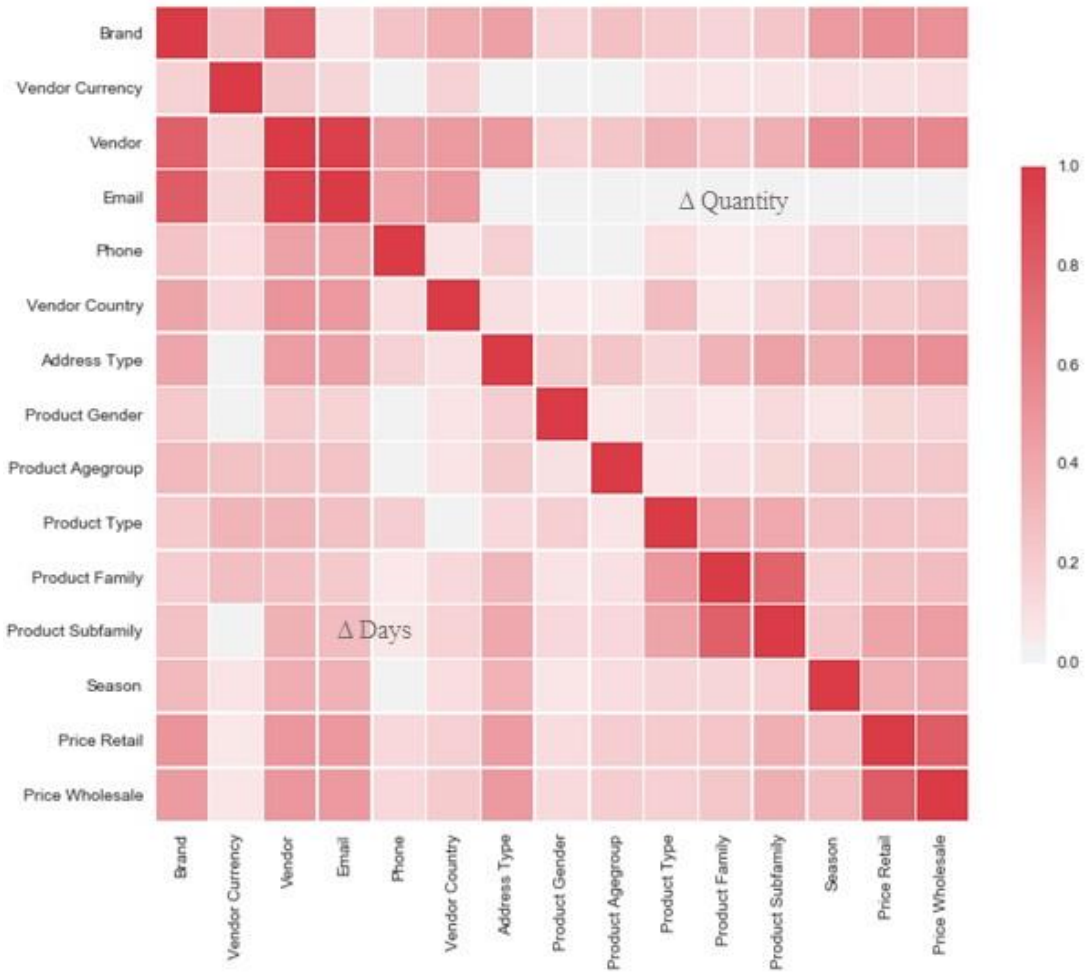


Figure 14 Features mutual information

Transforming features in a logic of one-hot-encoding led to the creation of new features that represent the presence (1) of absence (0) of a determined category of the original feature. Exemplifying: each vendor was transformed in a new feature, and if the purchase order (in the case of Δ Days dataset) or item (in the case of Δ Quantity) of the reception was delivered by that vendor, then the feature of vendor *i* is assigned a value 1 for that instance. Feature ranking was computed using these already encoded features. This type of encoding allowed the detection, by the mRMR method, of specific problematic vendors and materials. Table 8 shows the top 10 list of the most relevant and less redundant features ordered.

In fact, features that appear on top of the rank do not have a high mutual information between them. For instance, for the Δ Days dataset appear distinct vendors, but not their country, which by Figure 14 can be perceived to be a pair (vendor/vendor country) with a high mutual information.

Similarly, for the Δ Quantity feature rank, a vendor country is positioned on the top, but the vendors placed in this rank are not from the vendor country 103. Hence, it seems that the features in the ranking have a low mutual information between them, which was the expected, given the concept behind the mRMR criterion.

The presence of items with polyester in a certain reception was detected, by this method, to be an important factor to determine the value of the Δ Days target variable. On the other hand, one country was found to be a relevant feature for the value of the deviation of quantity from the agreed.

Table 8 Top 10 rank features by mRMR

Order	Δ Days	Δ Quantity
1	Polyester	Vendor country 103
2	Vendor 43	Spring Summer 2017
3	Acrylic	Trousers
4	Cupro	Nightwear
5	Vendor 154	Paper
6	Romper	Vendor 5
7	Autumn Winter 2016	Vendor 121
8	Vendor 126	Vendor 59
9	Paper	Vendor 130
10	Brand 23	Brand 53
…	…	…

This may be relevant information to report to the brand owner and provide insights concerning which are the critical variables for vendors' behavior. For warehouse management, this enables the use of this rank to the following methodology implementation: prediction.

5.2 Modeling and Evaluation

This subchapter introduces the prediction task performance measures as well as the impact of the elected feature subset selection method applied. Firstly, the error evolution with the number of features used and, afterwards, model comparison.

5.2.1 Feature Selection Analysis

This analysis aim was to focus on the evolution of the error with the subset of features/vectors used. As explained in chapter 4, a ranking method was applied and studying this behavior of the error as features or vectors are added to the model consisted in a test to this mRMR criterion applied.

Δ Days

Figure 15 shows the behavior of the error measure selected (RMSE) while incrementing the number of features used to train the models (see section 4.2.2).

Lasso model seems to be relatively constant in the prediction error between 41 and 201 features used for modeling and it is also noticeable from Figure 15 that adding features to the model increased the prediction error. Plus, the predicted value, excluding in the interval between 1 and 40 features, corresponded to the average days variance (target variable) of the train set. However, in this spectrum of feature numbers, three of the first four features were assigned non-null Lasso coefficients. A disadvantage of using these weights to perform feature selection is stated by the literature: the assumption of linearity between features and target variable behind Lasso modeling. On the other hand, mRMR is able to capture non-linear relationships.

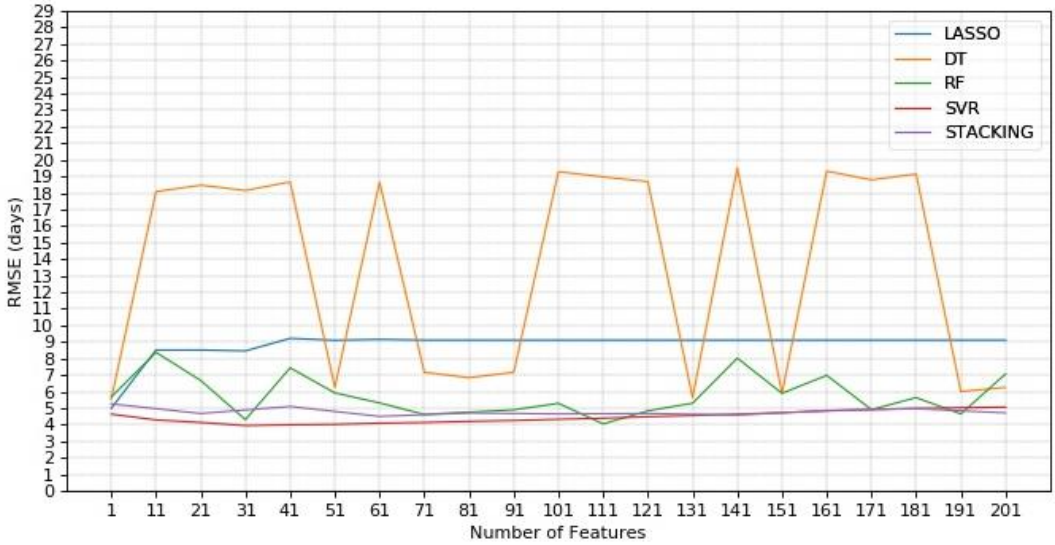


Figure 15 RMSE prediction versus number of features ΔDays

Decision tree is the model that exhibits the higher variance for the prediction error provided by adding new features. Thus, adding features to this model does not show a clear increment of the model performance. The best solution for this model uses only 1 feature (polyester). However, the ensemble of decision trees (Random Forest) is less sensible to the number of features used as an input and its best overall estimate was achieved after adding 111 features. One possible cause may be that Random Forest chooses randomly the features used and does not use all features for each decision tree of the ensemble. Hence, even though this is not conclusive concerning the feature ranking performed, it can be concluded that, in comparison with the previous models above discussed, adding features improved this model performance. One possible explanation for this behavior may be the fact that his ensemble is able to compensate bad performing decision trees with better performing ones.

Finally, both Support Vector Regressor and Stacking suggest a negative trend in the prediction error until the 21st (Stacking) and 31st (SVR) feature added. Stacking does not outperform (in terms of RMSE of this test sets) other models until 191 features added, which does not match the purpose for its application. One possible reason for this to happen is that the meta-regressor used (LASSO) was not able to properly learn which models are the best to predict certain instances.

Switching focus to the modeling performed using vectors generated by PCA, the scenario seems to be slightly different, and all models excluding SVR start with an improvement of the error.

Plus, the minimum error is achieved, in all models, after including more than 1 vector. Thus, even though there cannot be stated that it exists a direct relation between vectors ranking and this behavior, it can be concluded that there is a trend for error decrement while adding more vectors until a certain point. Figure 16 represents this error evolution with the number of vectors included.

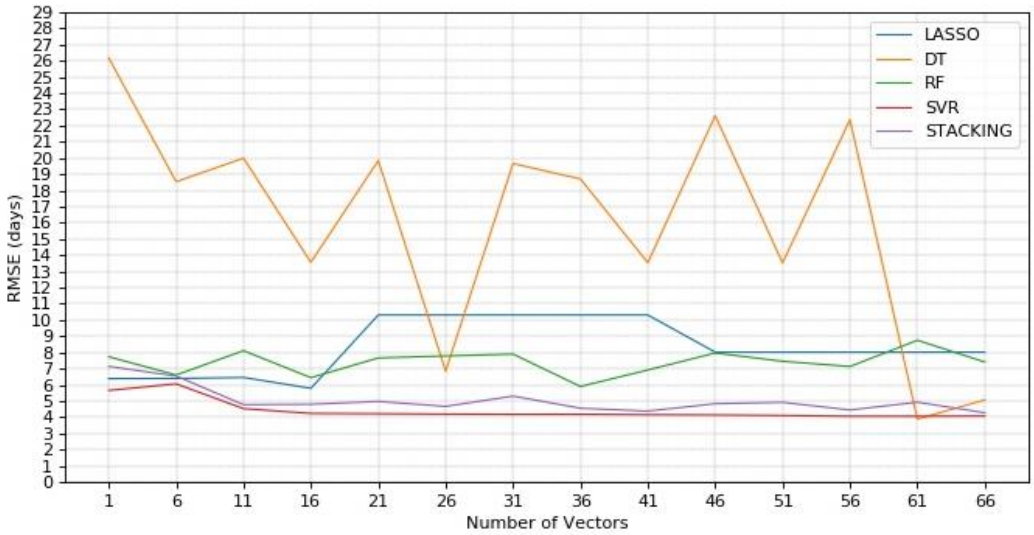


Figure 16 RMSE prediction versus number of vectors ΔDays

Δ Quantity

Following the same logic of analysis, features from this dataset were encoded in vectors using PCA. And, due to the dataset high numerosity, the more complex models trained (SVR with kernel RBF and Stacking) were just tested with vectors generated by PCA. Figure 17 shows the number of vectors included in the training dataset versus the RMSE of the prediction.

As Figure 17 illustrates, Lasso error prediction seems to be invariable with the number of vectors included. This will be analyzed more in-depth in section 5.2.2, but there is a possibility that can be pointed for this tendency: the model may not be using any vector to predict the output.

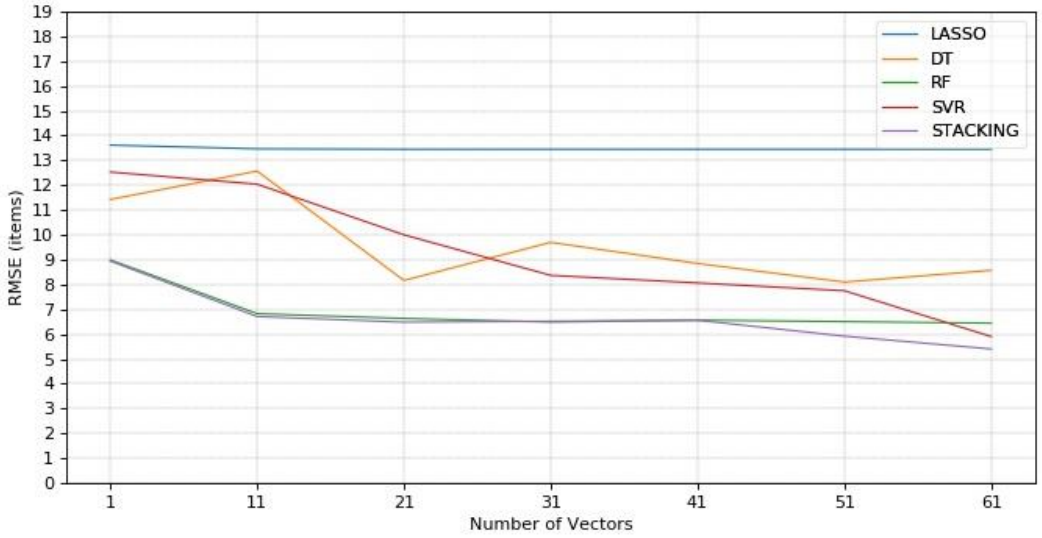


Figure 17 RMSE prediction versus number of vectors ΔQuantity

While Random Forest stabilizes after reaching the best prediction, a single decision tree points out a higher variability on the error. This can be related with the fact that decision trees aim to find the best division at each node of the tree. In contrast, Random Forest provides lower variance by training with different instances and number of vectors in each tree built (see 2.3.3).

Finally, both SVR and Stacking seem to improve their performance as vectors are added for training. However, Stacking outperforms all models (in terms of RMSE), using 61 vectors. The meta-learner was able to find a pattern from base-learners predictions and thus was able to

deliver more accurate predictions. The meta-regressor that proved o offer the best results was the SVR.

5.2.2 Prediction Results

After analyzing features and vectors behavior with the prediction performance, it may be also relevant to understand in which spectrum of the target variable values are more difficult or easier to predict, in each model.

Δ Days

Figure 18 shows the prediction versus the real data points values, for the days discrepancy prediction. Hence, the line in the Figures represent a perfect prediction, with a null error.

Lasso predictions seem to belong to the range of [-25, -7] for all data points of the test set, when real data points values' range is [-17.5, 0]. Thus, this model seems not to be able to distinguish, in their prediction value, a reception with -12.5 days or 0 days of variance between the agreed and actual delivery date. The other models tested, even predicting in a larger range of values and, for the test set, were able to predict more accurately than Lasso, Table 10 shows that the difference between the error predictions was not statistically significant.

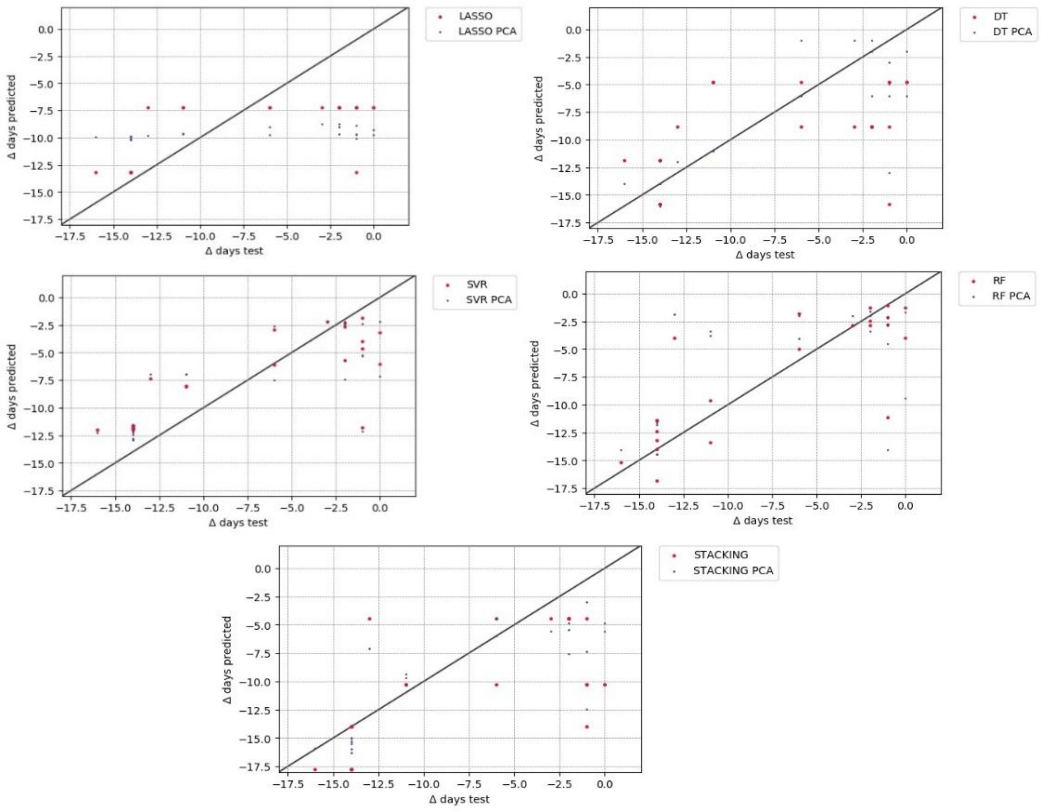


Figure 18 ΔDays actual versus predicted value

Focusing now on the Stacking model performance, it is also noticeable by Figure 18 that predicted data points vary in a range between -17.5 and -2.5, which is a smaller range when comparing with the Decision Tree, Random Forest and SVR. It can also be perceived that, for receptions with 0 days of variance, stacking predicts worse, in this test set, than the previously mentioned three models. Since the meta-learner that offered the best prediction (for stacking hyperparameter tuning) was a Lasso regressor, this model may have assigned a weight to each predictor and thus this prediction may be taking into account the first-level predictor Lasso.

However, it must be highlighted that this data (train and test sets) represent only 4% of the receptions. Plus, as the model complexity increases, the related explained variance (see Table 9) is increasing too, and an overfitting concern arises. Bias seems to be low, but variance can be considered as high. Thus, this trade-off between the two metrics is not being met. Variance reaches values too high (82%). Plus, in the business context, an error of 4 days is not valuable. The lack of data makes this analysis not conclusive and the models trained not reliable for production implementation. This means that with the current training data volume the model is not able to affect planning by now, what enhances the need of gathering more historical instances.

Δ Quantity

In contrast, quantity discrepancy dataset offered more reliable metrics, since more data was available. Figure 19 visually represents models' performance.

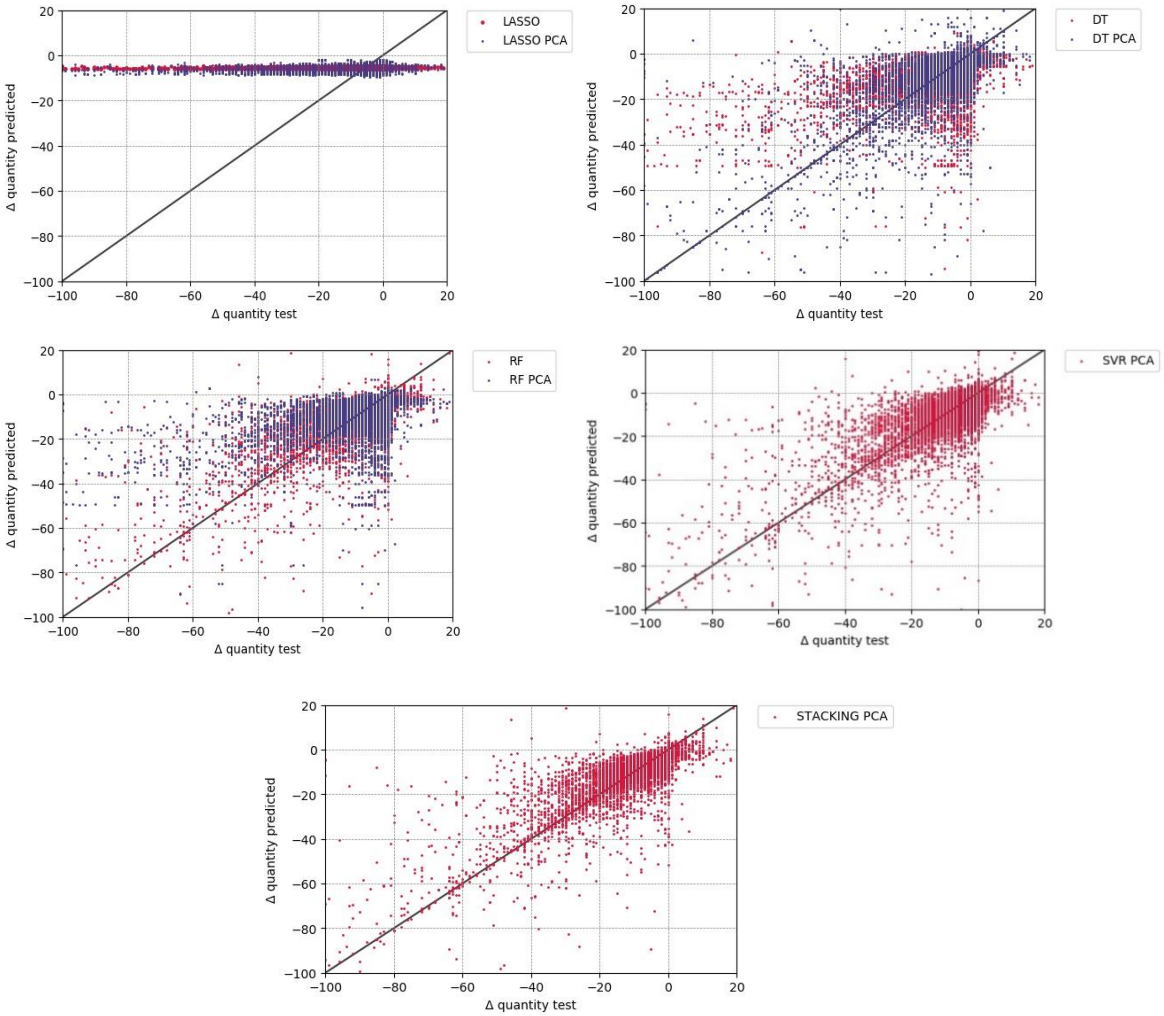


Figure 19 ΔQuantity actual versus predicted value

Data points predicted by Lasso algorithm reveal a tendency for the prediction value, in both type of encoding used (PCA or without PCA). These values predicted seem to be around the average of the target variable on the train set. Thus, features are almost not used for prediction. One possible reason for this behavior may be the fact that this algorithm is not able to capture non-linear interactions between features and the dependent variable. Adding more complexity, both Decision Tree and Random Forest offer better results and it is noticeable the increment of data points closer to the line of zero error. However, Random Forest with PCA

outperformed as illustrated in the graphic: points predicted by this model are closer to the actual value, from [-100, -40], when comparing to a single Decision Tree.

Stacking seems to perform better in the interval of [-20, 0] than SVR. And comparing with the Random Forest, Stacking seems to have more data points closer to the zero-error line in all spectrum of values.

The prediction results are shown in Table 9.

Table 9 Prediction Results

Target	**PCA**	**CV**			**Error Measures**				**Features**
		μ	σ	Explained Variance	MAE	R^2	Bias	RMSE	
ΔQuantity		(units)	(units)	(-)	(units)	(-)	(%)	(units)	
Lasso	Yes	12.6	7.3	0.03	-5.7	0.03	0.001	13.4	21
	No	12.8	7.3	0.01	-5.8	0.01	0.001	13.5	380
DT	Yes	8.1	1.1	0.57	-3.0	0.57	0.003	7.4	55
	No	9.11	2.8	0.45	-4.0	0.45	0.002	8.9	480
RF	Yes	7.1	2.9	0.68	-2.9	0.68	0.003	6.5	61
	No	9.1	3.0	0.45	-4.1	0.45	0.003	8.9	450
SVR	Yes	6.4	2.5	0.73	-2.5	0.73	0.002	5.9	61
	No	-	-	-	-	-	-	-	-
Stacking	Yes	5.9	2.3	0.76	-2.1	0.76	0.004	5.4	61
	No	-	-	-	-	-	-	-	-
ΔDays		(days)	(days)	(-)	(days)	(-)	(%)	(days)	
Lasso	Yes	15	19.6	0.08	-8.7	0.08	1.1	6.1	8
	No	14.9	20.8	0.14	-7.1	0.14	1	5.0	2
DT	Yes	18.0	20.12	0.43	-4.7	0.43	0.5	3.8	30
	No	14.3	20.6	0.82	-4.3	0.47	0.1	3.9	31
RF	Yes	17.1	21.3	0.18	-9.6	0.18	0.2	5.0	25
	No	15.8	21.3	0.23	-6.2	0.23	0.001	3.4	21
SVR	Yes	14.0	20.0	0.15	-5.7	0.34	0.2	4.0	61
	No	14.2	20.2	0.27	-5.9	0.26	0.02	3.7	31
Stacking	Yes	15.0	21.0	0.04	-7.0	0.04	0.9	4.1	61
	No	14.9	20.7	0.13	-7.4	0.13	1.4	5.7	141

Because model selection must take into account the statistical significance of the difference between two error estimates generated by the prediction of distinct learning models, an evaluation was performed concerning this.

With a significance level of 5%, a two-tailed Students' test was applied and Table 10 shows the values for this *t* score. The upper triangle refers to the ΔDays dataset and the lower triangle of the table to the ΔQuantity dataset. The bold cells are the ones that represent values

that reject the null hypothesis. And it is noticeable that, for the smallest dataset, there is no statistical significance that any model tested is better than any other (with a confidence of 95%).

In contrast, prediction errors from algorithms trained in ΔQuantity dataset are significantly different from each other, at a confidence of 95%. One possible reason for this distinct behavior appears related with the fact that there is a high variance in the smaller dataset and a lower one in the bigger dataset. This variance, stated in the literature as a signal for overfitting supports the previous conclusions that, for the ΔDays dataset, instances are insufficient to train models and thus applying those to predict new instances seems not reliable.

Table 10 Model selection

	Δ Days	**LASSO**		**DT**		**RF**		**SVR**		**Stacking**	
Δ Quantity	**PCA**	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No
LASSO	Yes	-	0.035	1.068	0.246	0.725	0.276	0.357	0.284	0.000	0.035
	No	0.194	-	1.071	0.205	0.739	0.305	0.312	0.241	0.034	0.000
DT	Yes	**6.096**	**6.366**	-	1.285	0.307	0.751	1.410	1.333	1.032	1.075
	No	**4.464**	**4.720**	**3.357**	-	0.945	0.506	0.104	0.035	0.238	0.205
RF	Yes	**7.002**	**7.257**	**3.224**	**4.986**	-	0.432	1.061	0.988	0.702	0.741
	No	**4.435**	**4.688**	**3.130**	0.024	**4.793**	-	0.616	0.545	0.267	0.303
SVR	Yes	**8.035**	**8.294**	**6.224**	**7.220**	1.828	**6.914**	-	0.070	0.345	0.313
	No	-	-	-	-	-	-	-	-	0.275	0.242
Stacking	Yes	**8.754**	**9.015**	**8.629**	**8.859**	**3.242**	**8.465**	1.472	-	-	0.034
	No	-	-	-	-	-	-	-	-	-	-

However, when studying quantity discrepancy, incrementing model complexity resulted in a statistically significant improvement of the model performance. A potential cause may be the amount of data available, allowing algorithms to have more information regarding interactions between features and the dependent variable. Model performance of Staking and SVR is not statistically different. Thus, these models were considered the two best performers.

5.3 Deployment

Implementing all the previous steps as a service was the final stage of this project. This enabled any user to perform a prediction request to the API where the output is the predicted value for the inserted input. Three user cases were considered: (i) data scientist, (ii) account managers and (iii) any stakeholder.

For a data scientist, it allows the request for the models' current performance. Testing the error state of the models using new data while varying the algorithm, encoding and target chosen is the main functionality.

Secondly, destined for stakeholders whose objective is to check previsions for certain reception(s) or item(s) about the target variables above defined, a *csv* file can be uploaded with the ids'. This file is sent in a JSON format and interpreted by the model script which is able to provide predictions in bulk. The result is a dataset with all predictions in a JSON format, but the Vendor API transforms this in a downloadable *csv* file. In the front-end webpage, the download is automatically triggered when the API sends the predictions output.

Finally, the more robust solution for implementation consisted in adding a second API that performs requests to the Vendor API every 12 hours if new receptions arrive to the system. I.e., if any new reception is inserted on the database, a request with the ID of the reception is triggered and the Vendor API will compute both quantity variance and days variance prediction, for each item of the reception. The output is saved in a local database via the API Prediction Request.

These predictions stored in the database are afterwards used by the warehouse allocation optimization algorithm. Warehouse planning is currently being tested and thus predictions are included in the planning report for every week. Operations manager is the final judge that decides how many resources to allocate per week. At the moment, it was possible to test that receptions handling is faster in 20%. This value was computed using the times from receptions delivered in an interval of 3 weeks and in comparison with the average time of handling a reception from the last 3 months before implementation. Before, planning was performed in a different way, thus the comparison is only reliable if done with the last 3 months.

Another improvement here, in comparison with the second user case, is the fact that since predictions are periodically generated, when predictions are requested in bulk, the runtime is faster.

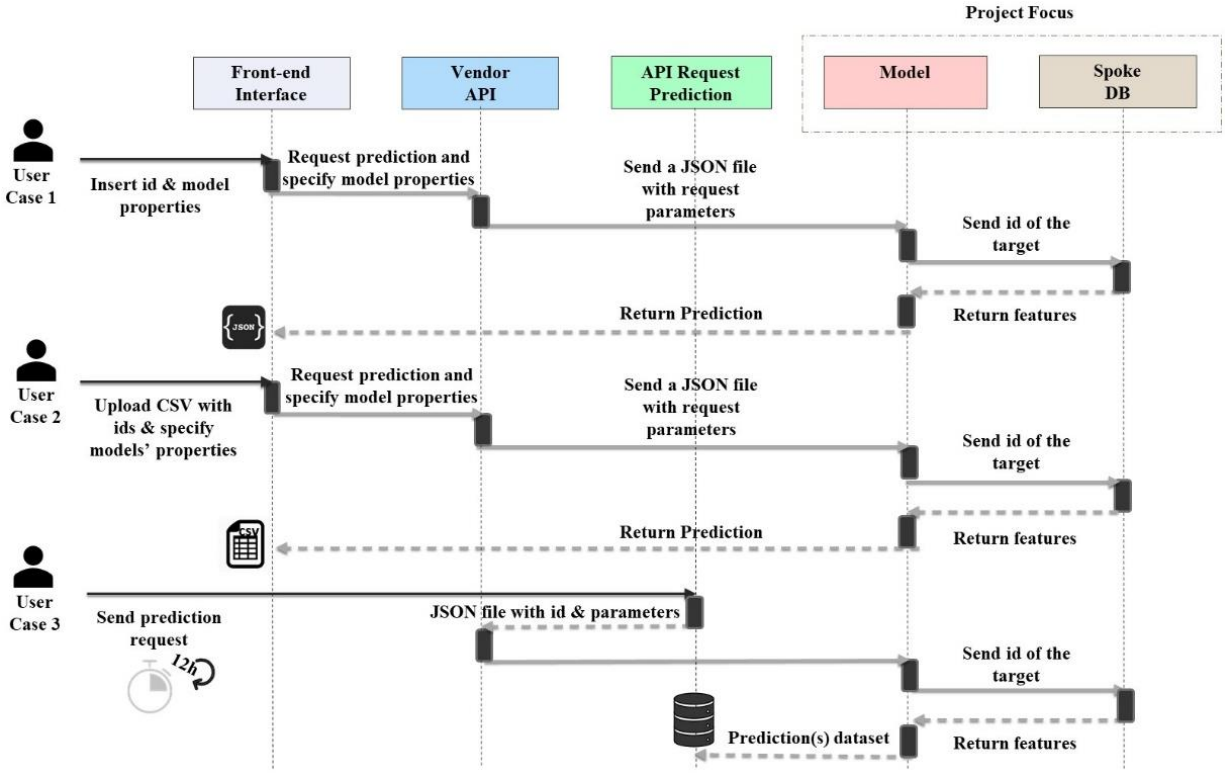


Figure 20 Sequence Diagram

In terms of computational specifications, *Python* was the elected programming language which was the basis for the construction of this implementation. Both APIs' were performed using the micro web framework *Flask*. To run this, all experiments were computed using an Intel® Core™ i7-4700MQ 2.4GHz processor with 8GB of RAM and a GeForce GT 740M graphics card.

A second part of this project consisted in scoring vendors, taking into account the four categories studied: returns, labeling, days and quantity discrepancy from the estimated. For that, a dashboard with these metrics was developed and Figure 21 illustrates an example of a vendor metrics evaluation.

Figure 21 Vendor Scores Dashboard

The visual representation was executed with *Power BI* where the drilldown criteria available are the vendor, brand and a temporal filter (date of purchase order creation).

6 Conclusions and Future Work

This dissertation addresses the implementation of predictability and visibility concerning vendors' service level, on a logistics company operating in the fashion industry. For the accomplishment of these implementations, a machine learning approach was followed.

Aiming to enable preventive actions by introducing predictability, this thesis work starts by detecting vendors impact on the business and afterwards gathering the necessary data to forecast and evaluate these entities' service level. Four categories were defined: (i) days discrepancy between the date agreed with the vendor and the actual goods' delivery date, (ii) quantity discrepancy between number of items agreed with the vendor and the actual number received, (iii) labeling rate and (iv) returns' rate due to vendor fault.

The first barrier found was linked to the insufficient amount of data, that restricted the prediction task. Only the first two indicators were used for forecast. Notwithstanding, the four categories were computed for scoring vendors and thus meet the goal defined of increasing visibility towards vendors performance.

Having already defined the target variables for prediction, variable definition was the following step. These features were afterwards ordered, by the mRMR criterion. This method consists in ordering features by minimum redundancy and maximum relevance with the target. Hence, provided insights concerning the vendors, materials and vendor countries that had the highest influence in the target variables behavior. Since dimensionality was considerable high for the computational resources available, PCA was applied and the vectors generated were ordered by the mRMR criterion. A model performance comparison between using features versus vectors generated by PCA was conducted.

An experimental approach for prediction is suggested in this work, which involves feature selection, hyperparameter grid search, single models' training and the development of a stacking learning model by the first learners trained.

Finally, the deployment stage for the prediction task comprised the development of two APIs': one that generates prediction requests if a new reception is inserted on the system, and another that receives periodically these requests and returns predictions. This was developed for three user cases: (i) Data Scientist, (ii) Account Manager and (iii) any stakeholder. The more robust solution developed included a local database that saves all predictions from all receptions present in the database.

An integration with the warehouse resources allocation algorithm was performed and these predictions are used as an input for this algorithm. Instead of using, as an input, the quantity that the vendor agreed to deliver, this quantity was replaced by the predicted quantity for delivery.

6.1 Main Results

The feature selection method applied (mRMR) provided better prediction results than using all features of the datasets. Plus, this method provided insights about the most important factors for the vendor behavior. For instance, items with polyester were found to be the most relevant for determining the number of days between the agreed with the vendor and the actual reception date. On the other hand, for the goods' quantity discrepancy between the agreed with the vendor and the actual delivered, there was a vendor country that showed to be the most important factor.

Concerning model's performance, for days discrepancy prediction, none of the models were statistically more accurate than any other and revealed a prediction error, at the moment,

too high for implementation. In fact, only 4% of receptions had available data for modeling. Thus, a possible cause detected for this behavior is the insufficient amount of data.

The scenario was different for the quantity discrepancy prediction: valuable results were achieved, in the context of the business. Two models were statistically equivalent in their performance: Stacking and SVR. Stacking was able to be a good performer and even achieve a lower RMSE than SVR, even though this difference was not statistically significant. In comparison with the other models tested, these two were the best performers. Lasso, Decision Tree and the Random Forest were the other models tested. Lasso showed a tendency for a constant prediction value, which was detected to be around the average target variable value on the training set. Thus, it was concluded that Lasso was almost not using any predictor. Random Forest outperformed the single Decision Tree, which can be related with the generalization power of the ensemble. Plus, this can also be related with its ability to compensate less accurate decision trees with another better performing ones.

Currently, quantity discrepancy prediction is being included as an input for the optimization algorithm that plans warehouse resources allocation. After this implementation, it was perceived a decrease of 20% of the time spent handling receptions of 3 weeks after integration, in comparison with the average from the 3 months before this implementation.

6.2 Future Work

Concerning data gathering, one of the main improvement opportunities found was the need to increase amount of data stored regarding estimated reception dates, labeling incidents and return causes. The focus to solve this would be in the integration of warehouse processes with *Spoke*, the company's central source of information (a platform that is linked to the database).

For the data transformation step of the material feature, an alarmistic for the filter built could be included. If a detected string was unknown to the filter, an alert would be triggered. If this string represented a new material, then this material would be added to the bag-of-words. If not, it would be checked whether it was another way of expressing a material that already exists or simply an insertion error.

Data encoding for multi-level categorical features was performed in a logic of one-hot-encoding. However, with the scalability of the business and the increase of number of category levels per feature, another type of encoding could be applied. In a computational effort point of view, target encoding could be a possible alternative.

Focusing now in the modeling phase of this work, the current first level learners of the Stacking model were the best performing ones, after a hyperparameter grid search, on the train set. However, they proved to be the best, but in average. A certain algorithm may be highly accurate in a range of values of the target variable, but incur in a high error in another range(s). Plus, if the meta-model is able to detect these patterns, including worse performing models as first-level learners could be an analysis of interest.

The developed models represent value to other stakeholders besides the ones defined above (HUUB's operational team and brand owners). In fact, HUUB's sales team can benefit from vendor faults predictions for brands on a before acquisition stage. This can help to predict unexpected operational costs, even before the contract is signed and to anticipate the P&L of that possible client. Keep in mind that each brand pays a custom price per item based on the season estimated operational cost. Hence, an integration with the results provided by this thesis work and the Marketing & Sales team can be defined as a future valuable implementation.

Lastly, the integration of the APIs developed with the *Spoke* platform could enhance user experience, since information would be even more easily accessible.

Bibliography

Aggarwal, Charu C. 2014. *Data Classification: Algorithms and Applications*. Chapman \& Hall/CRC

Aldave, Roberto and Jean-Pierre Dussault. 2014. Systematic Ensemble Learning for Regression.

Awad, Mariette and Rahul Khanna. 2015. *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*.

Basak, Debasish, Srimanta Pal, Dipak Ch and Ra Patranabis. 2007. "Support Vector Regression". Paper presented at Neural Information Processing

Brockwell, Peter J. and Richard A. Davi. 2002. *Introduction to Time Series and Forecasting*. 2 ed. New York: Springer.

Chopra, Sunil and Peter Meindl. 2004. *Supply chain management: Strategy, planning, and operation*. Upper Saddle River: N.J: Prentice Hall.

Christopher, Martin. 2011. *Logistics & Supply Chain Management*. 4 ed.: Financial Times Prentice Hall.

Cover, Thomas M. 1974. *The Best Two Independent Measurements Are Not the Two Best*. Vol. SMC-4.

Delen, Dursun and Haluk Demirkan. 2013. "Data, information and analytics as services". *Decision Support Systems* no. 55 (1):359-363.

Demirkan, Haluk and Dursun Delen. 2013. "Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud". *Decision Support Systems* no. 55 (1):412-421.

Ding, Chris and Hanchuan Peng. 2003. "Minimum Redundancy Feature Selection from Microarray Gene Expression Data". *Journal of bioinformatics and computational biology* no. 3 2:185-205.

Djouama, A., E. Zoechmann, S. Pratschner, M. Rupp and F. Y. Ettoumi. 2016. "Predicting CSI for Link Adaptation Employing Support Vector Regression for Channel Extrapolation". Paper presented at 20th International ITG Workshop on Smart Antennas, in Munich, Germany.

Doquire, Gauthier and Michel Verleysen. 2011. "An Hybrid Approach to Feature Selection for Mixed Categorical and Continuous Data".

Forker, L.B. 1997. "Factors affecting supplier quality performance". *Journal of Operations Management* no. 15 (4):243-269

Friedman, Jerome. 2002. "Stochastic Gradient Boosting". *Computational Statistics & Data Analysis* no. 38 (4):367-378.

García, Salvador, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez and Francisco Herrera. 2016. "Big data preprocessing: methods and prospects". *Big Data Analytics* no. 1 (1).

Ge, Zhiqiang, Zhihuan Song, Steven X. Ding and Biao Huang. 2017. "Data Mining and Analytics in the Process Industry: The Role of Machine Learning". *IEEE Access* no. 5:20590-20616.

GitHub. "Stacking Classifier". Accessed 27 Jun 2018. https://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier/#methods.

Han, Jiawei and Micheline Kamber. 2006. *Data Mining: Concepts and Techniques*. Edited by Jim Gray. Second ed.: Diane Cerra.

Hansen, J.V. 2000. *Combining Predictors: Meta Machine Learning Methods and Bias/variance & Ambiguity Decompositions*. Aarhus University, Computer Science Department.

Hastie, Trevor, Robert Tibshirani and Jerome Friedman. 2001. *The Elements of Statistical Learning*. 2 ed.

Hastie, Trevor, Robert Tibshirani and Martin Wainwright. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman & Hall/CRC

Hui Xiong, Gaurav Pandey, Michael Steinbach, Vipin Kumar. 2006. "Enhancing Data Analysis with Noise Removal". *IEEE Transactions on Knowledge and Data Engineering* no. 18 (3):304-319

Ian H. Witten, Eibe Frank 2006. *Data Mining: Practical Machine Learning Tools and Techniques*. Edited by Jim Gray. Diane Cerra

Jia, Peng, Kannan Govindan, Tsan-Ming Choi and Sivakumar Rajendran. 2015. "Supplier Selection Problems in Fashion Business Operations with Sustainability Considerations". *Sustainability* no. 7 (2):1603-1619.

Koprulu, Asli and M. Murat Albayrakoglu. 2007. *Supply Chain Management in the Textile Industry: a Supplier Selection Model with the Analytical Hierarchy Process*.

Krstajic, Damjan, Ljubomir J. Buturovic, David E. Leahy and Simon Thomas. 2014. "Cross-validation pitfalls when selecting and assessing regression and classification models". *Journal of Cheminformatics* no. 6 (1):10. https://doi.org/10.1186/1758-2946-6-10.

Lambert, Douglas M., Martha C. Cooper and Janus D. Pagh. 1998. "Supply Chain Management: Implementation Issues and Research Opportunities". *The International Journal of Logistics Management* no. 9 (2):1-20.

Maimon, Oded and Lior Rokach. 2005. *Data Mining and Knowledge Discovery Handbook*. Springer-Verlag New York, Inc.

Peng, Hanchuan, Fuhui Long and C. Ding. 2005. "Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy". *IEEE Transactions on Pattern Analysis and Machine Intelligence* no. 27 (8):1226 - 1238.

Pyle, Dorian. 1999. *Data Preparation for Data Mining*. San Francisco: MorganKaufmannPublishersInc.

Ren, Ye, Le Zhang and Ponnuthurai Nagaratnam Suganthan. "Ensemble Classification and Regression-Recent Developments, Applications and Future Directions [Review Article]". *IEEE Computational Intelligence Magazine* no. 11:41-53.

Rokach, Lior. 2009. "Ensemble-based classifiers". *Artificial Intelligence Review* no. 33 (1-2):1-39.

Sahay, B. S. and Jayanthi Ranjan. 2008. "Real time business intelligence in supply chain analytics". *Information Management & Computer Security* no. 16 (1):28-48.

Silva, Anthony Mihirana De and Philip H. W. Leong. 2015. *Grammar-Based Feature Generation for Time-Series Prediction*. Springer.

Tang, Jiliang, Salem Alelyani and Huan Liu. 2014. "Feature Selection for Classification: A Review". *Data Classification: Algorithms and Applications*:37.

Tiwari, Sunil, Hui M. Wee and Yosef Daryanto. 2018. "Big data analytics in supply chain management between 2010 and 2016: Insights to industries". *Computers & Industrial Engineering* no. 115:319-330.

Tsikriktsis, Nikos. 2005. "A review of techniques for treating missing data in OM survey research". *Journal of Operations Management* no. 24 (1):53-62.

Wang, Gang, Angappa Gunasekaran, Eric W. T. Ngai and Thanos Papadopoulos. 2016. "Big data analytics in logistics and supply chain management: Certain investigations for research and applications". *International Journal of Production Economics* no. 176:98-110.

Wirth, Rüdiger and Jochen Hipp. 2000. "CRISP-DM: Towards a Standard Process Model for Data Mining". Paper presented at Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining.

Zaki, Mohammed J. and Wagner Meira Jr. 2014. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press.

ANNEX A: Data Representation

Table 11 ΔDays and ΔQuantity datasets representation variables

#	Name	Values
0	Δ days	$\mathbb{Z}$
1	Δ quantity	$\mathbb{Z}$
2	Vendor	$\mathbb{N}$
3	VendorCountry	$\mathbb{N}$
4	VendorPhone	{0,1}
5	VendorEmail	{0,1}
6	VendorCurrency	$\mathbb{N}$
7	VendorAddressType	$\mathbb{N}$
8	Brand	$\mathbb{N}$
9	% per ProductGender	$\mathbb{N}$
10	% per ProductAgeGroup	$\mathbb{N}$
11	% per ProductType	$\mathbb{N}$
12	% per ProductFamily	$\mathbb{N}$
13	% per ProductSubFamily	$\mathbb{N}$
14	% material composition	$\mathbb{N}$
15	PriceRetail	$\mathbb{R}\geq 0$
16	PriceWholesale	$\mathbb{R}\geq 0$
17	Season	$\mathbb{N}$
18	Number_items_per_PO	$\mathbb{N}$
19	Number_quantity_updated	$\mathbb{Z}\geq 0$
20	Number_receptiondate_updated	$\mathbb{Z}\geq 0$

Table 12 Returns and labeling faults datasets representation

#	Name	Values
0	LabelFaultType	{0,1,2}
1	ReturnReason	{1, 2,…, 7, 8}
2	Vendor	$\mathbb{N}$
3	VendorCountry	$\mathbb{N}$
4	VendorCity	$\mathbb{N}$
5	VendorPhone	{0,1}
6	VendorEmail	{0,1}
7	SoCurrency	$\mathbb{N}$
8	VendorAddressType	$\mathbb{N}$
9	Brand	$\mathbb{N}$
10	BrandCountry	$\mathbb{N}$
11	BrandCity	$\mathbb{N}$
12	ProductGender	$\mathbb{N}$
13	ProductAgeGroup	$\mathbb{N}$
14	ProductType	$\mathbb{N}$
15	ProductFamily	$\mathbb{N}$
16	ProductSubfamily	$\mathbb{N}$
17	ProductModel	$\mathbb{N}$
18	ProductMaterial	$\mathbb{N}$
19	Carrier	$\mathbb{N}$
20	Season	$\mathbb{N}$
21	CustomerCountry	$\mathbb{N}$
22	CustomerCity	$\mathbb{N}$
23	SalesChannel	$\mathbb{N}$
24	PriceRetail	$\mathbb{R}\geq 0$
25	PriceWholesale	$\mathbb{R}\geq 0$
26	Number_items_per_SO	$\mathbb{N}$
27	Days_since_contract_supplier	$\mathbb{Z}\geq 0$
28	SalesorderDate	Y-M-D

ANNEX B: Best Prediction Models' Hyperparameters

Table 13 Prediction Models' Hyperparameters

	Hyperparameters					
ΔQuantity	λ	Criterion	Maximum depth	Number Estimators	Kernel	C
Lasso	1					
DT		MSE	70			
RF		MSE	80	20		
SVR					RBF	1
ΔDays	λ	Criterion	Maximum depth	Number Estimators	Kernel	C
Lasso	1					
DT		MSE	60			
RF		MSE	60	20		
SVR					RBF	10

Table 14 Stacking Hyperparameters

	Hyperparameters					
	Base-learners	Maximum Depth	Kernel	C	λ	Meta-learner
ΔQuantity	DT DT SVR	70 80	 RBF	 1		SVR RBF
ΔDays	DT SVR LASSO	60	 RBF	 10	 1	LASSO

ANNEX C: UML Local Database of API Request Prediction

Figure 22 UML API Request Prediction database