Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

# Using supervised machine learning for B2B sales forecasting: A case study of spare parts sales forecasting at an after-sales service provider

D. Rohaan[a,*], E. Topan[a], C.G.M. Groothuis-Oudshoorn[b]

[a] Industrial Engineering and Business Information Systems (IEBIS), Faculty of Behavioural Management and Social Sciences, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

[b] Health Technology and Services Research (HTSR), Faculty of Behavioural Management and Social Sciences, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

ARTICLE INFO

*Keywords:*
Supervised machine learning
Natural Language Processing (NLP)
B2B sales forecasting
Prioritization on sales potential
Information Extraction
Imbalanced data

ABSTRACT

In this paper, we present a method to use advance demand information (ADI), taking the form of request for quotation (RFQ) data, in B2B sales forecasting. We apply supervised machine learning and Natural Language Processing techniques to analyze and learn from RFQs. We apply and test our approach in a case study at a large after-sales service and maintenance provider. After evaluation we found that our approach identifies ~ 70% of actual sales (recall) with a precision rate of ~ 50%, which represents a performance improvement of slightly more than a factor 2.5 over the current labor-intensive manual process at the service and maintenance provider. Our research contributes to literature by giving step-by-step guidance on incorporating artificial intelligence in B2B sales forecasting and revealing potential pitfalls along the way. Furthermore, our research gives an indication of the performance improvement that can be expected when adopting supervised machine learning into B2B sales forecasting.

## 1. Introduction

In a business-to-business (B2B) environment, forecasting future demand is quite crucial as the entire production and supply process depends on these forecasts. There are several traditional forecasting methods mostly based on using past sales. With the advancements in information technologies, companies possess ever more data with the potential to be mined for valuable insights and/or utilized for advanced analytics applications, e.g. machine learning (ML). The majority, an estimated 80–90% of big data is unstructured data (e.g. emails), which is furthermore growing faster than any other type of data. Unstructured data is information that does not have a recognizable structure; it comes in many forms and thus is not a good fit for a mainstream database (Gandomi & Haider, 2015). A solution to analyzing such big unstructured data is natural language processing (NLP), which is a subfield of artificial intelligence that gives machines the ability to read, understand and derive meaning from human languages (Hirschberg & Manning, 2015). Furthermore, NLP is said to have the ability to automate data extraction from large volumes of unstructured text (Li & Elliot, 2019).

In this paper we focus on B2B sales forecasting using advance or future demand information coming from customers, taking the form of *requests for quotation* (RFQs). RFQs are uncommitted requests for a quote of spare parts and/or exchange of parts, by means of email containing unstructured text, that do not necessarily result in a sale. Despite the fact that they are uncommitted, RFQs can be used to predict future demand using artificial intelligence techniques, e.g. supervised machine learning and natural language processing.

The research in this paper is a case study carried out at a large after-sales service and maintenance provider, to which we will refer as *the service provider*. The service provider receives a large number of RFQs. Yet, the average ratio that an RFQ ever becomes a sale is only about 17%. Furthermore, these large number of RFQs exceed capacity of employees responsible for responding to the RFQs. This increases the respond time of the service provider to an RFQ, which is an important factor for the success of a sale. Consequently, customers may complain and even move to a competitor. Therefore, it is important for the service provider to pick up RFQs that have higher chance of sale as not to waste the efforts of sales employees, which are expensive and scarce. Thus there is a clear need/opportunity for the service provider to process its RFQs in a ’smart’ manner.

The objective of this research is two-fold. First, we propose a method to use advance demand information, taking the form of request for

[*] Corresponding author.
*E-mail address:* david.rohaan@hotmail.com (D. Rohaan).

https://doi.org/10.1016/j.eswa.2021.115925
Received 4 April 2021; Received in revised form 8 August 2021; Accepted 16 September 2021
Available online 13 October 2021

quotation (RFQ) data, to predict B2B sales. To achieve this goal, we use (i) supervised machine learning to predict likelihood of sale of RFQs and subsequently prioritize RFQs on sales potential, and (ii) natural language processing to automate this prioritization by automatically extracting required input data from RFQs and feeding it into the prediction model. Using our method, we enable the service provider to use its limited resources in the best way to generate maximum sales. Second, with our research we seek to bridge the gap between theory and practice by giving step-by-step guidance on incorporating supervised machine learning in B2B sales forecasting, revealing potential pitfalls along the way. Examples of such pitfalls are information leaking, due to various reasons, consequently making performance seem better than it actually is and complete loss of information due to the wrong choice of encoding.

The remainder of the paper is structured as follows: Section 2 presents the theoretical background and discusses our contribution to literature. Section 3 discusses our method. The results are presented in Section 4. In Section 5 we conclude our research and discuss the main findings of our work, indicate the limitations, and lay the foundation for further research. Finally, we discuss the ethical considerations of our research in Section 6.

## 2. Literature review

This section provides an overview of prior work relevant to achieving our research objective; B2B sales forecasting, imbalanced data classification, automation using NLP and spare part demand forecasting, including studies on advance demand information (ADI).

### 2.1. *Business-to-business sales potential forecasting*

Bohanec et al. (2015b) proposes a methodology for incorporating supervised machine learning in B2B sales forecasting (Fig. 1). Supervised Machine Learning is a variation of the machine learning paradigm where a classifier maps feature data, describing measurable properties or characteristics of a phenomenon being observed/analyzed, onto a class label. Here, a classifier is defined as an algorithm that identifies to which class an observation belongs, on the basis of a training data set containing (historical) observations whose class labels are known (Aggarwal, 2014).

The first step in this methodology is to create a sales opportunity representation based on historical sales data. This step requires identifying feature data describing a sales opportunity and deriving additional custom features, so called 'meta variables' (Mortensen, Christison, Li, Zhu, & Venkatesan, 2019). The second step is the data preparation, entailing data cleaning, -transformation and -splitting to prevent "Garbage in, Garbage out". The objective of the third step is to identify the model that is best in predicting sales potential. Here, we extended the existing methodology by incorporating a feature selection methodology (Bohanec et al., 2015a), hyperparameter tuning and classification threshold optimization. The final step uses ML techniques and visualizations to gain/emphasize insights which the sales department can use to adjust their current mental decision models.

Despite major progress within forecasting methods as a result of advancements in machine learning research, B2B sales forecasting methods experience little improvement (Bohanec et al., 2015b). Our research contributes to B2B sales forecasting in two ways. First, by giving step-by-step guidance on incorporating supervised machine learning in B2B sales forecasting and revealing potential pitfalls along the way, we seek to bridge the gap between theory and practice. Second, we give an indication of the performance improvement that can be expected when adopting supervised machine learning into B2B sales forecasting, showing its need for adoption if one is to stay ahead of competition.

### 2.2. *Imbalanced data classification*

Imbalanced data is characterized by an unequal frequency distribution of instances among classes and is present in our research since the average ratio of an RFQ ever becoming a sale is only about 17%. Classification with imbalanced data has encountered a significant drawback of the performance achieved by most standard classifier algorithms which assume a relatively balanced class distribution and equal misclassification costs (Sun, Wong, & Kamel, 2009). Since minority class instances occur less frequent, classification rules predicting the minority class(es) tend to be rare, undiscovered or ignored and consequently, samples belonging to the minority class(es) are misclassified more often than those belonging to the majority class(es) (Sun et al., 2009).

Theoretical and experimental studies indicate that, besides an unequal class frequency distribution, the following factors influence the modeling of a capable classifier in identifying rare events (Sun et al., 2009):

- *Sample size*. When sample size is limited, discovering patterns corresponding to the minority class is unreliable. Experimental observations indicate that as the size of the training set increases, the error rate caused by the imbalanced class distribution decreases (Japkowicz & Stephen, 2002).
- *Class separability*. Referring to the degree of discriminative patterns within classes. Research shows that the unequal frequency distribution of instances among classes by itself is less worrisome, but combined with overlapping discriminative patterns between classes, it can significantly decrease the number of minority class instances correctly classified (Prati & Batista, 2004).
- *Within-class imbalance*. In many classification problems, a single class is composed of various subclasses. Within-class imbalance corresponds to an imbalanced class distribution among subclasses and worsens the imbalance distribution problem in two ways: (1) increased learning complexity and (2) within-class subclasses are usually not apparent (Japkowicz, 2001).

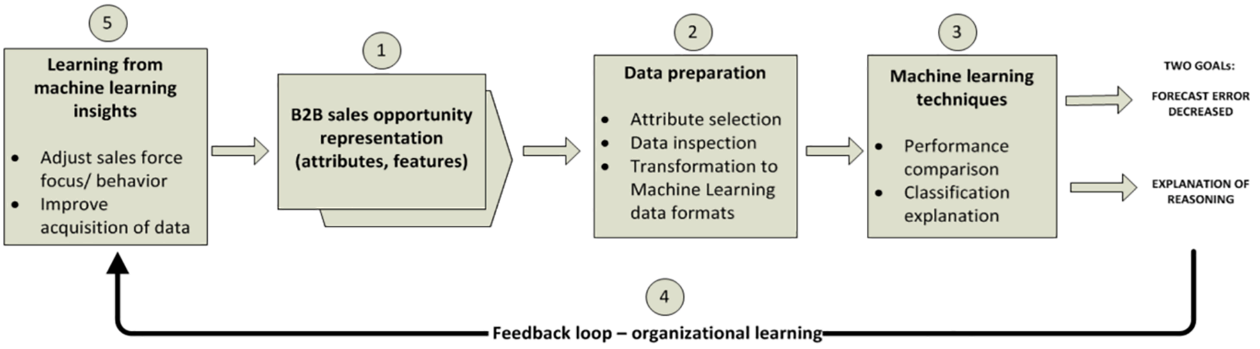

5
Learning from machine learning insights
Adjust sales force focus/ behavior
Improve acquisition of data
1
B2B sales opportunity representation (attributes, features)
2
Data preparation
Attribute selection
Data inspection
Transformation to Machine Learning data formats
3
Machine learning techniques
Performance comparison
Classification explanation
TWO GOALs:
FORECAST ERROR DECREASED
EXPLANATION OF REASONING
4
Feedback loop – organizational learning

**Fig. 1.** Predicting B2B sales potential methodology (Bohanec et al., 2015b).

### 2.3. Automation using NLP

Automation of B2B sales forecasting is vital for a number of reasons: achieving greater sales productivity and sales go-to-market alignment (Lawrence et al., 2010), improved planning (Lu & Kao, 2016), higher efficiency and sale prioritization (Duncan & Elkan, 2015), effective allocation of resources (D'Haen & Van den Poel, 2013) and understanding of the driving factors behind successful sale (Bohanec, 2017; Lambert, 2018).

The solution to automated B2B sales forecasting lies in the field of Natural Language Processing, more specifically in its sub-field Information Extraction. Information extraction concerns extraction of structured information from unstructured or semi-structured text in machine-readable documents (Jiang, 2012). Two fundamental tasks of information extraction are named entity recognition (NER) and relation extraction. First, a named entity is a sequence of words that refers to a real-world entity. NER identifies named entities from unstructured text and classifies them into a set of predefined types. Usually, NER cannot be simply accomplished by string/pattern matching against pre-compiled dictionaries, so-called entity ruler, because (1) named entities usually do not form a closed set and (2) named entities can be context dependent. In such cases a NER model could provide a solution, which identifies and categorizes entities in unstructured data, on the basis of unstructured labelled training data. Second, relation extraction is the task of detecting and characterizing the semantic relations between entities in text, which is less relevant for our research objective.

### 2.4. Spare parts demand forecasting

There are several papers on forecasting spare parts demand. The main focus of this stream of research is to forecast slow moving erratic, lumpy, or intermittent demand patterns, which are typical characteristics of spare parts demand. One of the seminal works is Croston (1972). Several other papers propose approaches to extend this work e.g., Syntetos and Boylan (2005), Teunter, Syntetos, and Babai (2011) and Babai, Dallery, Boubaker, and Kalai (2019). We refer to Boylan and Syntetos (2010) and Pinçe, Turrini, and Meissner (2021) for reviews. Although our aim is to forecast spare parts demand also, we differ from this literature in three ways:

1. Unlike these papers, who focus on point forecast of demand using demand historical sales data, we estimate the likelihood of an RFQ to be successful.
2. In this regard, we focus on the likelihood of each individual RFQ, and therefore, our predictions are coupled with each individual piece of information (RFQ).
3. Therefore, our methodology is also different. Unlike those paper on forecasting spare parts, which predominantly use time series models, we use machine learning and natural language processing to forecast spare part demand based on unstructured data retrieved from RFQs.

Furthermore, particularly because of (1) and (2), our paper is much closer to advance demand information (ADI) since RFQs can be considered ADI (Topan, Tan, van Houtum, & Dekker, 2018). Yet, different from those papers on ADI, we are presenting a case study to show how RFQs can be used as an ADI and how this information is used to estimate the success probability that an RFQ will become a sale.

## 3. B2B sales potential forecasting model

In this section we explain the methodology used to create the machine learning model step-by-step, structured according to the framework of supervised machine learning (Fig. 2).

Supervised machine learning (Fig. 2) starts by collecting historical labelled data (data whose class is known) of instances, reflecting features (that could be) of importance in predicting class label. Then, data cleaning takes place after which the cleaned data is split into a training- and testing data set (usually ratio 70/30 or 80/20 respectively). Next, different classifiers are trained on different feature subsets of the training data and afterwards applied to the testing data. Accordingly, a performance estimation can be derived by comparing the predicted outcomes of the testing data with the true known outcomes of the testing data. Finally, the combination of feature subset and classifier that yields the best prediction performance is adopted and applied to future/unseen cases.

### 3.1. Data acquisition

#### 3.1.1. From RFQ to sales order

The process from RFQ to sales order at the service provider is shown in Fig. 3. If an RFQ is deemed to have the potential to result in a sale, a quote is created containing the offer (price, lead- time, etc.) in response to the RFQ. A quote may contain multiple (quote) lines, each representing the requested quantity for a specific part by the corresponding customer. Consequently, each quote line may or may not convert into a sales order.

RFQs that were not pursued by the service provider cannot be considered in our research since these lack a class label (Section 3). Therefore, we work with quote line data in our research. The inevitable consequence of this is that the input data, and therefore the model and predictions will initially contain a bias induced through earlier RFQ assessments of the service provider.

#### 3.1.2. Data set

The data set used for training- and evaluating our model covers features from multiple data sources, describing the customer, the requested part, the (requested) part vendor, the corresponding (or non-existing) sales order and concerned quote, linked together on primary keys. Note, primary key features are unique identifiers and therefore solely used to link data sources rather than for model learning. Furthermore, note that quote features cannot be used for learning since these are not available at the point in time an RFQ arrives.

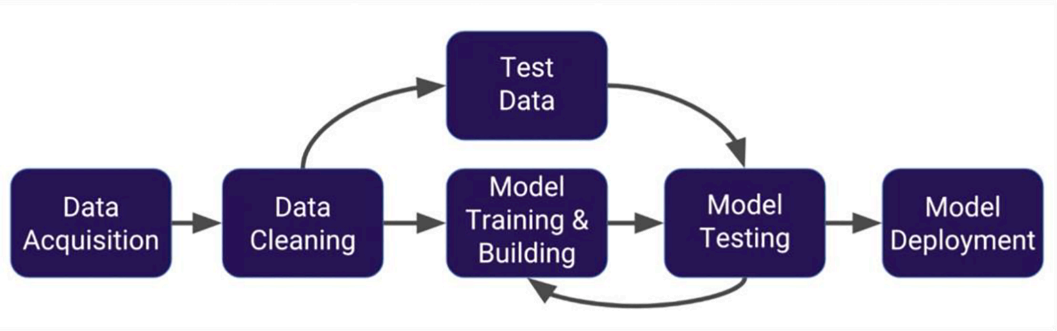


**Fig. 2.** Supervised Machine Learning framework.

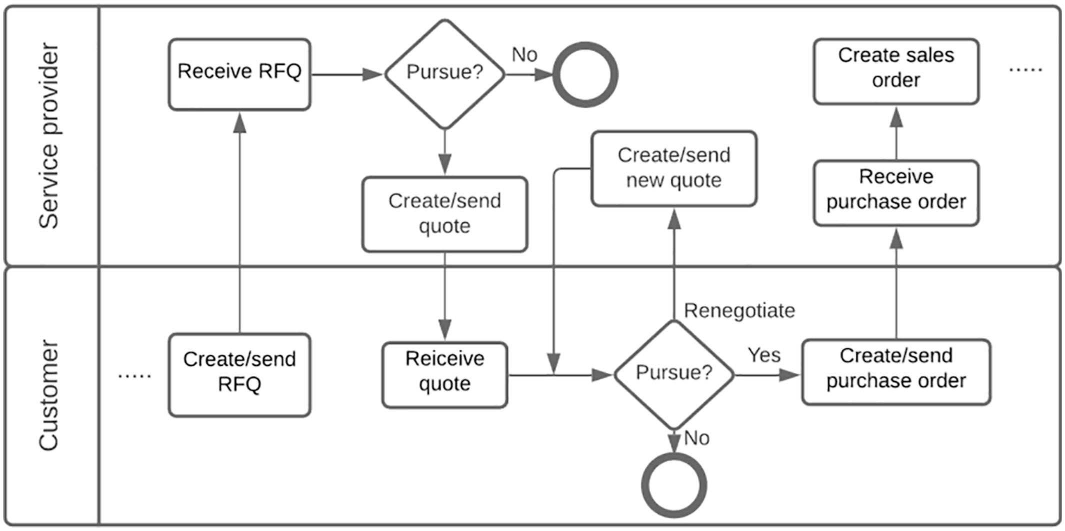


**Fig. 3.** Process from RFQ to sales order at the service provider.

We created 5 additional custom features (*meta*-features), as example cases of B2B sales prediction with ML have shown that these can potentially capture influence which the default features do not, and can be significant predictors (Mortensen et al., 2019). These meta features are the following:

*Frequency part*, frequency count of a part number, and thus part, in the sales order data at the time that the quote was issued.

*Frequency customer*, frequency count of an account number, and thus customer, in the sales order data at the time that the quote was issued.

*Hit rate account*, the percentage of the total quoted value for a customer account that converted to a sale at the time that the quote was issued.

*Hit rate part*, the percentage of the total quoted value for a part that converted to a sale at the time that the quote was issued.

*Stock*, whether the requested part is on stock. This *meta*-feature was created to capture response time, which is believed to be an important factor for the success of a sale. This feature was approximated in consultation with process experts at the service provider since stock level data was not recorded over time.

### 3.1.3. Linking data sources

*3.1.3.1. Linking sales orders to quotes.* The quote line- and sales order data sources are linked on features account number, part number and sales type with a one to one-or-many relation. This relation is in line with reality because a quote may be ordered multiple times by the customer within a certain time frame in which the quote is (still) active. Hence, a quote line is linked to 0, 1 or more than one sales order(s). There exist cases in which a sales order is created without a quote line. However, these are rare. Yet, there exists no unique identifier which a quote line and sales order have in common. Therefore, quote-sales order linking is approximated by requiring an account number-, part number- and sales type feature value combination match as well as a valid timeline in terms of both chronological order and the number of days between issuance of quote and sales order. Regarding the latter, we make a distinction between regular customers and governments, where the subsequent is allowed more time due to bureaucratic approvals.

*3.1.3.2. Linking customer- and part number data to quotes.* Customer/part- and quote data sources are linked on features account number/part number respectively, representing unique customers/parts, with a relation one to one-or-many. Yet, this link is valid only when the customer- and part in the quote occur in the customer- and part data source, which by their design only contain customers/parts that have been sold (to) at least once in the past. Hence, when a quote contains a customer/part that has not been sold in the past, all feature values will be missing, and by the design of the data sources, this quote will not have converted into a sale and this poses a data leak. We eliminate this leak by removing all rows for which all feature values of the customer- and/or part data source is missing.

## 3.2. Data cleaning

After establishing the ML data set, data cleaning took place comprising encoding of categorical features and handling missing values.

### 3.2.1. Encoding categorical features

Categorical features were encoded using either label encoding, one hot encoding (OHE) or our own developed ABC encoding, depending on their number of categorical values (Zheng & Casari, 2018).

Categorical features with two categorical values were label encoded, where one categorical value gets replaced with ‘0′ and the other with ‘1′. Be aware that, when applied to features with more than two categorical values, label coding misunderstands data to be in some kind of order. Instead, categorical features with more than two categorical values were one hot encoded, where the original categorical feature is transformed into a number of dummy features equal to the number of categorical values within the original categorical feature. Here, each categorical value is represented by a dummy feature containing ‘0′ and ’1′ entries depending on the presence of the categorical value. Note that OHE automatically induces perfect multicollinearity, which can be resolved by simply removing one random dummy feature.

However, a downside of OHE is that when dealing with too many categorical values e.g., for primary key features, a huge sparse matrix will result which can cause a memory error. Therefore, we developed our own *ABC encoding*, which uses a form of ABC classification (Teunter, Babai, & Syntetos, 2010) where each categorical value is assigned a bin based on their cumulative contribution to the total sales order revenue. For each ABC encoded feature, the top X categorical values (contributing to the total sales order revenue) remain original, whereas the other categorical values are replaced by the (cumulative) bin they were assigned. The number of top X categorical values and bins are decided upon per feature based on the distribution of categorical values amongst bins. ABC encoding is applied to primary key features and has shown to vastly reduce the number of categorical values while retaining (most of) their informativeness.

### 3.2.2. Handling missing values

We determined the percentage missing values (MVs) per feature and addressed each feature based on the nature of their missingness.

There exist three types of missing data: Missing At Random (MAR), Missing Completely At Random (MCAR) and Missing Not At Random (MNAR). First, MAR means that the distribution of MVs depends on the observed data and unknown parameter(s), but not on any missing data. Second, MCAR means that the distribution of MVs is independent of both the observed- and missing data and depends entirely on some unknown parameter(s). Finally, MNAR means that the distribution of MVs is dependent on the MVs itself and therefore signifies meaning (García, Luengo, & Herrera, 2015).

After verifying the nature of missingness for each feature with process experts at the service provider, we differentiate between two approaches for handling MVs. First, instances with $\geq 1$ MAR/MCAR feature values are discarded. We discard these instances because there is plenty of data and imputation, even a sophisticated method such as multiple imputation, would induce unnecessary uncertainty. Second, missing values of MNAR features are replaced by ‘unknown’.

## 3.3. Data splitting

Historically 17% of RFQs turn into a sale, meaning we are dealing with unequal distribution of instances amongst classes, i.e. class imbalance. When dealing with imbalanced data, it is best to take either a balanced- or stratified sample. Both types of samples force a certain composition to the to-be predicted target feature, where in a balanced sample this composition is predefined and in a stratified sample the composition follows the natural class distribution of the original data set.

In our research, we use (random) stratified splitting, ensuring equal frequency distribution of classes in the data used for model training and -evaluation (Kuhn & Johnson, 2019). Moreover, we combine the stratified splitting with k-fold cross validation to prevent overfitting. This means that data is split into k folds with equal class distribution, the model is trained on k-1 folds and evaluated on the remaining fold. This process is repeated for multiple splits, in which the training/testing fold allocation differ. The final performance estimation is the average performance estimation of all k testing folds over all splits.

## 3.4. Model training & building

### 3.4.1. Outlier detection

Outliers only exist in non-categorial features. To detect outliers, Z-scoring method is applied to the numerical features. This method assumes that values of the concerned features are normally distributed. In this method, any instance containing a numerical feature value more than 3 standard deviations away from the mean is discarded.

### 3.4.2. Resampling

Solutions to address class imbalance are adapting existing algorithms, boosting, cost-sensitive learning and data resampling (Sun et al., 2009). In our research we address class imbalance via boosting (considering the Gradient Boosting Classifier algorithm), cost-sensitive learning (indirectly via the F1-score performance measure) and data resampling.

Regarding data resampling, for our case study, there are two approaches: under- and oversampling. This because we want both the majority- and minority class to have an equal recognition rate. Oversampling entails the replication/creation of new instances (usually minority class) from existing ones, and under-sampling the exclusion of instances (usually majority class). Between the two methods, oversampling yields the highest performance increase as it increases the minority class recognition rate without sacrificing the majority class recognition rate (Batuwita & Palade, 2010). Yet, there are also practical issues to take into consideration when choosing between both methods, such as model training time, especially when incorporating cross-validation. Therefore, we prefer under-sampling in our paper.

## 3.5. Model improvement

In our research, we focused our improvement efforts on metric F1-score, which represents the harmonic mean of precision, the percentage correctly predicted sales over the total number predicted to-be sale, and recall, the percentage of correctly predicted sales over the total number of actual sales. We focused on the F1-score since both wrongly predicted sales (concerning precision), which waste time, and wrongly predicted non-sales (concerning recall), which causes sales to be missed, ought to be minimized.

### 3.5.1. Feature selection

A major problem for large data sets, in which many potential predictor features are present, is the curse of dimensionality, i.e., the consequence of high dimensionality of the input that increases the size of the search space in an exponential manner. Feature selection is used to address the curse of dimensionality. This speeds up computation time, improves input data quality, and potentially increases model performance while simultaneously decreasing model complexity. We use a feature selection methodology by Bohanec et al. (2015a). This method consists of the following steps:

1) Ranking features according to importance.
2) Adding the features with the highest importance and monitoring model performance each time until the optimal cut-off point is obtained.
3) Eliminating noise/redundancy using a wrapper method.

It should be noted that literature recommends feature selection to be carried out only if the number of events per variable (EPV), which is the smallest of the number of positive/negative cases (sales/non-sales), divided by the number of independent features, is at least 50 (Heinze & Dunkler, 2016). In the training data, used below for feature selection, the EPV is 979.89.

*3.5.1.1. Feature importance.* Feature ranking was carried out using 5 different filter methods (Inf. Gain, Gain Ratio, Logistic Regression, Chi-Square and Random Forest) via Orange datamining suite (Orange Data Mining, 2020). Here, non-encoded training data was used since we are interested in the importance of features and not the dummy features (categories). Note that these filter methods require categorical features and that Orange datamining suite by default applies equal-frequency discretization (4 intervals) to numerical features.

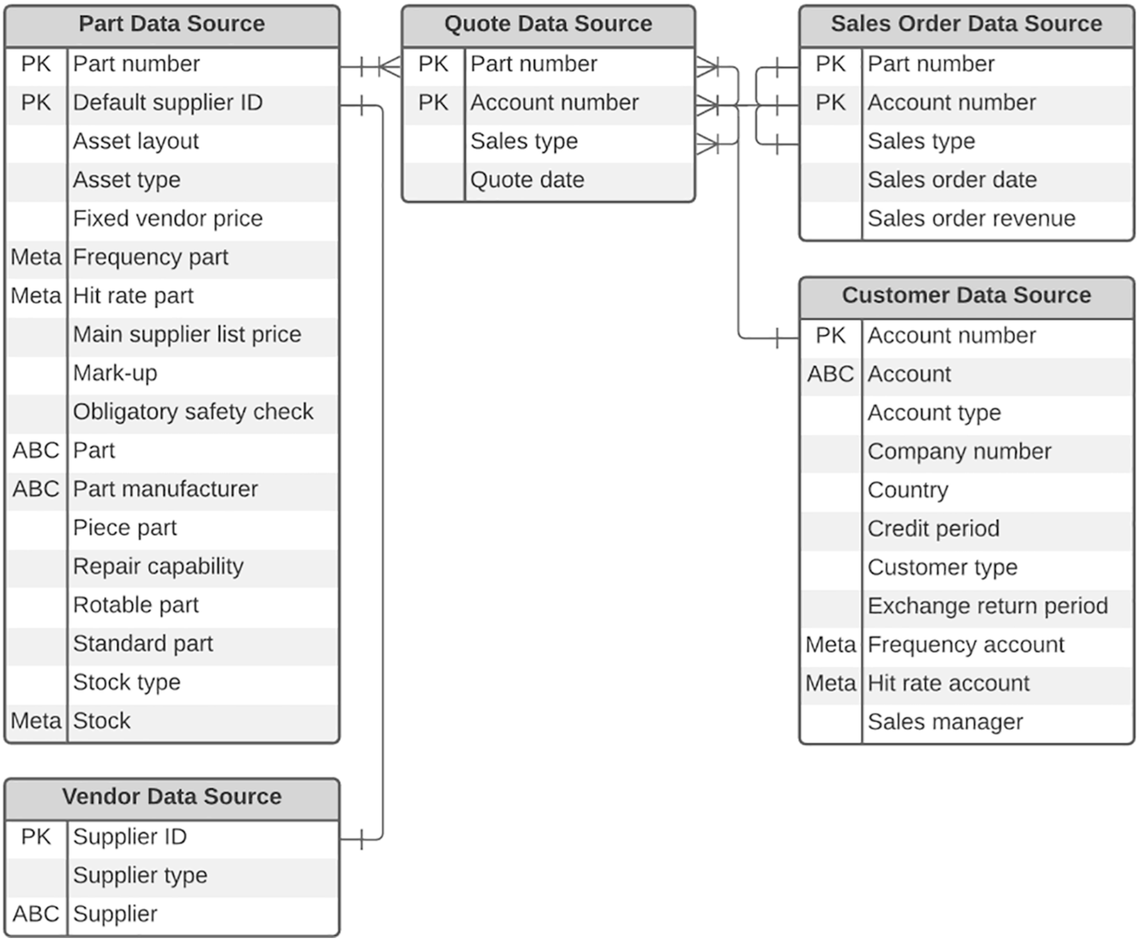
Solely training data is used for feature selection. The testing data is not included to prevent bias in performance estimators (most important features would not necessarily be generalizable). Using the feature ranking, we select the top 15 features per filter method for predicting sales potential (Table 9).

*3.5.1.2. Monitoring performance to detect optimal cut-off point.* The features are added one-by-one according to the feature ranking (from highest- to lowest) and F1 score is monitored. The results are shown in Figs. 5-7. Note that each filter method produces a different ranking. The classifiers we used are Gradient Boosting Classifier, Random Forest and Logistic Regression.

From these figures, we identify the optimal cut-off point for each classifier according to the highest overall F1-score and this yields the following results:

For Gradient Boosting Classifier, top 13 features are selected using filter method Random Forest and this yields an F1 score of 48.84%.

For Random Forest, top 14 features are selected using filter method Random Forest and this produces an F1 score of 53.82%.



| Part Data Source | |
|---|---|
| PK | Part number |
| PK | Default supplier ID |
| | Asset layout |
| | Asset type |
| | Fixed vendor price |
| Meta | Frequency part |
| Meta | Hit rate part |
| | Main supplier list price |
| | Mark-up |
| | Obligatory safety check |
| ABC | Part |
| ABC | Part manufacturer |
| | Piece part |
| | Repair capability |
| | Rotable part |
| | Standard part |
| | Stock type |
| Meta | Stock |

| Quote Data Source | |
|---|---|
| PK | Part number |
| PK | Account number |
| | Sales type |
| | Quote date |

| Sales Order Data Source | |
|---|---|
| PK | Part number |
| PK | Account number |
| | Sales type |
| | Sales order date |
| | Sales order revenue |

| Customer Data Source | |
|---|---|
| PK | Account number |
| ABC | Account |
| | Account type |
| | Company number |
| | Country |
| | Credit period |
| | Customer type |
| | Exchange return period |
| Meta | Frequency account |
| Meta | Hit rate account |
| | Sales manager |

| Vendor Data Source | |
|---|---|
| PK | Supplier ID |
| | Supplier type |
| ABC | Supplier |

**Fig. 4.** Data source linking. Label "PK", "ABC" and "Meta" refer to primary key feature, ABC-encoded feature (Section 3.2.1) and *meta*-feature respectively.

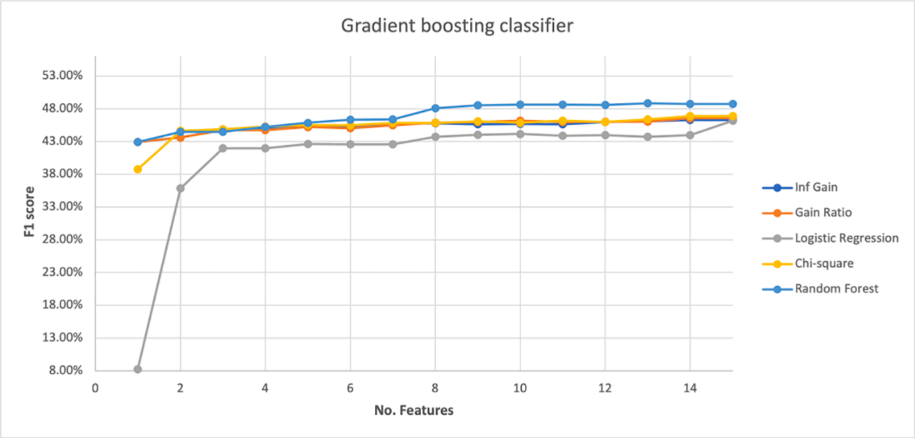


**Fig. 5.** Monitoring F1 score performance Gradient Boosting Classifier using multiple filter methods.

For Logistic Regression, top 15 features are selected using filter method Logistic Regression and this produces an F1 score of 46.05%.

*3.5.1.3. Eliminating noise and redundancy.* Figs. 5-7 show negative or no difference in F1 score during incremental addition of certain features. This indicates the presence of noisy and/or redundant features. The noisy and/or redundant features are eliminated using a wrapper method. For the optimal cut-off point per classifier, features were excluded one-by-one from highest- to lowest ranked and permanently excluded when exclusion increases F1 score performance (Tables 10-12). This results in the following feature selection:

For Gradient Boosting Classifier, 13 features are selected (Table 10). This results in an F1-score of 48.84%.

For Random Forest, 13 features are selected (Table 11). This results in an F1-score of 54.26%.

For Logistic Regression, 12 features are selected (Table 12). This results in an F1-score of 46.69%.

Comparison of the algorithm F1-score performance before- and after the feature selection methodology (Table 1) reveals the presence- and severity of the curse of dimensionality in our dataset. It can be seen in

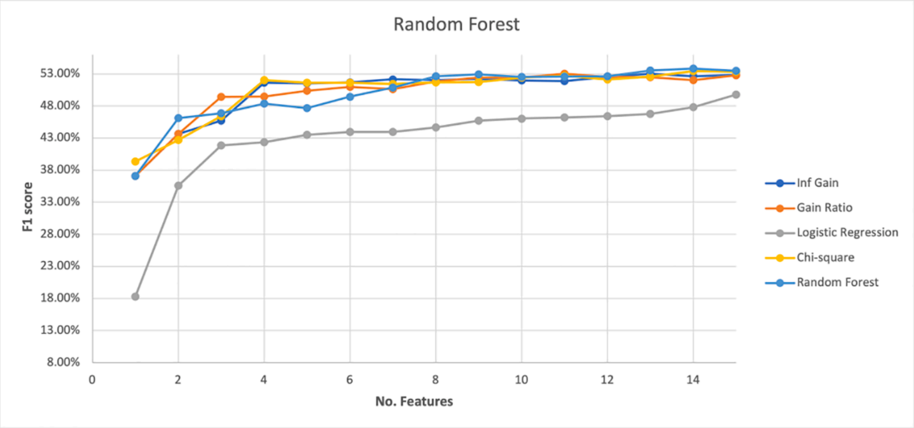


**Fig. 6.** Monitoring F1 score performance Random Forest using multiple filter methods.

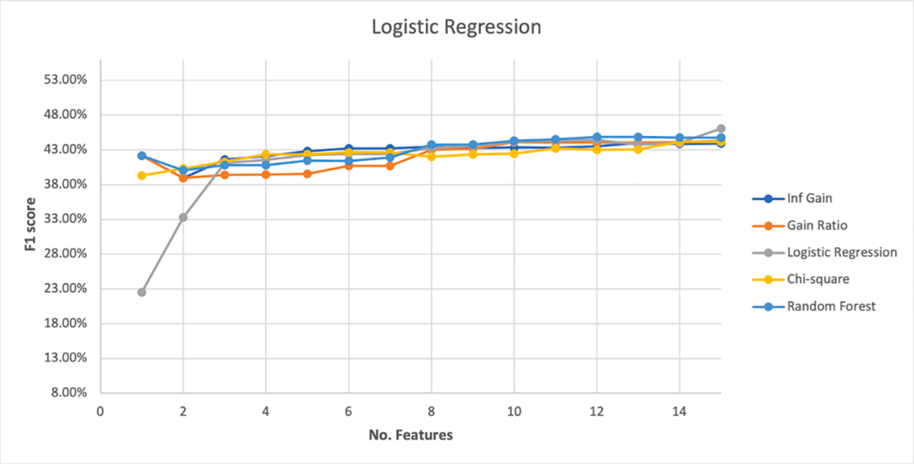


**Fig. 7.** Monitoring F1 score performance Logistic Regression using multiple filter methods.

**Table 1**
Comparison F1-score performance of algorithms before- and after feature selection.

| Algorithm | Before feature selection | | After feature selection | |
|---|---|---|---|---|
| | F1-score | No. features | F1-score | No. features |
| Gradient Boosting Classifier | 48.89% | 28 | 48.84% | 13 |
| Random Forest | 54.14% | 28 | 54.26% | 13 |
| Logistic Regression | 46.80% | 28 | 46.69% | 12 |

Table 1 that reducing the number of features by more than half has little effect on the performance, e.g., F1-score performance differs 0.05–-0.12%.

*3.5.1.4. Model performance comparison.* Next, we compared the resulting best performing models using a k-fold cross validation approach (Tan, Steinbach, & Kumar, 2006), where a confidence interval is calculated for the true difference in performance estimation between two models. This confidence interval (CI) can be calculated as $d_{true}^{CV} = \overline{d} \pm \sigma_{CV}{}^{*}t_{(1-a)(k-1)}$, where $\sigma_{CV}^{2} = \frac{\sum_{j=1}^{k}(d_j-\overline{d})^2}{k(k-1)}$.

Here, $\overline{d}$ is the average performance difference of all k folds,$d_j$ is the difference in performance estimation in the $j^{th}$ fold, $\sigma_{CV}^{2}$ represents the variance of the true performance difference between the models, and coefficient $t_{(1-a)(k-1)}$ can be obtained from the probability table of the t-distribution.

From a 5-fold cross validation comparison it followed that Random Forest performs significantly better than Gradient Boosting Classifier, whereas Gradient Boosting Classifier performs significantly better than Logistic Regression at a confidence level of 99% (see Table 2). This can be concluded for both comparisons since zero does not lie in either of the confidence intervals of the true differences in performance estimates.

### 3.5.2. *Hyper-parameter tuning*

Hyper-parameters are the intrinsic parameters of a (machine learning) model which are set before- and used to control learning (Bishop, 2006). Hyper-parameter tuning is therefore the problem of choosing a set of hyper-parameter values that yields the best (generalizable) performance improvement. To obtain such a generalizable performance improvement an additional (stratified) split, resulting in a validation data set, is required. This because if hyper-parameter tuning would be conducted using solely a train- and test set, there would be a risk of overfitting since parameters can be tweaked until the model performs optimally on the test set. That way, knowledge about the test set leaks and consequently evaluation metrics do no longer report on general performance (Hastie, Tibshirani, & Friedman, 2009).

The most widely used strategies for hyper-parameter optimization are manual- and grid search. In manual search, the operator adjusts the parameters, possibly incorporating knowledge about the effect of adjustments on both the behavior of the model and estimation procedure, whereas grid search is an exhaustive search over a pre-defined grid, returning the best configuration. Yet, literature rather urges the use of

**Table 2**
Model performance comparison using k-fold cross validation approach.

| Random Forest minus Gradient Boosting Classifier | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Fold No. | 1 | 2 | 3 | 4 | 5 | Average | Varlence | 99% Cl |
| Difference | 0.054 | 0.056 | 0.052 | 0.052 | 0.056 | 0.054 | 6.7666E-07 | [0.053,0.055] |
| Gradient Boosting Classifier minus Logistic Regression | | | | | | | | |
| Fold No. | 1 | 2 | 3 | 4 | 5 | Average | Varlence | 99% Cl |
| Difference | 0.022 | 0.016 | 0.028 | 0.023 | 0.020 | 0.022 | 3.6413E-06 | [0.0019,0.025] |

random search, which evaluates a number of random configurations over a pre-defined grid, as it has shown that random search yields equally good or better configurations than grid search in a fraction of the computational time (Bergstra & Bengio, 2012). The idea behind this phenomenon is that not all hyper-parameters are equally important and grid search allocates too many experiments to explore hyper-parameters that have less impact, and therefore it suffers from poor coverage for hyper-parameters that have larger impact.

In our research, we use two sequential runs of random grid search, where the results from the first "exploratory search" are used to create a more narrowed down grid for the second "targeted search". Hyper-parameter values from the grid of the exploratory search are excluded in the targeted search if their frequency of occurrence in the top 20 configurations (Table 13) is below a certain relative threshold. The grid values of the exploratory- and targeted random grid search are shown in Table 3. Fig. 8 shows the average F1 score (3-fold CV) ordered against the number of iterations of the exploratory- and targeted random search. Here, we see a steep- and flat improvement for the exploratory- and targeted random search respectively, confirming the need of an initial exploratory random search and suggesting that increasing the number of iterations in the targeted random search is not likely to result in a better configuration.

From our hyper-parameter tuning methodology followed that the hyper-parameter configuration shown in Table 4 results in the highest performance.

#### 3.5.3. Classification threshold optimization

Classification is based on a probability that the classifier assigns to an observation using the inferences learned from the training data set. By default, the classification threshold is set to 0.5 meaning in this case, that an observation with a predicted probability greater than 0.5 is classified as "Sale" and $\leq$ 0.5 as "No sale". Yet, the classification threshold can be altered and therefore optimized. The classification threshold can be perceived as a hyper-parameter and should be treated accordingly. From the validation curves plotted in Fig. 9, we find that the optimal threshold lies within [0.55, 0.65]. After zooming-in on this area, we determine that the optimal classification threshold, given the optimal feature subset and -hyper parameter configuration, is 0.62.

**Table 3**
Values hyper parameter grid random searches.

| Hyper-parameter | Grid values exploratory search | Grid values targeted search |
|---|---|---|
| Criterion | ['Gini', 'Entropy'] | ['Gini', 'Entropy'] |
| No. estimators | [75, 100, 150, 200, 300, 400, 500, 600, 700, 800] | [75, 100, 150, 200, 300, 400, 500] |
| Max depth | [None, 10, 20, 30, 40, 50] | [None, 40, 50] |
| Min samples split | [2, 5, 10, 15, 20] | [2, 5, 10] |
| Min samples leaf | [1, 2, 5, 10, 15, 20] | [1] |
| Max features | [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] | [4, 5, 6, 7, 8, 9, 10, 11, 12, 13] |
| Bootstrap | [True,False] | [True,False] |

### 3.6. Model testing

We selected five performance measures to evaluate the performance of our prediction model. These are recall, precision, confusion matrix, F1-score, and AUC, and are explained in the remainder of this subsection (Tharwat, 2020). *Precision* is the percentage of correctly predicted sales over the total number of (correctly or falsely) predicted sales. *Recall* is the percentage of correctly predicted sales over the total number of actual sales. A *confusion matrix* summarizes the classification performance of a classifier with respect to some test data. It is a n-dimensional matrix, where n is equal to the number of classes, indexed in one axis by the true class of an observation and in the other axis by the class that the classifier assigns. *F1-score* is the harmonic mean of the precision and the recall. Finally, the *AUC,* which stands for Area Under the ROC (Receiver Operating Characteristics) Curve, indicates the extent to which the prediction model can distinguish between classes. The AUC metric takes values on the interval [0.5, 1], where a score of 1 means the prediction model can perfectly distinguish between classes and a score of 0.5 means the prediction model cannot differentiate between classes.

### 3.7. Model automation

In this section we investigate automation of our prediction model and corresponding sales forecasts.

#### 3.7.1. Discrepancy RFQ data and customer data source

The prediction model takes input features which can all be acquired if the account number (customer) and (requested) part number are known. Each RFQ contains, amongst other things, the requested part number(s), the name of the requesting company and the customer email address, where the latter two can theoretically both be linked to the feature account number. However, in practice there appears to be a significant discrepancy between customer company names as they occur in the customer data source and as they occur in the RFQs consisting of different wording rather than punctuation, capital letters, etc. Furthermore, the service provider did not collect/store customer email address data, corresponding to the RFQs received, due to which this information cannot be used for linking. Consequently, until either one of these issues is resolved, prioritization cannot happen automatically nor autonomously.

However, to demonstrate the feasibility/opportunity we decided to create a proof of concept (POC) for the automation part instead. Within this POC we argue for the use of partial customer email address data, from "@" until the end of the email address, which primarily contains the customer company name. The reason for this being that a partial customer email addresses, representing the company email address domain, will rarely change over time, whereas customer company name can differ (or lack) in each RFQ and is prone to human error.

#### 3.7.2. NLP proof of concept

The service provider RFQ email inbox is accessed after which the most recent emails are retrieved and put in a data frame consisting of columns containing their corresponding conversation ids, sender email addresses and text bodies.

The part number(s) and partial customer email address are extracted

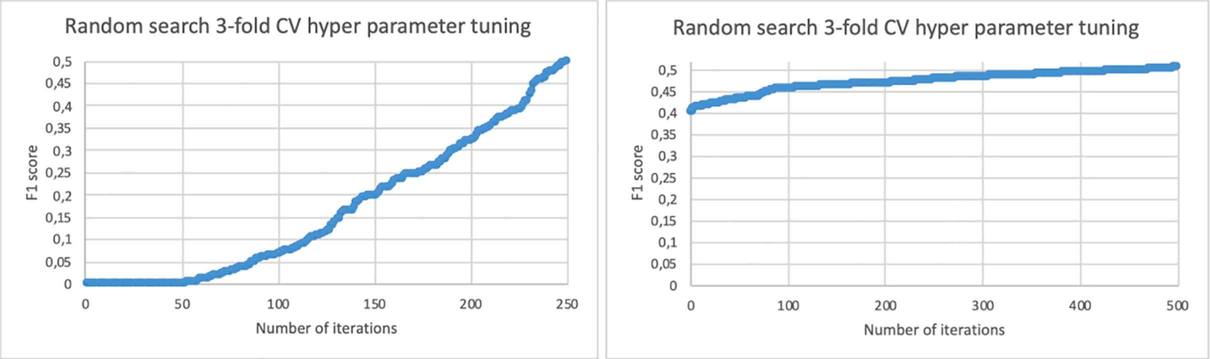


**Fig. 8.** Results exploratory- (left) and targeted (right) random grid search. CV stands for cross-validation.

**Table 4**
Best performing hyper-parameter configuration.

| Hyper-parameter configuration 13-feature Random Forest model | | | | | | |
|---|---|---|---|---|---|---|
| No. estimators | Min samples split | Min samples leaf | Max features | Max depth | Criterion | Bootstrap |
| 400 | 2 | 1 | 13 | None | Gini | False |

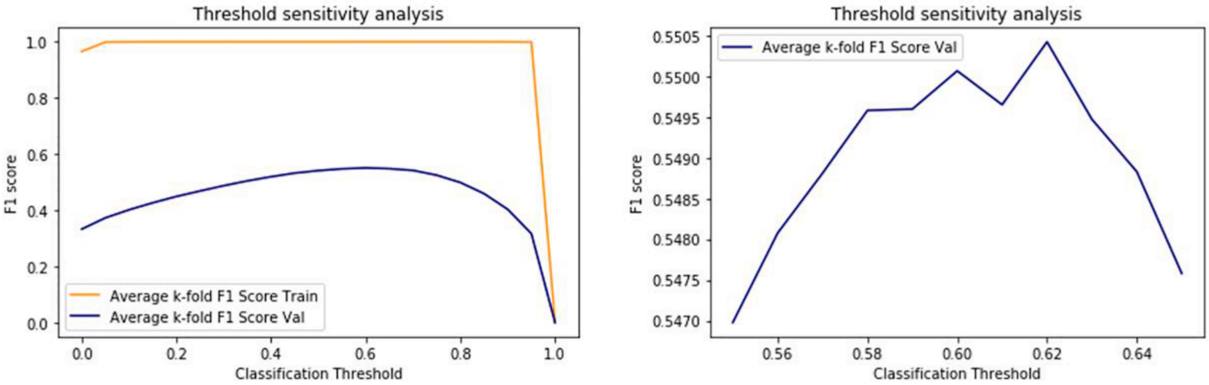


**Fig. 9.** Validation curves classification threshold optimization (k = 3).

from RFQ text bodies. First, part numbers are extracted using an Entity Ruler, which is a matcher based on a dictionary of patterns and corresponding (entity) labels. The Entity Ruler identifies entities in texts which can be called upon per label. The Entity Ruler is given a dictionary of all part numbers (approximately 1 million) as seen in the part data source (Fig. 4) with label "part". Now, when the Entity Ruler is applied to historical and/or future RFQs, it identifies any part number in the RFQ that also occurs in the part data source. We defined a custom tokenizer for extracting part numbers as a standard tokenizer interprets "-" to be an infix and as a consequence splits part numbers, which often contain multiple '-' symbols, into multiple tokens allowing the entity ruler to falsely identify part numbers within the requested part number (s).

Second, the (partial) customer email address data will be recognized/extracted similarly using the entity ruler. Once the service provider collects its historical customer email address data, partial customer email addresses is fed into the Entity Ruler with label "customer email". Now, when the Entity Ruler is applied to historical and/or future RFQs, it identifies any partial customer email address in the RFQ that also occurs in the service providers data. In addition, within extraction of partial customer email addresses a distinction is made between direct- and indirect customer RFQs, the latter entailing (online) spare part marketplaces. For the direct RFQs the partial customer email address is extracted from the sender email address and for the indirect RFQs the partial customer email address is extracted from the RFQ text body.

## 4. Results

In this section we evaluate the performance of both the prediction model and NLP POC and provide insights into the prediction model.

### 4.1. Prediction model

The creation of the prediction model started with the collection of a dataset consisting of ~ 180,000 historical quote lines from 2012 to 2019, describing the features shown in Fig. 4. Next, data cleaning took place, comprising of feature encoding and handling missing values (Section 3.2). Features were encoded using either label encoding, OHE or our custom ABC encoding, depending on their number of categorical values. Quote lines with $\geq$ 1 MAR/MCAR feature values were discarded, whereas MNAR feature values were replaced by 'unknown'. The maximum percentage of missing values amongst MAR/MCAR features is 3.5% and there are two MNAR features with ~ 70% and ~ 35% MVs. Then, data was split using stratified k-fold cross validation (k = 5) in a training- and testing data set. We have used splitting ratio 80/20 for training- and testing data respectively. Afterwards, outlier detection and under-sampling were carried out on the training data (Section 3.4).

Investigation of flagged outlier samples in the ERP system of the service provider revealed that these were caused by human error, confirming the necessity. Under-sampling was carried out to address the class imbalance in the dataset (Sun et al., 2009) since the historical average ratio that an RFQ ever becomes a sale is only about 17%. In our dataset 19% of quote lines converted into a sale. After under-sampling, the data used for training the model consists of 54,874 quote lines, equally distributed amongst classes "Sale"/"No Sale".

Next, we train the prediction model on the remaining 54,874 historical quote lines. The prediction model, following from Section 3.5, is a 13-feature Random Forest model with hyper-parameter configuration as seen in Table 4 and a classification threshold of 0.62. Finally, we apply the trained prediction model to the testing data set and compare the predicted class outcomes with the true known class outcomes. From this follows that the performance is 56.24% for the F1 score, 66.9% for the recall, 48.5% for the precision, 83% for the AUC and, following from the confusion matrix, ~80% of non-sales are correctly predicted (Fig. 10, Table 5).

Thus, our model predicts 48.5% correctly to be a sale. Compared to 19% quote-to-sales order conversion rate in our dataset, resulting from manual handling at the service provider, this represents a performance increase of + 155.3% (slightly more than a factor 2.5 times the performance using manual handling). The need for the prediction model sprung, amongst other things, from the fact that the service provider receives more RFQs than it can process. Using the prediction model in the future allows the service provider to pursue RFQs in order of descending predicted probability of sale, until capacity restrictions are met. Assuming the same distribution of "Sale"/"No sale" amongst the RFQs that previously could not be processed, this will enable the service provider to generate more sales.

### 4.2. *NLP proof of concept*

Performance of the NLP POC has been manually evaluated on 100 random historical RFQs, due to time constraints, from which follows

For 100 out of 100 RFQs the correct part number(s) are recognized.

For 99 out of 100 RFQs a single correct partial customer email address has been recognized. For 1 out of 100 RFQs two partial customer email addresses have been recognized, the reason being that the customer stated two different email addresses, yet still correctly.

However, in addition to the correct part numbers, "other" numbers occurring in the RFQs were falsely recognized as part numbers. These numbers are in fact existing part numbers but not in the context of the RFQ (e.g. the street number of the service provider). This issue is addressed via an array in which frequently falsely identified (part) numbers can be stated which will then not be given to the Entity Ruler and therefore no longer (falsely) recognized. Yet, the Entity Ruler remains the best choice for the task at hand due to its high accuracy performance. This is desired because the extracted part number(s) and partial customer email address will serve as input for the prediction model in a fully autonomous/automated flow, meaning that any unrecognized partial customer email address and/or part number by definition will never become a sale. Note that training a NER model (Section 2.3), which does account for context, was also attempted but had shown high losses during training indicating, and following from performance, that the NER model was unable to learn properly and thus unsuitable for the task at hand.

**Table 5**
Performance metrics sales potential prediction model (5-fold cross validation).

| | Precision | Recall | F1-score |
|---|---|---|---|
| Fold 1 | 0.487 | 0.672 | 0.565 |
| Fold 2 | 0.486 | 0.670 | 0.563 |
| Fold 3 | 0.485 | 0.670 | 0.563 |
| Fold 4 | 0.484 | 0.670 | 0.562 |
| Fold 5 | 0.483 | 0.665 | 0.559 |
| **Average** | **0.485** | **0.669** | **0.562** |
| **Max Δ fold** | **0.004** | **0.007** | **0.006** |

### 4.3. *Model insights*

In this section we want to give insight into the prediction model, which we expect will be perceived by most to be a "black-box", with the objective to foster adoption in practice. First, the 13 most informative features which the prediction model uses to achieve its performance can be seen below in Table 6 with a description, ranked in descending order of importance. Table 6 shows that ABC encoded features (4 out of 13) and custom *meta*-features (3 out of 13) cover more than half of the most informative features. The former, implies that our ABC encoding method (Section 3.2.1), which we apply to primary key features, is effective at retaining the informativeness at a significantly lower computational cost. To illustrate, using our ABC encoding method, the number of categorical values from features account- and part number were reduced from 1,288 and 47,131 to 14 and 29 respectively, yet are part of the most informative features (Account (ABC) and Part (ABC)). Furthermore, the custom *meta*-features from Table 6 are currently not collected by the service provider. Therefore, we recommend the service provider to start collecting- and/or retroactively create these *meta*-features.

Next, we want to give insight into patterns in the training dataset, by extracting association rules. Association rules are if-then statements that help show the probability of relationships between observations in the dataset and can be used to reveal preconditions for class outcomes

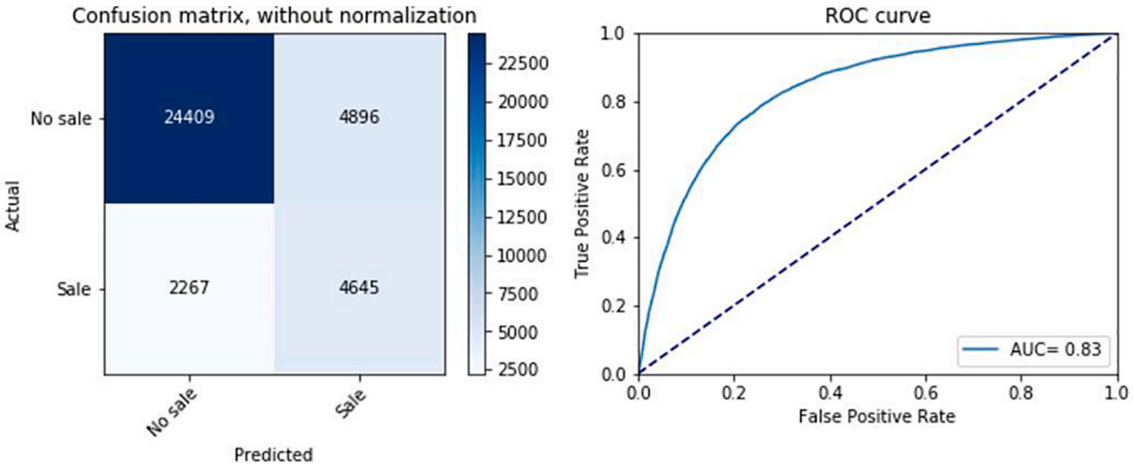


**Fig. 10.** Confusion matrix and AUC sales potential prediction model.

**Table 6**
Features used by the 13-feature Random Forest model to predict sales potential.

| Rank | Feature name | Feature description |
|---|---|---|
| 1 | Hit rate account | The percentage of the total quoted value for an account that converted to a sale at the time that the quote was issued. |
| 2 | Frequency account | Frequency count of an account number, and thus customer, in the sales order data at the time that the quote was issued. |
| 3 | Main Supplier List Price | Unnegotiated supplier part price which is visible to the entire world. This feature is an indication of the purchase/sales price as the service provider can sometimes negotiate a better price. |
| 4 | Frequency part | Frequency count of a part number, and thus part, in the sales order data at the time that the quote was issued. |
| 5 | Account (ABC) | This feature represents ABC classified feature account number. |
| 6 | Customer type | The customer company type. Note: different sales managers might categorize customers differently as there are no categorization rules. |
| 7 | Stock | Whether the requested part was on stock. This feature was approximated in consultation with process experts at the service provider. Here, we assume that if a part was on stock, its delivery window would be ≤ 7 days. |
| 8 | Part (ABC) | This feature represents ABC classified feature part number. |
| 9 | Supplier (ABC) | This feature represents ABC classified feature (default) supplier ID. |
| 10 | Part manufacturer (ABC) | This feature represents ABC classified feature part manufacturer ID. |
| 11 | Rotable part | Binary feature representing whether the part is a rotable part or not. Rotable parts are parts that can be repaired (second-hand). |
| 12 | Sales manager | The regional sales manager. |
| 13 | Supplier type | The type of supplier. |

**Table 7**
Association rules corresponding to the training data set. Sale = 0/Sale = 1 translate to "No sale"/"Sale" respectively.

| No. | Supp | Conf | Lift | Antecedent | Consequent |
|---|---|---|---|---|---|
| 1 | 0.249 | 0.663 | 1.327 | Account (ABC)=(80,100] | Sale=0.0 |
| 2 | 0.193 | 0.691 | 1.383 | Account (ABC)=(80, 100], Part (ABC)=(80,100] | Sale=0.0 |
| 3 | 0.173 | 0.635 | 1.269 | Account (ABC)=(80, 100], Stock=1.0 | Sale=0.0 |
| 4 | 0.164 | 0.654 | 1.307 | Account (ABC)=(80,100], Supplier type=Unknown | Sale=0.0 |
| 5 | 0.163 | 0.62 | 1.241 | Account (ABC)=(80, 100], Rotable part=0 | Sale=0.0 |
| 6 | 0.157 | 0.617 | 1.234 | Customer type=AIRLINE, Stock=1.0 | Sale=1.0 |
| 7 | 0.156 | 0.608 | 1.216 | Customer type=AIRLINE, Rotable part=0 | Sale=1.0 |
| 8 | 0.156 | 0.633 | 1.266 | Rotable part=1 | Sale=0.0 |
| 9 | 0.139 | 0.651 | 1.303 | Account (ABC)=(80,100], Part (ABC)=(80,100], Rotable part=0 | Sale=0.0 |
| 10 | 0.132 | 0.827 | 1.654 | Customer type=BROKER | Sale=0.0 |
| 11 | 0.129 | 0.69 | 1.38 | Account (ABC)=(80,100], Supplier type=Unknown, Part (ABC)=(80,100] | Sale=0.0 |
| 12 | 0.128 | 0.652 | 1.305 | Account (ABC)=(80,100], Part (ABC)=(80,100], Stock=1.0 | Sale=0.0 |
| 13 | 0.12 | 0.633 | 1.266 | Sales manager=MLS | Sale=1.0 |
| 14 | 0.12 | 0.618 | 1.237 | Rotable part=1, Stock=1.0 | Sale=0.0 |
| 15 | 0.12 | 0.67 | 1.34 | Customer type=DEFENSE | Sale=1.0 |
| 16 | 0.115 | 0.628 | 1.256 | Account (ABC)=(80, 100], Supplier type=Unknown, Stock=1.0 | Sale=0.0 |
| 17 | 0.114 | 0.677 | 1.354 | Customer type=DEFENSE, Rotable part=0 | Sale=1.0 |
| 18 | 0.114 | 0.638 | 1.557 | Customer type=DEFENSE | Rotable part=0, Sale=1.0 |
| 19 | 0.111 | 0.619 | 1.237 | Account (ABC)=(80, 100], Supplier type=Unknown, Rotable part=0 | Sale=0.0 |
| 20 | 0.108 | 0.848 | 1.696 | Account (ABC)=(80,100], Customer type=BROKER | Sale=0.0 |
| 21 | 0.108 | 0.679 | 2.729 | Customer type=BROKER | Account (ABC)=(80, 100], Sale=0.0 |
| 22 | 0.106 | 0.653 | 1.307 | Customer type=DEFENSE, Part (ABC)=(80,100] | Sale=1.0 |
| 23 | 0.103 | 0.663 | 1.326 | Customer type=AIRLINE, Rotable part=0, Stock=1.0 | Sale=1.0 |
| 24 | 0.103 | 0.664 | 1.328 | Customer type=DEFENSE, Part (ABC)=(80,100], Rotable part=0 | Sale=1.0 |
| 25 | 0.103 | 0.64 | 1.561 | Customer type=DEFENSE Part (ABC)=(80, 100] | Rotable part=0, Sale=1.0 |

(Witten, Frank, & Hall, 2011). Each association rule is represented with three standard evaluation metrics: support, confidence and lift. The support measures the proportion of cases in the data set which contain the antecedent of a rule. The confidence reflects the proportion of the cases in which the antecedent and consequent are satisfied. The lift reports the ratio of the observed support to the expected antecedent and consequent being independent. Table 7 shows the top 25 association rules which were extracted from the training dataset, under the restrictions of ≥ 10% and ≥ 60% for minimal support and -confidence respectively. From Table 7 follows for example that selling non-rotable parts from stock to airlines (line 23) and selling low contributing non-rotable parts to defense (line 24) often result in a sale.

Finally, we want to give insight into the prediction algorithm, Random Forest, itself. We do this through visualization of a single random tree from the Random Forest (Fig. 11), upon which is zoomed-in on levels 1–2. The model consists of 400 trees similar to the one shown in Fig. 11, from which the majority vote determines the prediction outcome.

## 5. Conclusion

In this section, we discuss the main findings of our work, indicate the limitations, and lay the foundation for further research. The research in our paper, a combination of supervised machine learning and natural language processing, has shown to increase B2B sales forecasting performance by + 155.3% over prior manual handling and demonstrated the feasibility of automation through a proof of concept. The need for the sales forecasting model sprung, amongst other things, from the fact that the service provider receives more RFQs than it is able to process. Using the model in the future will allow the service provider to generate more sales, assuming the same distribution of "Sale"/"No sale" amongst the RFQs that previously could not be processed, by responding to RFQs in order of descending predicted probability of sale.

Regarding the limitations and future research direction, first, our prediction model predicts solely sales potential but neglects the value of the corresponding RFQ. This means that a low-cost part (e.g. bolt) could be prioritized over a high value part (e.g. engine) with neglectable difference in sales potential. Second, in our research we focused on optimizing the F1 score, the harmonic mean between precision and recall.

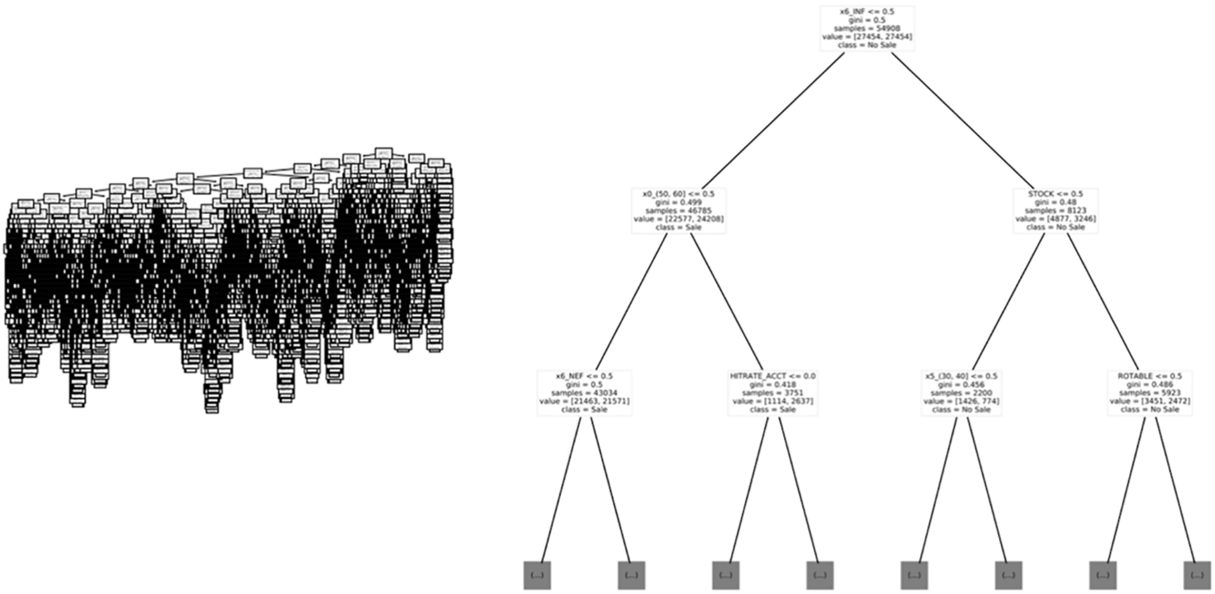


**Fig. 11.** Single tree from the Random Forest (left), zoomed-in upon levels 1–3 (right).

Yet, in practice it is unlikely that a missed sale and wasted employee time are equally valuable, and therefore investigation into an optimal (subjective) precision-recall trade-off is recommended. Third, we recommend retraining the model regularly and monitoring performance over time. Feature selection, hyper-parameter tuning and classification threshold optimization were statically executed. If performance drops over time, e.g., potentially due to change in importance of features, our research should be repeated. Fourth, our model can only be applied to RFQs from existing customers, concerning parts in the product range of the service provider. We elaborate further on this limitation in Section 6. Finally, the NLP POC falsely recognizes certain numbers occurring in the RFQs as part numbers. These numbers were actual part numbers but not in the context of the RFQ (e.g. street number of the service provider). To diminish the extend of this issue we created an array in which frequently falsely identified (part) numbers can be specified, which will then not be fed to the Entity Ruler and therefore no longer (falsely) recognized. Here, we recommend the service provider to investigate the revenue corresponding to such frequently falsely identified parts and decide whether to exclude them.

## 6. Ethical considerations

AI presents three major areas of ethical concern for society: privacy and surveillance, bias and discrimination, and perhaps the deepest, most difficult philosophical question of the era, the role of human judgment (xThe Harvard Gazette, 2020). In our research we encounter ethical considerations in the latter two areas, which we discuss below.

First, the prediction model prioritizes RFQs based on estimated sales potential and may therefore end up labelling legitimate requests with low sales potential (“No sale”). This may be perceived as if the prediction model is introducing ethical issues. However, in the current system, because of limited sales personnel, a similar amount of RFQs is discarded based on manual assessment which may also contain legitimate requests. The prediction model thus solely shifts the grounds upon which RFQs are assessed from subjective to statistical.

Furthermore, as earlier mentioned in Section 3.1.1, the historical quote line data used to train- and evaluate our model contains a bias stemming from earlier RFQ assessments by the service provider. We would like to emphasize that this bias is already present at the service provider and will thus not be introduced by our prediction model. On the contrary, after deployment of the prediction model at the service provider we expect this bias to be slowly phased out over time. The reason for this is that the model in the future will point the service provider toward RFQs with high sales potential, based on statistical inferences from the training data. Thus, over time the model will be retrained on more and more historical RFQs that were labelled by the model itself, and from which, if mistaken, it will learn.

Finally, the prediction model requires i.a. the requested part number (s) and the requesting account number (customer) to prioritize an RFQ on sales potential (see Fig. 4). In practice this means that the prediction model cannot be used for new customers and/or part number(s) outside the product range of the service provider. For part numbers this does not pose an issue since the service provider solely sells from its product range. On the other hand, the inability of the prediction model to handle new customers does pose an issue that must be addressed. To resolve this issue, we propose that the service provider dedicates a fixed fraction of capacity to (manually) handling new customers. Here, we recommend the service provider to use the extracted association rules (Table 7). Fortunately, once a single RFQ from a new customer is processed and the prediction model is subsequently retrained, future requests from this customer can be automatically prioritized by the model.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.eswa.2021.115925.

## References

Aggarwal, C. C. (2014). *Data Classification: Algorithms and Applications*. New York, USA: CRC Press.

Babai, M. Z., Dallery, Y., Boubaker, S., & Kalai, R. (2019). A new method to forecast intermittent demand in the presence of inventory obsolescence. *International Journal of Production Economics, 209*, 30–41.

Batuwita, R., & Palade, V. (2010). Efficient resampling methods for training support vector machines with imbalanced datasets. Barcelona: IEEE.

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research, 13*, 1–25.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bohanec, M., Borštnar, M. K., & Robnik-Šikonja, M. (2015). Feature subset selection for b2b sales forecasting. Bled: The 13th International Symposium on Operations Research.

Bohanec, M., Borštnar, M. K., & Robnik-Šikonja, M. (2015). Integration of machine learning insights into organizational learning: a case of B2B sales forecasting. Bled.

Bohanec, M., Borštnar, M. K., & Robnik-Šikonja, M. (2017). Explaining machine learning models in sales predictions. Expert Systems with Applications.

Boylan, J. E., & Syntetos, A. A. (2010). Spare parts management: A review of forecasting research and extensions. *IMA journal of management mathematics, 21*(3), 227–237.

Croston, J. D. (1972). Forecasting and stock control for intermittent demands. *Operational Research Quarterly, 23*(3), 289–303.

D'Haen, Jeroen, & Van den Poel, Dirk (2013). Model-supported business-to-business prospect prediction based on an iterative customer acquisition framework. *Industrial Marketing Management, 42*(4), 544–551.

Duncan, B. A., & Elkan, C. P. (2015). Probabilistic Modeling of a Sales Funnel to Prioritize Leads. International Conference on Knowledge Discovery and Data Mining.

Gandomi, Amir, & Haider, Murtaza (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management, 35*(2), 137–144.

García, S., Luengo, J., & Herrera, F. (2015). *Data Preprocessing in Data Mining.* Springer International Publishing.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction.* Springer.

Heinze, G., & Dunkler, D. (2016). Five myths about variable selection. *Transplant International, 30*(1), 1–6.

Hirschberg, Julia, & Manning, Christopher D. (2015). Advances in natural language processing. *Science, 349*(6245), 261–266.

Japkowicz, N. (2001). Concept-learning in the presence of between-class and within-class imbalances. *Canadian Conference on AI. Springer.*

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. Intelligent Data Analysis, 31.

Jiang, J. (2012). Information extraction from text. In C. C. Aggarwal, & C. (. Zhai, Mining text data. Springer.

Kuhn, M., & Johnson, K. (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models* ((1 ed.).). CRC Press.

Lambert, M. (2018). Sales Forecasting: Machine Learning Solution to B2B Sales Opportunity Win-Propensity Computation.

Lawrence, Rick, Perlich, Claudia, Rosset, Saharon, Khabibrakhmanov, Ildar, Mahatma, Shilpa, Weiss, Sholom, … Kumar, Shiva (2010). Operations Research Improves Sales Force Productivity at IBM. *Interfaces, 40*(1), 33–46.

Li, Andrew Yu, & Elliot, Nikki (2019). Natural language processing to identify ureteric stones inradiology reports. *Journal of Medical Imaging and Radiation Oncology, 63*(3), 307–310.

Lu, Chi-Jie, & Kao, Ling-Jing (2016). A clustering-based sales forecasting scheme by using extreme learning machine and ensembling linkage methods with applications to com- puter server. *Engineering Applications of Artificial, 55*, 231–238.

Mortensen, S., Christison, M., Li, B., Zhu, A., & Venkatesan, R. (2019). *Predicting and Defining B2B Sales Success with Machine Learning.* Charlottesville: IEEE.

Orange Data Mining. (2020). Retrieved from https://orangedatamining.com.

Pinçe, Ç., Turrini, L., & Meissner, J. (2021). Intermittent Demand Forecasting for Spare Parts: A Critical Review. *Omega*, 1–30.

Prati, R. C., & Batista, G. E. (2004). Class imbalances versus class overlapping: an analysis of a learning system behavior. Mexican International Conference on Artificial Intelligence. Springer.

Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of imbalanced data: A review. International Journal of Pattern Recognition and Artificial Intelligence, 3.

Syntetos, Aris A., & Boylan, John E. (2005). The accuracy of intermittent demand estimates. *International Journal of forecasting, 21*(2), 303–314.

Tan, P.-N., Steinbach, M., & Kumar, V. (2006). *Classification: Basic Concepts, Decision Trees, and Model Evaluation. In Introduction to Data Mining* (pp. 25–44). Pearson Addison-Wesley.

Teunter, R. H., Babai, M. Z., & Syntetos, A. A. (2010). ABC Classification: Service Levels and Inventory Costs . Production and Operations Management, 343-352.

Teunter, R. H., Syntetos, A. A., & Babai, M. Z. (2011). Intermittent demand: Linking forecasting to inventory obsolescence. European Journal of Operational Research, 606-615.

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*.

The Harvard Gazette. (2020, 10 26). Ethical concerns mount as AI takes bigger decision-making role in more industries. Retrieved from The Harvard Gazette: https://news.harvard.edu/gazette/story/2020/10/ethical-concerns-mount-as-ai-takes-bigger-decision-making-role/.

Topan, Engin, Tan, Tarkan, van Houtum, Geert-Jan, & Dekker, Rommert (2018). Using imperfect advance demand information in lost-sales inventory systems with the option of returning inventory. *IISE Transactions, 50*(3), 246–264.

Witten, I. H., Frank, E., & Hall, M. A. (2011). Chapter 4 - Algorithms: The Basic Methods. In I. H. Witten, E. Frank, & M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques (pp. 85-145).

Zheng, A., & Casari, A. (2018). *Feature engineering for machine learning.* O'Reilly Media Inc.