



Expertise
and insight
for the future

Débora Guimarães de Sousa

Using Machine Learning to Predict On-Time Delivery

Metropolia University of Applied Sciences

Master's Degree

Degree Programme in Business Informatics

Master's Thesis

30/11/2022

Author Title	Débora Guimarães de Sousa Using Machine Learning to Predict On-Time Delivery
Number of Pages Date	78 pages + 36 appendices 30 November 2022
Degree	Master of Business Administration
Degree Programme	Business Informatics
Instructor	Antti Hovi, Senior Lecturer
<p>Guide to abstract writing:</p> <p>The objective of this study is to create a ML algorithm to predict the On-Time Delivery (OTD) for Wärtsilä Global Logistics (WGLS). WGLS uses OTD to measure service efficiency and improve customer satisfaction. However, WGLS has no advanced tools to predict OTD.</p> <p>This thesis used an applied research family with mixed research methods. It mainly relied on the qualitative data gathered via internal document analysis, observations, and interviews, but it also used quantitative data for developing the ML algorithm.</p> <p>The proposal was built by examining the literature review and evaluating the current state analysis. The data was extracted from EDW. The activities executed in the pre-processing stage includes data collection, outliers' exclusion, missing data handling, feature encoding, data imbalance analysis, feature scaling and analysis of the correlations. After pre-processing, the data was loaded into seven different ML algorithms baseline: Random Forest, Neural Network, Bagging Classifier, Support Vector Machine, Gradient Boosting Machines, Logistic Regression and K-nearest neighbour. The three best algorithms were tuned by simulating different hyperparameters and feature selection. The best model after tuning was the Random Forest.</p> <p>The final model was validated using ML metrics and stakeholder feedback. The algorithm was loaded into a Power BI file and shared with the case company.</p> <p>It is expected that this tool will reduce the number of late deliveries. Late delivery is a problem for both customers and operations. Late deliveries lead to more claims and costs for the business. Therefore, on-time delivery service is crucial in maintaining high customer retention rates and satisfying customers.</p>	
Keywords	Machine Learning, Data science, OTD, On-Time deliveries, KPI, Data analytics

Contents

1	Introduction	1
1.1	Business Context	1
1.2	Business Challenge, Objective and Outcome	2
1.3	Thesis Outline	2
2	Method and Material	4
2.1	Research Approach	4
2.2	Research Design	6
2.3	Data Collection and Analysis	8
3	Existing Knowledge and Best Practice on Building the ML Models / Algorithms	10
3.1	Definition of Machine Learning	10
3.2	Types of Machine Learning	12
3.2.1	Linear Regression and Logistic Regression	15
3.2.2	Random Forest	17
3.2.3	K Nearest Neighbour	19
3.3	Building a Machine Learning Algorithm	22
3.3.1	Data collection and data pre-processing	24
3.3.2	Model training and testing	31
3.3.3	The Machine Learning Canvas	38
3.4	Conceptual Framework of This Thesis	41
4	Current State Analysis of OTD Prediction in the Case Organization	43
4.1	Overview of the Current State Analysis	43
4.2	Description of OTD Calculation	43
4.3	Data selection for the ML Algorithm	44
4.4	Business requirements for the ML Algorithm	46
5	Building the ML Algorithm for the Case Organization	50
5.1	Overview of the Building Stage	50
5.2	Data Collection and Data Preparation	50
5.2.1	Data collection	51
5.2.2	Data pre-processing	52

5.2.3	Feature Encoding	61
5.2.4	Analysis of correlations	62
5.2.5	Feature Scaling	64
5.2.6	Data imbalance analysis	65
5.3	Model Training and Testing	66
5.4	Random Forest	67
5.5	Neural Network	70
5.6	Bagging Classifier	71
5.7	Model selection	72
6	Algorithm Validation	73
6.1	Overview of the Validation Stage	73
6.2	Developments to the Proposal	74
6.3	Final Proposal	75
7	Conclusion	76
7.1	Executive Summary	76
7.2	Thesis Evaluation / Research Quality Criteria for This Thesis	77
	References	1
	Appendices	
	Appendix 1. SQL Script to generate dataset	
	Appendix 2. Data pre-processing python script	
	Appendix 3. Random Forest – Final Model	
	Appendix 4. Dashboard Snapshot	

List of figures

Figure 1.	The action research spiral	5
Figure 2.	Research design of this study.	7
Figure 3.	Supervised Learning (Auger, 2021).....	12
Figure 4.	Unsupervised Learning. (Auger, 2021).....	13
Figure 5.	Reinforcement learning. (Auger, 2021).....	13
Figure 6.	Simplified convolutional neural network layers (Di, 2018).....	14
Figure 7.	Regression (Verdhan, 2020)	15
Figure 8.	Decision Tree (Géron, 2017).....	18
Figure 9.	Random Forest. (Christopher, 2021).....	19
Figure 10.	KNN Algorithm. (Navlani, 2018).....	21
Figure 11.	K-nearest neighbour algorithm (Natlat, 2016)	22
Figure 12.	General ML research approach. (Kamiri, 2021)	23
Figure 13.	Impute missing values with Mean/Median. Left: Age column before Imputation, Right: Age column after imputation by the mean value. (Kumar, 2020)	25
Figure 14.	One hot encoding. (Tripathi, 2019)	26
Figure 15.	Label encoding. (Tripathi, 2019)	26
Figure 16.	Frequency Encoding (Tripathi, 2019).....	27
Figure 17.	Target Encoding. (Tripathi, 2019)	27
Figure 18.	Normal Distribution (Math is fun, 2022).....	28
Figure 19.	Imbalanced x balanced dataset. (Badr, 2019).....	29
Figure 20.	Undersampling x balanced dataset (Badr, 2019).	30
Figure 21.	Feature Scaling Techniques (Roy, 2020).....	31
Figure 22.	Randomly split the input data into train, valid, and test set (Agrawal, 2021). 32	
Figure 23.	Train Valid Test Dataset after sorting the data (Agrawal, 2021).	33
Figure 24.	Evaluation metrics for classification and regression models (Ladkar, 2020) 34	
Figure 25.	ROC Curve (Ladkar, 2020)	36
Figure 26.	AUC (Ladkar, 2020).....	37
Figure 27.	The Machine Learning Canvas	39
Figure 28.	Framework for creating the ML Algorithm to predict OTD in the case company 41	
Figure 29.	The Machine Learning Canvas for OTD prediction in the case company47	
Figure 30.	Dataset info summary.....	51
Figure 31.	Dataset cardinality	52

Figure 32.	Graph of late delivery frequency per year and month	53
Figure 33.	Graph of late delivery by POD	54
Figure 34.	Deliveries by Ship to Country.....	55
Figure 35.	Top 10 countries by number of deliveries in the sample.	55
Figure 36.	PCM Team	56
Figure 37.	MRP Profile	57
Figure 38.	Shipping Type	57
Figure 39.	Shipping condition	58
Figure 40.	Incoterms 1.....	58
Figure 41.	Consolidation Key.....	59
Figure 42.	Shipping Point	59
Figure 43.	C1 Blocked	60
Figure 44.	Dataframe summary after data pre-processing.....	61
Figure 45.	Correlogram	62
Figure 46.	Feature Scaling – Lead Time.....	65
Figure 47.	Feature Scaling – Lead Time.....	66
Figure 48.	Accuracy of Baseline Classification models	67
Figure 49.	Random Forest - ROC Curve	69
Figure 50.	Neural Network - ROC Curve	70
Figure 51.	Bagging Classifier - ROC Curve	72
Figure 52.	Random Forest – Validation Metrics	73

List of tables

Table 1. Details of qualitative data collections used in this study.....	8
Table 2. Details of quantitative data collections used in this study.....	9
Table 3. Confusion Matrix (Gollapudi,2016).	35
Table 4. Data selection for this study.	44
Table 5. Top 10 correlations.....	63
Table 6. Top 10 correlations with target	64
Table 7. Random Forest Accuracy per feature amount	67
Table 8. Chi-square Test score for top 25 features	68
Table 9. Random Forest Classification report	69
Table 10. Neural Network Classification report	70
Table 11. Neural Network Classification report	71
Table 12. Model Selection Performance	72
Table 13. Expert suggestions for the Initial proposal.....	74

Glossary

KPI	Key Performance Indicator. Measurable value to gauge a company's performance
AI	Artificial Intelligence. It is a branch of computer science that uses mechanisms and machines to simulates human intelligence processes.
ML	Machine Learning. It is an area of artificial intelligence that provides algorithms that can learn learns to recognize patterns automatically.
OTD	On time delivery. It is the metric used to estimate the delivery performance.
KNN	K nearest neighbour. It is a type of ML algorithm that uses basic intuition technique to find the nearest neighbour of a data point.
RF	Random Forest. It is a type of ML algorithm that uses several decision trees to make predictions.
NN	Neural Network. It is a type of ML algorithm that operates similarly to the human brain. It uses nodes or neurons in a layered structure.
BAG	Bagging Classifier. It is a type of ML algorithm that uses voting schema to reduce the prediction's variance.
LR	Logistic Regression. is a type of ML algorithm that uses concepts of statistics and probability in classifying data.
SVM	Support Vector Machine. It is a type of ML algorithm that seeks to maximize the distance between the closest points to classify the data
GBM	Gradient Boosting Machines. It is a type of ML algorithm which predicts the data based on a set of weak prediction models.

1 Introduction

According to MIT Computer Science Professor Aleksander Madry, "Machine learning is changing or will change, every industry, and leaders need to understand the basic principles, the potential, and the limitations" (Brown, 2021).

Machine Learning (ML) is a subcategory of AI. It analyses a large amount of data using specific statistical methods and algorithms to find patterns in the database. Machine learning is the machine's ability to imitate human intelligence. It accomplishes tasks similar to how humans solve problems. (Brown, 2021.) Companies are using ML to improve efficiency. The use of ML helps to break the bottlenecks of complex digital processes. It can, for example, bring valuable insights more quickly and make data more meaningful. (Wilson et al. 2016.)

Another example is the recommendation algorithms behind Netflix and social media. The ML algorithms are helping companies to understand customer preferences. Many companies deploy online chatbots, where customers do not talk to humans but interact with a machine. These algorithms use ML to get appropriate responses. Fraud detection is another case when companies use ML. Algorithms can identify consumption patterns and potentially fraudulent credit card transactions, login attempts or spam emails. There are several other applications of ML in business, such as automatic helplines or chatbots, and autonomous cars, among others. (Brown, 2021.)

This study focuses on developing a Machine Learning (ML) algorithm for Wärtsilä Global Logistics (WGLS) to predict the probability of a delivery being on time or late. The goal is to map all potential late deliveries and act before the delay happens.

1.1 Business Context

The case company of this thesis is Wärtsilä, "a Finnish corporation which produces technologies and solutions for the Marine and Energy markets. Wärtsilä employs about 19 000 people and operates globally in over 200 locations in more than 80 countries." The corporation consists of two businesses: Marine and Energy. (Wärtsilä, 2019.)

Wärtsilä Energy business provides solutions in engines, solar, storage technologies and software. In the first half of 2019 Energy business represented 35% of total Wärtsilä Net

Sales (Wärtsilä, 2019). The *Marine business* covers the following segments: gas carriers, traditional merchant, navy, cruise & ferry, and unique vessels. In the first half of 2019, the Marine business represented 65% of total Wärtsilä Net Sales. (Wärtsilä, 2019.)

This thesis is done to benefit the case organization of the student, which is *Wärtsilä Global Logistics* (WGLS). The unit employs directly about 500 people and indirectly 200 people in 23 countries. It operates the entire logistics chain of Wärtsilä Spare Parts, from order intake to customer delivery. WGLS operates in more than thirty countries with the central spare part distribution center in Kampen, NL, and satellite warehouses in Singapore; Vaasa, FIN; Ft. Lauderdale, Houston; New Orleans, US; Vancouver, CA; Quito, EC; and Niteroi, BRA.

1.2 Business Challenge, Objective and Outcome

WGLS uses *On-Time Delivery* (OTD) to measure service efficiency and improve customer satisfaction. According to Dalin-Kaptzan (2022), OTD is the Key-Performance Indicator (KPI) that shows whether an organization meets its goals regarding promised delivery times. The late delivery is a problem for both customers and operations. Lower OTD rates lead to more claims and costs for the business. Customers don't care about the reason that might cause order delays. They want the orders to arrive on time. Therefore, on-time delivery service is a crucial factor in maintaining high customer retention rates and keeping customers satisfied. (Dalin-Kaptzan, 2022)

However, WGLS has no advanced tools to predict OTD. Therefore, the company needs a tool to help the managers make decisions. As ML has become more powerful and accessible, therefore, this study aims to use this approach *to create an automated algorithm to predict the OTD*. The outcome of the thesis is the Machine Learning algorithm.

1.3 Thesis Outline

The scope of this study focuses on improving the current tools for measuring OTD in WGLS. Once Machine Learning can analyse an enormous amount of data, all countries and products are included in the study. However, the thesis concentrates on GLS Plants. GLS Plants comprise WGLS CDC Kampen, WGLS Nuclear, WGLS SPC Americas and WGLS SPC Asia. According to SAP (2022), "The delivering plant refers to the plant from

which the goods are delivered to the customer, within a specific sales organization and distribution channel.”

The thesis is organized as follows. Firstly, Section 1 is the introduction giving the background and objective of the thesis. Section 2 describes the methods and material used in this study. Section 3 is the literature review. Its goal is to understand how, and which ML models can be used in helping to reach the thesis objectively. Section 4 contains the analysis of the current situation, describing all the relevant tools and databases used to predict OTD in Wärtsilä Global Logistics presently. The next step, Section 5, contains the development of ML algorithms. Then, Section 6 is the report on the validation results for the proposed algorithm and provides a proposal for the case company. Last, Section 7 concludes the thesis.

2 Method and Material

This section describes the research approach, research design, data collection and analysis methods used in this Thesis.

2.1 Research Approach

In terms of the *research family*, there is basic and applied research. *Basic research* increases business and management understanding processes, resulting in universal principles linking the process to its relationship and outcomes. In comparison, *applied research* aims to improve understanding of a specific business or management problem, leading to a solution to the problem. (Saunders et al. 2007).

According to Adams (2014), qualitative and quantitative research methods are the two most common types of research methods used in studies. *Quantitative methodology* is based on data collection and analysis, in which various mathematical and statistical analysis methods are applied. On the other hand, *qualitative research* relies on non-quantifiable data and focuses on exploring such phenomena as social relations and descriptions of reality as experienced by the respondents. There is also mixed research possible that utilises both types of methods. Furthermore, the data collection can be primary data (field study) and secondary data (desk study). *Primary data* are collected to solve the research problem, while *secondary data* are collected for purposes other than the research problem. (Saunders et al. 2007).

Regarding the research strategy, the most common ones in the field of business include case studies (for qualitative research), survey and experimental research (for quantitative research) and action research.

The *case study* is most often used in exploratory and explanatory research as it investigates a particular contemporary phenomenon within its real-life context using multiple sources of evidence. (Saunders et al. 2007, p.139).

Survey research is widely used in social science, healthcare, and education studies. It relies on asking people standardised questions and analysing them using statistical techniques. On the other hand, experimental research relies on hypothesis testing. (Leavy, 2017.)

Action research aims to solve specific organisational problems by doing a cyclical process. Figure 1 shows the action research spiral, starting with the research objective (context and purpose). The next step is to analyse the problem (Diagnosing), which will support the creation of a plan and help in the decision process. Finally, these actions are evaluated and considered in the subsequent cycles. (Saunders et al. 2007, p.141)..

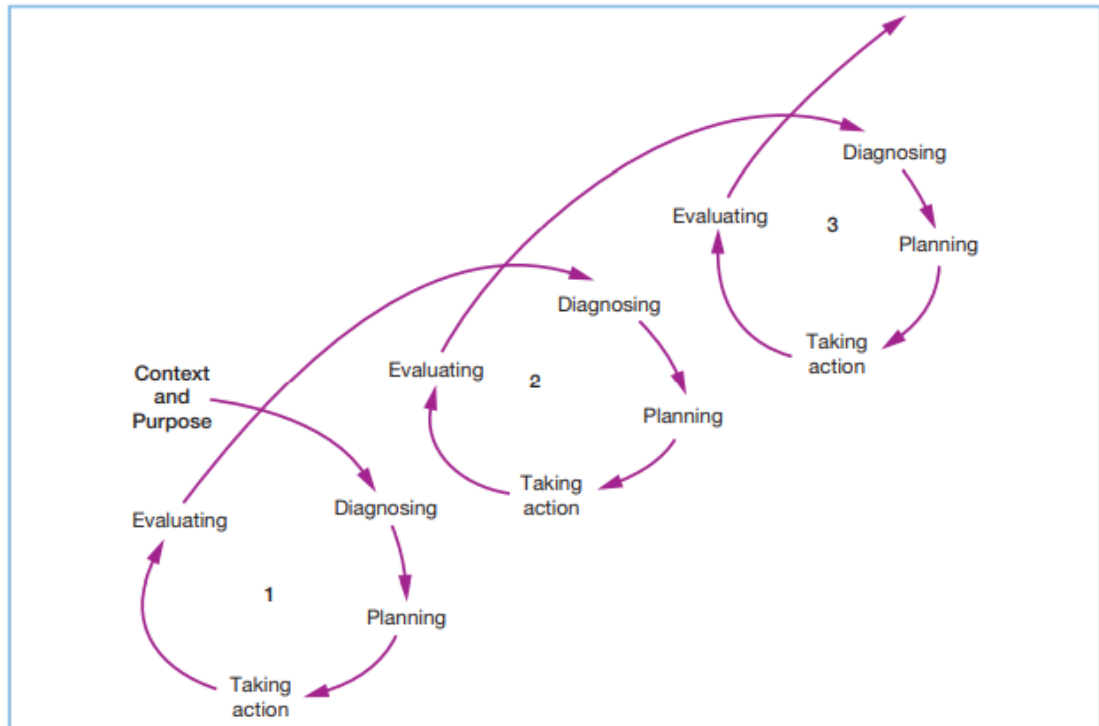


Figure 1. The action research spiral

Additionally, there are other possible approaches. One of the recent ones is applied action research, also called design research. Applied action research combines various research methodologies. It can be used in different situations, such as improving products, services or processes. The research usually has only one interaction, focusing on the practical result. It is “conducted according to the research conventions: data is carefully collected, documented, and analyzed using research methods, which means the methods that produce reliable and novel (for this context) results”. (Kananen 2013, p. 13-22)

This thesis belongs to the applied research family. It is a field study since it collects qualitative and quantitative data from the organization. This thesis uses mixed research methods. It mainly relies on the qualitative data gathered via internal document analysis, observations, and interviews; it also uses quantitative data for developing the ML model/algorithm. As for the research strategy, this thesis relies on using the Applied action research (Design research) that was found most suitable for developing a practical proposal for the case company.

2.2 Research Design

As seen in Figure 2, the study starts with setting up the objective and continues to be exploring literature and best practice. The literature review aims to analyse the various machine learning models for identifying, selecting, and applying the relevant ones to suit the company's needs in the subsequent stages, or for using the available knowledge and best practice for guiding in the development of own solution. The outcome is the conceptual framework for developing the ML models/algorithms.

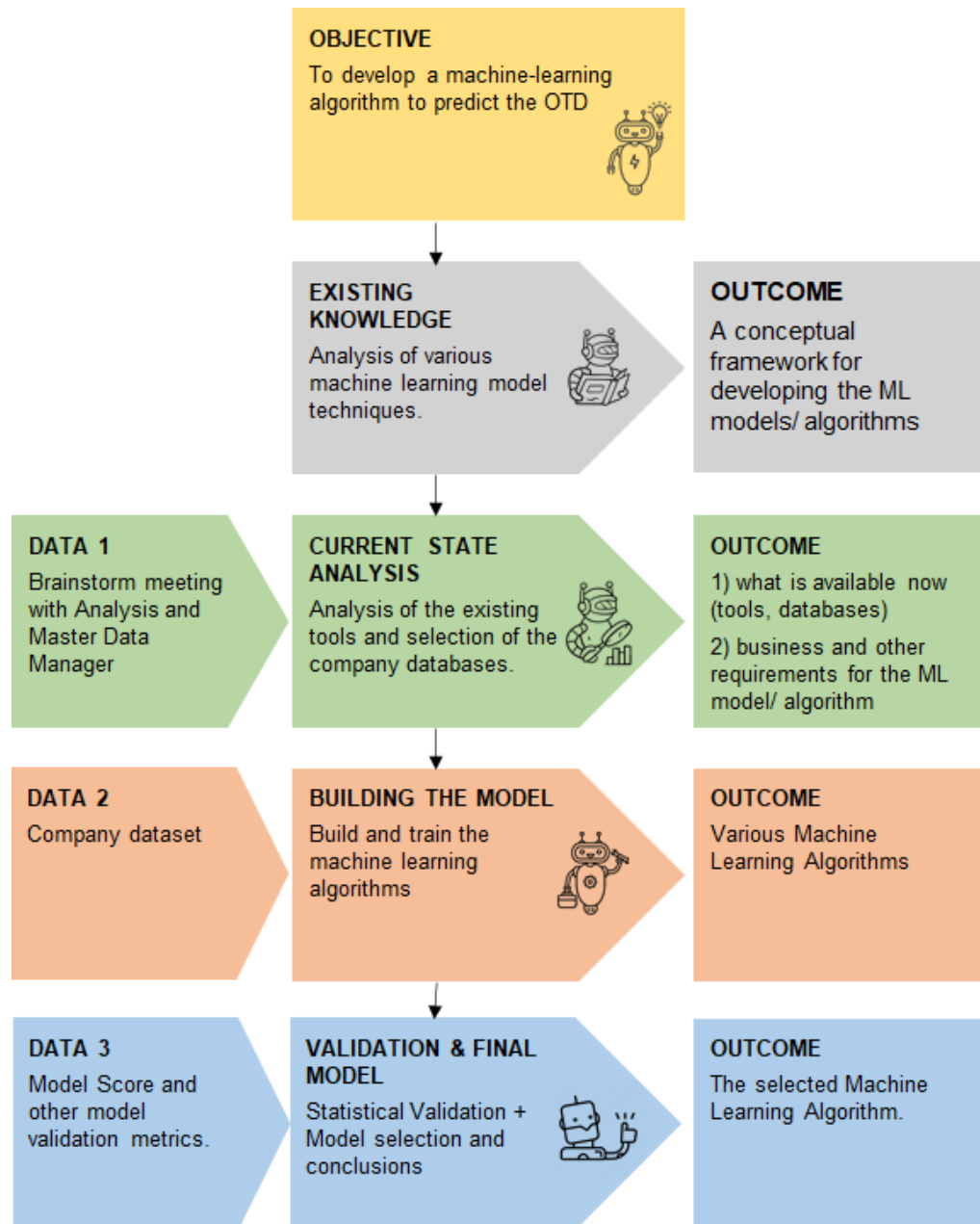


Figure 2. Research design of this study.

As soon as the relevant knowledge and best practice for guiding the next stages in the thesis are identified, the current state analysis is conducted. The objective is to understand the current situation and identify all the existing tools and datasets helpful in developing the ML algorithm and compare the company's current tools and practices against the knowledge acquired from literature and best practice. The data for the analysis is collected from the enterprise data warehouse (EDW). The outcome is (a) the understanding of available tools and databases at present, and (b) business and other requirements for the ML model/ algorithm.

The next stage is the proposal building, where the goal is to define, build and validate the machine learning algorithms. The outcome is a range of machine learning models.

Next, as a result of validation, the study finishes the model selection. The last stage focuses on analysing all the models and selecting the best ML model/algorithm based on statistical validation.

2.3 Data Collection and Analysis

This study relied on gathering and analysing qualitative and quantitative data collection. As seen in Table 1, the source of qualitative data is interviews and brainstorming sessions with the key stakeholder.

Table 1. Details of qualitative data collections used in this study.

	Participants / role	Data type	Topic, description	Date, length	Documented as
Data 1, for the Current state analysis (Section 4)					
1	Respondent 1 - Analyses and Master data Manager (Former Manager)	Brainstorming session	List of all factors that could affect the OTD based on the respondent experience.	September 2021, 30 minutes	Microsoft Team notes
2	Respondent 1 - Analyses and Master data Manager (Former Manager)	Discussion	Team discussion to develop understand the business requirements	July 2022, 60 minutes	Machine Learning Canvas
	Respondent 2 - Analyses and Master data Manager				
	Respondent 3 - The General Manager Operations Support				
	Respondent 4 - Data scientist				

The first data source was a brainstorming session with the Analyses and Master Data Manager. The goal of the brainstorming was to understand what factors could affect the

delivery to be on time or late. The outcome of the brainstorming session was a list of all variables to be loaded into the ML algorithm. The outcome was documented in Microsoft Teams notes.

The second data source was a meeting conducted with three stakeholders. The objective this time was to understand all requirements to predict the OTD in the case company. The outcome of this session was documented as a Machine Learning Canvas.

In addition, table 2 shows that the company's Datawarehouse was used as input for the machine learning model.

Table 2. Details of quantitative data collections used in this study.

	Data type	Topic, description	Schedule	Documented as
1	Data collection from different sources	Datawarehouse and dataflow mapping to get the list of all relevant fact tables.	April 2022	Oracle SQL Script

All variables listed in the brainstorming meeting were collected and analysed. The dataset consists of sales and delivery data. The complete description of the dataset is available in chapter 4. SQL Script was used to document this stage. SQL is the programming language used to extract data from the data warehouse.

The model validation was done using statistical metrics described in session 3.4.

3 Existing Knowledge and Best Practice on Building the ML Models / Algorithms

This section gives an overview of available knowledge and best practice on building the ML algorithms. It will attempt to answer the following questions: What is machine learning? Which and how many machine learning techniques were considered suitable for the given task? How to build a machine learning algorithm?

3.1 Definition of Machine Learning

It is important to look back at its history to understand machine learning better. In part, Machine Learning is based on a model of brain cell interaction. This model was introduced in 1949 by Donald Hebb in a book titled "The Organization of Behavior". The book presents Hebbian theory about the adaptation of brain neurons during the learning process. According to Hebb, the relationship between two neurons weakens if activated separately and strengthens if activated simultaneously. (Foote, 2019.)

The term "Machine Learning" first came up by Arthur Samuel. Samuel developed a computer program for playing checkers and designed several mechanisms to improve his program. Samuel defined machine learning as "a field of study that gives computers the ability to learn without being explicitly programmed". (Samuel, 1959.)

Based on Hebb's model combined with Samuel's Machine Learning efforts, Frank Rosenblatt created 1957 the perception, the first successful neurocomputer. Even if it looked promising, it failed to recognise many types of visual patterns, causing frustration, and stalling neural network research. Later, it was discovered that using multilayers in the perceptron offered significantly more processing power than a perceptron using one layer. As a result, various neural networks were created and continue to expand. One example is the backpropagation, developed in the 1970s, nowadays used to train deep neural networks. (Foote, 2019.)

Machine Learning continued to be developing over the years. In 1967, the nearest neighbour algorithm was conceived. This algorithm was used for mapping routes and was one of the first algorithms used to solve the travelling salesman problem approximately. Using it, a salesperson enters a selected city and repeatedly, the program determines the nearest towns until all have been visited. (Foote, 2019.)

Boosting, another Machine Learning concept was introduced by Robert Schapire in 1990. In machine learning, boosting is a supervised learning algorithm used to reduce bias that converts weak learners into strong ones. It attempts to build a robust model based on weak models in series. Weak learning is a classifier that is slightly better than random guessing. In 1997, Jürgen Schmidhuber and Sepp Hochreiter developed a new machine learning model focused on Speech Recognition called Long-Short-Term Memory. (Foote, 2019.)

In 1997 Tom Michell proposed a new definition for Machine Learning. "Machine Learning is the study of computer algorithms that improve automatically through experience. Applications range from data-mining programs that discover general rules in large data sets to information filtering systems that automatically learn users' interests." (Michell, 1997.)

Another ML definition is given by Verdhan (2020): "In Machine Learning, we study statistical/mathematical algorithms to learn the patterns from the data which are then used to make predictions for the future."

Based on the mentioned ML definitions, it is possible to conclude that ML consists of analysing a large amount of data using specific statistical methods and various algorithms to find patterns in the database. The goal is to discover functional relationships between different data. Machine learning can make inferences about behaviour and new cases.

Nowadays, ML is responsible for some of the most significant advancements in technology. Some examples of the use of ML to improve business are Algorithms to understand customer preferences, fraud detection, chatbots, development of products like autonomous cars, among others. (Brown, 2021.)

The next session will explore the most common types of Machine Learning. The objective is to understand what machine learning type fits best for this study.

3.2 Types of Machine Learning

The most common Machine Learning types are supervised learning, unsupervised learning, reinforcement learning and deep learning. A short explanation of each type is given as follows.

Supervised learning models are ML models trained on examples. The input is added to the model without the additional desired output and is requested to predict the output. The model must be adjusted if the output differs from the desired one. The process continues until an accurate output is generated. (Skilton, 2018.)

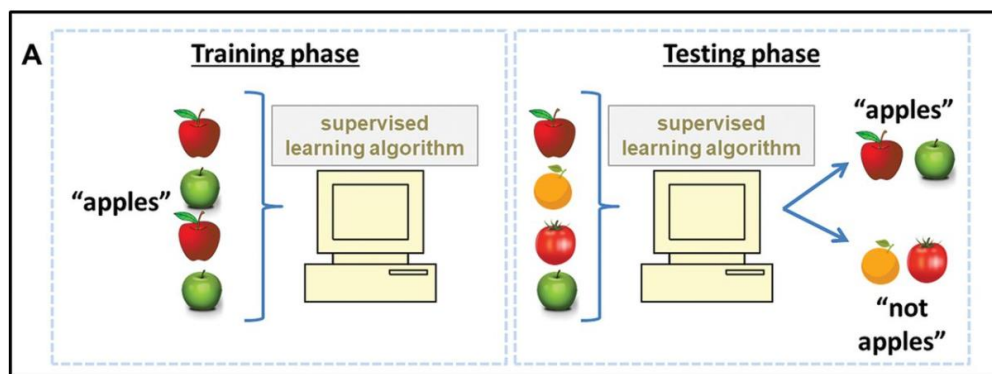


Figure 3. Supervised Learning (Auger, 2021).

As seen in Figure 3, first, the algorithm processes labelled images of apples. Then it learns to predict the category of unseen image data. Next is the test, which provides pictures of different fruits, and the model predicts if the output is an apple or not an apple.

According to Skilton (2018), *unsupervised learning* is a machine learning model used to train data required to recognise similarities between inputs. Figure 3 shows an example of unsupervised learning, where the algorithm processes unlabelled fruits images.

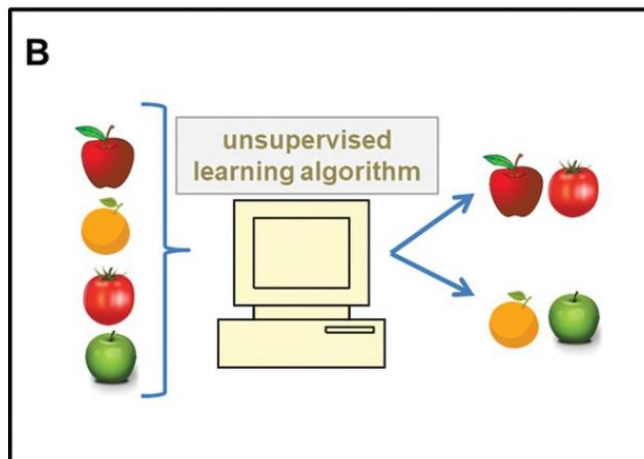


Figure 4. Unsupervised Learning. (Auger, 2021)

Figure 4 shows an example of unsupervised learning. The database is loaded without instructions on what is apple, orange, or tomato. The algorithm groups the data according to its similarities. The output is the group of fruits that look more similar.

Reinforcement learning is a technique that produces an action based on inputs or observations from the environment in which the algorithm operates.

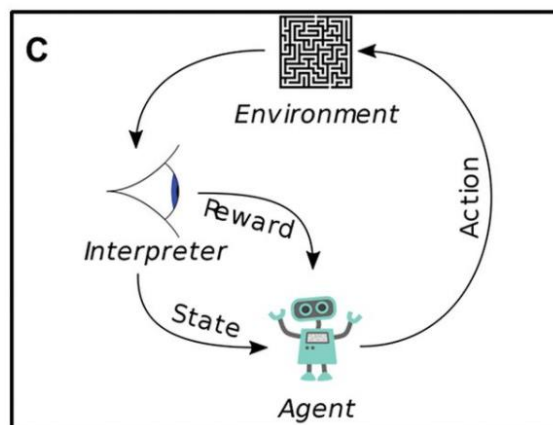


Figure 5. Reinforcement learning. (Auger, 2021)

As seen in Figure 5, The data is processed and looks at the consequences of that action and then "performs" or punishes the pair of actions. It makes the same action more or less likely for future scenarios where a similar condition exists. Multiple cycles of the same process maximize the total rewards. (Auger, 2021.)

Deep learning tries to imitate the layered activity of neurons in the neocortex. It learns hierarchical structures and levels of representation and abstraction to understand data patterns from various sources, such as text, sound, videos, and images. It usually consists of multiple layers to build an enhanced resource space. One of the best benefits is its ability to learn complex functions without too many dependencies on human-made resources. (Di, 2018.)

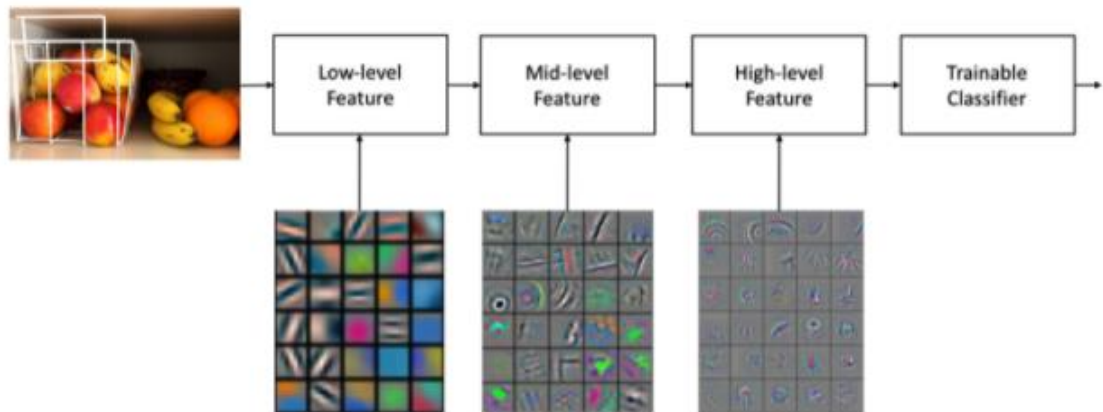


Figure 6. Simplified convolutional neural network layers (Di, 2018)

Figure 6 shows a simplified convolutional neural network (CNN). The first layer learns functions such as colours and frames. The second layer learns functions like corners. Third layer learns about textures. The layers are loaded in the algorithm that will classify the image.

After analysing the most common ML types, the one that fits best for this study is supervised learning. In supervised learning, the algorithm's goal is to learn a general rule that maps inputs to outputs correctly. The input in this case study is the historical deliveries database. The output is the probability of delivery being late.

There are many different types of ML algorithms. The recommendation is to create and compare the accuracy of at least four algorithms, and then select the algorithm with the highest score. (Verdhan, 2020.)

This session will explore all the different types of ML algorithms used on this study.

3.2.1 Linear Regression and Logistic Regression

The purpose of *linear regression* is to obtain a mathematical equation that measures the relationships between the data. This equation can be used to predict unseen data. The equation for linear regression is given as follows. (Verdhan, 2020.)

$$Y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \varepsilon_i$$

where

Y_i = The target variable which the model wants to predict

x_1 = The predicting variables used to predict the value of Y

β_0 = The value of Y when the value of x_1 is zero (Population Y intercept)

β_1 = The expected change in the value of Y by a unit change in x_1 (Population slope coefficient)

ε = The random error term in the model

In linear regression, there can be multiple lines to represent the relationship. ML is used to find "the equation which gives the minimum loss for the data at hand". (Verdhan, 2020.)

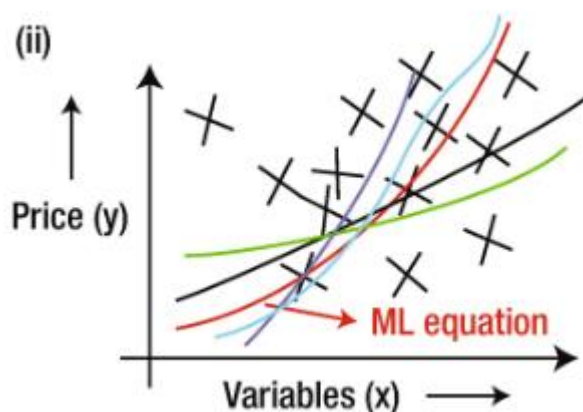


Figure 7. Regression (Verdhan, 2020)

Figure 7 shows an example of linear regression for price prediction. The many different linear equations are represented by the colours blue, red, green, and black. The ML algorithm had selected the equation that best fits its proposal, resulting in the minimum loss for the data.

There are three linear regression types: simple regression, polynomial regression, and multiple regression. Simple regression occurs when only one independent variable is used to predict the dependent variable, and the model fits in a linear relationship. Polynomial regression happens when the relationship is non-linear. Multiple regression occurs when more than one independent variable is used for predicting the dependent variable. (Gollapudi, 2016.)

It is necessary to analyse the correlations for multiple regression models before adjusting the model. The reason is that adding more independent variables creates “more relationships, and it can cause dependency between the independent variables themselves”. It is called *multicollinearity*. The tool to identify multicollinearity is the correlation matrix. The correlation matrix measures the correlation coefficient between all variables. “The closer the value to 1 or -1 , the higher is the correlation.” (Gollapudi, 2016)

Linear regression is used to predict continuous data. For example, linear regression could be used in this case study to estimate the most probably delivery date. This prediction can be made using linear models because the output is continuous data (The lead time in days from the order creation date and the proof of delivery date).

To calculate the probability of a delivery being late or not, linear regression is not the recommended option. In this case the dependent variable is a binary data (1 = late, 0 = on-time)

Logistic regression is an option to predict binary data. “Logistic regression is an extension of linear regression where the dependent variable is a categorical variable that is responsible for the classification of the observations.” (Gollapudi, 2016)

There are two steps in the logistic regression. The first is to find the probability of the observation belonging to a specific class. In this study, the first step is to calculate the probability of the delivery being late $P(Y=1)$. The second step is to select the cut-off value

to assign the prediction to one of the classes. For example, suppose the cut-off is 0,5. In that case, all deliveries with a probability of being late above 0,5 will be classified as late. Consequently, the other ones will be classified as on time. (Gollapudi, 2016)

According to Brannick (2022), the logistic regression equation is:

$$P_i = \frac{1}{1 + e^{-(a+bx_i)}}$$

where,

Y_i = The target variable which the model wants to predict

x_1 = The predicting variables used to predict the value of Y

a and b = The model parameters

e = The base of the natural logarithm (approximately equal to 2.718281828459)

Another ML model used in this thesis is the Random Forest.

3.2.2 Random Forest

Random Forest is a supervised ML where a large number of decision trees operate together to generate predictions. (Yiu, 2019).

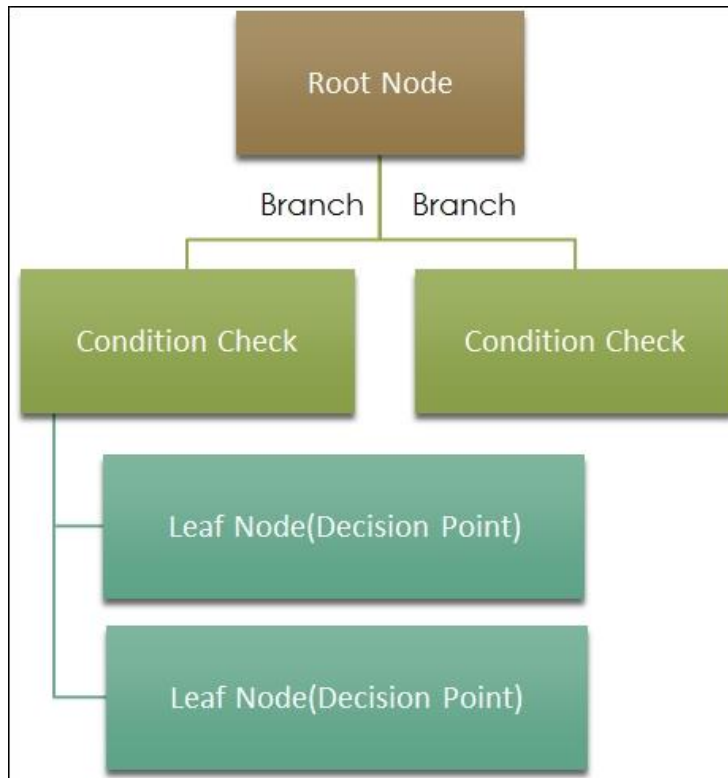


Figure 8. Decision Tree (Géron, 2017).

Figure 8 shows an example of decision tree. To make a decision, the flow starts at the root nodes, then navigates to the arcs where a condition is met, and if satisfied, the flow follows a branch until reaching a leaf node, each leaf represents the value of the target attribute.

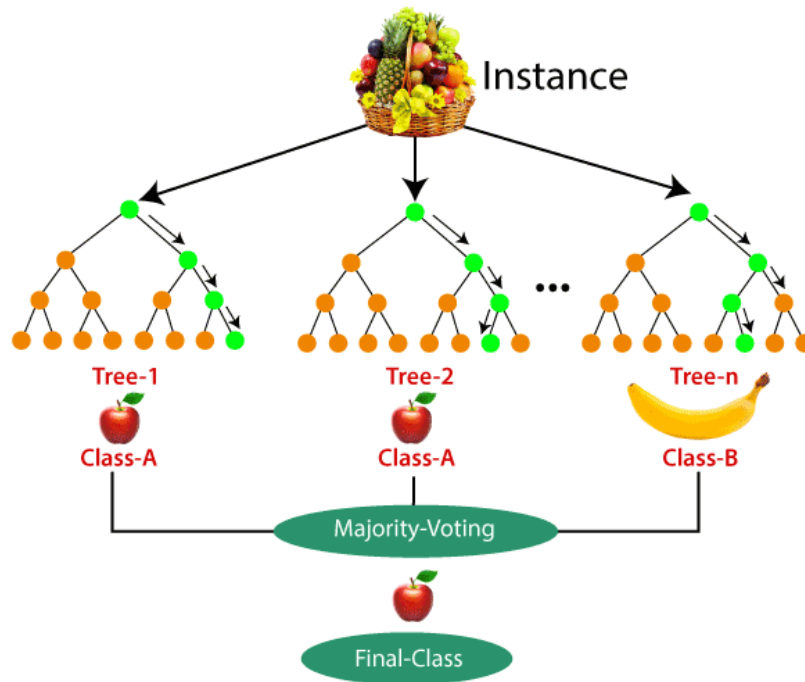


Figure 9. Random Forest. (Christopher, 2021).

Figure 9 shows an example of a random forest. During the training phase, each decision tree predicts a result. It is possible that some decision trees may predict the correct output, while others may not. Based on most results, *the Random Forest* classifier predicts the final decision.

The decision trees models are easy to understand and require little experimentation. Furthermore, decision trees do not require complex data preparation and can handle both categorical and numerical data. Another advantage is that this model can operate large datasets and handle highly dimensional data. (Géron, 2017.)

Another ML model used in this study is the K Nearest Neighbour.

3.2.3 K Nearest Neighbour

The *K nearest neighbour (KNN)* uses basic intuition technique to find the nearest neighbour of a data point. The algorithm uses measures such as Euclidean distance and Hamming distance to find the closest neighbour. The Euclidean distance treats all

the dimensions equally. The disadvantage of this measure is its sensitivity to the extreme values within a single attribute. (Géron, 2017.) The Euclidean distance formula is:

$$D(x, x^i) = \sqrt{\sum |x_d - x_d^i|^2}$$

where,

- (x, x^i) are the coordinates of one point.
- d is the distance

Hamming distance measures if two attributes are equal or not, therefore it is the default option when dealing with categorical data. When they are equal, the distance is 0, otherwise, it is 1. (Géron, 2017.) The formula is:

$$D(x, x^i) = \sum_d 1_{x_d \neq x_d^i}$$

Where,

- (x, x^i) are the coordinates of one point
- d is the distance

In KNN, K is the number of neighbours that the model will analyse to make the decision. Ideally, K should be an odd number to avoid a tie. Once the value of K is defined, the model will choose the most frequent value among the k nearest neighbours. It is illustrated in Figure 10 below.

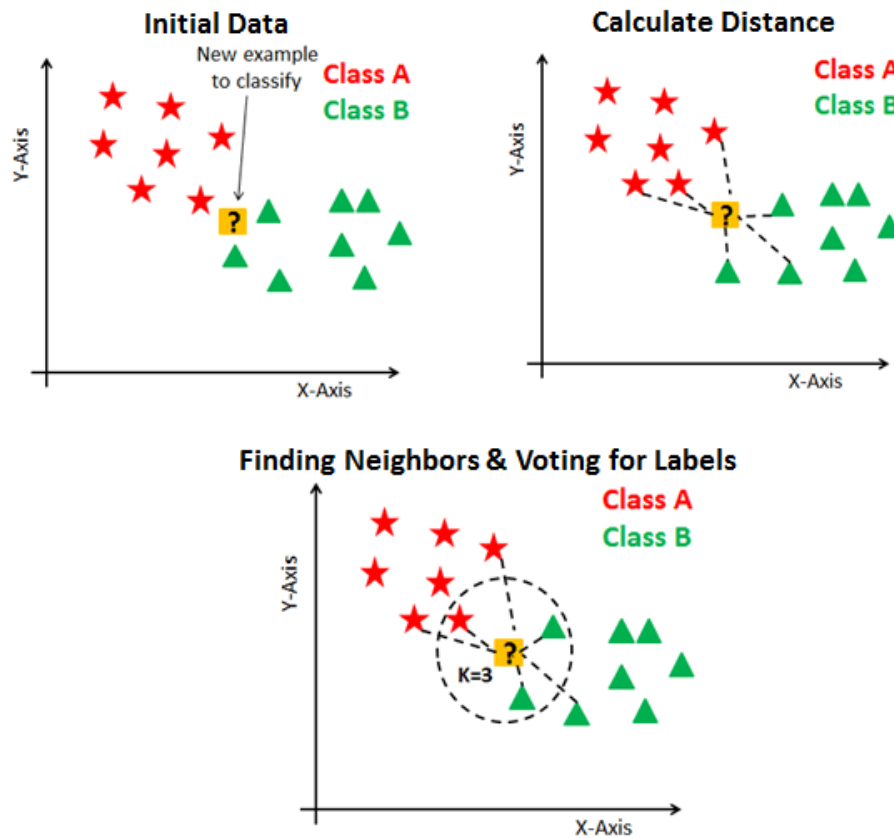


Figure 10. KNN Algorithm. (Navlani, 2018)

Figure 10 shows an example of KNN, in this example a new data is added in the dataset represented by the question mark and the distance between the other data are calculated so it can find the K-nearest neighbours. And then, the algorithm will start the "voting system". In this case most of the classes are triangle therefore the new data point (Question mark) will be assigned to triangle.

Figure 11 shows another example of KNN.

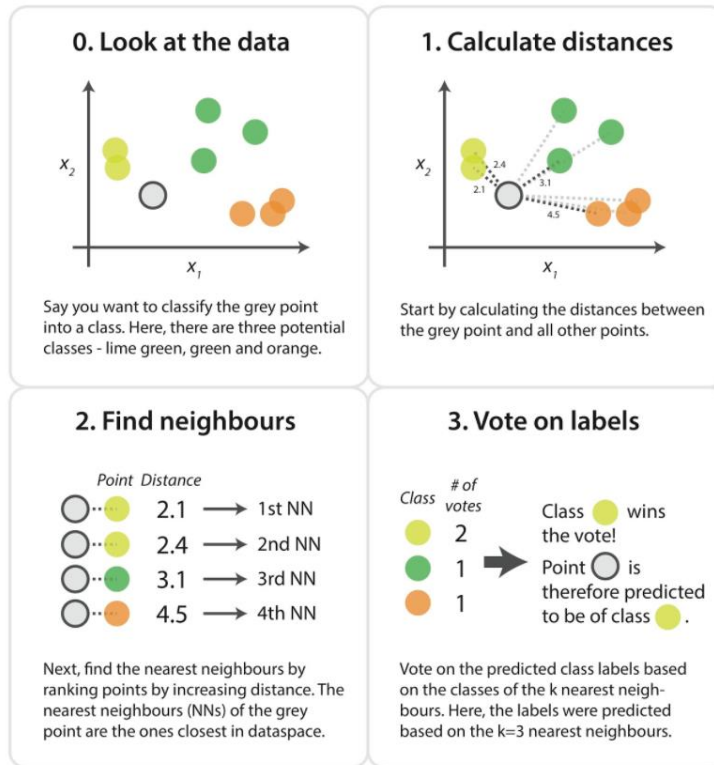


Figure 11. K-nearest neighbour algorithm (Natlat, 2016)

Figure 11 shows another example of a k-nearest neighbour algorithm. First, the algorithm will calculate the distance between the data it wants to predict (the grey point) and the historical data of all categories (The yellow, the green and the orange points). The third step is to identify the nearest neighbour by sorting the distances (The yellow is the nearest because it has a smaller distance). The last step is to vote on the label with the most frequent class. If the $K=3$, the algorithm will select the three closest neighbours, and the most frequent class among them is the predicted category. In this case, the predicted value is yellow because two of the three nearest neighbours are yellow.

The next chapter will discuss the Machine learning pipeline.

3.3 Building a Machine Learning Algorithm

In computer science an algorithm is a tool for solving computational problems. It consists of procedure that takes a set of values as input and produce another set of values as output. In other words, algorithm is “A sequence of steps that transform the input into the output”. (Cormen, 2003, p. 5)

Another definition for algorithm is “An Algorithm is a set of rules that a machine follows to achieve a particular goal. An algorithm can be considered as a recipe that defines the inputs, the output and all the steps needed to get from the inputs to the output. Cooking recipes are algorithms where the ingredients are the inputs, the cooked food is the output, and the preparation and cooking steps are the algorithm instructions.” (Molnar 2022, p.15)

The Machine Learning algorithm will result in a Machine Learning model. “A Machine Learning Model is the learned program that maps inputs to predictions.” (Molnar 2022, p.15)

A study conducted in 2021 by Murang’a University of Technology, analyzed 100 research articles in the field of ML and concluded that the most common approach for conducting experiments in ML studies. is as follows: Data collection, Data pre-processing, Model training, model testing and model evaluation. The general method is shown in Figure 12. (Kamiri, 2021)

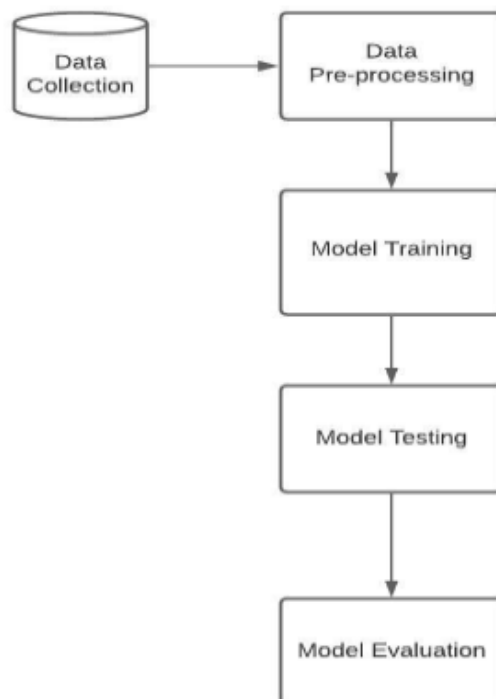


Figure 12. General ML research approach. (Kamiri, 2021)

Figure 12 shows the general research approach used in Machine Learning. Each step will be explained in the next sessions.

3.3.1 Data collection and data pre-processing

The first step is *data collection*. Machine learning requires many training data, and the quality and quantity of data will be the key to how good the model might be. (Rijmenam, 2019). A separate team in large companies usually does the data preparation. Accessing a large volume of data and knowing how can combine it with other data sources will help bring new insights. (Dean, 2014)

The next step is *to perform exploratory data analysis* (data pre-processing). It is essential to understand the data and the relationships between the variables. In addition to the knowledge of data miners, it is also necessary to work together with domain experts. Domain experts will answer whether the findings from the data have been known for some time. (Dean, 2014.) Some examples of exploratory analysis include understanding what kind of data is available, removing outliers, encoding categorical values, and defining the features and target variables. (Shin, 2020)

This thesis will apply the data pre-processing activities because since ML learns from data it directly affects the results of the model. A good data pre-processing can result in a more accurate prediction for the OTD estimation and a better computer performance since it cleans all unnecessary data. Therefore, the next sessions will explore the most frequent activities performed in data pre-processing.

3.3.1.1 Handling Missing data

In *the data pre-processing step*, resource imputation techniques are used to fill in missing values. This is important because most machine learning models do not work when there is missing data in the dataset. (Shin, 2020)

A more straightforward way *to handle missing data* is to delete the rows or columns with null values. Columns with more than 50% null rows can be completely discarded, and rows containing one or more null values can also be discarded. Removing the missing data will result in a more robust model, but as much information is lost, this technique is not recommended if the percentage of missing values is excessive compared to the entire dataset. (Kumar, 2020)

One way to avoid data loss is to assign missing values with the Mean, Median or Mode. In this method, columns with missing numeric values can be replaced by the average remaining values in the column. Figure 13 shows how it can be done.

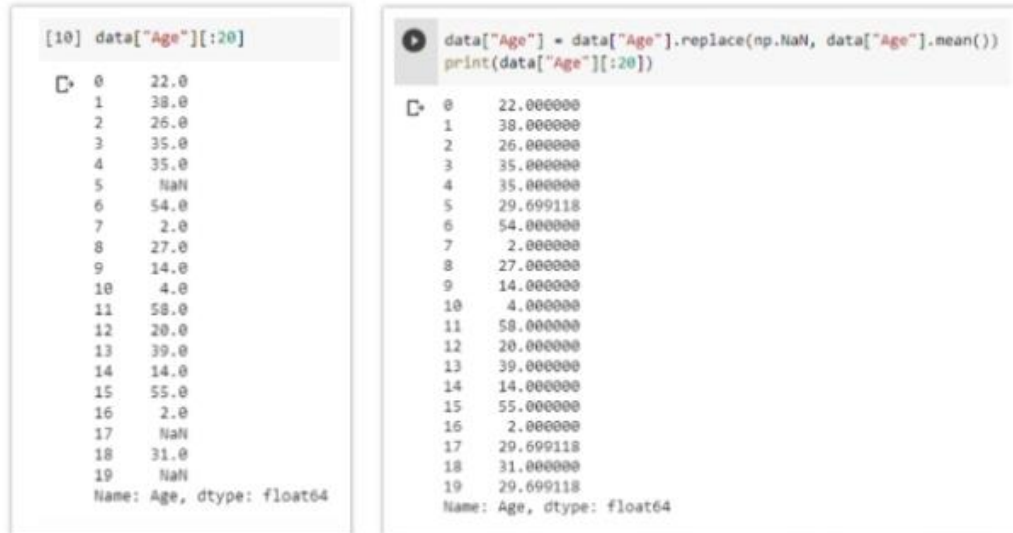


Figure 13. Impute missing values with Mean/Median. Left: Age column before Imputation, Right: Age column after imputation by the mean value. (Kumar, 2020)

Figure 13 shows an example where the mean was used to replace the missing values. Likewise, the median or the mode can be used to replace the values. This technique works well with small datasets and is easy to implement. However, it only works with continuous numeric variables and can cause data leakage. For categorical values, similar techniques can be adopted in which the most frequent ones replace missing values. If the number of missing values is too large, a new category is created. (Kumar, 2020.)

Next session will explore the different techniques for feature encoding.

3.3.1.2 Feature Encoding

Feature encoding is the process of turning categorical variables into numbers (Shin, 2020). Categorical variables are typically stored as text values, and as ML is based on mathematical equations, we need to encode categorical values into numeric to apply ML.

The *One-Hot-Encoding* is a method that creates a binary array for each category. It is shown in Figure 12 blow.

User	City
1	Roma
2	Madrid
1	Madrid
3	Istanbul
2	Istanbul
1	Istanbul
1	Roma

→

User	Istanbul	Madrid
1	0	0
2	0	1
1	0	1
3	1	0
2	1	0
1	1	0
1	0	0

Figure 14. One hot encoding. (Tripathi, 2019)

As shown in Figure 12, the number of vectors depends on the number of classes. The disadvantage of this method is that if the data has many categories, this method will produce many columns and consequently significantly affect the model performance. (Tripathi, 2019.)

Another way to encode data with high cardinality is *label encoding*.

CAT73	CAT73 label_encoded
A	1
A	1
C	3
B	2
A	1
C	3
B	2

Figure 15. Label encoding. (Tripathi, 2019)

As shown in Figure 13, in the label encoding technique each category receives a value from 1 to N, where N is the total number of classes in the column. This technique can be used for ordinal data, where the order of the category matters. (Tripathi, 2019.)

Another way to encode data is through *frequency encoding*.

A	0.44 (4 out of 9)
B	0.33 (3 out of 9)
C	0.22 (2 out of 9)

Feature	Encoded Feature
A	0.44
A	0.44
A	0.44
A	0.44
B	0.33
B	0.33
B	0.33
C	0.22
C	0.22

Figure 16. Frequency Encoding (Tripathi, 2019).

As shown in Figure 14, in Frequency encoding a value from 0 to 1 is assigned to each category according to the relative data frequency. For example, the A feature is available in 44% of the dataset therefore the encoding will be 0,44 for the A category.

Another popular encoding approach is *the target encoding*. It is shown in Figure 17 below.

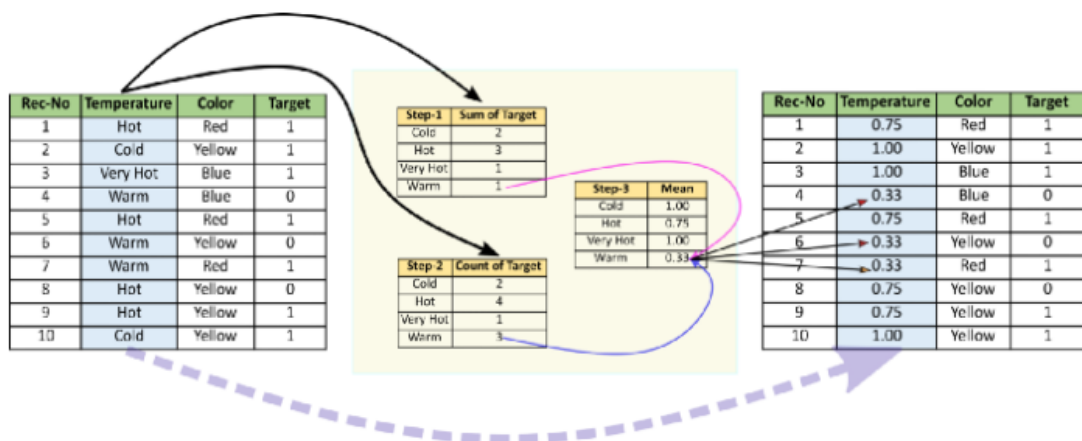


Figure 17. Target Encoding. (Tripathi, 2019)

As shown in Figure 17, In the Target Encoding method, the category is replaced with the mean of the target. This technique can be used with high cardinality data since it does

not increase the column amount and helps in faster learning. It is also possible to use other metrics for encoding like the average of the target. (Tripathi, 2019.)

Next session will explore different ways to handle outliers.

3.3.1.3 Outlier

Unlike other data, *an outlier* is a discrepant value outside the expected range. If left untreated, outliers can result in incorrect predictions, depending on the ML model. Therefore, before training ML, it is essential to address outliers. (Brownlee, 2018.)

To better understand how to handle outliers, the concept of a Gaussian distribution needs to be introduced first, it is also called a normal distribution. Figure 18 shows an example of a normal distribution, the Gaussian distribution is a symmetric probability distribution in which the data tend to centre around the mean.

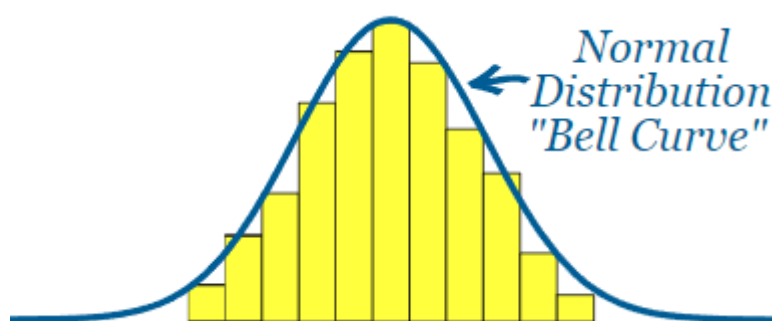


Figure 18. Normal Distribution (Math is fun, 2022).

For a sample with a Gaussian distribution, the standard deviation can be used as a cut-off point to identify outliers. In this type of distribution, the standard deviation of the mean can be used to reproduce the percentage of sample values reliably. Three standard deviations from the mean in a large sample or two standard deviations in smaller samples are the common cut-off point for identifying outliers in a Gaussian distribution. (Brownlee, 2018.)

The method for non-Gaussian data is *the Interquartile Range Method*. This method is calculated as the difference between the 75th and 25th percentiles of the data multiplied by 1,5. (Brownlee, 2018.)

Next session will explain how to handle data imbalance.

3.3.1.4 Data imbalance

Data imbalance happens when there are classes with values out of proportion in a dataset.

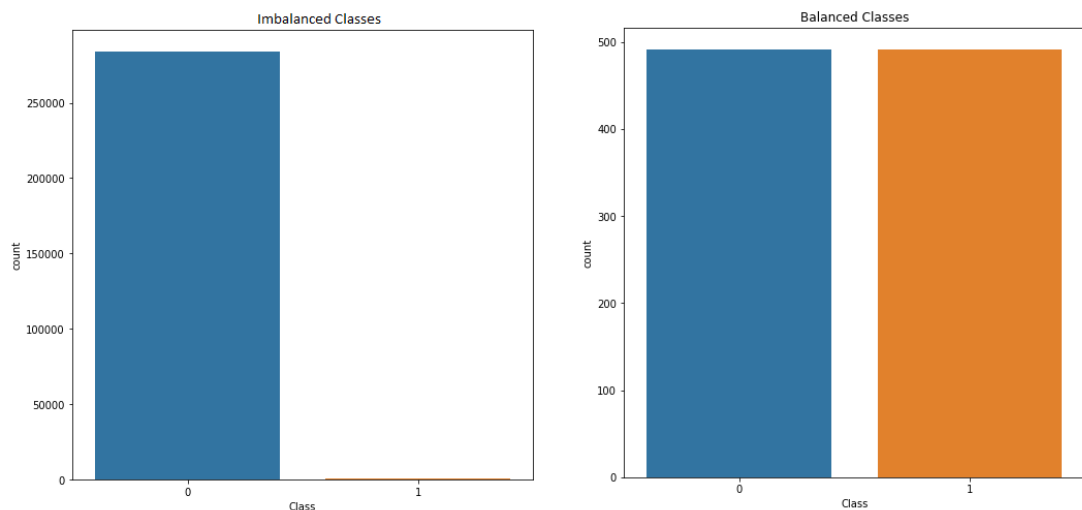


Figure 19. Imbalanced x balanced dataset. (Badr, 2019)

The chart on the left in Figure 19 shows that the dataset example is imbalanced because class 1 has much fewer observations than class 0. This mismatch must be corrected as it affects the correlations between resources and can generate a biased model. The chart on the right shows an example of a balanced dataset where both classes have the same number of observations. (Badr, 2019.)

There are two main techniques to solve this problem, *Undersampling* and *Oversampling*. They are shown in Figure 20 below.

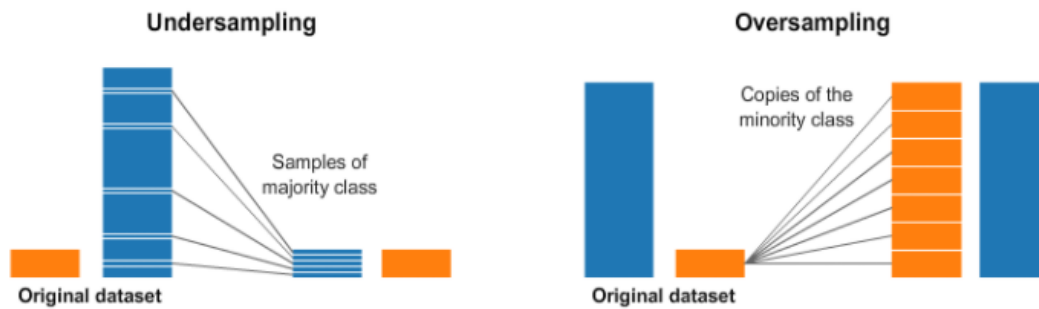


Figure 20. Undersampling x balanced dataset (Badr, 2019).

As seen Figure 20, in *Undersampling*, values of the largest dice are randomly deleted to match the number of dice of the smallest class. In *Oversampling*, the data of the smaller class is randomly copied until the set equals the amount of data of the larger class. (Badr, 2019.)

Another important step of data pre processing is to identify the needs of feature scaling. Next session will explain when and how to do feature scaling.

3.3.1.5 Feature Scaling

In many machine learning algorithms, when there are numerical variables with very different scales, it is convenient to normalize or scale them so that a significant number does not affect the model just because of its large magnitude. (Roy, 2020)

Suppose there are two variables, weight and price, a price of \$10 and a weight of 10 grams represent two completely different things, but the models will treat the two as the same. For this reason, it is needed to convert the values to the same scale. (Roy, 2020)

Figure 21 shows the most common techniques for feature scaling.

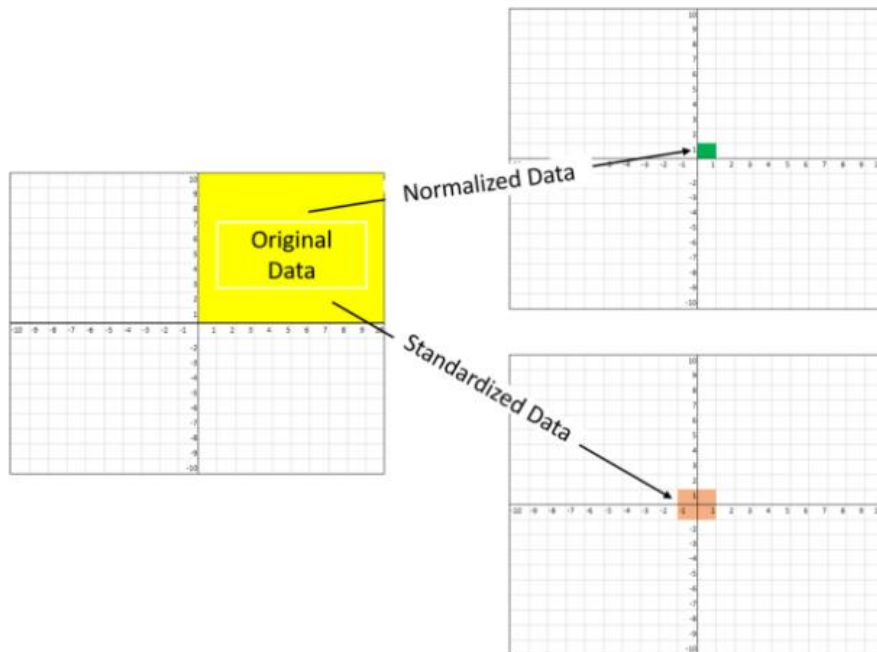


Figure 21. Feature Scaling Techniques (Roy, 2020).

As seen from Figure 21, normalization limits the values between two numbers. While Standardization transforms the data to have a mean of zero and a variance of 1. (Roy, 2020.)

However, not all ML algorithms need scaling. Scaling is essential in algorithms that calculate the distance between data, such as KNN and Principal Component Analysis. Rule-based algorithms such as Random Forests, Gradient Boosted and Decision Trees do not need to be normalized. (Roy, 2020.)

This thesis uses KNN algorithm and therefore Feature scaling is needed.

Next session will explore the different techniques to split the dataset into training and testing.

3.3.2 Model training and testing

Countless models can be used for many different purposes. When selecting the model, it is necessary to understand how much preparation the model requires and if the model meets the business objective. (Rijmenam, 2019.)

The objective will not necessarily be achieved with the first model. Until the first model is created, the potential impact of the model is unknown. Thomas Edison’s quote: “I have not failed. I’ve just found 10,000 ways that won’t work”. Therefore, building the first model is a reality check for performance and future expectations. (Dean, 2014.)

3.3.2.1 Training

Training the data is to incrementally improve the model's predictions, updating the weights and biases at each training cycle. (Rijmenam, 2019.)

A common problem when training the model is when it works very well on the data used in training but fails to achieve the same performance on new, unseen data. This type of problem is called *overfitting*. Another problem is when the model also presents a bad result in the data used to train it. This type of problem is called *underfitting*. *Overfitting* and *Underfitting* can happen for a variety of reasons. Creating different samples of data to train and test the model is the most common approach to help identify this type of problem. (Myrianthous, 2021.)

The fastest and most common way to split data is to split it randomly. In this technique, the database is randomly divided between training, testing and validation. It is shown in Figure 22 below.

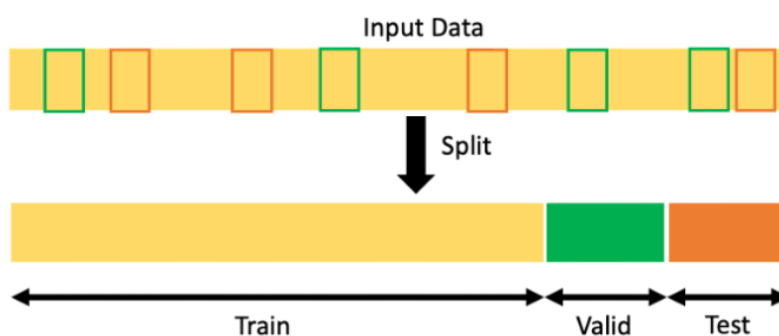


Figure 22. Randomly split the input data into train, valid, and test set (Agrawal, 2021).

Figure 22 shows an example, where the Input data was randomly split into valid (green), test (red) and train (yellow).

For temporal data, a better way to split the data is to include date variables. In this technique, the data is sorted by the variable date. The training dataset will contain the oldest data, the validation dataset the next data, and the test dataset the most recent data. The advantage of this technique is to validate if the model will work well in the next periods. It is shown in Figure 23 below.

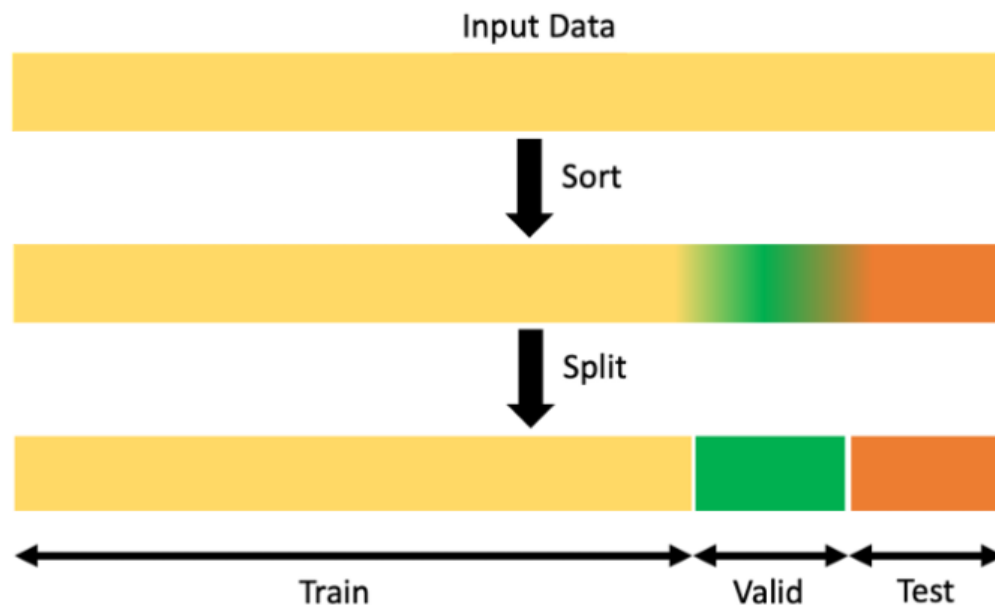


Figure 23. Train Valid Test Dataset after sorting the data (Agrawal, 2021).

Figure 23 shows an example where the input data was sort by a period and in sequence split into train (yellow), valid (green) and test (red).

Next step is to build models iteratively. The goal is to create several different models and compare it to get select the model with best result.

This step is a feedback cycle in which the researcher will create and compare a model to the first one. If the current model is better than the first model, but the project goals still need to be met, another model is created. (Dean, 2014.)

The criteria to select the best model are discussed in next session.

3.3.2.2 Evaluation

There are different evaluation metrics for each type of machine learning model.

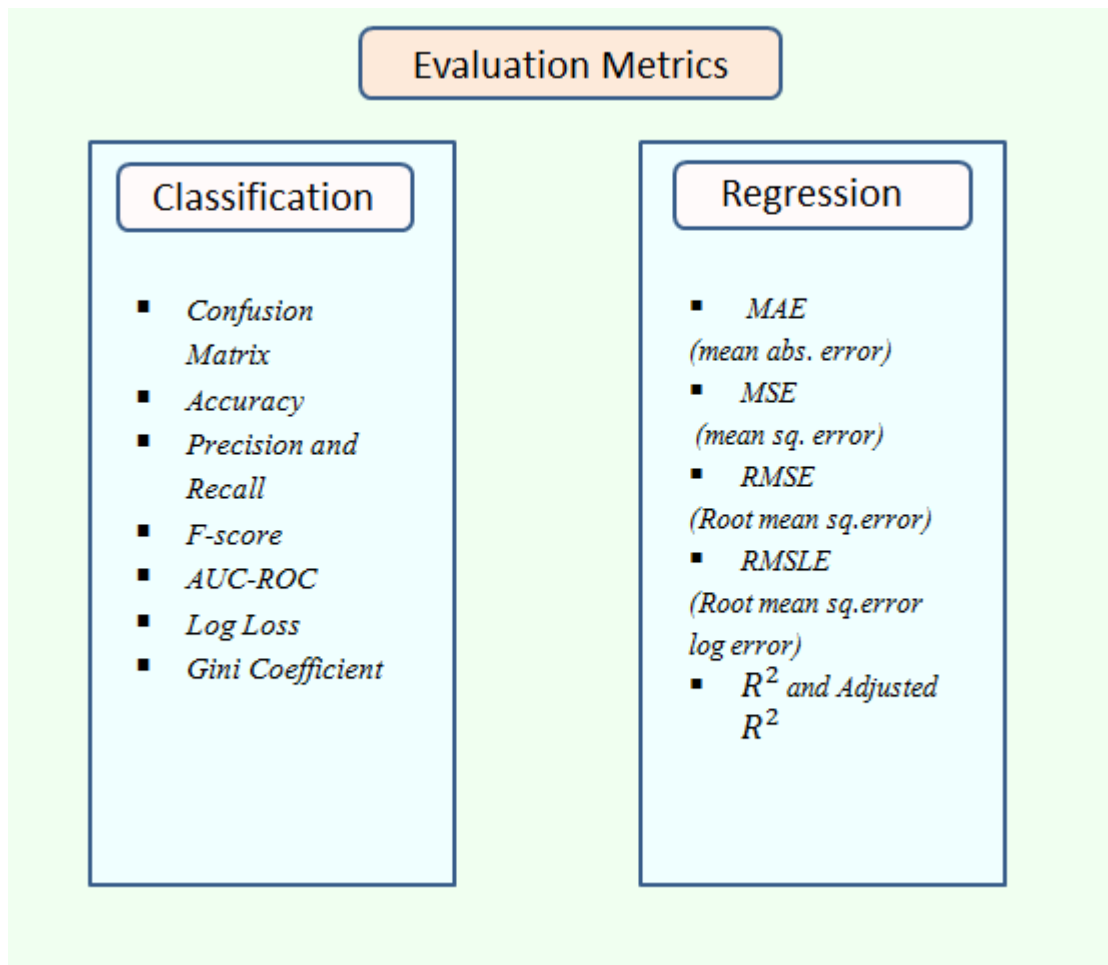


Figure 24. Evaluation metrics for classification and regression models (Ladkar, 2020)

Figure 24 shows the many different evaluation metrics according to the ML type. For Classification the metrics are Confusion Matrix, Accuracy, Precision and Recall, F score, AUC-ROC, Log Loss and Gini Coefficient. For Regression the metrics are: Mean absolute error (MAE), Mean square error (MSE), Root mean square error (RMSE), Root mean square error log error (RMSLE), R squared (R^2) and R squared adjusted (R^2 adjusted).

The *confusion matrix* is a comparative table of the values that an algorithm predicted against the actual values that occurred. It provides good insight into which classes were predicted correctly and incorrectly and what mistakes are being made. (Ladkar 2020)

Action	Predicted (that there will be a buy)	Predicted (that there will be no buy)
Actually bought	TP: 500	FN: 400
Actually did not buy	FP: 100	TN: 9000

Table 3. Confusion Matrix (Gollapudi,2016).

Table 3 shows an example of a Confusion Matrix. In this example, the algorithm is trying to predict if the customer will buy or not a product. The Columns are the predicted values, and the rows are the actual values. The amount of correctly predicted and bought is 500 customers. 100 customers were predicted to buy and did not buy. 400 customers predicted not to buy, and they bought. 9 000 customers were predicted to no buy and did not buy.

Accuracy is the proportion of true results over the total number of cases analyzed. (Ladkar 2020). The accuracy formula is:

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FP+FN+TN)}$$

Where,

TP = True Positive, positive predicted positive data.

TN = True Negative, negative predicted negative data.

FP = False Positive, negative predicted positive data.

FN = False Negative, positive predicted negative data.

Precision is the proportion of predicted Positives that are truly Positive. *Recall*, also called sensitivity, is the rate of actual positives correctly classified (Ladkar 2020). The precision and Recall formula are given as follows:

Precision = $(TP)/(TP+FP)$.

Recall = $(TP)/(TP+FN)$

The *F1 score* is a hybrid evaluation metric useful for unbalanced classes because it is the harmonic mean between precision and recall. (Ladkar 2020) The F1 score formula is:

$F1 = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$

The receiver operating characteristic curve (ROC curve) is a graph that plots two parameters: True Positive Rate and False Positive Rate. The closer the chart is to the top and left-hand borders, the more accurate the test. Figure 4 shows an example of a ROC curve. (Ladkar 2020)

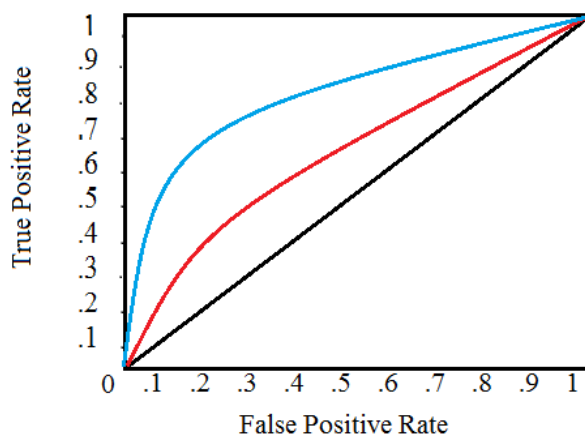
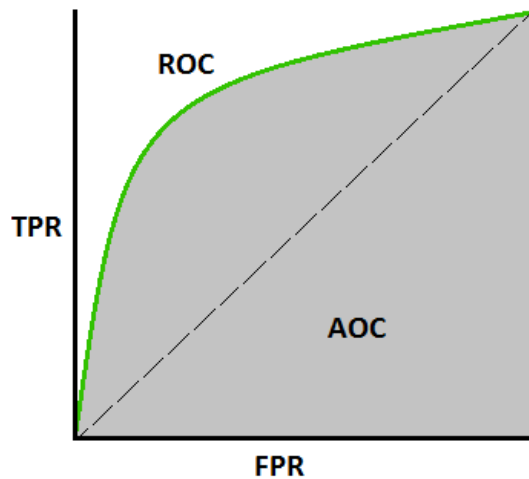


Figure 25. ROC Curve (Ladkar, 2020)

The *Area Under the ROC Curve* (AUC) is a performance metric used for evaluating the ML algorithms. The closest to one is the ROC curve, the better is the model's prediction. (Figure 5)



shaded area represent AUC whereas green line indicates ROC

Figure 26. AUC (Ladkar, 2020)

In regression models the prediction is a continuous variable. Therefore, the validation metrics are different. As classification models, for regression the validation is obtained by comparing the prediction values versus the actual values. (Gollapudi, 2016)

One of the metrics for regression model is the *Mean absolute error* (MAE). MAE is the most straightforward evaluation metric, and it is easy to interpret. It is the average of the absolute value of the errors. (Ladkar 2020) The formula is:

$$MAE = \frac{\sum_{i=1}^n |y - \hat{y}|}{n}$$

Where,

n = number of observations

y = prediction

\hat{y} = true value

Another metric is the *Mean squared error* (MSE).

The *Root Mean Square Error* (RMSE) takes the difference between observed and predicted values. Lower RMSE values are better. We can use the RMSE to compare any predicted value with an actual measurement. (Ladkar 2020) Here is the formula:

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y - \hat{y})^2}{n}}$$

Where

n = number of observations

y = prediction

\hat{y} = true value

All the metrics listed on this session will be used on this study to select the best ML Model.

The next session will explore the Machine Learning Canvas, “A framework to connect the dots between data collection, machine learning, and value creation”. (Dorard, 2016)

3.3.3 The Machine Learning Canvas

To support the creation of Machine learning Algorithms, Louis Dorard (2016) created a methodology named “Machine Learning Canvas”. It is a powerful tool that helps to understand all requirements to build accurate ML models.

The ML Canvas is derived from the Business Model Canvas. Figure 27 shows the template for the ML Canvas proposed by Dorard.

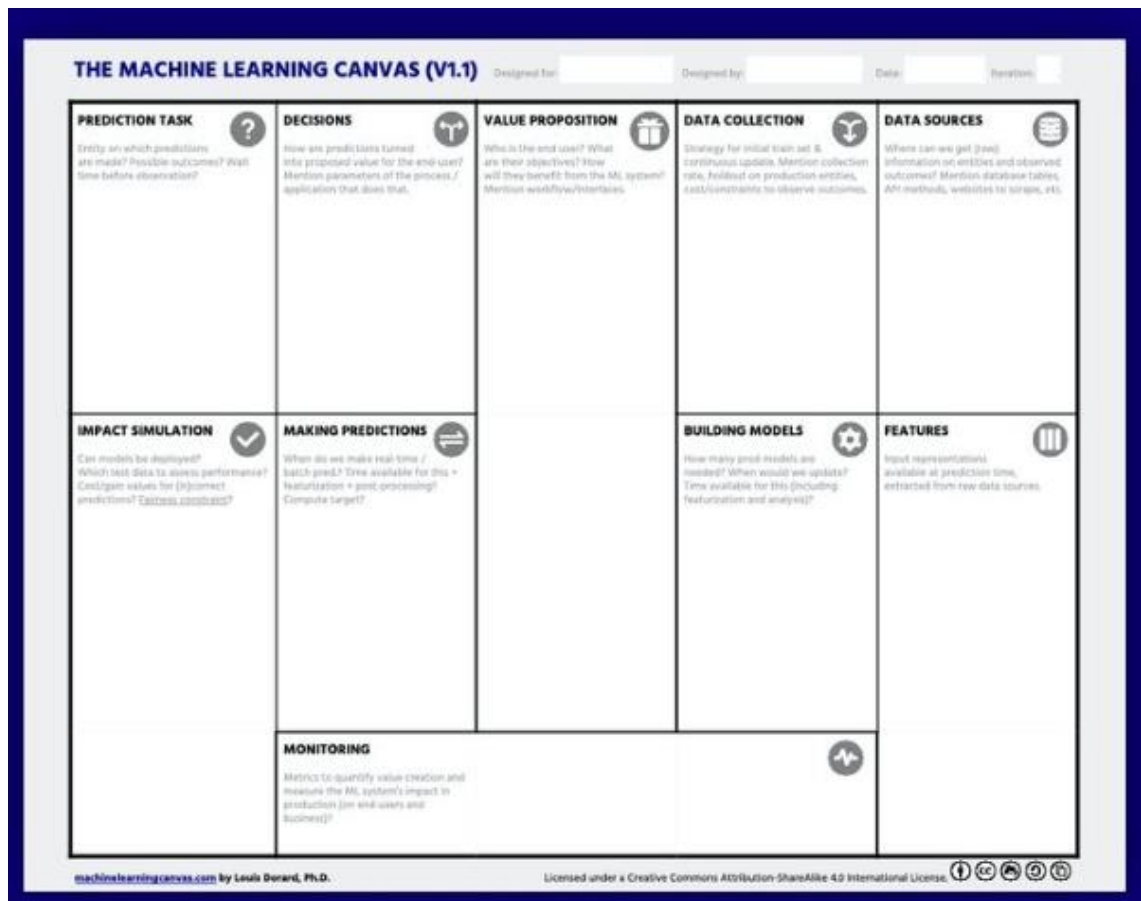


Figure 27. The Machine Learning Canvas

In the centre of the canvas is the *Value Proposition*, a list of the values the ML Algorithm will add to the business. It describes the importance, the objective and who the stakeholders are. (Dorard, 2016)

The next block is the *Data Sources*. It is a preliminary step in analysing which data sources will be used in the algorithm. The objective is to identify dependencies with other ML models and anticipate problems related to data source constraints. For example, some sources might be the outputs of other machine learning systems. (Dorard, 2016)

The *Collecting data* block is used to list the data collection strategy. The dataset used in the algorithm is critical to the model's success. Therefore, verifying that the collected data source represents a real scenario in which the model must make predictions is essential. In addition, it is recommended to analyze the possibilities of long-term data collection. (Dorard, 2016)

The list of all variables to be studied must be described in the "*Features*" block. This session helps the data engineer understand what data needs to be extracted from the data source. (Dorard, 2016)

The *Building models* block describes the frequency that the algorithm must be updated, as well as explains when the model will need to be updated. There are two main reasons why a model is updated. The first is to add more data and generate better models. The second reason is when model dynamics change. In addition, this section is also used to specify other constraints, such as the size of the model. Knowing all the constraints is essential for choosing the right ML algorithms before implementing them. (Dorard, 2016)

The ML Task section shows the ML problem type and the predicted inputs and outputs. It is recommended to list all possible outputs (for classification) or their range (for regression). The objective is to identify metrics to consider when evaluating the model's accuracy.

The decision block describes how the prediction will be used in the business and all steps that should be taken to generate value for the users. (Dorard, 2021)

The Making predictions session aims to identify the technical settings for the models. Here are listed the frequency at which the prediction should be calculated, the volume of predictions and any other constraints the model has. (Dorard, 2016)

The Impact simulation block list all scenarios that can occur when the model predicts the data correctly or incorrectly. The objective is to understand the cost and gain values associated with each decision. (Dorard, 2021)

The Monitoring session describes all metrics and tools that will be used to quantify the value creation of the ML Model. (Dorard, 2021)

3.4 Conceptual Framework of This Thesis

This session will present the conceptual framework of this study.

The framework of this thesis combines the ML Canvas proposed by Dorard (2016) and the research methods in Machine Learning proposed by Kamiri (2021). Figure 28 illustrates this approach.

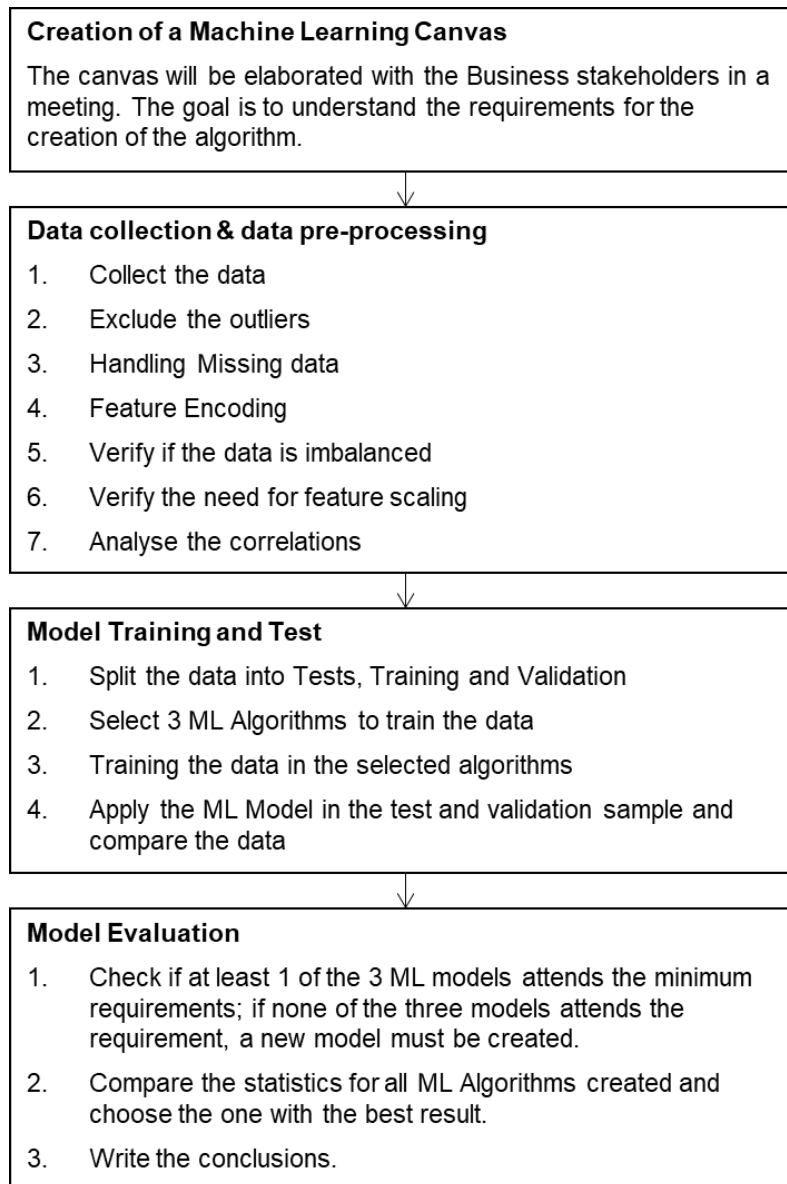


Figure 28. Framework for creating the ML Algorithm to predict OTD in the case company

The first step is to create an ML canvas. The objective of the canvas is to support the creation of the machine learning algorithm because it gives a clear description of the requirements for the algorithm.

The next step is data collection and data pre-processing. In this stage, a sequence of activities is performed. The list of all activities is shown in figure 28.

The third step is training and testing the model. Three data samples will be analysed, one for training, one for testing and one for validation. Finally, the ML algorithm will be applied to the samples.

The last step is to evaluate the model. In this stage, the best algorithm will be selected.

The next session will explore the current state analysis.

4 Current State Analysis of OTD Prediction in the Case Organization

This section of the thesis presents the results from the current state analysis of the current OTD prediction within the case company.

4.1 Overview of the Current State Analysis

The current state analysis objective is to evaluate the current situation and identify the requirements for the ML algorithm. Online meetings and data from EDW were the primary data collection sources. The current state analysis was conducted in three steps.

First, a brainstorming session was conducted virtually over Microsoft Teams. The manager of analysis and master data was selected for this session because she is the most experienced employee in the case company on this topic. She was asked to list all variables that make a delivery on-time or late.

Next, an ML Canvas was created, where all the requirements were identified for the ML algorithm. The canvas was created in an online meeting with three stockholders. They were asked about the expectations and the desired output.

Finally, a script was created to extract the data from EDW. This data is the input for the Machine Learning algorithm.

Next session will explain how the OTD KPI is calculated.

4.2 Description of OTD Calculation

On-time Delivery (OTD) measures the percentage of deliveries processed according to the promise made to a customer. It is calculated as the number of deliveries on-time versus total orders delivered.

Where,

- On-time: $\text{POD Date} \leq \text{Combined Delivery Date}$

- Late: $\text{POD Date} > \text{Combined Delivery Date}$
- POD Date is when the proof of delivery (POD) was confirmed for delivery.
- Combined Delivery date (or promised delivery date) can be either the first input date (Original) or the last input date (Active). That depends on the company's delay indicator and delivery blocks combinations. When the company's delay indicator exists, the measure is against the *original date* unless there is a customer-related delivery block. In that case, the calculation is based on the *active date*.

Currently there is no tool to predict the OTD in the case company.

The next session will explore all variables selected to input into the ML Algorithm.

4.3 Data selection for the ML Algorithm

A brainstorming on Microsoft Teams chat was conducted with the Area Manager responsible for the OTD Analysis to select all the variables for the ML model/algorithm.

The stakeholder was asked to list all variables she believed could impact the OTD performance.

Table 4 shows the data selected for this study. The first column is the Variable name. The second column shows the possible values, and the third column is the variable definition.

Table 4. Data selection for this study.

Variable Name	Values	Definition
Is OTD	1 = On-time 0 = Late	Indicator if OTD is on time.
Is Late	1 = Late 0 = On-time	Indicator if OTD is late.
POD Date	Date	Date when the proof of delivery (POD) was confirmed for a delivery.

PCM Team	alphanumeric string	Internal team that handled the order.
Ship to Country	alphanumeric string	Country destination of the delivery
Order ID	integer	Unique number of a sales document
Order Item	integer	Item number from a sales order
MRP Profile	alphanumeric string	Material requirements planning (MRP) parameters.
Shipping Type	alphanumeric string	Shipping type (for example, by road or rail) that has been selected for the transportation of the goods for the shipment legs.
Shipping Condition	alphanumeric string	Type of shipping strategy for the delivery.
Incoterm	alphanumeric string	Trading terms established by the International Chamber of Commerce
Consolidation key	alphanumeric string	Text to identify grouped deliveries.
Plant	alphanumeric string	Key uniquely identifying a plant.
Shipping Point	alphanumeric string	The physical location from which the item is shipped.
Combined Delivery Data	Date	Delivery date requested by the customer.
Schedule Line Delivery Data	Date	The date requested by the customer, or the earliest date proposed by the system.
First delivery date	Date	The first Schedule Line Delivery date.
Lead Time (OCD x FDD)	Date	Lead time in working days between Order Creation date and First delivery date.
Affected by COVID	1 = Yes 0 = No	Indicator if the document was delayed due to the pandemic situation in 2019.
C1 Task	1 = Yes 0 = No	Workflow task, indicates delivery released.
T Task	1 = Yes 0 = No	Workflow tasks starting with T, technical requirement tasks.

The next session will explore the ML Canvas.

4.4 Business requirements for the ML Algorithm

A meeting was conducted with the stakeholders to discuss their expectations and requirements for the ML Algorithm.

Was present in the meeting:

- This thesis researcher
- Former manager of Analysis and Master Data
- Current manager of Analysis and Master Data
- General Manager Operations Support
- Data scientist

The output of the meeting was the ML Canvas for the OTD Prediction. The ML Canvas has ten topics: Value proposition, data collection, data sources, prediction task, decisions, impact simulator, making predictions, building the model, features, and monitoring.

Figure 29 shows the ML Canvas for the OTD prediction in the case company.

The Machine Learning Canvas











<p>Prediction Task </p> <p>Entity: Order Item Outcomes: "on time" or "late".</p> <p>Binary Classification where 1 = late and 0= "on-time"</p>	<p>Decisions </p> <p>The Power BI report will execute the following tasks:</p> <ol style="list-style-type: none"> 1. Filter out orders already delivered. (Where POD date is not empty) 2. Filter out predicted "on-time" 3. Sort remaining by the probability of "late" in descending order. <p>The report is automatically refreshed daily.</p>	<p>Value Proposition </p> <ul style="list-style-type: none"> • Objective Anticipate delivery issues by Identifying whether the order will be delivered on time or late. • Question to answer What is the probability that this order will be delivered late? • How Users will access the data in a Power Bi report; from there, they will be able to see the list of all deliveries they should focus on. <p>Based on this list, they can negotiate new delivery dates with the Customers and/or closely monitor the delivery workflow.</p> <ul style="list-style-type: none"> • Who is the stakeholder PCM Managers and Operations Support 	<p>Data Collection </p> <p>Data is extracted weekly from EDW by running SQL script in Power BI Dataflow.</p>	<p>Data Sources </p> <p>The source of the data is internal from SAP. Only GLS Plants are included. Filter POD delivery date >= 01/01/2019. Exclude all documents affected by the pandemic. Exclude all documents delivered between June-September - 2021</p>
<p>Impact Simulation </p> <p>Correct classification of late deliveries (True positive) will anticipate issues and help the stakeholder to make quicker decisions, thus reducing costs, improving operations, and providing better customer services.</p> <p>Classification of a late delivery as on time (False negative) can cause higher costs and lower customer satisfaction.</p> <p>Classifying on-time delivery as late delivery (False positive) can waste stakeholders' time and increase costs.</p>	<p>Making Predictions </p> <ul style="list-style-type: none"> • Every week run re-classification for all orders. • Load the results into Power BI. 		<p>Building Models </p> <p>Every month run the model from the previous 24 months of data. The minimum accepted accuracy is 70%, and if the accuracy doesn't meet the target a new model must be created.</p>	<p>Features </p> <ul style="list-style-type: none"> • Is Late • POD Date • PCM Team • Ship to Country • Order ID • Order Item • MRP Profile • Shipping Type • Shipping Condition • Incoterm • Consolidation Key • Plant • Shipping Point • Combined Delivery data • Schedule Line Delivery data • Affected by Covid • C1 Task • T Task
<p>Monitoring </p> <ul style="list-style-type: none"> • Report usage • Stakeholders feedback • OTD 				

Figure 29. The Machine Learning Canvas for OTD prediction in the case company

First it was discussed the *value proposition*. The stakeholders said that the algorithm's objective is to anticipate delivery issues by identifying whether the Order would be delivered on time or late. The goal is to have a Power BI report where users will see a list of all potentially late deliveries. Based on this list, they can negotiate new delivery dates with the customers and closely monitor the delivery workflow.

Next, there was a discussion about *data collection* methods. The decision was to extract the data from EDW by running SQL script in Power BI Dataflow. *The data sources* are tables from SAP where only GLS Plants should be included. The way to calculate OTD changed in 2019 due to adding new fields on SAP. Therefore, all documents delivered before 01/01/2019 should be excluded. It was also decided to remove all documents from June to September 2021; during this period, there was a system change in the distribution centre, and the KPI calculation was inaccurate. The last filter discussed was the documents affected by the Covid 19 pandemic. In 2019 when many countries closed their borders, there was an impact on the delivery time for some countries; since the goal is to predict the OTD for the post-pandemic period, it was decided to exclude all documents flagged with "Affected by Covid-19".

A binary classification was selected for the *prediction task* where the *entity* is the Order Item, and the outcome is on-time (1) or late (0). This step was important to give clarity to the level of analysis. Since the entity is the Order item, all the selected features must be available before creating the delivery id. So, when selecting the data from SAP, the tables linked to Order should be prioritised over Delivery tables. Some tables on SAP are duplicated for Delivery and Order; this happens when a document changes. E.g., The Order was requested for a specific address, but the delivery was changed to another address. This will result in mismatching data between Order and Delivery tables on SAP.

Next, the *impacts of the simulation* were analysed. The conclusions were that correct classification of late deliveries (True positive) will anticipate issues and help the stakeholder to make quicker decisions, thus reducing costs, improving operations, and providing better customer services. While the classification of a late delivery as on time (False negative) will cause higher costs and lower customer satisfaction and classification of on-time delivery as late delivery (False positive) can waste stakeholders' time and increase costs.

In the *decision* process, it was decided that after loading the ML model into a Power BI report, the following tasks must be executed every day:

1. Filter out orders already delivered. (Where POD date is not empty)
2. Filter out predicted "on-time."
3. Sort the remaining by the probability of "late" in descending Order.

This process should be automated by using PowerBI resources.

For *making the predictions*, weekly, the ML algorithm will recalculate the model and the predictions, and all orders will be re-classified.

A data scientist will run the model from the previous 24 months of data every month. The minimum accepted accuracy is 70%, and a new model must be created if the accuracy doesn't meet the target.

To *monitor* the results, the case company will check the report usage metrics, the stakeholders' feedback and the OTD calculation. The report usage metrics will give good insights if the ML model is adding value to the business or not. If the report has low usage, feedback from the stakeholders must be collected. An increase in the OTD results is expected. Therefore, the OTD KPI will be monitored.

Finally, the features discussed in the brainstorming session were presented. The stakeholders had the opportunity to add new features, but no new features were requested.

The next session will explore the algorithm creation.

5 Building the ML Algorithm for the Case Organization

This section presents all the steps applied to create the machine learning algorithm.

5.1 Overview of the Building Stage

The process of creating the ML algorithm happened in seven steps. The first step was data exploration to handle outliers and missing data properly. Secondly, all the categorical data were encoded, and their correlations were analysed. The next step was feature scaling, where features were transformed to fit within a specific scale. The fourth step was to perform the data imbalance analysis. In the fifth step, the dataset was split into samples for training, testing and validation. Next, seven baseline classification models were trained and tested. Finally, the best tree models were tuning and evaluated.

5.2 Data Collection and Data Preparation

The data collection and preparation consist of an exploratory data analysis. The activities executed in this session are data collection, outliers' exclusion, missing data handling, feature encoding, data imbalance analysis, feature scaling and analysis of the correlations.

This study uses Python 3.9.7 and Jupiter Notebook to describe, train and test the models. The following libraries were used in the project:

- *Pandas* is a library written for manipulating numerical tables and time series.
- *NumPy* is a library for adding support for multi-dimensional arrays and matrices.
- *Matplotlib* is a library for making the graphic representation of data in Python.
- *Seaborn* is based on the Matplot library and provides high-level graphics interfaces in Python.
- *Sklearn* is a library for the automation of Machine learning algorithms in Python.

The next session explains how the data was collected.

5.2.1 Data collection

The data was collected from the Enterprise Data Warehouse (EDW). The time range of study is January 2019 to June 2022. Only GLS plants were loaded into the report. As the requirement in the ML Canvas, documents affected by the pandemic or delivered between June to September 2021 were excluded. All human errors were also excluded from the model. Examples of human error are proof of delivery date in the far future or delivery date earlier than the order creation date.

The dataset was loaded using panda's library. The preliminary analysis of the data frame is shown in Figure 30.

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2162689 entries, 0 to 2162688
Data columns (total 17 columns):
 #   Column                Dtype
---  -
 0   Key                    int64
 1   PCM Team               int64
 2   Ship to Country        object
 3   MRP Profile            object
 4   Shipping type          int64
 5   Shipping conditon      object
 6   Incoterms1             object
 7   C1 block reason        object
 8   T Task                 int64
 9   ConsKey                object
10   Shipping Point         object
11   Is Late                 int64
12   POD                    datetime64[ns]
13   Combined Delivery date datetime64[ns]
14   Schedule line delivery datetime64[ns]
15   Lead Time (OCD x FDD) float64
16   First Date              datetime64[ns]
dtypes: datetime64[ns](4), float64(1), int64(5), object(7)
memory usage: 280.5+ MB
```

Figure 30. Dataset info summary

Figure 30 shows that the data frame has 17 columns and 2,16 million rows. The data types are integer (int64), date (datetime64), decimal (float64) and text (object).

Figure 31 shows the cardinality (number of distinct values) for each column in the dataset.

```

Key                2162689
PCM Team           44
Ship to Country    198
MRP Profile        11
Shipping type      22
Shipping conditon  10
Incoterms1         16
C1 block reason    15
T Task             2
ConsKey            52
Shipping Point     2
Is Late            2
POD                1200
Combined Delivery date 1588
Schedule line delivery date 1445
Lead Time (OCD x FDD) 1120
First Date         1521
dtype: int64

```

Figure 31. Dataset cardinality

As seen in Figure 31, the column “Key” is the unique field that identifies the order, and therefore the cardinality has the same length as the dataset. The columns “POD”, “Combined delivery date”, “First date” and “Schedule Line delivery date” have high cardinality because they are date fields composed of day, month, and year dimensions. The column “Lead Time (OCD x FDD)” has high cardinality because it is a decimal number. The column “Ship to Country” is the text column with the highest cardinality because there are 200 different countries in the dataset.

Next session explores each feature listed in the dataset.

5.2.2 Data pre-processing

This session explores the data. The goal was to understand each model column and adequately handle the outliers and missing data. As the literature review shows, an

outlier can skew the data, leading to incorrect inferences. And missing data can be a critical causing failure of the ML model. Therefore, each feature (Column) of the dataset was analysed. The result of this analysis is given as follows.

Figure 32 shows the relative frequency of late delivery from January 2019 to July 2022. The Y axis is the percentage of late deliveries (hidden due to sensitive data).

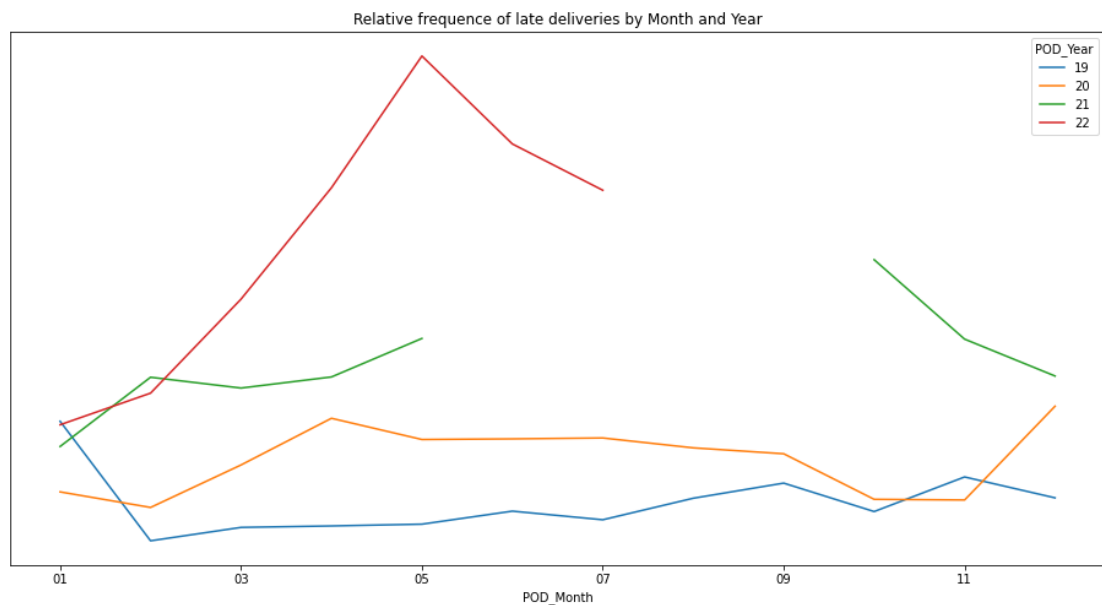


Figure 32. Graph of late delivery frequency per year and month

As seen in figure 32, there is no clear evidence of seasonality, but it is possible to see a trend because the frequency of late deliveries has increased over the years. The data from June to September 2021 is missing because it is excluded from the sample, as explained in the previous session.

The following data analysed was the weekly frequency of late deliveries. The boxplots (Figure 33) give a clear overview of the data. The line in the middle of each boxplot represents the median of the late deliveries.

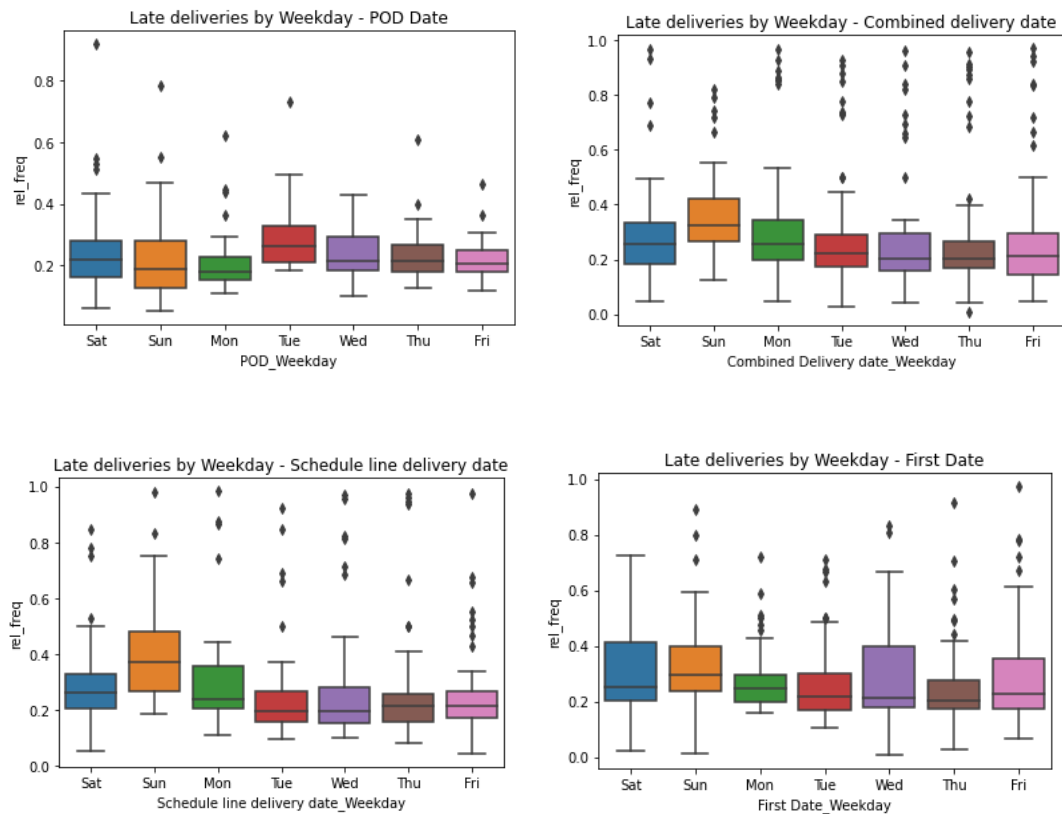


Figure 33. Graph of late delivery by POD

As shown in Figure 33, when the POD is on Tuesday or when the Combined delivery date or the Schedule line delivery date is on Sunday, the delivery tends to be slightly later than on the other weekdays. The First date boxplot doesn't show any significant impact of the weekday in the late delivery. It is also possible to see in the boxplot some indications of outliers in the dataset.

The following data analysed was the geographical distribution of the dataset. Figure 34 shows the sample distribution map. The bubble's size is related to the number of deliveries in the dataset.

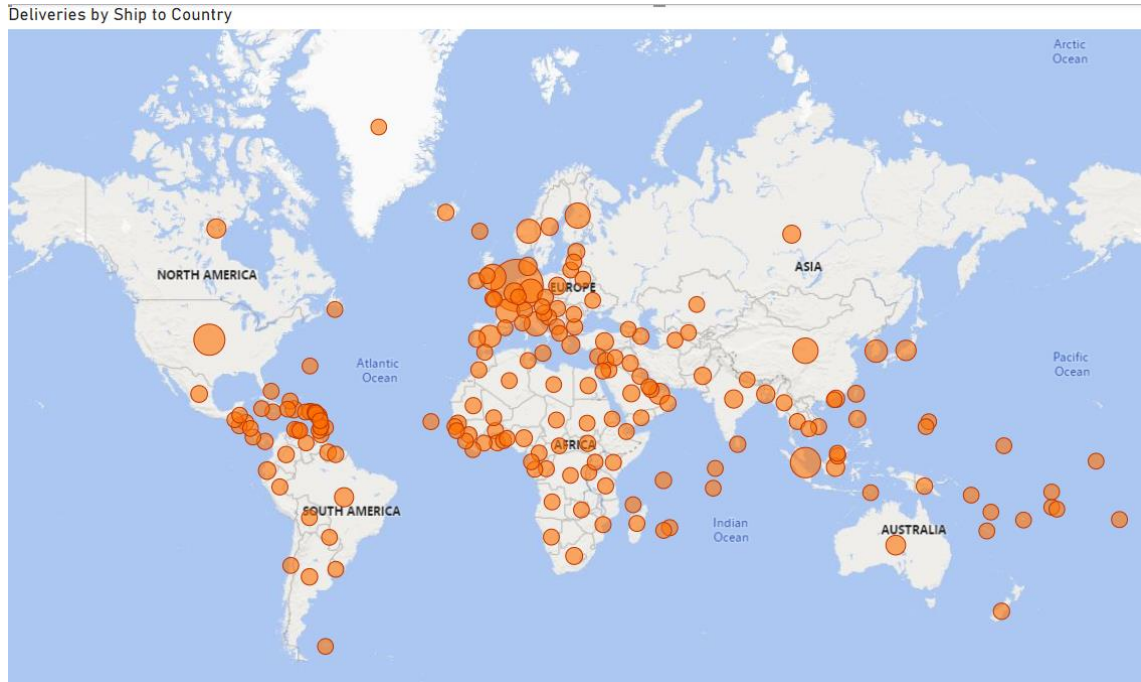


Figure 34. Deliveries by Ship to Country

Figure 34 shows that the sample is distributed globally, with higher number of deliveries in Europe.

There are 32 countries in the sample with less than 50 observations. Figure 35 shows the top 10 countries by the number of deliveries.

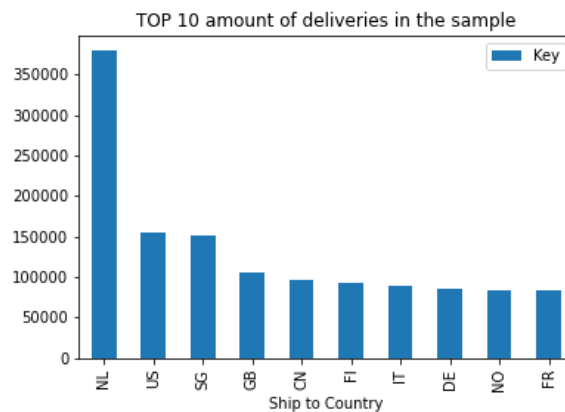


Figure 35. Top 10 countries by number of deliveries in the sample.

As seen in figure 35, the top 10 countries with the most significant number of observations are the Netherlands (NL), United States (US), Singapore (SG), United

Kingdom (GB), China (CN), Finland (FI), Italy (IT), Germany (DE), Norway (NO) and France (FR).

There are 198 countries in the dataset. Therefore, some countries were grouped to reduce cardinality. The code to reduce cardinality was adapted from Sangani (2022). First, the countries were sorted in descending order according to the number of deliveries. Then, all countries below 75% of the total number of deliveries were grouped as “others”.

Another column with high cardinality is the Parts Management Team responsible for the delivery (PCM Team). One of the reasons is that the PCM Team structure changed in 2020, and because of this change, the old codification needed to be replaced by the new codes. Figure 36 shows the number of deliveries by the PCM Team.

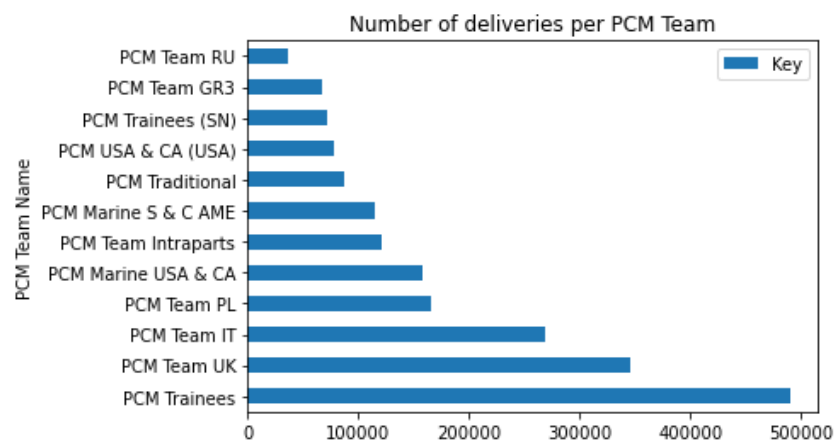


Figure 36. PCM Team

Figure 36 shows that PCM Trainees have the highest number of documents. There is also possible to see some outliers: PCM Team BR, PCM Marine IC&O, PCM Team SUEACN, PCM Team NO, PCM Center GR, PCM Team NL and PCM Energy S & C AME. Those teams are old teams that other teams had incorporated. Since assigning those documents to the correct team is impossible, these values were removed from the dataset. The total number of excluded observations is 17 693, representing 0.82% of the entire data frame (2,16M).

The following data analysed was the Material Requirements Planning Profile (MRP Profile). Figure 37 shows the analysis of the data frame by MRP profile.

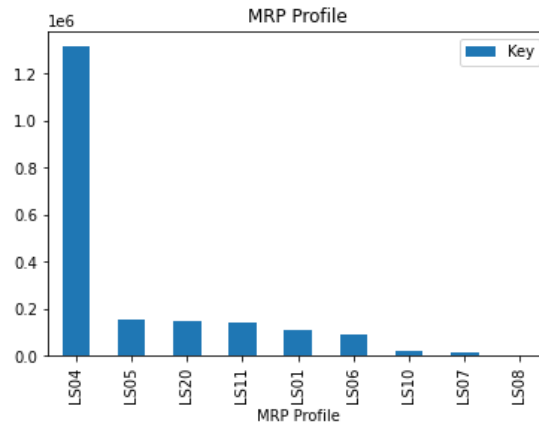


Figure 37. MRP Profile

Figure 37 shows that the class with the most significant number of documents is LS04 representing 63% of the database. The types LS03 and LR20 have less than ten documents each and therefore are not representative and were excluded.

The following data analysed was the shipping type. Figure 38 shows the distribution frequency of shipping type.

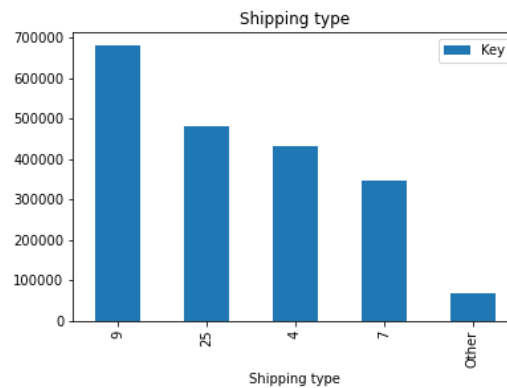


Figure 38. Shipping Type

As seen in Figure 38, the shipping type most frequent in the sample is type 9 (Courier), representing 33% of the total values. There is also a representative amount of data for shipping types 25 (To be collected), 4 (Airfreight) and 7 (Truck). All other shipping types were grouped in the category "Other".

The following data analysed was the Shipping Condition. Figure 39 shows the number of observations by shipping condition.

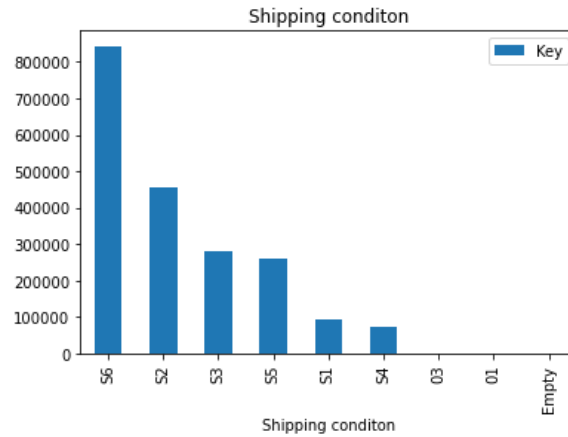


Figure 39. Shipping condition

Figure 39 shows that class S6 is the most frequent shipping condition. The shipping condition '03', '01' and 'empty' have less than 50 observations, and because there are not enough data, they were deleted from the dataset.

The following column analysed was the Incoterms 1. Incoterms are terms of sales used in international trade. Figure 40 shows the incoterms 1 and the number of deliveries in the sample.

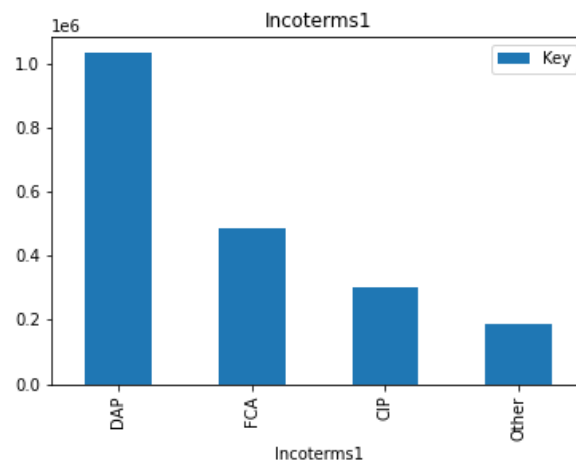


Figure 40. Incoterms 1

Figure 40 shows that the most frequent terms found in the dataset are DAP, FCA and CIP. The other terms were grouped as "Other" to reduce dimensionality.

The following column analysed was the Consolidation Key. The consolidation key indicates when several different orders need to be delivered together. Figure 41 shows the number of deliveries by consolidation key.

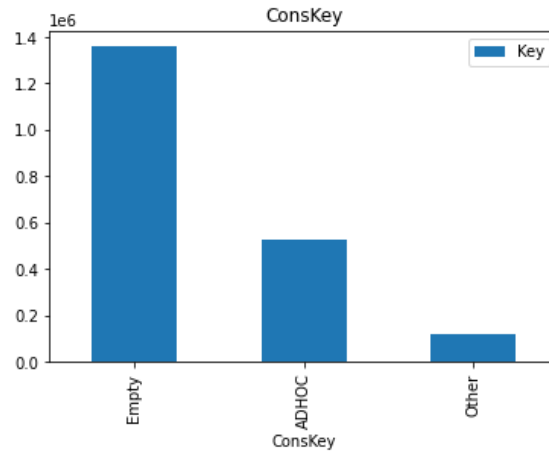


Figure 41. Consolidation Key

As seen in figure 41, most orders don't have a consolidation key. The Adhoc is the most frequent consolidation key, and all the other keys were grouped as "Other".

The following column analysed was the shipping point. Figure 42 shows the number of deliveries by shipping point.

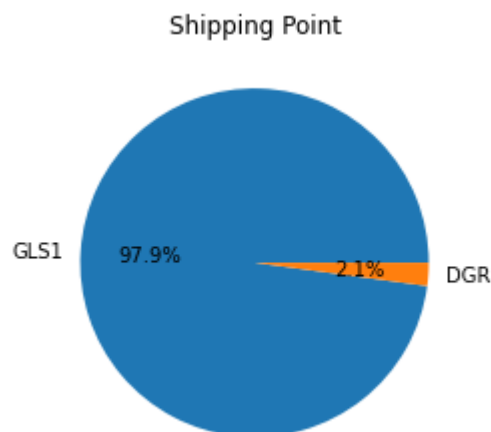


Figure 42. Shipping Point

As Figure 42 shows, there are only two classes of shipping points. GLS1 represents 97,9% of the database, and DGR represents 2,1%. Since there is not enough data for all classes, this feature was removed from the dataset.

The following column analysed was the C1 block reason. C1 block reasons are tasks created when a document was blocked, for example, Information pending from a customer or missing delivery address. This column was grouped into “Is C1 blocked”, where:

1 = It has a C1 block reason

0 = It doesn't have a C1 block reason

Figure 43 shows the C1 blocked distribution.

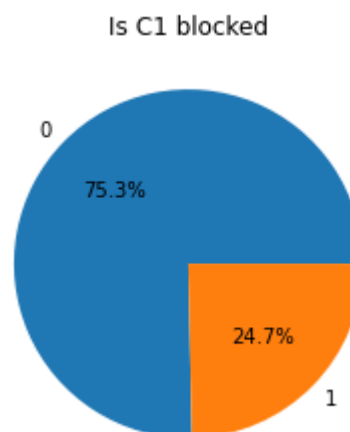


Figure 43. C1 Blocked

As seen in figure 43, 24,7% were blocked by the C1 task.

Another task analysed was the “T Task”. They are technical requirement tasks. However, from the total amount of documents, only 0,2% had a T Task linked to it, and since the amount is not representative, this column was excluded from the dataset.

Figure 44 shows the dataset summary after applied data pre-processing.

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 2002646 entries, 0 to 2020434
Data columns (total 14 columns):
#   Column                                Dtype
---  -
0   Ship to Country                       object
1   MRP Profile                           object
2   Shipping type                         object
3   Shipping conditon                     object
4   Incoterms1                            object
5   ConsKey                               object
6   Shipping Point                       object
7   Is Late                               int64
8   Lead Time (OCD x FDD)                float64
9   POD_day                              object
10  Schedule line delivery date_Year      object
11  FD Weekday                            object
12  PCM Team Name                        object
13  Is C1 blocked                        int32
dtypes: float64(1), int32(1), int64(1), object(11)
memory usage: 221.5+ MB

```

Figure 44. Dataframe summary after data pre-processing

Figure 44 shows that the dataset after the pre-processing consists of 14 columns. The column “T Task” and “Shipping point” were excluded because they didn’t have enough data for all classes. The column “Key” was deleted because it is the unique id that identifies each order and can’t be loaded into the ML algorithm. One hundred sixty thousand rows were removed from the dataset. They were related to outliers or missing data.

The next session shows the process of feature encoding.

5.2.3 Feature Encoding

Feature encoding is needed to convert categorical variables into numbers. The variables in this study are not ordinal; therefore, the feature encoding technique chosen was one-hot encoding. For this process, the panda's library function `get_dummies` was used.

The one hot encoding was applied for all categorical columns. After this process, the dataset increased to 72 columns.

The next step is to analyse the correlations.

5.2.4 Analysis of correlations

The analysis of correlation is important to detect multicollinearity and for feature engineering. The first step was to create a correlation matrix graph looking for any visually apparent correlation. Figure 45 shows the matrix.

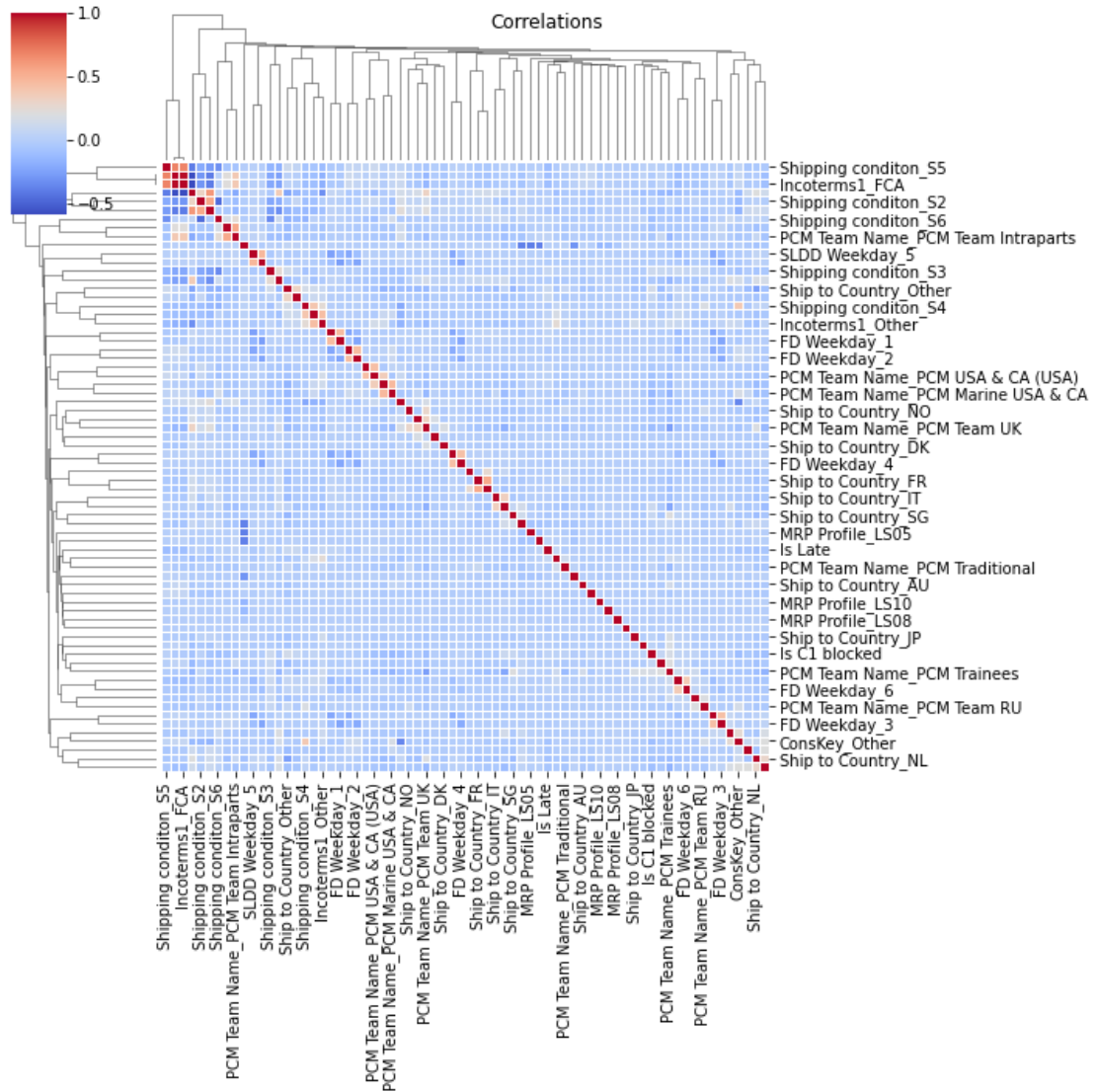


Figure 45. Correlogram

As shown in figure 45, it is possible to see a concentration of data points with darker blue or red areas, which indicates the presence of multicollinearity in the dataset.

Table 5 shows the top 10 correlations.

Table 5. Top 10 correlations

	FEATURE_1	FEATURE_2	CORRELATION
3198	Shipping type_25	Incoterms1_FCA	0.983491
465	Shipping conditon_S5	Shipping type_25	0.684663
468	Shipping conditon_S5	Incoterms1_FCA	0.682217
3127	Shipping type_9	Incoterms1_DAP	0.605508
3338	Incoterms1_DAP	Incoterms1_FCA	0.584863
3197	Shipping type_25	Incoterms1_DAP	0.578564
1666	Ship to Country_FI	PCM Team Name_PCM Team Intraparts	0.538382
254	Shipping conditon_S2	Shipping type_9	0.536752
1734	Ship to Country_FR	PCM Team Name_PCM Team GR3	0.534243
908	SLDD Weekday_5	FD Weekday_5	0.470098

Table 5 shows that Shipping type_25 and Incoterms1_FCA are the highest correlated features, where 98% of the variance in Incoterms1_FCA is explained by Shipping type_25. Shipping type_25 is also high correlated with Shipping condition_S5. Therefore, to avoid multicollinearity, this feature was excluded from the dataset.

Table 6 shows the correlation between the target variable and the other features.

Table 6. Top 10 correlations with target

	FEATURE_1	FEATURE_2	CORRELATION
6	Is Late	Shipping conditon_S5	0.120534
48	Is Late	Incoterms1_FCA	0.102910
45	Is Late	Shipping type_25	0.100614
32	Is Late	Ship to Country_Other	0.088751
62	Is Late	PCM Team Name_PCM Trainees (SN)	0.084199
49	Is Late	Incoterms1_Other	0.083742
30	Is Late	Ship to Country_NL	0.078535
4	Is Late	Shipping conditon_S3	0.074665
51	Is Late	ConsKey_Other	0.072925
15	Is Late	Ship to Country_BD	0.069060

As shown in table 6, the highest correlation is Shipping Condition_S5, where Shipping Condition_S5 explains 12% of the variance in late deliveries.

The next step will explain how the feature scaling was handled.

5.2.5 Feature Scaling

In the KNN algorithm, it is necessary to apply feature scaling. This step evolves to transform the dataset to fit within a specific scale.

The 'Lead time (OCD x FDD)' is the only variable that is not binary or categorical; therefore, the feature scaling max_min method was applied to this variable.

Figure 46 shows the 'Lead time (OCD x FDD)' before and after scaling.


```
#Before Scaling
df2['Lead Time (OCD x FDD)'].describe().round(2)
```

```
count    1954561.00
mean      2387.23
std       46970.91
min       -585995.00
25%        3.00
50%        6.00
75%       26.00
max       793677.00
Name: Lead Time (OCD x FDD), dtype: float64
```

```
#After Scaling
df['Lead Time (OCD x FDD)'].describe().round(2)
```

```
count    1954561.00
mean      0.43
std       0.03
min       0.00
25%       0.42
50%       0.42
75%       0.42
max       1.00
Name: Lead Time (OCD x FDD), dtype: float64
```

Figure 46. Feature Scaling – Lead Time

As seen on figure 46, the Lead Time was adjusted to fit in a range between 0 and 1.

Next step was the data imbalance analysis.

5.2.6 Data imbalance analysis

The data imbalance is a problem in classification models because it can impact the ability to represent samples from the minority class and consequently reduce the accuracy of the model.

Figure 47 shows the sample distribution between on-time and late deliveries.

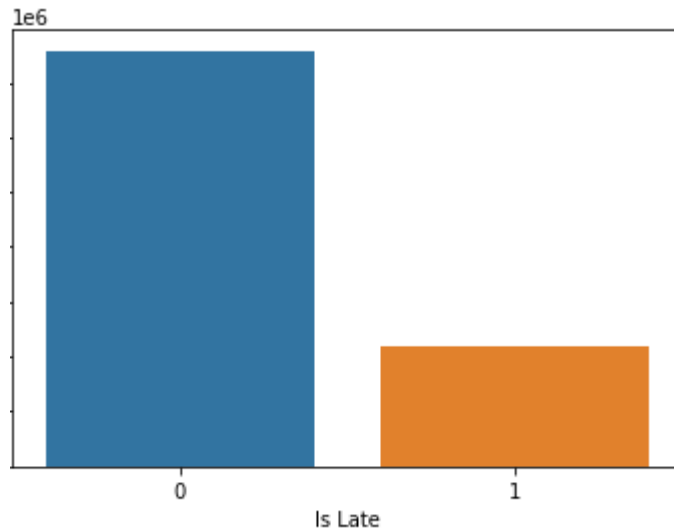


Figure 47. Feature Scaling – Lead Time

As seen in figure 47, the sample size for class 0 (On time) is much bigger than for class 1 (Late). Therefore, the oversampling technique was used to solve this issue. This technique takes random copies of the smaller class until it achieves the same size as the other class.

The next session will show how the model was trained, tested, and validated.

5.3 Model Training and Testing

This session explains how the model was trained, tested, and validated. First, the dataset was split into samples. Secondly, it ran the baseline for the most common ML algorithms, and the top 3 best accuracies were selected for tuning and validation.

It was necessary to run tests on data not used in constructing the model to infer the model's behaviour on new data. So, the dataset was split randomly into 80% in the training dataset and 20% in the remaining dataset. Then the remaining dataset was randomly divided in half for the Valid and Test dataset.

After partitioning the data into training, valid and test sets, the sample was loaded into various baseline classification models. Each model was split into ten smaller sets on this algorithm and trained three times.

Figure 48 shows the result of this analysis.

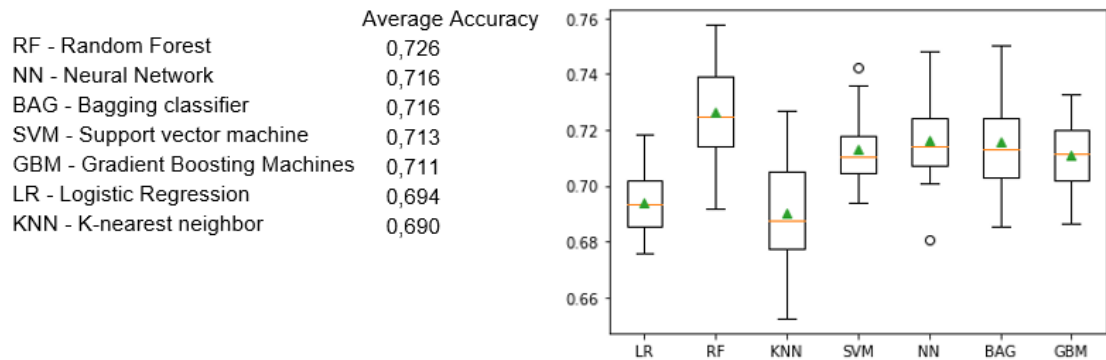


Figure 48. Accuracy of Baseline Classification models

Figure 48 illustrates the similarities between the model performances. The model with the worst accuracy was the K-nearest neighbour, and the model with the best accuracy was the Random Forest. The three best models were selected for tuning.

5.4 Random Forest

The next step was select the features for the Random Forest model, where all the features were ranked and sorted by the chi-square test score. Table 7 shows the performance of the Random Forest model with a different number of features.

Table 7. Random Forest Accuracy per feature amount

Amount of Features	Test Score	Train Score	Valid Score
10	0.6108	0.609200	0.6020
15	0.6325	0.630737	0.6227
20	0.6384	0.643050	0.6344
25	0.6588	0.669512	0.6490
30	0.6796	0.700875	0.6750
35	0.6994	0.738313	0.6920
40	0.7186	0.767912	0.7150
45	0.7324	0.792800	0.7274
50	0.7385	0.815362	0.7353

Figure 49 indicates that the accuracy of the test increases according to the number of features added to the model, but it gets distance from validation, and test sets, which indicates that some features correlate with each other (multicollinearity).

Based on this analysis, the top twenty-five features were selected. Table 8 shows the selection and the respective chi-square score.

Table 8. Chi-square Test score for top 25 features

Feature	Score
Shipping conditon_S5	20141.482325
Incoterms1_FCA	11453.495296
ConsKey_Other	7841.823243
Ship to Country_NL	7346.591137
Ship to Country_Other	6833.260155
PCM Team Name_PCM Trainees (SN)	6747.607868
Incoterms1_Other	6733.648166
Shipping conditon_S3	5284.308643
PCM Team Name_PCM Team PL	4893.211226
Ship to Country_DE	4483.048223
MRP Profile_LS11	4276.606559
Ship to Country_BD	4144.534075
Ship to Country_SG	3796.442151
PCM Team Name_PCM Team Intraparts	2331.105508
PCM Team Name_PCM Team UK	2325.182608
Shipping type_Other	2033.708208
Ship to Country_CA	2012.911751
Ship to Country_US	1800.975668
FD Weekday_1	1650.857785
MRP Profile_LS20	1573.098976
Is C1 blocked	1057.848285
PCM Team Name_PCM USA & CA (USA)	1053.995403
FD Weekday_6	974.948056
Shipping conditon_S4	819.834757
Shipping conditon_S6	741.712674

Table 8 shows the features' chi-square test score compared with the Late delivery. Pearson's Chi-Square test score is used to compare two categorical variables and check if they are homogeneous with each other.

The Random Forest final algorithm scored 73,4% of accuracy in the training sample, 72,58% in the valid sample and 72,72% in the test database. Table 9 shows the classification report.

Table 9. Random Forest Classification report

	precision	recall	f1-score	support
0	0.73	0.72	0.73	43989
1	0.72	0.74	0.73	43803
accuracy			0.73	87792
macro avg	0.73	0.73	0.73	87792
weighted avg	0.73	0.73	0.73	87792

Table 9 shows that the precision and recall ranged from 72% to 74%. 0 is the deliveries on time, and 1 is the late deliveries.

Figure 49 shows the ROC curve of the model.

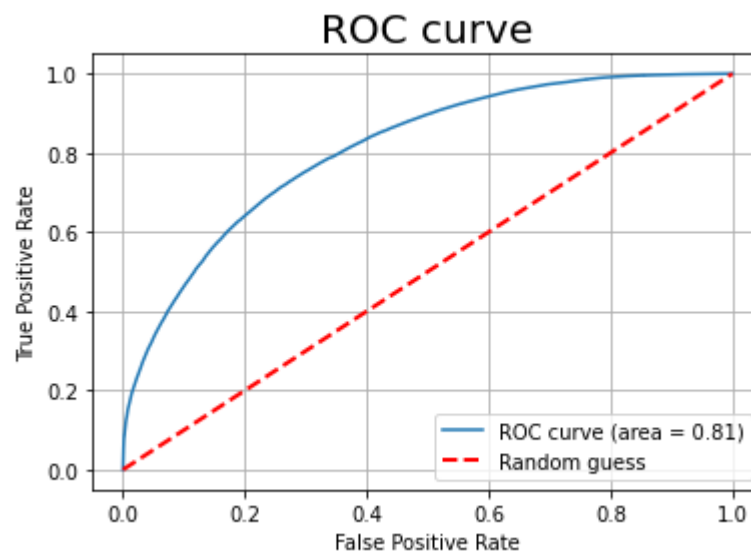


Figure 49. Random Forest - ROC Curve

Figure 49 shows that the ROC curve is 0,81. The closest the area is to 1 better is the model; therefore, a curve of 0,81 is a good result. The chart also shows the probability of getting the correct results by random guessing (red line).

The second-best model was the neural network.

5.5 Neural Network

The Neural Network algorithm scored 71,5% accuracy in the training sample, 71,01% in the validation sample and 70,1% in the test database. Table 10 shows the classification report.

Table 10. Neural Network Classification report

	precision	recall	f1-score	support
0	0.71	0.72	0.71	43851
1	0.71	0.70	0.71	43941
accuracy			0.71	87792
macro avg	0.71	0.71	0.71	87792
weighted avg	0.71	0.71	0.71	87792

Table 10 shows that the precision and recall ranged from 70% to 71%. 0 are the deliveries on time, and 1 is the late deliveries.

Figure 50 shows the ROC curve of the model.

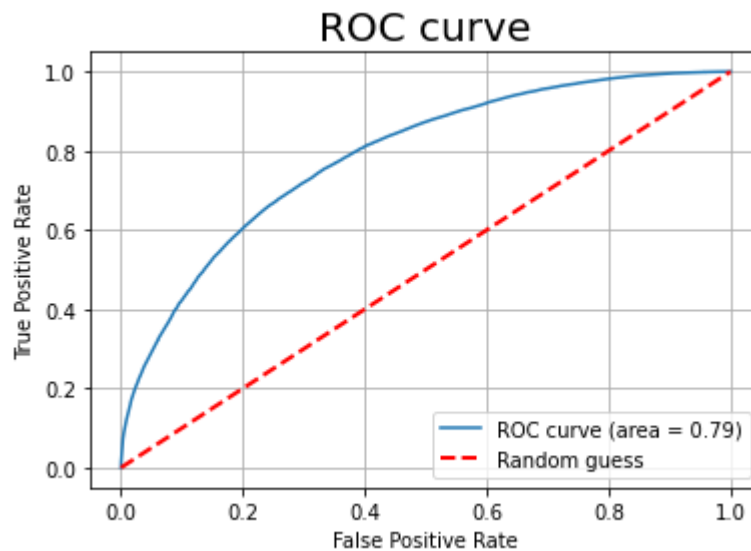


Figure 50. Neural Network - ROC Curve

Figure 50 shows that the ROC curve is 0,79. The chart also shows the probability of getting the correct results by random guessing (red line).

The last algorithm analysed was the bagging classifier.

5.6 Bagging Classifier

The Bagging Classifier final algorithm scored 73,41% accuracy in the training sample, 72,3% in the validation sample and 72,4% in the test database. Table 11 shows the classification report.

Table 11. Neural Network Classification report

	precision	recall	f1-score	support
0	0.73	0.71	0.72	43797
1	0.72	0.74	0.73	43995
accuracy			0.72	87792
macro avg	0.72	0.72	0.72	87792
weighted avg	0.72	0.72	0.72	87792

Table 11 shows that the precision and recall ranged from 72% to 74%. 0 represents the deliveries on time, and 1 illustrates the late deliveries.

Figure 50 shows the ROC curve of the model.

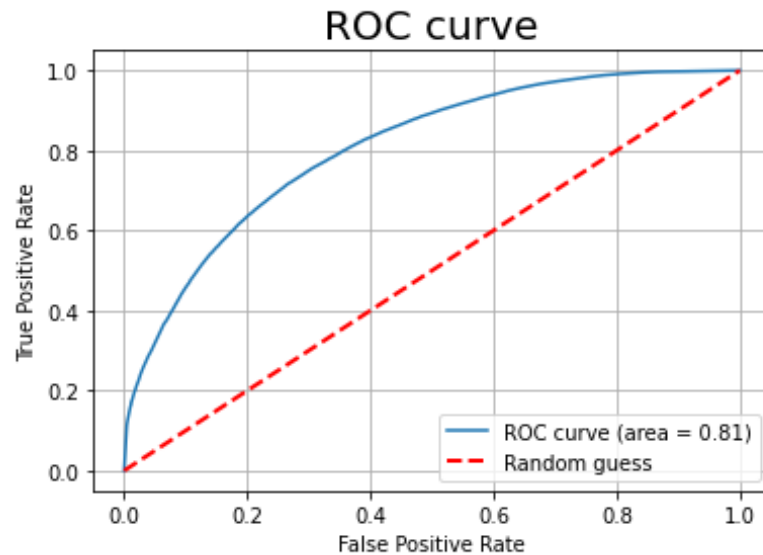


Figure 51. Bagging Classifier - ROC Curve

Figure 51 shows that the ROC curve is 0,81. The chart also shows the probability of getting the correct results by random guessing (red line).

The next step is to select the best model.

5.7 Model selection

After selecting the top three models from the baseline and conducting a deeper analysis, it is possible to conclude that the best model is the Random Forest.

Table 12 shows the comparison of the models in the test sample.

Table 12. Model Selection Performance

	Accuracy	ROC
Random Forest	73,39 %	80,99 %
Neural Network	71,07 %	78,71 %
Bagging Classifier	72,30 %	80,64 %

As seen in table 12, Random Forest is the model with the highest accuracy and ROC.

The next session will explore more validation metrics for the Random Forest model and the feedback stakeholder feedback.

6 Algorithm Validation

This section reports on the results of the validation stage and points to further developments to the initial Proposal. The final proposal and recommendations are presented at the end of this section.

6.1 Overview of the Validation Stage

The validation happened in three steps. First the statistical validation, where the ML validation metrics were used to validate the algorithm. The next step was to compare the results with the ML Canvas requirements presented in Session 4.4. Finally, the selected model was presented to the stakeholders, and their feedback was collected.

Figure 52 shows the summary of the ML validation metrics for the Random Forest Algorithm in the validation sample.

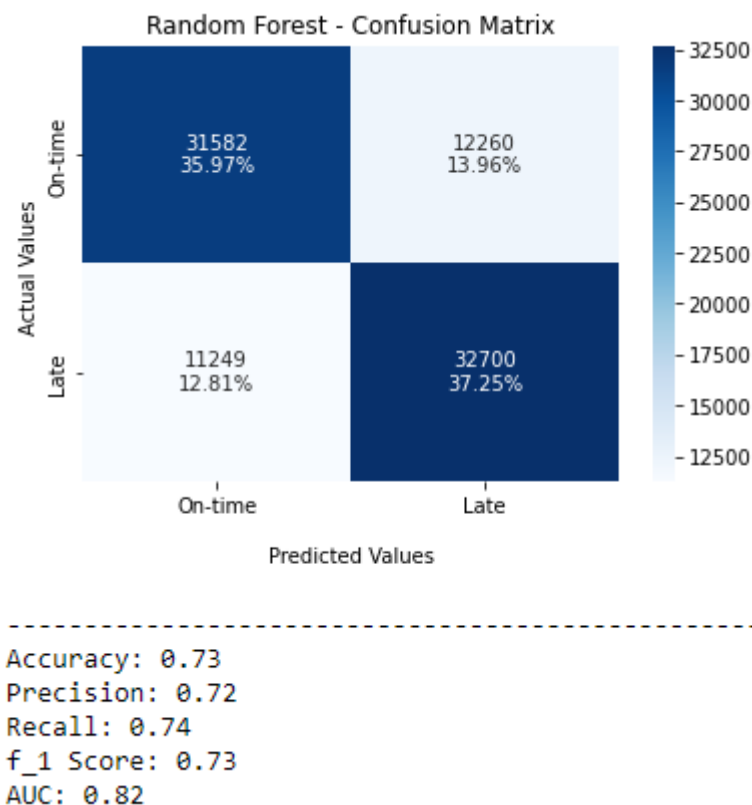


Figure 52. Random Forest – Validation Metrics

Figure 52 shows that 73% of the predictions were correct (Accuracy). The precision and recall were also similar. The precision indicates that for every 100 deliveries classified as late, 72 were correct. At the same time, the recall indicates that for each 100 actual late deliveries, 74 were correctly classified as late. The F1 metric is the harmonic mean between recall and precision. The F1 score was 73%. The last metric analysed was the area under the ROC curve (AUC). The model scored 82% for this metric. The AUC measures how well the model distinguishes between late and on-time deliveries, the closer to 1, the better the model.

According to the ML canvas presented in session 4.4, the stakeholders expected a minimum accuracy of 70%. Therefore, the algorithm attends to the requirements proposed for this thesis because the model achieved 73% of accuracy and similar scores in the other metrics.

Another requirement requested by the stakeholders was the creation of a dashboard for tracking the predictions. As a result, the dashboard was created and presented to the stakeholders in a meeting.

6.2 Developments to the Proposal

The collected feedback of the initial proposal is presented in Table 13.

Table 13. Expert suggestions for the Initial proposal.

<i>Proposal area</i>	<i>Requests</i>
1. Dashboard	<ul style="list-style-type: none"> a) The experts suggested to create a new graph to show the predictions by month b) The experts suggested to create filters based on teams to export the data
2. Algorithm	<ul style="list-style-type: none"> a) The experts suggested to automate all scripts in a same file.

As seen in Table 5, the experts suggested some changes in the algorithm and the dashboard. All these changes were made, and the new files were shared with them.

6.3 Final Proposal

The proposed algorithm is the Random Forest. The Random Forest attends to all the requirements, and the model gives the best accuracy compared to the other studied models. The algorithm was loaded into a Power BI file and shared with the case company.

Future recommendations include revalidation of the model from period to period because the dataset shows the presence of a trend once the frequency of late deliveries has increased over the years. In addition, as discussed in session 4, if the model's accuracy scores less than 70%, the data team should create a new algorithm. Another recommendation is to check the report usage metrics. For example, suppose the users are accessing the report with frequency. In that case, it is one indication that the algorithm is given business values, if users are not accessing the report, the data team have to collect feedback, and the report and algorithms should be revalidated. Finally, this tool is expected to help increase the number of deliveries on time, so the users must also follow the OTD KPI.

7 Conclusion

This section summarizes the key findings of this study from the objective definition to the outcome.

7.1 Executive Summary

The case company of this thesis is Wärtsilä Global Logistics, a Finnish corporation that operates the entire logistics chain of Wärtsilä Spare Parts, from order intake to customer delivery. WGLS operates in more than thirty countries globally.

WGLS uses On-Time Delivery (OTD) to measure service efficiency and improve customer satisfaction. OTD is the KPI that shows whether an organisation meets its goals regarding promised delivery times. However, WGLS has no advanced tools to predict OTD. Therefore, this study proposed to build a tool using ML to predict the OTD.

This thesis used an applied research family with mixed research methods. It mainly relied on the qualitative data gathered via internal document analysis, observations, and interviews, but it also used quantitative data for developing the ML model/algorithm.

The literature review covered the definition of machine learning, types of machine learning, machine learning building process, validation metrics, machine learning canvas and conceptual framework. The current state analysis evaluated the case company's current situation and the ML algorithm's requirements. The requirements discussed include the construction of a Power BI report where users should see a list of all predicted late deliveries. A binary classification was selected for the prediction task. The minimum accepted accuracy defined was 70%.

The data was extracted from EDW. The activities executed in the pre-processing stage includes data collection, outliers' exclusion, missing data handling, feature encoding, data imbalance analysis, feature scaling and analysis of the correlations.

After pre-processing, the data was loaded into seven different ML algorithms baseline: Random Forest, Neural Network, Bagging Classifier, Support Vector Machine, Gradient Boosting Machines, Logistic Regression and K-nearest neighbour. The model with the

best performance was the Random Forest with 72,6% accuracy. Neural Network and Bagging Classifier were the second best. Both had the same accuracy of 71,6%.

The three best algorithms were tuned by simulating different hyperparameters and feature selection. The best model after tuning was the Random Forest.

The final model was validated first by using ML metrics. The model predicted corrected 73% of data (Accuracy). The precision indicates that for every 100 deliveries classified as late, 72 were correct. At the same time, the recall indicates that for each 100 actual late deliveries, 74 were correctly classified as late. The F1 metric is the harmonic mean between recall and precision. The F1 score was 73%. The last metric analysed was the area under the ROC curve (AUC). The model scored 82% for this metric. The AUC measures how well the model distinguishes between late and on-time deliveries, the closer to 1, the better the model.

The algorithm attended to the requirements proposed for this thesis because the model achieved 73% accuracy and similar scores in the other metrics, while the minimum requirement was 70% accuracy. The algorithm was loaded into a Power BI file and shared with the case company.

It is expected that this tool will reduce the number of late deliveries. The late delivery is a problem for both customers and operations. Late deliveries lead to more claims and costs for the business. Therefore, on-time delivery service is a crucial factor in maintaining high customer retention rates and keeping customers satisfied.

7.2 Thesis Evaluation / Research Quality Criteria for This Thesis

The objective of the thesis was to create Machine Learning Algorithm to predict the OTD. The outcome of this study is the ML Algorithm and a Power BI report to track the predictions. Therefore, the outcome met the study's objective.

The EDW was the primary source of data for this study. In addition to the company's data warehouse, meeting notes were collected to ensure the study's validity. The dataset consisted of enough amount of data. The dataset had 17 features (columns) and 2,16 million observations (rows).

The dataset was divided into three samples: one for training, the second for tests and the third for validation. This splitting of samples increases the probability that the chosen model accurately reflects actual data.

The selected algorithm performed similarly in all three samples, which indicates that the model will probably achieve the same performance when new data is added to the dataset.

The ML Algorithm scored good results from the statistical perspective and also attended to the stakeholder's requirements. In addition to algorithm development, a Power BI report was created. The objective was to display the results of the predictions. The validation of the report was done by meeting with stakeholders.

The main challenge faced in this study was the complexity of the topic and the number of different tools and computer languages used. Usually, the whole process includes, in addition to the stakeholders, a data engineer, a data scientist and sometimes a machine learning engineer and a data analyst. However, in this thesis, all the functions were executed by the author of this thesis, from writing the SQL code and extracting the data from SAP to creating the ML Algorithm, reporting, and training the team.

Future recommendations include revalidation of the model from period to period. Another recommendation is to check the report usage metrics, if users are not accessing the report, the data team must collect feedback, and the report and algorithms should be revalidated. Finally, this tool is expected to help increase the number of deliveries on-time, so the users must also track the OTD KPI in long term.

References

- Agrawal, Samarth (2021). How to split data into three sets (train, validation, and test) And why? Available from: <https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c> (Accessed 28 February 2022)
- Auger SD, Jacobs BM, Dobson R, et al. Big data, machine learning and artificial intelligence: a neurologist's guide. *Practical Neurology* 2021;21:4-11. Available from: <https://pn.bmj.com/content/21/1/4> (Accessed 10 August 2021)
- Badr, W. (2019) Having an Imbalanced Dataset? Here Is How You Can Fix It. Available from: <https://towardsdatascience.com/having-an-imbalanced-dataset-here-is-how-you-can-solve-it-1640568947eb> (Accessed 28 February 2022)
- Brown, S. Machine learning, explained. MIT Management Sloan School. Available from: <https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained> (Accessed 28 February 2022)
- Brannick, M.T. Logistic Regression. Available from: <http://faculty.cas.usf.edu/mbrannick/regression/Logistic.html> (Accessed 22 May 2022)
- Brownlee, J. (2018). How to Remove Outliers for Machine Learning. Available from: <https://machinelearningmastery.com/how-to-use-statistics-to-identify-outliers-in-data/> (Accessed 28 February 2022)
- Christopher, Antony. (2021). Random Forest: <https://medium.com/analytics-vidhya/random-forest-4a2981aab4f7> . (Accessed 28 February 2022)
- Cormen, T., Leiserson C., Rivest R. & Stein C. 2003. *Introduction to Algorithms*. Second Edition. MIT Press.
- Dalin-Kaptzan, Zahava. On Time Delivery: Understanding, Calculating and Improving Your KPI. *Bringg*. Available from: <https://www.bringg.com/blog/delivery/on-time-delivery/#:~:text=On%20time%20delivery%2C%20or%20OTD,performance%20and%20maintaining%20customer%20satisfaction.> (Accessed 28 February 2022)
- DataTechNotes. How to Create ROC Curve in Python: <https://www.datatechnotes.com/2019/11/how-to-create-roc-curve-in-python.html> . (Accessed 28 February 2022)
- Dean, J. & Hassibi, K. 2014. *Big Data, Data Mining, and Machine Learning: Value Creation for Business Leaders and Practitioners*. John Wiley & Sons, Incorporated.

- Di, W., Jianing Wei, Wei Di & Wei, J. 2018. Deep Learning Essentials. Packt Publishing, Limited.
- Dorard, Louis 2016. From Data to AI with the Machine Learning Canvas. <https://medium.com/louis-dorard/from-data-to-ai-with-the-machine-learning-canvas-part-i-d171b867b047> (Accessed 01 May 2022)
- Dorard, Louis 2021. Machine Learning Canvas v1.1: change log. Available from: <https://www.ownml.co/blog/machine-learning-canvas-v11-change-log> (Accessed 11 September 2022)
- Géron, A. 2017. Hands-on machine learning with Scikit-Learn and TensorFlow: Concepts, tools, and techniques to build intelligent systems. First edition. Sebastopol, CA: O'Reilly Media, Inc.
- Gollapudi, S. & Laxmikanth, V. 2016. Practical Machine Learning. Packt Publishing.
- Foote, K. D. (2019). A Brief History of Machine Learning. Available from: <https://www.dataversity.net/a-brief-history-of-machine-learning/> (Accessed 5 July 2021).
- Fuatakal. Visualization and Interpretation of the Confusion Matrix. Available from: <https://www.thegoodclass.net/post/visualization-and-interpretation-of-confusion-matrix> (Accessed 15 October 2021).
- Kumar, S. (2020). 7 Ways to Handle Missing Values in Machine Learning. Available from: <https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e> (Accessed 15 February 2022).
- Kananen, J. 2013. Design research (applied action research) as thesis research: *A practical guide for thesis research*. Jyväskylä: Jyväskylän ammattikorkeakoulu.
- Kamiri, J. (2021). Research Methods in Machine Learning: A Content Analysis. Available from: <https://www.ijcit.com/index.php/ijcit/article/view/79/31> (Accessed 15 February 2022).
- Ladkar, S. (2020). Most Popular Machine Learning Performance Metrics-Part 1. Available from: <https://medium.com/@ladkarsamisha123/most-popular-machine-learning-performance-metrics-part-1-ab7189dce555> (Accessed 15 February 2022).
- Leavy, P. Research (2017) Design: Quantitative, Qualitative, Mixed Methods, Arts-Based, and Community-Based Participatory Research Approaches. Guilford Press, The.
- Rijmenam, M.V. (2019). 7 Steps to Machine Learning: How to Prepare for an Automated Future. Available from: <https://medium.com/dataseries/7-steps-to-machine->

- [learning-how-to-prepare-for-an-automated-future-78c7918cb35d](#) (Accessed 10 February 2022).
- Roy, Baijyanta. (2020). All about Feature Scaling. Available from: <https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35> (Accessed 10 February 2022).
- Minaie, Nick. (2019). Evaluating Machine Learning Classification Problems in Python: 6+1 Metrics That Matter. Available from: <https://towardsdatascience.com/evaluating-machine-learning-classification-problems-in-python-5-1-metrics-that-matter-792c6faddf5> (Accessed 10 October 2022).
- Myriantous, Giorgios. (2021). How to Split a Dataset into Training and Testing Sets with Python. Available from: <https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35> (Accessed 10 February 2022).
- Natlat (2016). Implement your own k-nearest neighbour algorithm using Python. Available from: <https://cambridgecoding.wordpress.com/2016/01/16/machine-learning-under-the-hood-writing-your-own-k-nearest-neighbour-algorithm/> (Accessed 13 August 2021).
- Navlani (2018). KNN Classification Tutorial using Scikit-learn. Available from: <https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn> (Accessed 13 March 2022).
- Math is fun (2022). Normal Distribution. Available from: <https://www.mathsisfun.com/data/standard-normal-distribution.html> (Accessed 18 April 2022).
- Mitchell, T.M. (1997). Machine Learning. [McGraw-Hill].
- Molnar, C. (2022). Interpretable Machine Learning: A Guide for Making Black Box Models Explainable (2nd ed.). Available from: christophm.github.io/interpretable-ml-book/ (Accessed 11 September 2022)
- Samuel, Arthur L. (1959). Some Studies in Machine Learning Using the Game of Checkers. Available from: <https://www.cs.virginia.edu/~evans/greatworks/samuel1959.pdf> (Accessed 16 May 2022).
- Saunders, M., Lewis, P. & Thornhill, A. 2007. Research methods for business students. 4th ed. Harlow: Prentice Hall.
- SAP (2022). Delivery Plants. Available from: https://help.sap.com/docs/SAP_ERP_SPV/a428aae377ba4a1199c3ecc8b7f5f33d/168bc95360267214e10000000a174cb4.html?version=6.06.24&locale=en-US (Accessed 15 May 2022).

- Skilton, M. & Hovsepian, F. 2018. The 4th industrial revolution: Responding to the impact of artificial intelligence on business. Cham, Switzerland: Palgrave Macmillan.
- Shin, T. (2020). How To Prepare Your Data for Your Machine Learning Model. Available from: <https://towardsdatascience.com/how-to-prepare-your-data-for-your-machine-learning-model-b4c9fd4e7ea> (Accessed 15 February 2020).
- Tripathi, H.(2019). Different Type of Feature Engineering Encoding Techniques for Categorical Variable Encoding. Available from: <https://medium.com/analytics-vidhya/different-type-of-feature-engineering-encoding-techniques-for-categorical-variable-encoding-214363a016fb> (Accessed 15 February 2022).
- Verdhan. 2020. Supervised Learning with Python: Concepts and Practical Implementation Using Python. Apress
- Wärtsilä. (2019). Wärtsilä website. About. Available from: www.wartsila.com (Accessed 07 December 2019).
- Wilson, J. and Sachdev, S. and Alter, A. (2016). How Companies Are Using Machine Learning to Get Faster and More Efficient Harvard Business Review. Available from: <https://hbr.org/2016/05/how-companies-are-using-machine-learning-to-get-faster-and-more-efficient> (Accessed 25 October 2019).
- Yiu, Tony (2019). Understanding Random Forest. Available from: <https://towardsdatascience.com/understanding-random-forest-58381e0602d2#:~:text=The%20random%20forest%20is%20a,that%20of%20any%20individual%20tree>. (Accessed 28 February 2022).

SQL Script to generate dataset

```

WITH pcm_team
  AS (SELECT orderdocument,
            CASE
              WHEN ERCC.coctr_level5_code IS NOT NULL THEN er_org_unit
              WHEN SRCC.coctr_level5_code IS NOT NULL THEN sr_org_unit
              WHEN CBCC.coctr_level5_code IS NOT NULL THEN
                created_by_org_unit
              ELSE NULL
            END AS PCM_Team_ID
  FROM wbr_ods.v_tf_coordinators C
  LEFT JOIN wbr_ods.v_ts_vbak
    ON orderdocument = vbeln
  ---Check if is PCM Team
  LEFT JOIN wbr_ods.v_td_cost_center_custom ERCC
    ON er_cc = ERCC.coctr_id
    AND ERCC.coctr_level11_code = 'ALTER_BUS'
    AND ERCC.coctr_level5_code = '5040-113'
  LEFT JOIN wbr_ods.v_td_cost_center_custom CBCC
    ON created_by_cc = CBCC.coctr_id
    AND CBCC.coctr_level11_code = 'ALTER_BUS'
    AND CBCC.coctr_level5_code = '5040-113'
  LEFT JOIN wbr_ods.v_td_cost_center_custom SRCC
    ON sr_cc = SRCC.coctr_id
    AND SRCC.coctr_level11_code = 'ALTER_BUS'
    AND SRCC.coctr_level5_code = '5040-113'
  WHERE ( SRCC.coctr_level5_code IS NOT NULL
        OR ERCC.coctr_level5_code IS NOT NULL
        OR CBCC.coctr_level5_code IS NOT NULL ) --
  Remove non PCM Team
  -- AND ORDERDOCUMENT like '%2841466'
)
SELECT VBAP.vbeln AS "Order",
       VBAP.posnr AS "Item",
       VBAK.kunnr AS "Sold to",
       CASE
         WHEN VBAK.zzconskey LIKE 'A%' THEN 'ADHOC'
         ELSE VBAK.zzconskey
       END AS "ConsKey",
       To_date(Min(VBAK.erdat), 'yyyymmdd') AS "Order Creation Date",
       Min(VBPA_PARVW.land1) AS "Ship to Country",
       Min(MARC_dispr) AS "MRP Profile",
       Min(VBKD.vsart) AS "Shipping type",
       Min(VBAK.vsbed) AS "Shipping conditon",
       Min(VBKD.inco1) AS "Incoterms1",
       Min(pcm_team_id) AS "PCM Team",
       CASE
         WHEN Max(OT.combi_item_otd_late_work_days) IS NULL THEN -1
         WHEN Max(OT.combi_item_otd_late_work_days) > 0 THEN 0
         ELSE 1
       END AS "Is OTD",
       Min(otd_datetime) AS POD,
       VBAP.zztplnumber AS "Installation ID",
       VBAP.matnr AS "Material Number",
       VBAP.werks AS "Plant",
       VBAP.vstel AS "Shipping Point",
       VBAP.lprio AS "Delivery Priority",
       To_date(Min(VBEP.edatu), 'yyyymmdd') AS
       "Schedule line delivery date",

```

```

CASE
  WHEN VBAP.gewei IS NULL THEN NULL
  WHEN VBAP.gewei = 'KG' THEN VBAP.brgew
  WHEN VBAP.gewei = 'LB' THEN VBAP.brgew / 2.20462
  WHEN VBAP.gewei IN ( 'TO', 'TON' ) THEN VBAP.brgew * 1000
  WHEN VBAP.gewei = 'G' THEN VBAP.brgew / 1000
  WHEN VBAP.gewei = 'KT' THEN VBAP.brgew * 1000000
  WHEN VBAP.gewei = 'MG' THEN VBAP.brgew / 1000000
END
CASE
  WHEN appl_category IS NOT NULL
  AND VBAP.zzrefeng IS NOT NULL THEN appl_category
  || ' '
  || VBAP.zzrefeng
  WHEN VBAP.zzrefeng IS NULL THEN 'Missing PRT'
  ELSE VBAP.zzrefeng
END
AS "Prod Ref. Type",
VBAP.zztplnumber AS "Installation",
To_date(Min(VBEP_FD.fd), 'yyyymmdd') AS "First Date",
To_date(Max(mat_available_date), 'yyyymmdd') AS
"Material Availability Date",
reschedule_times AS "Reschedule Times"
FROM wbr_ods.v_ts_vbap VBAP
LEFT JOIN wbr_ods.v_ts_vbak VBAK
  ON VBAP.vbeln = VBAK.vbeln
LEFT JOIN wbr_ods.v_ts_vbpa VBPA_PARVW
  ON VBPA_PARVW.vbeln = VBAP.vbeln
  AND VBPA_PARVW.parvw LIKE 'WE'
  AND VBPA_PARVW.posnr = '000000'
LEFT JOIN wbr_ods.ts_marc MARC
  ON ekgrp IS NOT NULL
  AND MARC.matnr = VBAP.matnr
  AND VBAP.werks = MARC.werks
LEFT JOIN (SELECT vbeln,
  Max(inco1) INCO1,
  Max(inco2) INCO2,
  Max(vsart) AS VSART,
  Max(zterm) AS ZTERM
  FROM wbr_ods.v_ts_vbkd
  GROUP BY vbeln) VBKD
  ON VBKD.vbeln = VBAP.vbeln
LEFT JOIN pcm_team
  ON pcm_team.orderdocument = VBAP.vbeln
LEFT JOIN wbr_ods.v_tf_ot_delivery OT
  ON OT.order_doc = VBAP.vbeln
  AND OT.order_item = VBAP.posnr
--Keep only Confirmed documents
INNER JOIN (SELECT sales_doc_number VBELN,
  sales_doc_item POSNR,
  Max(schedule_line_delivery_date) AS EDATU,
  Max(mat_available_date) AS MAT_AVAILABLE_DA
  FROM wbr_ods.v_ts_vbep_mod
  WHERE confirmed_quantity > 0
  AND met_chg_ind <> 'D'
  GROUP BY sales_doc_number,
  sales_doc_item) VBEP
  ON VBAP.vbeln = VBEP.vbeln
  AND VBAP.posnr = VBEP.posnr
LEFT JOIN wbr_ods.v_td_installation I
  ON VBAP.zztplnumber = I.inst_id
LEFT JOIN (SELECT sales_doc_number VBELN,
  sales_doc_item POSNR,
  Min(schedule_line_delivery_date) AS FD,

```

```

                Count(*)
                AS RESCHEDULE_TIMES
            FROM wbr_ods.v_ts_vbep_mod
            WHERE met_chg_ind <> 'D'
            GROUP BY sales_doc_number,
                    sales_doc_item) VBEP_FD
            ON VBEP_FD.vbeln = VBAP.vbeln
            AND VBEP_FD.posnr = VBAP.posnr
-- LEFT JOIN WBR_ODS.V_TS_LIPS LIPS
-- ON VBAP.VBELN = LIPS.VGBEL AND VBAP.POSNR = LIPS.VGPOS
WHERE VBAP.werks LIKE '%GLS%'
      AND ( VBAK.ERDAT >= 20170101
            OR otd_datetime >= '31-DEC-17 01.01.01.000000001' )
      AND VBAK.met_chg_ind <> 'D'
      AND VBAK.auart IN ( 'ZSAS', 'ZSI2', 'ZSIC', 'ZTFR',
                        'ZTOC', 'ZTOR', 'ZTOS' ) -- Only Orders
GROUP BY VBAP.vbeln,
         VBAP.posnr,
         VBAP.zztplnumber,
         VBAP.matnr,
         VBAK.kunnr,
         VBAK.zzconskey,
         VBAP.werks,
         VBAP.vstel,
         VBAP.lprio,
         VBAP.gewei,
         VBAP.brgew,
         VBAP.zzrefeng,
         appl_category,
         VBAP.zztplnumber,
         reschedule_times

```

Data pre-processing python script

OTD Data collection and data pre-processing

Set the libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
import scipy.stats
from sklearn.preprocessing import MinMaxScaler
#Removing warnings:
pd.options.mode.chained_assignment = None
```

1. Collect the Data

General settings

```
#Set the file location
Mainfolder = 'G:/My Drive/Thesis/raw'
dataframe = Mainfolder + '/OTD Study - Raw data.csv'

#Set table size
pd.set_option('display.max_rows', 99999)
pd.set_option('display.max_columns', 9999)
pd.set_option('display.width', 1000)
```

Read data

```
df = pd.read_csv(dataframe, low_memory=False)
```

Convert date fields

```
df['POD'] = df['POD'].str[:10]
df['Combined Delivery date'] = df['Combined Delivery date'].str[:10]
df['Schedule line delivery date'] = df['Schedule line delivery date'].str[:10]
df['First Date'] = df['First Date'].str[:10]
```

```
df['POD'] = pd.to_datetime(df['POD'])
df['Combined Delivery date'] = pd.to_datetime(df['Combined Delivery date'])
df['Schedule line delivery date'] = pd.to_datetime(df['Schedule line delivery date'])
df['First Date'] = pd.to_datetime(df['First Date'])
```

Convert date fields into date format

```
df.info()
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2162689 entries, 0 to 2162688
Data columns (total 17 columns):
 #   Column                                Dtype
---  -
 0   Key                                    int64
 1   PCM Team                               int64
 2   Ship to Country                       object
 3   MRP Profile                           object
 4   Shipping type                          int64
 5   Shipping conditon                     object
 6   Incoterms1                            object
 7   C1 block reason                       object
 8   T Task                                 int64
 9   ConsKey                                object
10   Shipping Point                        object
11   Is Late                                int64
12   POD                                    datetime64[ns]
13   Combined Delivery date                datetime64[ns]
14   Schedule line delivery date           datetime64[ns]
15   Lead Time (OCD x FDD)                 float64
16   First Date                            datetime64[ns]
dtypes: datetime64[ns](4), float64(1), int64(5), object(7)
memory usage: 280.5+ MB

```

Check cardinality

```
df.apply(pd.Series.nunique)
```

```

Key                2162689
PCM Team           44
Ship to Country    198
MRP Profile        11
Shipping type      22
Shipping conditon  10
Incoterms1        16
C1 block reason    15
T Task             2
ConsKey            52
Shipping Point     2
Is Late            2
POD                1200
Combined Delivery date 1588
Schedule line delivery date 1445
Lead Time (OCD x FDD) 1120
First Date         1521
dtype: int64

```

1.1 POD

```

df['POD_day'] = df['POD'].apply(lambda x: x.strftime('%d'))
df['POD_Month'] = df['POD'].apply(lambda x: x.strftime('%m'))

```

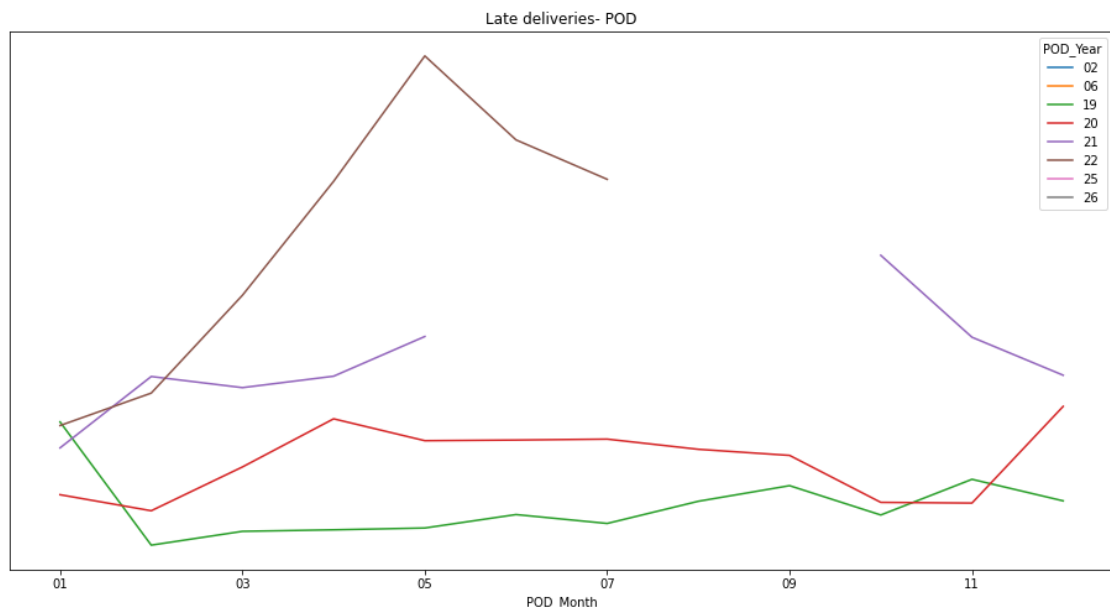
```

df['POD_Year'] = df['POD'].apply(lambda x: x.strftime('%y'))
df['POD_Month_Year'] = df['POD'].apply(lambda x: x.strftime('%Y-%m'))
df['POD_Weekday'] = df['POD'].apply(lambda x: x.strftime('%a'))

df_temp = df.pivot_table(index=['POD_Month_Year', 'POD_Month', 'POD_Year'],
                          columns='Is Late', values='Key',
                          aggfunc='count').reset_index()
df_temp['rel_freq'] = df_temp[1]/(df_temp[0]+df_temp[1])

df_temp.pivot(index='POD_Month', columns='POD_Year', values='rel_freq')
).plot(title='Late deliveries- POD', figsize=(16,8))
plt.tick_params(labelleft=False, left=False) #Remove sensitive data

```



Settings for boxplot

```

columns_order = ['Sat', 'Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri']

df_temp1 = df_temp1.rename(columns={"POD_Month_Year": "Month_Year", "POD_Weekday": "Weekday"})
df_temp1['class'] = 'POD'

```

Delete trasformed date columns

```

del df['POD']
del df['POD_Month_Year']

```

Delete columns not used in the ML model

```

del df['POD_Weekday']
del df['POD_Month']
del df['POD_Year']

```

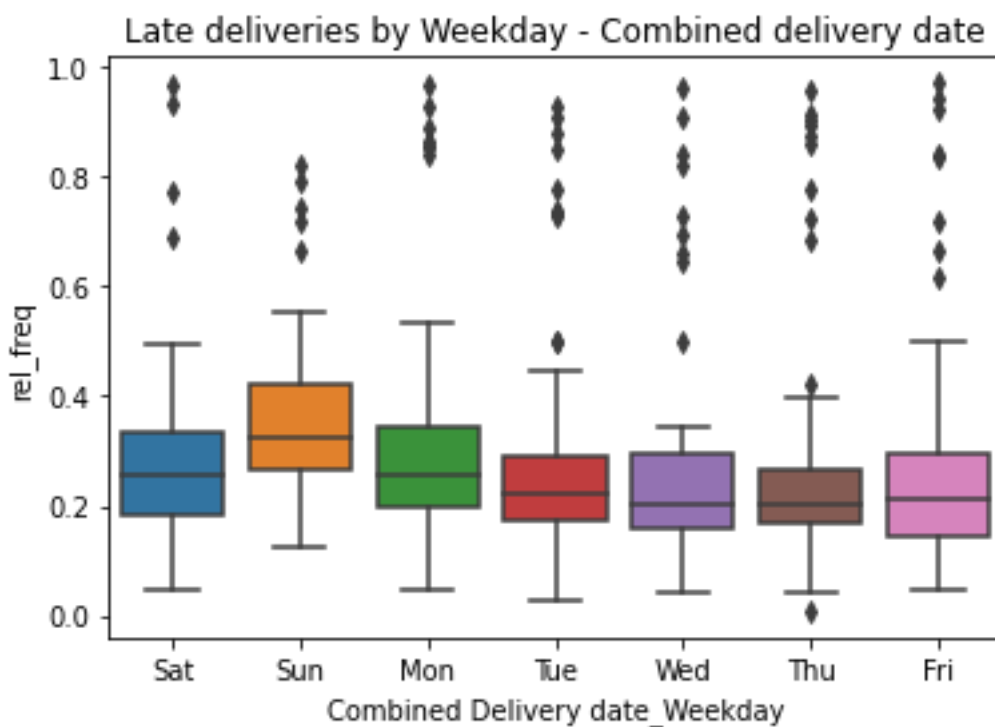

1.2 Combined delivery date

```
df['Combined Delivery date_Month'] = df['Combined Delivery date'].apply(
lambda x: x.strftime('%m'))
df['Combined Delivery date_Year'] = df['Combined Delivery date'].apply(
lambda x: x.strftime('%y'))
df['Combined Delivery date_Month_Year'] = df['Combined Delivery date']
.apply(lambda x: x.strftime('%Y-%m'))
df['Combined Delivery date_Weekday'] = df['Combined Delivery date'].ap
ply(lambda x: x.strftime('%a'))

df_temp2 = df.pivot_table(index=['Combined Delivery date_Month_Year',
'Combined Delivery date_Weekday'], columns='Is Late', values='Key',
aggfunc='count').reset_index()
df_temp2['rel_freq'] = df_temp2[1]/(df_temp2[0]+df_temp2[1])

sns.boxplot(x = df_temp2['Combined Delivery date_Weekday'], y = df_tem
p2['rel_freq'],
order = columns_order).set(title='Late deliveries by Weekda
y - Combined delivery date')

[Text(0.5, 1.0, 'Late deliveries by Weekday - Combined delivery date')
]
```



```
df_temp2 = df_temp2.rename(columns={"Combined Delivery date_Month_Year": "Month_Year", "Combined Delivery date_Weekday": "Weekday"})
df_temp2['class'] = 'Combined Delivery date'
```

Delete trasformed date columns

```
del df['Combined Delivery date']
del df['Combined Delivery date_Month']
```

Delete columns not used in the ML model

```
del df['Combined Delivery date_Year']
del df['Combined Delivery date_Month_Year']
del df['Combined Delivery date_Weekday']
```

1.3 Schedule Line Delivery date

```
df['Schedule line delivery date_Month'] = df['Schedule line delivery date'].apply(lambda x: x.strftime('%m'))
df['Schedule line delivery date_Year'] = df['Schedule line delivery date'].apply(lambda x: x.strftime('%y'))
df['Schedule line delivery date_Month_Year'] = df['Schedule line delivery date'].apply(lambda x: x.strftime('%Y-%m'))
df['Schedule line delivery date_Weekday'] = df['Schedule line delivery date'].apply(lambda x: x.strftime('%a'))
```

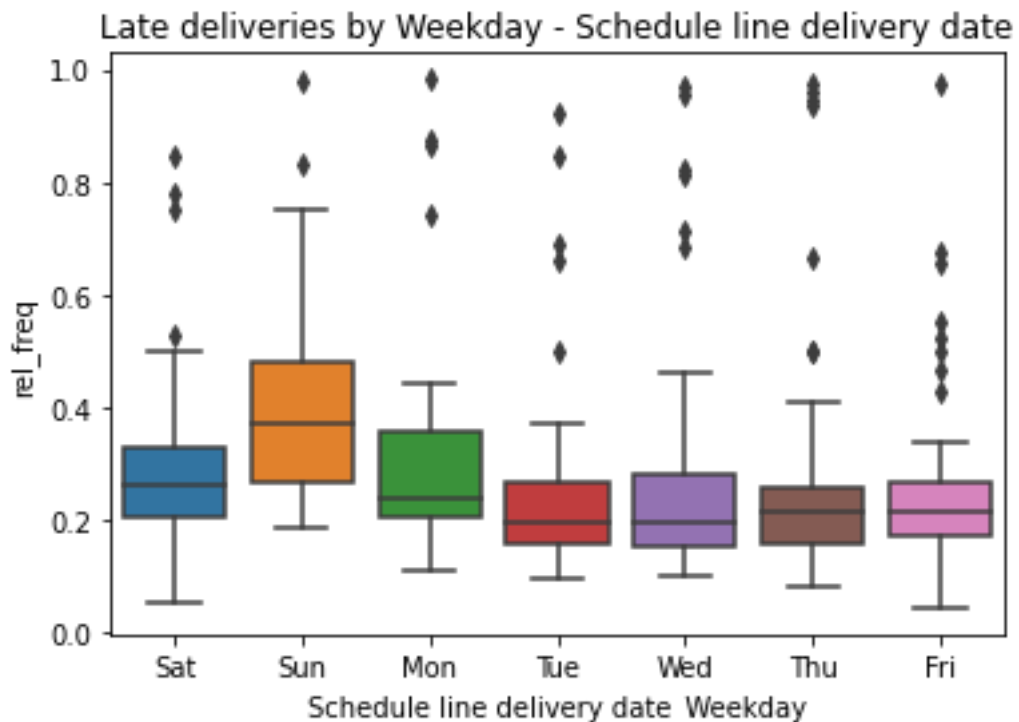
```
df_temp3 = df.pivot_table(index=['Schedule line delivery date_Month_Year', 'Schedule line delivery date_Weekday'], columns='Is Late', values='Key',
```

```
aggfunc='count').reset_index()
```

```
df_temp3['rel_freq'] = df_temp3[1]/(df_temp3[0]+df_temp3[1])
```

```
sns.boxplot(x = df_temp3['Schedule line delivery date_Weekday'], y = df_temp3['rel_freq'],
            order = columns_order).set(title='Late deliveries by Weekday - Schedule line delivery date')
```

```
[Text(0.5, 1.0, 'Late deliveries by Weekday - Schedule line delivery date')]
```



```
df_temp3 = df_temp3.rename(columns={"Schedule line delivery date_Month
_Year": "Month_Year", "Schedule line delivery date_Weekday": "Weekday"
})
df_temp3['class'] = 'Schedule line delivery date'
```

Delete trasformed date columns

```
del df['Schedule line delivery date']
del df['Schedule line delivery date_Month']
```

Delete columns not used in the ML model

```
del df['Schedule line delivery date_Year']
del df['Schedule line delivery date_Month_Year']
del df['Schedule line delivery date_Weekday']
```

1.4 First Date

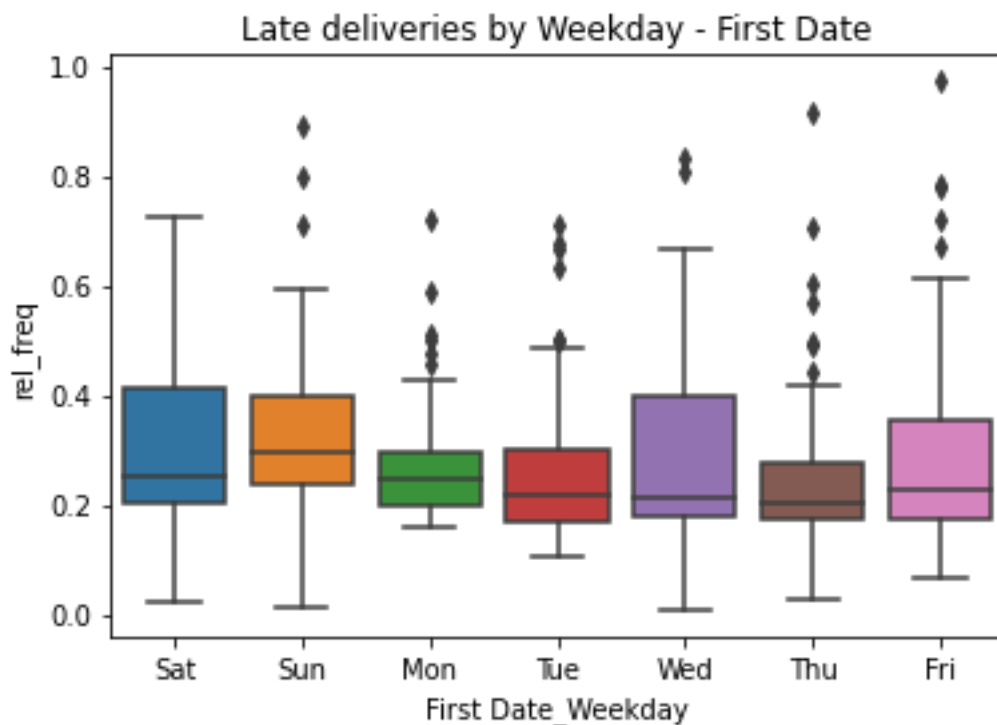
```
df['First Date_Month'] = df['First Date'].apply(lambda x: x.strftime('%m'))
df['First Date_Year'] = df['First Date'].apply(lambda x: x.strftime('%y'))
df['First Date_Month_Year'] = df['First Date'].apply(lambda x: x.strftime('%Y-%m'))
df['First Date_Weekday'] = df['First Date'].apply(lambda x: x.strftime('%a'))

df_temp4 = df.pivot_table(index=['First Date_Month_Year', 'First Date_Weekday'],
columns='Is Late', values='Key',
aggfunc='count').reset_index()
```

```
df_temp4['rel_freq'] = df_temp4[1]/(df_temp4[0]+df_temp4[1])

sns.boxplot(x = df_temp4['First Date_Weekday'], y = df_temp4['rel_freq'],
            order = columns_order).set(title='Late deliveries by Weekday - First Date')

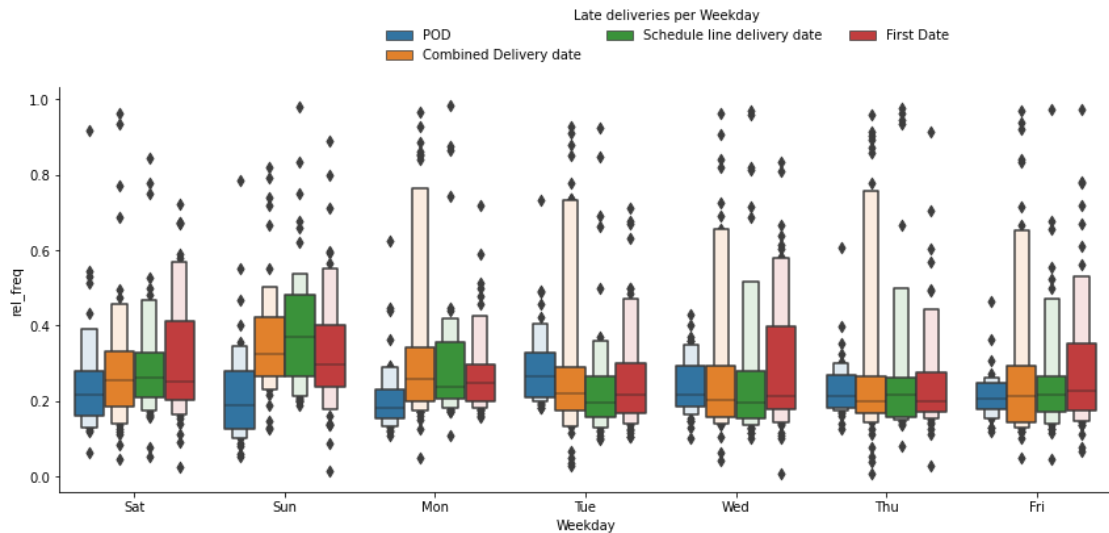
[Text(0.5, 1.0, 'Late deliveries by Weekday - First Date')]
```



```
df_temp4 = df_temp4.rename(columns={"First Date_Month_Year": "Month_Year", "First Date_Weekday": "Weekday"})
df_temp4['class'] = 'First Date'

df_temp = pd.concat([df_temp1, df_temp2, df_temp3, df_temp4])
ax = sns.catplot(data=df_temp, x='Weekday', y='rel_freq', order = columns_order, hue='class', kind="boxen", aspect=20/8.27)
sns.move_legend(ax, "lower center", bbox_to_anchor=(.5, 1), ncol=3, title='Late deliveries per Weekday', frameon=False)

plt.show()
```



```
df['FD Weekday'] = df['First Date'].apply(lambda x: x.strftime('%w'))
```

Delete transformed date columns

```
del df['First Date']
del df['First Date_Month']
```

Delete columns not used in the ML model

```
del df['First Date_Year']
del df['First Date_Month_Year']
del df['First Date_Weekday']
```

1.5 Ship to Country

```
feature_of_analysis = 'Ship to Country'
```

```
#Call the function with a threshold of 75%
```

```
transformed_column,new_category_list=cumulatively_categorise(df[feature_of_analysis],0.75,return_categories_list=True)
```

```
df[feature_of_analysis] = transformed_column
```

```
df[feature_of_analysis].unique()
```

```
array(['Other', 'FI', 'DE', 'NO', 'DK', 'NL', 'GB', 'BE', 'ES', 'FR',
      'IT', 'SG', 'US', 'JP', 'KR', 'CN', 'AE', 'BR', 'AU', 'ID', 'CA',
      'BD'], dtype=object)
```

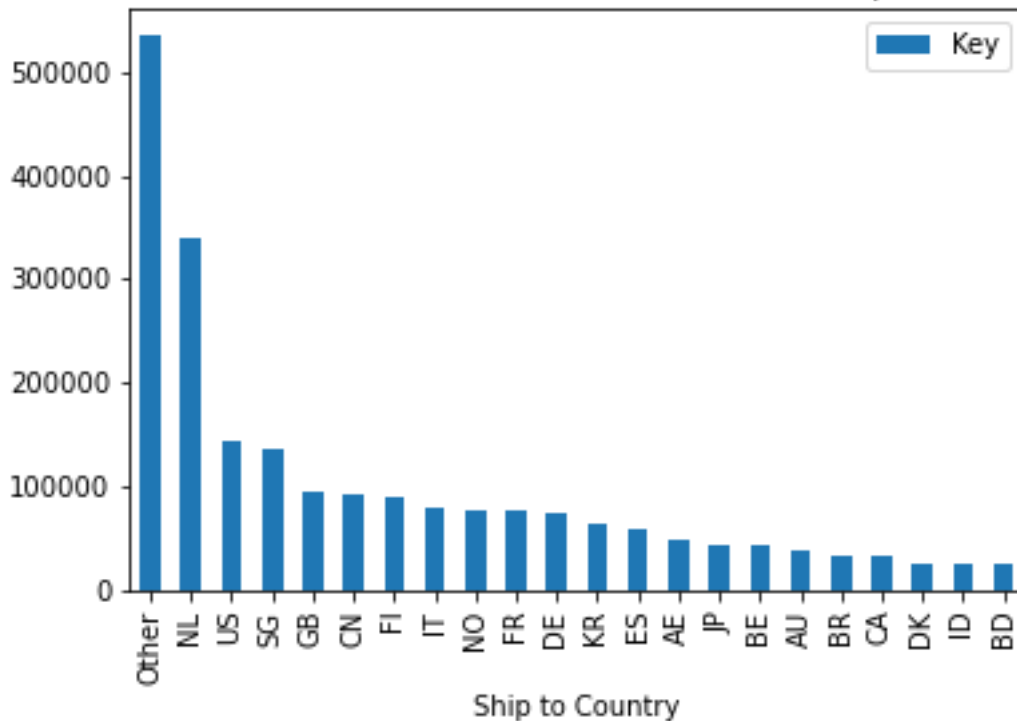
```
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(ascending=False).reset_index()
```

```
df_temp.set_index(feature_of_analysis).sort_values(
    by=['Key'],ascending=False).head(200).plot(kind='bar', title='TOP
10 amount of deliveries in the sample')
```

```
print("There are", len(df_temp[df_temp['Key'] < 50]), "countries with  
less than 50 deliveries in the sample." );
```

There are 0 countries with less than 50 deliveries in the sample.

TOP 10 amount of deliveries in the sample



1.6 PCM Team

```
#Set the file location
```

```
Mainfolder = 'G:/My Drive/Thesis/raw'
```

```
dataframe_PcmTeam = Mainfolder + '/PCM Team.csv'
```

```
columns_to_read = ['ORG_NAME', 'PCM Team Name' ]
```

```
pcm_team = pd.read_csv(dataframe_PcmTeam, usecols=columns_to_read)
```

```
pcm_team.columns = ['PCM Team', 'PCM Team Name']
```

```
pcm_team.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 82 entries, 0 to 81
```

```
Data columns (total 2 columns):
```

#	Column	Non-Null Count	Dtype
0	PCM Team	82 non-null	object
1	PCM Team Name	81 non-null	object

```
dtypes: object(2)
```

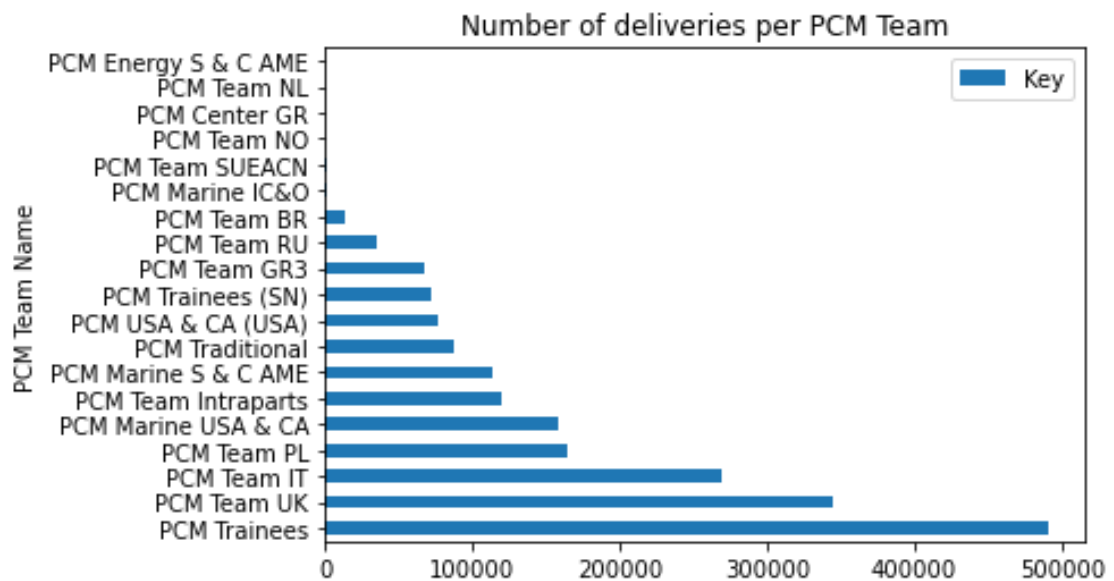
```
memory usage: 1.4+ KB
```

```
df['PCM Team'] = df['PCM Team'].map(str)
df = pd.merge(df, pcm_team)

feature_of_analysis = 'PCM Team Name'

df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(
ascending=False).reset_index()

df_temp.set_index(feature_of_analysis).sort_values(
by=['Key'],ascending=False).plot(kind='barh', title='Number of del
iveries per PCM Team');
```



```
names = ['PCM Team BR', 'PCM Marine IC&O', 'PCM Team SUEACN', 'PCM Team
NO', 'PCM Center GR', 'PCM Team NL', 'PCM Energy S & C AME' ]
df = df[~df[feature_of_analysis].isin(names)]
```

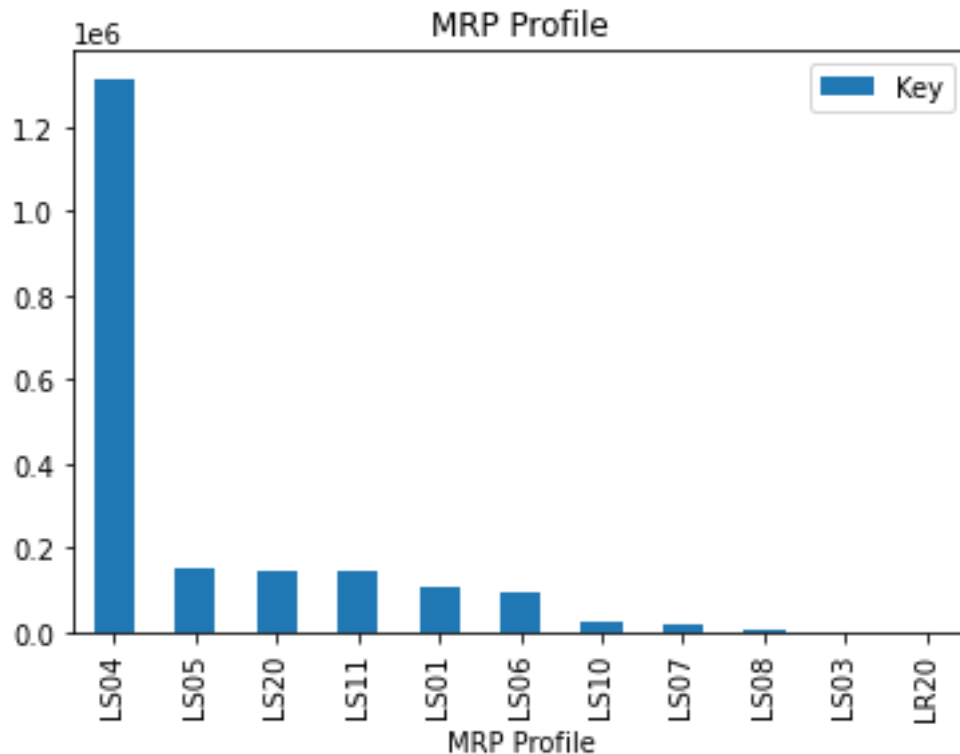
```
del df['PCM Team']
```

1.7 MRP Profile

```
feature_of_analysis = 'MRP Profile'

df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(
ascending=False).reset_index()

df_temp.set_index(feature_of_analysis).sort_values(
by=['Key'],ascending=False).head(200).plot(kind='bar', title=featu
re_of_analysis);
```



```
df_temp = df.pivot_table(index=[feature_of_analysis], columns='Is Late',
                          values='Key',
                          aggfunc='count').reset_index()
df_temp['rel_freq'] = df_temp[1]/(df_temp[0]+df_temp[1])

names = ['LS03', 'LR20']
df = df[~df[feature_of_analysis].isin(names)]
```

1.8 Shipping Type

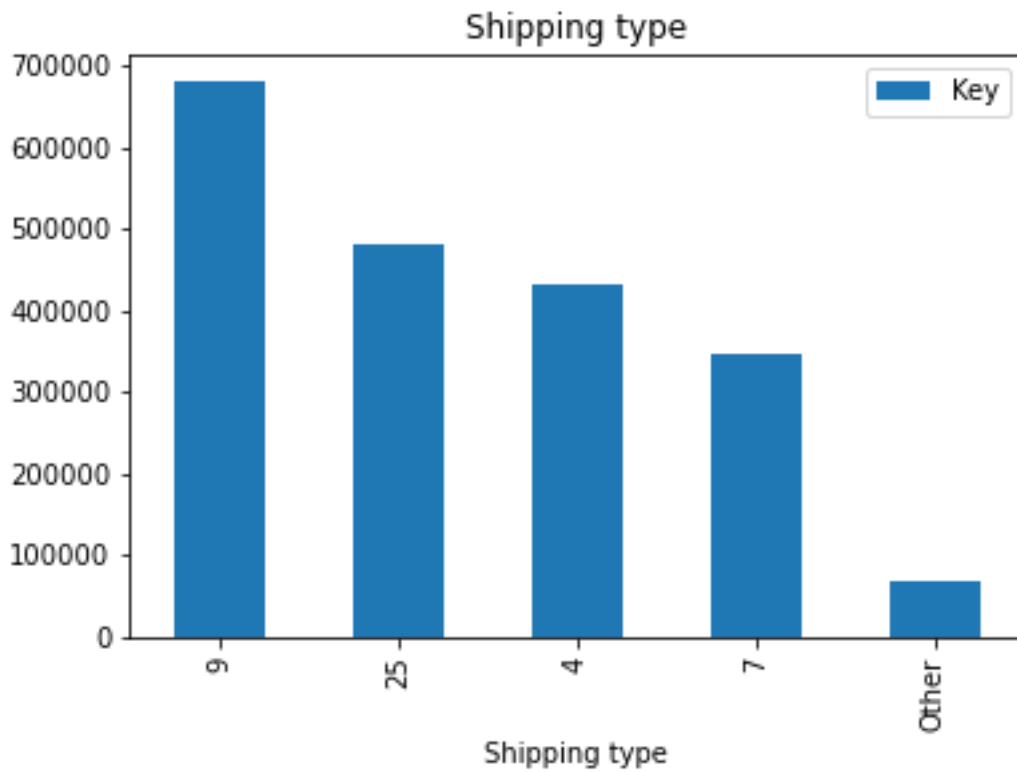
```
feature_of_analysis = 'Shipping type'

#Call the function with a threshold of 80%
transformed_column,new_category_list=cumulatively_categorise(df[feature_of_analysis],0.80,return_categories_list=True)
df[feature_of_analysis] = transformed_column
df[feature_of_analysis].unique()

array([9, 7, 4, 25, 'Other'], dtype=object)

df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(ascending=False).reset_index()

df_temp.set_index(feature_of_analysis).sort_values(
    by=['Key'],ascending=False).head(200).plot(kind='bar', title=feature_of_analysis);
```

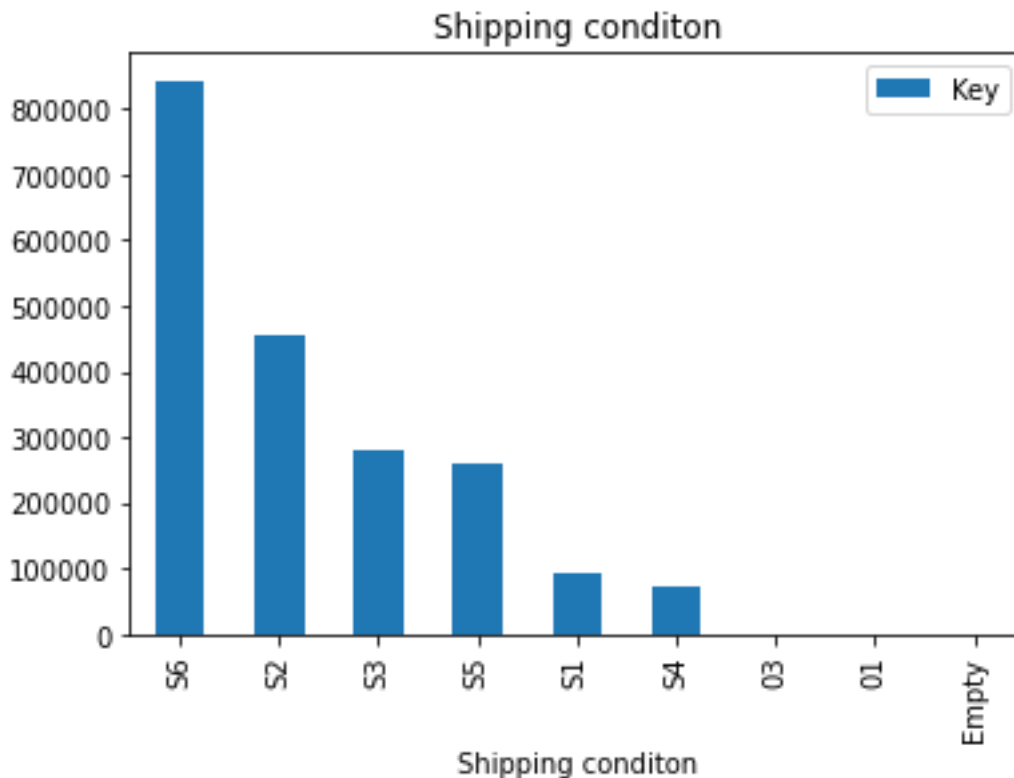



1.9 Shipping Condition

```
feature_of_analysis = 'Shipping conditon'
```

```
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(  
ascending=False).reset_index()
```

```
df_temp.set_index(feature_of_analysis).sort_values(  
by=['Key'],ascending=False).head(200).plot(kind='bar', title=featu  
re_of_analysis);
```



```
names = ['03','01', 'Empty']
df = df[~df[feature_of_analysis].isin(names)]
```

1.10 Incoterms 1

```
feature_of_analysis = 'Incoterms1'
```

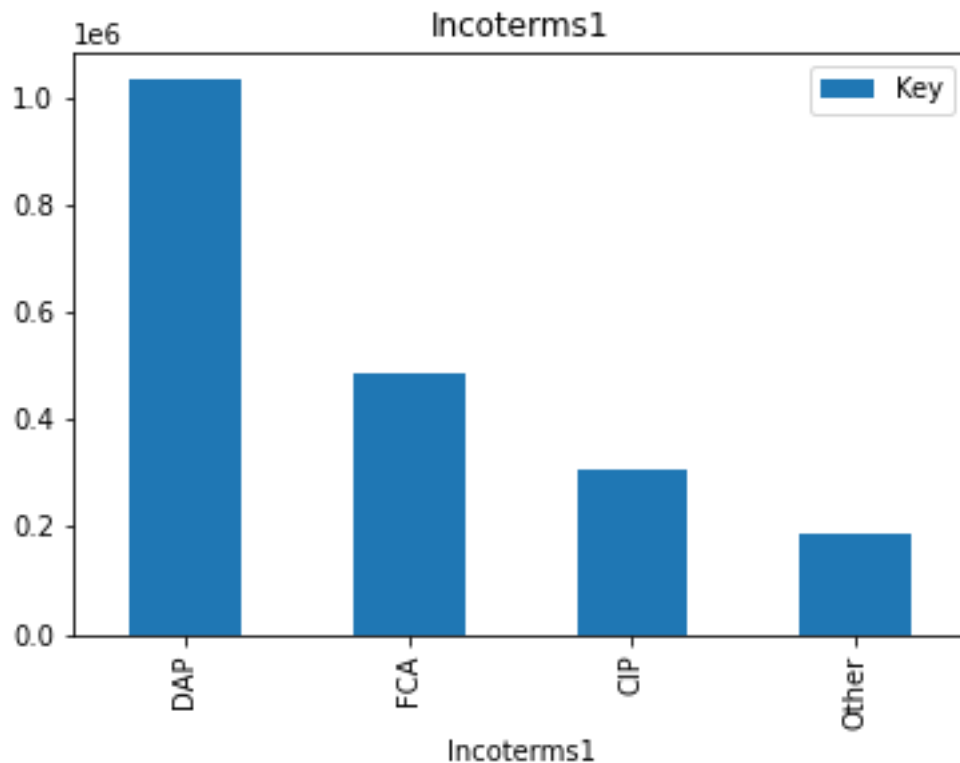
#Call the function with a threshold of 80%

```
transformed_column,new_category_list=cumulatively_categorise(df[feature_of_analysis],0.80,return_categories_list=True)
df[feature_of_analysis] = transformed_column
df[feature_of_analysis].unique()
```

```
array(['DAP', 'CIP', 'Other', 'FCA'], dtype=object)
```

```
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(ascending=False).reset_index()
```

```
df_temp.set_index(feature_of_analysis).sort_values(
    by=['Key'],ascending=False).head(200).plot(kind='bar', title=feature_of_analysis);
```



1.11 Consolidation Key

```
feature_of_analysis = 'ConsKey'
```

```
#Call the function with a threshold of 80%
```

```
transformed_column,new_category_list=cumulatively_categorise(df[feature_of_analysis],0.80,return_categories_list=True)
```

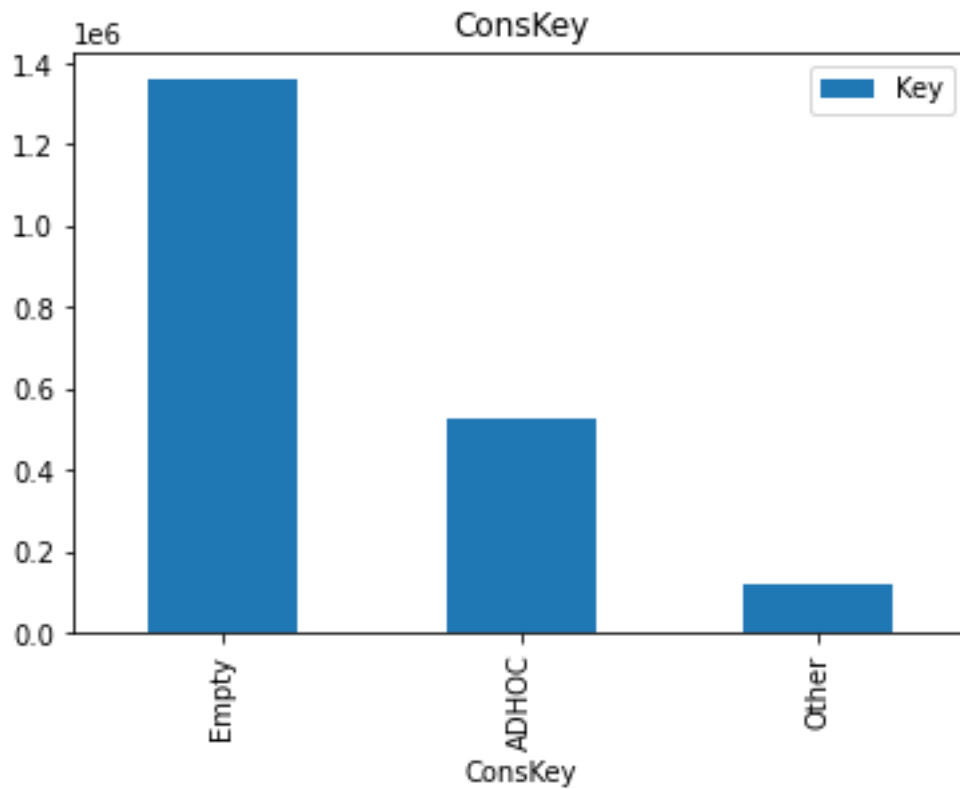
```
df[feature_of_analysis] = transformed_column
```

```
df[feature_of_analysis].unique()
```

```
array(['Empty', 'ADHOC', 'Other'], dtype=object)
```

```
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(ascending=False).reset_index()
```

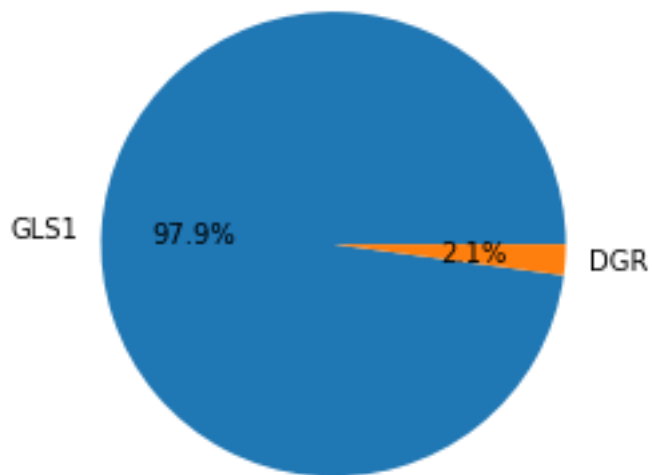
```
df_temp.set_index(feature_of_analysis).sort_values(
    by=['Key'],ascending=False).head(200).plot(kind='bar', title=feature_of_analysis);
```



1.12 Shipping Point

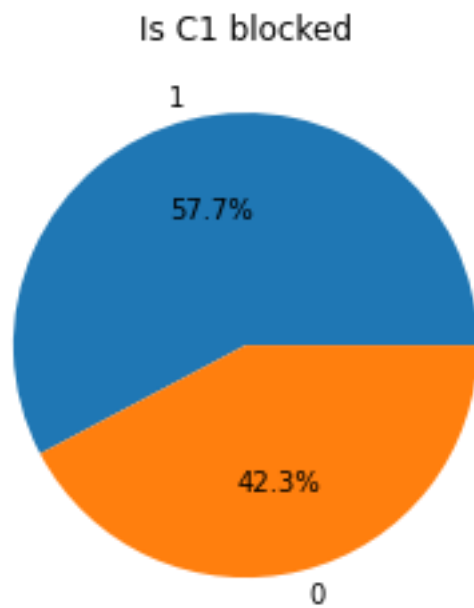
```
feature_of_analysis = 'Shipping Point'  
  
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(  
ascending=False).reset_index()  
plt.title(feature_of_analysis)  
plt.pie(df_temp['Key'], labels = df_temp[feature_of_analysis], autopct  
='%1.1f%%');
```

Shipping Point



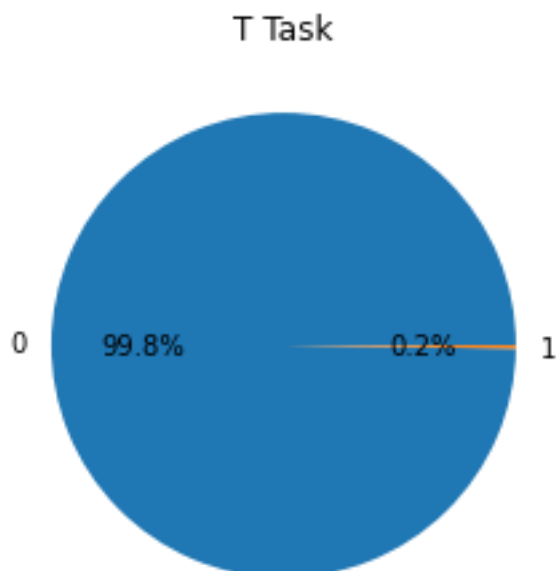
1.13 C1 block reason

```
feature_of_analysis = 'C1 block reason'  
df['Is C1 blocked'] = np.where(df[feature_of_analysis]!="Empty", 1 ,  
0)  
del df['C1 block reason']  
feature_of_analysis = 'Is C1 blocked'  
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(  
ascending=False).reset_index()  
plt.title(feature_of_analysis)  
plt.pie(df_temp['Key'], labels = df_temp[feature_of_analysis], autopct  
='%1.1f%%');
```



1.14 T Task

```
feature_of_analysis = 'T Task'  
  
df_temp = df.groupby([feature_of_analysis])['Key'].size().sort_values(  
ascending=False).reset_index()  
plt.title(feature_of_analysis)  
plt.pie(df_temp['Key'], labels = df_temp[feature_of_analysis], autopct  
='%1.1f%%');
```



```
del df['T Task']
```

Summary

```
del df['Key']
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2002646 entries, 0 to 2020434
Data columns (total 14 columns):
#   Column                                Dtype
---  -
0   Ship to Country                       object
1   MRP Profile                           object
2   Shipping type                          object
3   Shipping conditon                     object
4   Incoterms1                            object
5   ConsKey                                object
6   Shipping Point                        object
7   Is Late                                int64
8   Lead Time (OCD x FDD)                 float64
9   POD_day                                object
10  Schedule line delivery date_Year      object
11  FD Weekday                             object
12  PCM Team Name                          object
13  Is C1 blocked                          int32
dtypes: float64(1), int32(1), int64(1), object(11)
memory usage: 221.5+ MB
```

```
df.apply(pd.Series.nunique)
```

```
Ship to Country                22
MRP Profile                     9
Shipping type                   5
Shipping conditon               6
Incoterms1                     4
ConsKey                         3
Shipping Point                  2
Is Late                         2
Lead Time (OCD x FDD)          1020
POD_day                         31
Schedule line delivery date_Year 6
FD Weekday                      7
PCM Team Name                   12
Is C1 blocked                   2
dtype: int64
```

1.15 Checking Missing data

Check what columns has rows missingvalues

```
df.isnull().sum()
```

```

Ship to Country          0
MRP Profile              7159
Shipping type            0
Shipping conditon       0
Incoterms1              0
ConsKey                  0
Shipping Point          0
Is Late                  0
Lead Time (OCD x FDD)   41078
POD_day                  0
Schedule line delivery date_Year 0
FD Weekday              0
PCM Team Name           0
Is C1 blocked           0
dtype: int64

```

```

amount_null = df.isna().sum().sum()
size_table = len(df)

```

```

print("Rows to be deleted:" , amount_null , " - " , round(amount_null
/ size_table ,4)*100 , "% of " , size_table , "total rows" )

```

```

Rows to be deleted: 48237 - 2.41 % of 2002646 total rows

```

delete rows:

```
df = df.dropna()
```

```
df2 = df.copy()
```

2. Explore the Data

Label encoding

Use one-hot-encoder to convert categorical data into numbers.

```

df = pd.get_dummies(df, columns=['Shipping conditon','Ship to Country'
, 'MRP Profile', 'Shipping type', 'Incoterms1', 'ConsKey', 'Shipping P
oint', 'PCM Team Name', 'FD Weekday' ], drop_first=True)

```

Correlation

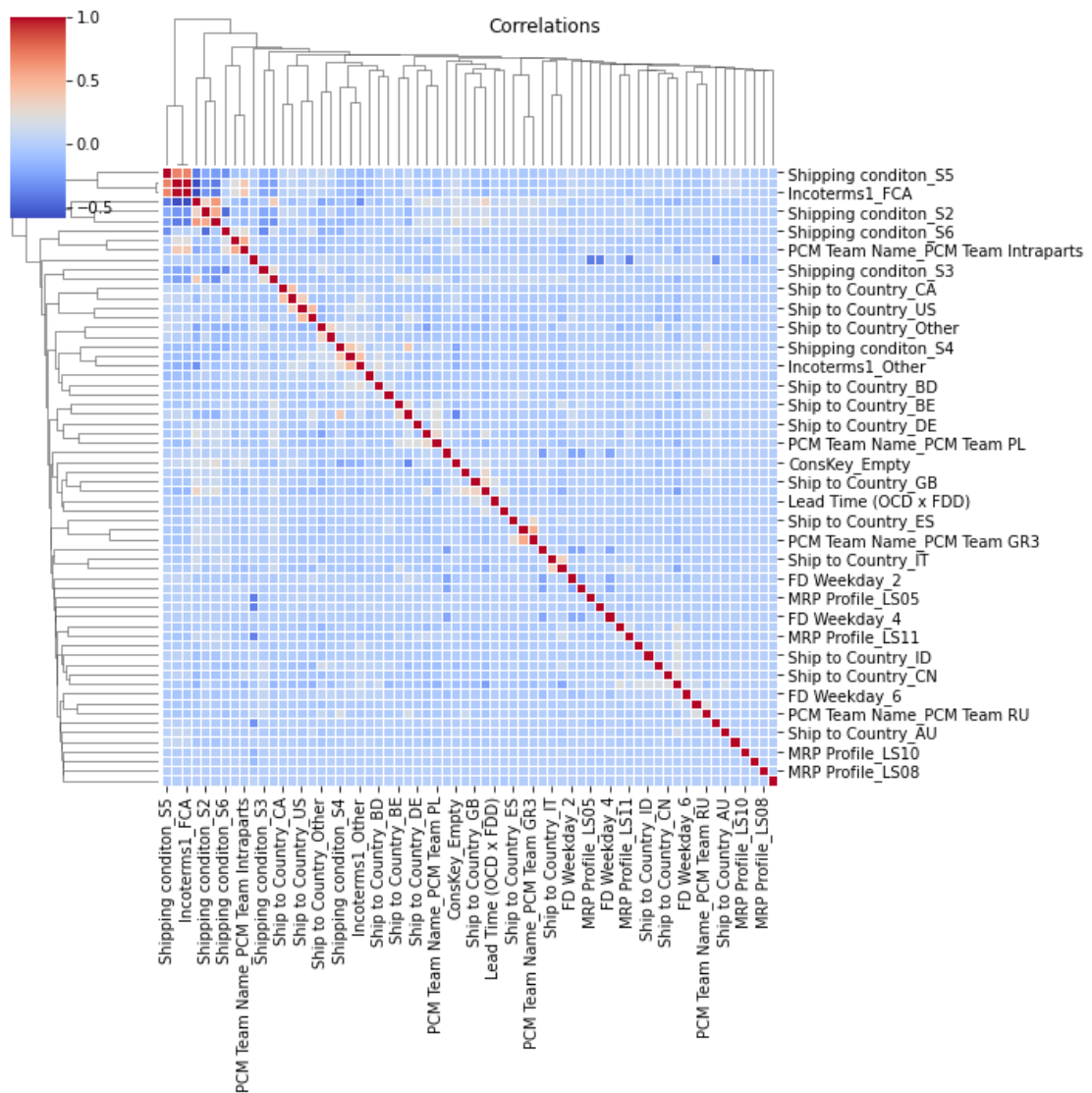
```
corr = df.corr()
```

```

cm = sns.clustermap(corr, cmap='coolwarm', linewidth=1)
cm.fig.suptitle('Correlations')

```

```
Text(0.5, 0.98, 'Correlations')
```

```
pd.set_option('display.expand_frame_repr', False)
```

```
from scipy.stats import kendalltau, pearsonr, spearmanr
import itertools
```

```
df = df.drop(['PCM Team Name_PCM Trainees', 'Ship to Country_ID', 'MRP
Profile_LS08', 'POD_day', 'MRP Profile_LS07', 'Ship to Country_FR', 'S
hip to Country_JP' ], axis=1)
```

Check correlation between target and others

```
del df['Shipping type_25']
```

Feature Scaling

```
scaler = MinMaxScaler()
```

```
df['Lead Time (OCD x FDD)'] = scaler.fit_transform(df[['Lead Time (OCD x FDD)']])
```

```
df['Schedule line delivery date_Year'] = scaler.fit_transform(df[['Schedule line delivery date_Year']])
```

#After Scaling

```
df['Lead Time (OCD x FDD)'].describe().round(2)
```

```
count    1954561.00
mean         0.43
std         0.03
min         0.00
25%         0.42
50%         0.42
75%         0.42
max         1.00
```

```
Name: Lead Time (OCD x FDD), dtype: float64
```

Feature selection

```
X = df.drop(columns = ['Is Late']).copy()
y = df['Is Late']
```

Data Imbalance

```
response = 'Is Late'
```

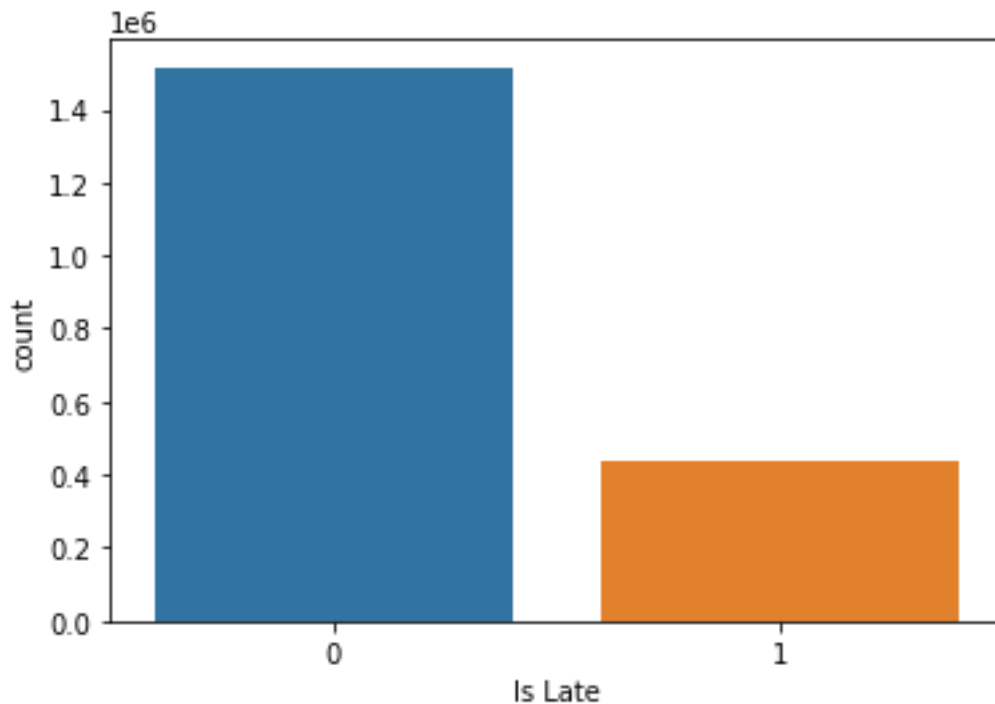
```
y = df[[response]]
```

```
features = list(df.columns)
```

```
features.remove(response)
```

```
sns.countplot(x = response, data = df)
```

```
<AxesSubplot:xlabel='Is Late', ylabel='count'>
```



Random undersampling

```
# class count
y = df[response]
X = df[features]

class_count_0, class_count_1 = df[response].value_counts()

# Separate class
class_0 = y[y==0]
class_1 = y[y==1]

# print the shape of the class
print('class 0:', class_0.shape)
print('class 1:', class_1.shape)

class 0: (1515604,)
class 1: (438957,)

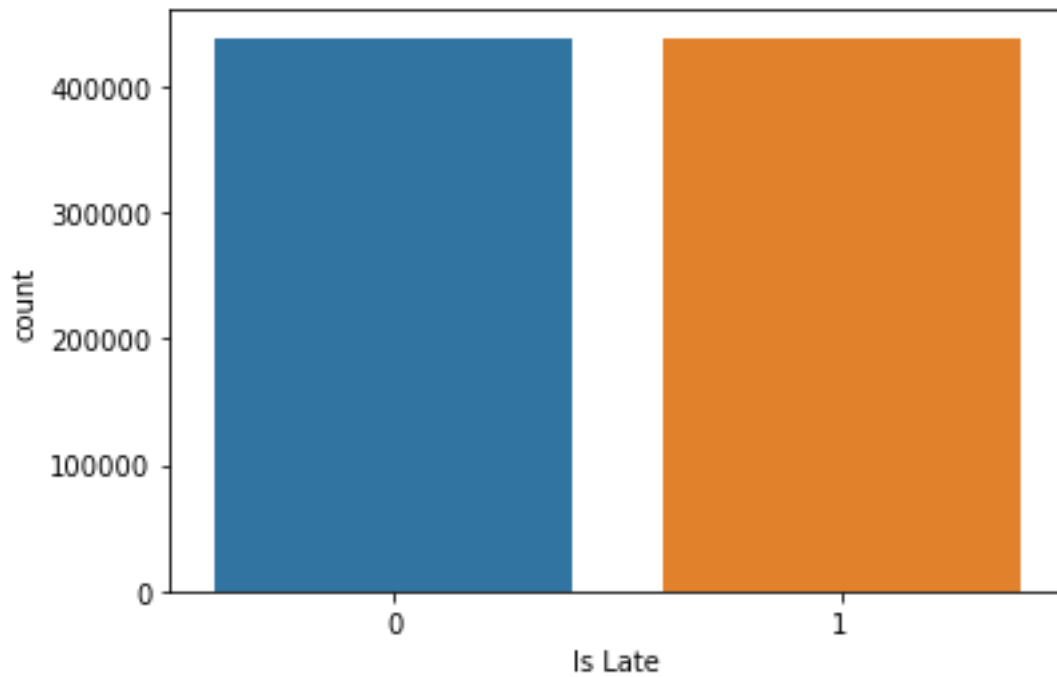
y_1 = y[y==1]
y_0 = y[y==0].sample(n=len(y_1),replace=True, random_state = 2)

print('class 0:', y_1.shape)
print('class 1:', y_0.shape)

y_novo = pd.concat([y_1,y_0])
X_novo = X.loc[y_novo.index]

class 0: (438957,)
class 1: (438957,)
```

```
df_sample = pd.concat([X_novo,y_novo],axis=1)
len(df_sample)
877914
sns.countplot(x = response, data = df_sample)
<AxesSubplot:xlabel='Is Late', ylabel='count'>
```



Random Forest – Final Model

Set the libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
from sklearn import metrics
import scipy.stats
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
```

#Set the file location

```
dataframe = 'G:\My Drive\Thesis\pre-
processing\late_deliveries_pre_processing_sample.csv'
```

#Set table size

```
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
```

Read data

```
df = pd.read_csv(dataframe)
```

```
X = df.drop(columns = ['Is Late']).copy()
y = df['Is Late']
```

Select top 25 features

```
from sklearn.feature_selection import SelectKBest
```

```
bf = SelectKBest(score_func=chi2, k=40)
fit = bf.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

```
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Feature','Score'] #naming the dataframe
kbest_features = X.columns[bf.get_support(indices=True)].tolist()
print(featureScores.nlargest(40,'Score')) #print 20 best features
```

	Feature	Score
5	Shipping conditon_S5	20141.482325
34	Incoterms1_FCA	11453.495296
36	ConsKey_Other	7841.823243
20	Ship to Country_NL	7346.591137

22	Ship to Country_Other	6833.260155
46	PCM Team Name_PCM Trainees (SN)	6747.607868
35	Incoterms1_Other	6733.648166
3	Shipping conditon_S3	5284.308643
42	PCM Team Name_PCM Team PL	4893.211226
13	Ship to Country_DE	4483.048223
29	MRP Profile_LS11	4276.606559
8	Ship to Country_BD	4144.534075
23	Ship to Country_SG	3796.442151
41	PCM Team Name_PCM Team Intraparts	2331.105508
44	PCM Team Name_PCM Team UK	2325.182608
32	Shipping type_Other	2033.708208
11	Ship to Country_CA	2012.911751
24	Ship to Country_US	1800.975668
48	FD Weekday_1	1650.857785
30	MRP Profile_LS20	1573.098976
1	Is C1 blocked	1057.848285
47	PCM Team Name_PCM USA & CA (USA)	1053.995403
52	FD Weekday_6	974.948056
4	Shipping conditon_S4	819.834757
6	Shipping conditon_S6	741.712674
19	Ship to Country_KR	737.610369
40	PCM Team Name_PCM Team IT	731.691425
10	Ship to Country_BR	686.846783
50	FD Weekday_3	496.675844
9	Ship to Country_BE	464.222647
14	Ship to Country_DK	454.253119
2	Shipping conditon_S2	331.220269
49	FD Weekday_2	289.068580
7	Ship to Country_AU	246.183890
21	Ship to Country_NO	206.634237
51	FD Weekday_5	188.505739
45	PCM Team Name_PCM Traditional	180.869995
31	Shipping type_7	144.050385
39	PCM Team Name_PCM Team GR3	103.191708
28	MRP Profile_LS10	100.035591

Random Sampling

Creating a copy of the dataframe before sampling it

```
df_copy = df.copy()
```

Prepare the Data

defining response and predictive variables

```
#X = df.drop(columns = ['Is Late']).copy()
X = df[kbest_features]
y = df['Is Late']
```

Split the data

```
# Splitting the data into training, test and validation
X_train, X_rem, y_train, y_rem = train_test_split(X,y, train_size=0.8)

test_size = 0.5
X_valid, X_test, y_valid, y_test = train_test_split(X_rem,y_rem,
test_size=0.5)

print("Rows on x_train:" , len(X_train))
print("Rows on y_train:" , len(y_train))
print(" ")
print("Rows on x_valid:" , len(X_valid))
print("Rows on y_valid:" , len(y_valid))
print(" ")
print("Rows on x_test:" , len(X_test))
print("Rows on y_test:" , len(y_test))

Rows on x_train: 702331
Rows on y_train: 702331

Rows on x_valid: 87791
Rows on y_valid: 87791

Rows on x_test: 87792
Rows on y_test: 87792
```

8 Model Training and Validation

Train the model

```
model = RandomForestClassifier().fit(X_train, y_train)
```

Model Score

```
#Training
model.score(X_train, y_train)
```

```
0.7339231786721645
```

```
#Valid
model.score(X_valid, y_valid)
```

```
0.7322162864074905
```

```
#Test
model.score(X_test, y_test)
```

```
0.7269341170038273
```

Calculate the predictions

```
y_test_pred = model.predict(X_test)
```

Probability

```
y_test_proba = model.predict_proba(X_test)
```

Show classification report

```
print(classification_report(y_test, y_test_pred))
```

	precision	recall	f1-score	support
0	0.73	0.72	0.72	43862
1	0.72	0.74	0.73	43930
accuracy			0.73	87792
macro avg	0.73	0.73	0.73	87792
weighted avg	0.73	0.73	0.73	87792

Calculate ROC and AUC

```
#Code adapted from DataTechNotes (2022)
```

```
predY = model.predict_proba(X_test)
```

```
fpr, tpr, thresh = metrics.roc_curve(y_test, predY[:,1])
```

```
auc = metrics.auc(fpr, tpr)
```

```
print("AUC:", auc)
```

```
plt.plot(fpr, tpr, label='ROC curve (area = %.2f)' %auc)
plt.plot([0, 1], [0, 1], linestyle='--', lw=2, color='r',
label='Random guess')
plt.title('ROC curve', size=20)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.grid()
plt.legend()
plt.show()
```

```
AUC: 0.8092541542482178
```

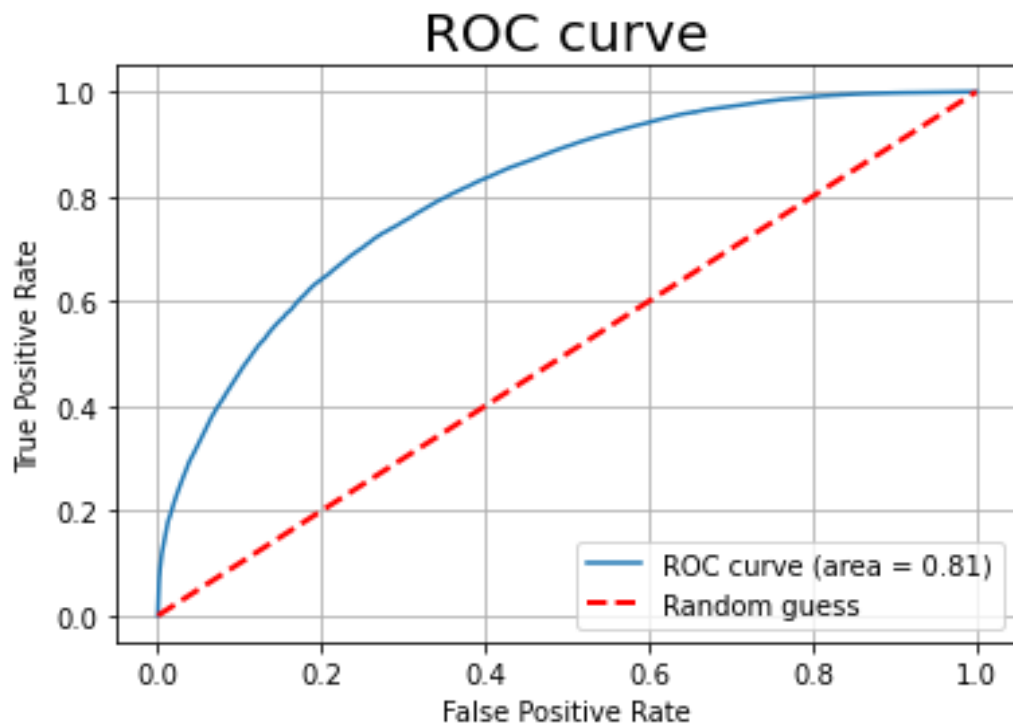



Table 14. Threshold decision

```

test_y = y_test
predicted_y_probs = y_test_proba

prob_df = pd.DataFrame(data={
    'prob_0': predicted_y_probs[:,0],
    'prob_1': predicted_y_probs[:,1],
    'Is Late': y_test.values.ravel()
})

prob_df['prob_bin'] = pd.cut(prob_df['prob_1'], bins = 10)

prob_df.head()

   prob_0  prob_1  Is Late  prob_bin
0  0.529354  0.470646     1  (0.4, 0.5]
1  0.957762  0.042238     0  (-0.001, 0.1]
2  0.000000  1.000000     1  (0.9, 1.0]
3  0.198337  0.801663     1  (0.8, 0.9]
4  0.771197  0.228803     0  (0.2, 0.3]

prob_df.groupby('Is Late')['prob_1'].hist(alpha=0.7)

Is Late
0  AxesSubplot(0.125,0.125;0.775x0.755)
1  AxesSubplot(0.125,0.125;0.775x0.755)
Name: prob_1, dtype: object

```

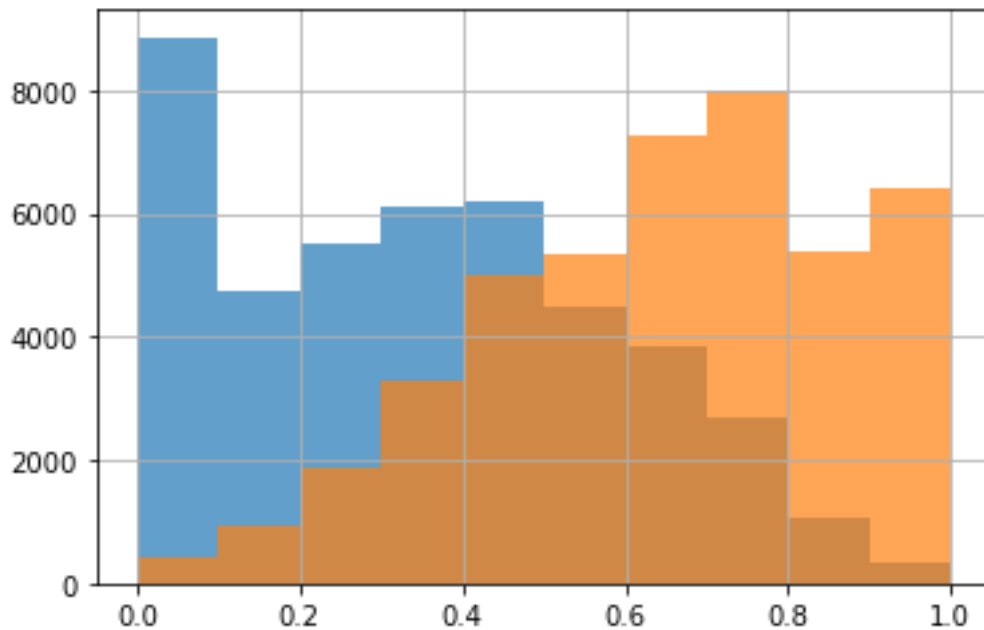


Table 15. Checking Feature Importance

9 PREDICTING

Selecting entire database to include also future deliveries

```
df_valid = X_valid.copy()
df_valid['Is Late'] = y_valid.copy()
```

Calculate prediction

```
predict_prob = model.predict_proba(X_valid)
```

```
y_pred = model.predict(X_valid)
```

```
proba = pd.DataFrame(predict_prob)
proba.columns = [['pred_0', 'pred_1']]
#proba
```

```
df_valid['Pred'] = y_pred
```

```
p0 = predict_prob[:,0]
p1 = predict_prob[:,1]
```

```
df_valid['Prob_1'] = p1
df_valid['Prob_0'] = p0
```

```
subset = df_valid[['Is Late', 'Pred', 'Prob_1', 'Prob_0']]
```

```
subset_copy = subset.copy()
```

```
subset['Is Late'].unique()
```

```

array([1, 0], dtype=int64)

subset = subset.loc[(subset["Is Late"] != -1)
                    ]

# Creating a function to report confusion metrics
# Code adapted from Minaie (2019)

def confusion_metrics (cm):
# save confusion matrix and slice into four pieces
    TP = cm[1][1]
    TN = cm [0][0]
    FP = cm [0][1]
    FN = cm [1][0]

    accuracy = (float (TP+TN) / float(TP + TN + FP + FN))
    sensitivity = (TP / float(TP + FN))
    specificity = (TN / float(TN + FP))

    precision = (TN / float(TN + FP))
    recall = TP/(TP+FN)
    conf_f1 = 2 * ((precision * sensitivity) / (precision +
sensitivity))

    print('-'*50)
    print(f'Accuracy: {round(accuracy,2)}')
    print(f'Precision: {round(conf_precision,2)}')
    print(f'Recall: {round(recall,2)}')
    print(f'f_1 Score: {round(f1,2)}')

#AUC
    fpr,tpr, thresh = metrics.roc_curve(actual, subset['Prob_1'])
    auc = metrics.auc(fpr, tpr)
    print(f'AUC: {round(auc,2)}')

actual = subset['Is Late']
predicted = subset['Pred']

# Confusion matrix. Code adapted from Fuatakal (2022)

cm = metrics.confusion_matrix(actual, predicted)

cm_display = metrics.ConfusionMatrixDisplay(cm = cm, display_labels =
['On-Time', 'Late'])

group_c = [{"0:0.0f}".format(value) for value in
            cm.flatten()}
group_p = [{"0:.2%}".format(value) for value in

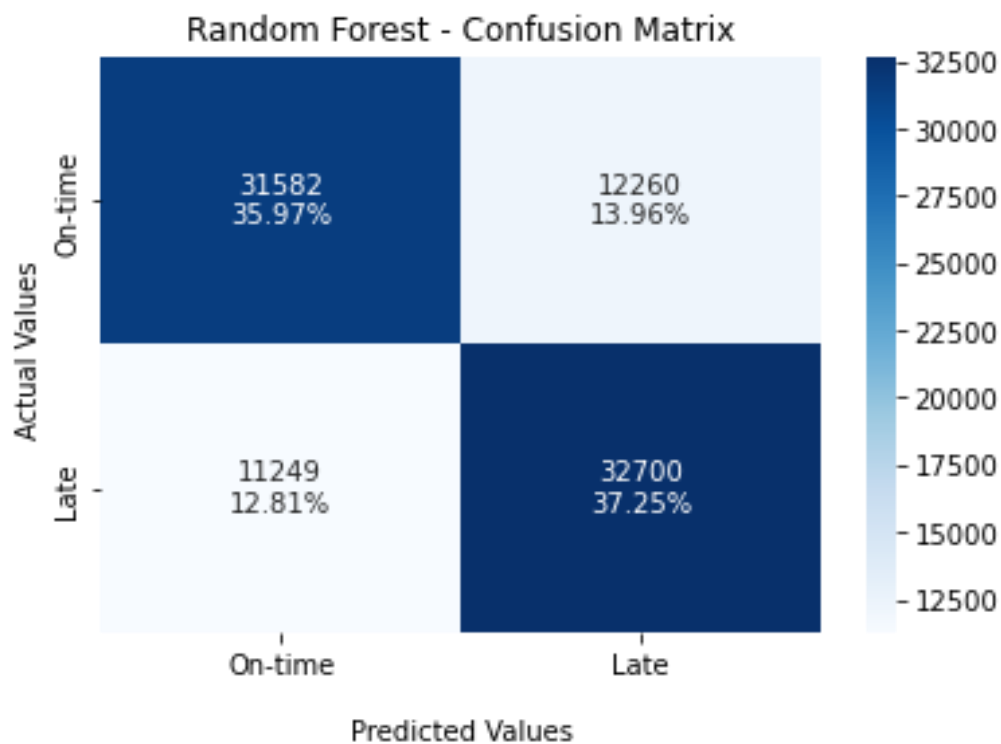
```

```

cm.flatten()/np.sum(cm)]
lb = [f"{v1}\n{v2}" for v1, v2 in
      zip(group_c,group_p)]
lb = np.asarray(lb).reshape(2,2)
ax = sns.heatmap(cm, annot=labels, fmt='', cmap='Blues')

ax.set_title('Random Forest - Confusion Matrix');
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values ');
ax.xaxis.set_ticklabels(['On-time','Late'])
ax.yaxis.set_ticklabels(['On-time','Late'])
plt.show()
confusion_metrics (cm)

```



```

-----
Accuracy: 0.73
Precision: 0.72
Recall: 0.74
f_1 Score: 0.73
AUC: 0.82

```

```
print(classification_report(y_true, y_pred))
```

	precision	recall	f1-score	support
0	0.74	0.72	0.73	43842

1	0.73	0.74	0.74	43949
accuracy			0.73	87791
macro avg	0.73	0.73	0.73	87791
weighted avg	0.73	0.73	0.73	87791

Dashboard Snapshot

