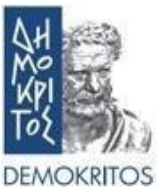

Infusing business optimization processes with machine learning and expert knowledge

by

Theodoros Adamantidis

Submitted
in partial fulfilment of the requirements for the degree of

Master of Artificial Intelligence

at the

UNIVERSITY OF PIRAEUS

July 2022

Author: Theodoros Adamantidis

II-MSc “Artificial Intelligence”

July, 2022

Certified by: Georgios Giannakopoulos

Georgios Giannakopoulos,
Researcher,
Thesis Supervisor

Certified by: Theodoros Giannakopoulos

Theodoros Giannakopoulos,
Researcher,
Member of Examination Committee

Certified by: Maria Diagioglou

Maria Dagioglou,
Researcher,
Member of Examination Committee

Infusing business optimization processes with machine learning and expert knowledge

By

Theodoros Adamantidis

Submitted to the II-MSc "Artificial Intelligence" on July, 2022, in partial fulfillment of the
requirements for the MSc degree

Abstract

Leaving in the 4th industrial revolution, the digitization in manufacturing in conjunction with the rapid increase of AI applications have raised new opportunities. In the context of this work, the infusion of machine learning on a real word manufacturing optimization knapsack-like problem will be researched and addressed by focusing on interpretability and generalization of the models. The problem will be described, modelized and formulated mathematically, the data collection and dataset construction will be presented, and machine learning techniques will be used and compared based on which achieves to extract the most information out of the data. Also, the explanation of the trained models is discussed, and the most efficient way of data constructions is checked in terms of generating a model that generalizes better.

Thesis Supervisor: Georgios Giannakopoulos
Title: Researcher

Acknowledgement

I would like to thank my supervisor Researcher (ELE) B Grade Mr. Georgios Giannakopoulos, his assistant Ph.D. student Mr. Dimitrios Kaklis and in general REPR for their invaluable assistance, guidance, comments, and insights and for helping me undertake this research. I would also like to thank Greycon, for their contributions to data collection and their support. Finally, I would like to thank my parents, for their love and encouragement, without whom I would never have enjoyed so many opportunities.

Contents

List of Figures

List of Tables

1 Introduction

The first industrial revolution began between 1760 and 1840, a period in which railroads constructed and steam engines invented. Then in the late 19th century and in the early 20th century we had the second industrial revolution, in which the electricity made mass productions possible. Later, in around the 1960s, began the third industrial revolution, which was also called *digital revolution*, because it was connected with the development of semiconductors, mainframes, personal computing and the internet. In Germany from 2011 started the fourth industrial revolution. (Schwab, 2016, pp. 15, [16]).

Today, we live in the fourth industrial revolution, which enables the smart factories and tries to create a world in which virtual and physical systems co-operate with each other in a flexible way. (Schwab, 2016, pp. 16, [16]).

On the other hand, Machine Learning (ML) and Artificial Intelligence (AI) in general, has become more and more part of the products that are used daily. We can find applications of ML in many areas such as:

- image/speech recognition,
- self-driving cars,
- traffic prediction,
- product recommendations,
- decision support, etc..

The rapid increase in AI applications, in conjunction with the evolution and the digitization of manufacturing, raised new opportunities.

In the context of this work we conducted a research of ML applications in manufacturing production planning.

Production planning is the future of production and is a key component in the process of business manufacturing. Its purpose is to minimize the production time and cost while in parallel help to increase profit.

In the next sections we will present a data driven tool that operates as a decision support alongside with an already implemented and existing combinatorial algorithm that minimizes the production waste by solving a knapsack-like problem. In contrast with pertinent literature around the content, we approached the problem as a black box and we focused more on the connection between the input and the output, leaving out any clue related to the algorithm that solves it.

The goal of the tool employed in this work, is to help manufacturing organisations to increase their productivity, by reducing the required time to develop a production plan. This tool, will act as an experienced user - expert in the area, that even not familiar with the optimization procedure, he can still make assumptions for the function that connects the input and the output. This tool could also be utilized as the base to build a more complex architecture that would potentially improve the computational performance.
The work presented in this document is organized as follows:

- Presentation of background and related work,
- Problem definition,
- Experiments,
- Conclusions.

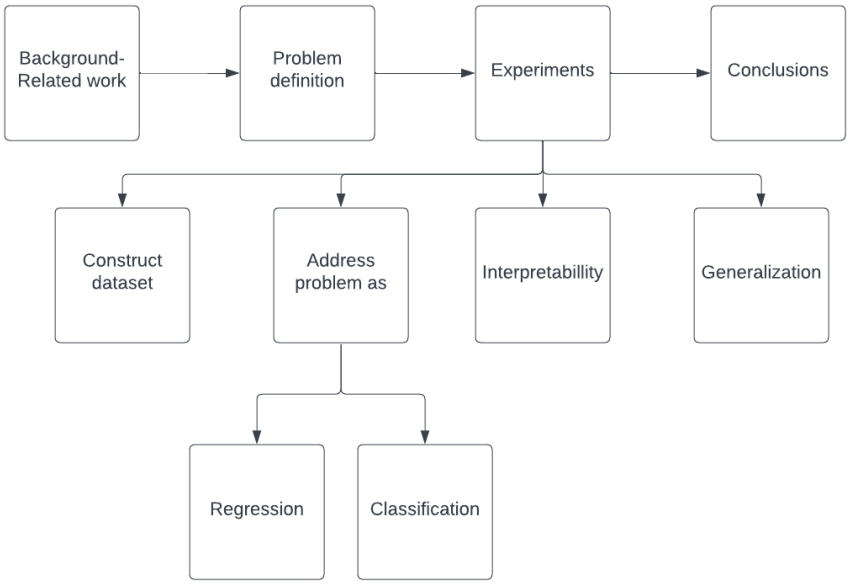

Figure 1: Document structure

The experiments include the following four big topics:

- **Dataset construction**: Understand the problem, explore the metadata and construct datasets.
- **Modeling the problem**: Model and address the problem both as regression and classification, perform experiments using ML and simple NN models and find the most efficient combination of dataset (set of features) - approach.
- **Interpretability**: As it is necessary to be able to explain a prediction, the impact, the integration, inference time etc), the model's interpretability is explored and further checked.
- **Generalization**: The dataset utilized for training cannot take into account the whole space of the parameters of this optimization problem. To this end we split the dataset in train and test sets and we cross examine

the accuracy and generalization capabilities of the ML models trained in these parts.

Our contribution - the questions that will be answered are:

- Which is the best way to model a dataset in this knapsack-like problem. Which information helps the ML models perform better? Which samples can describe a bigger part of the input space?
- Which ML model performs the best and why?
- Can we explain the predictions?

2 Background

In this section we will present the required background knowledge, like what KBS (Knowledge Based Systems)/ML/Regression/Classification/NN/Optimization problems are.

2.1 KBS/Machine learning

Based on Ahmed et al., 2019 [2] a Knowledge-Based System (KBS) is described as a system which helps experts transfer knowledge into information systems. The basic components of a KBS are the following:

- **Knowledge base**, contains the necessary knowledge for modeling the problem,
- **Inference Engine**, brain of the system, extracts the methodology for reasoning,
- **Knowledge Acquisition**, ability to enlarge knowledge base,
- **Explanation Facility**, ability to interpret results,
- **User interface**, interface which the user can interact with.

While the KBS requires the experts to pass the base knowledge, ML extracts patterns from data and constructs the rules. ML is part of AI and can be defined as a set of computational methods which use experience in order to try to improve their performance or to make accurate predictions. As experience refers to information - data, which are used to train these computational methods. ML consists of designing efficient and accurate prediction algorithms which learn and extract patterns from the input data. (Mohri et al, 2017, pp. 1, [12]). ML consists of designing efficient and accurate prediction algorithms which learn and extract patterns from the input data. (Mohri et al, 2017, pp. 1, [12]). ML can be used in any type of problem such as:

- Predict a house's price, or predict a student's grade,
- Document classification, or mail spam filtering,
- Natural language processing,
- Speech recognition applications,
- Computer vision applications, etc..

ML problems can be distinguished into two big categories, the supervised and unsupervised. The difference in these two big categories, is that in the first one we provide to the algorithms the results of the data which gets trained on while in the second, we ask from the algorithm to construct teams/categories. In supervised learning models the algorithm gets as an input a labeled set of

examples and makes predictions for unseen points. In Unsupervised learning, the algorithms gets as input an unlabeled set of examples and makes predictions for unseen points. (Mohri et al, 2017, pp. 6, [12]).

The two most famous types of supervised problems are the regression and classification. The data utilized in the context of this work have labels. This means that the problem is supervised, and approached as regression and classification. Regression consists of problems in which, an accurate value tries to be predicted, such as predict a house's price. Classification consists of problems in which, a class tries to be predicted, such as a mail is spam or not. (Mohri et al, 2017, pp. 2, [12]). Regression can be also defined as a problem in which an algorithm tries to describe the space of the data, while classification as a problem in which the algorithm tries to divide the space of the data.

2.2 Regression

As already mentioned, regression consists of using data to predict a value, as closely as possible. Regression models construct a function which maps the inputs X with the outputs Y.

In the context of this work we used, linear regression and rule based models. Linear regression models, are models which try to describe the data in a linear way and all of them can directly or indirectly be written in the form of equation 1. (Kuhn and Johnson, 2013, pp. 95, [9]).

$$y_i = b_0 + b_1 x_{i_1} + b_2 x_{i_2} + ... + b_p x_{i_p} + e_i \tag{1}$$

On the other hand, there are the rule-based models, which consist of one or more nested if-then statements and describe the space using rules, as shown above.

```
if FeatureA <= ValueA:
    if FeatureB <= ValueB:
        return Prediction1
    else:
        return Prediction2
else:
    return Prediction3
```

This type of models can describe more complex than linear problems because it is not necessary the data to be linearly described.

Regarding the evaluation metrics in the context of this work we focused on MAE, (2), which indicates the mean absolute difference between the predicted value and the actual. MAE provides a mechanism to measure, the model's accuracy. As closest to zero the MAE is, the better the model can predict a value.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |x_i - x| \tag{2}$$

Both types of regression models are easily interpretable and explainable. In the first case the weights of each feature can be extracted and a function which gets as input the value for each feature and give as output the prediction can be made, while in the second the rules of the conditions can get extracted.

2.3 Classification

Classification consist of problems that use data to predict a class or a value in general, out of a set of finite and discrete values. An example of a classification problem is a feature for an email provider, which we want to learn which emails are okay and which emails are spam. Classification models construct a function which maps the inputs X with a value Y_u out of a finite set of Y. (Han and Kamber, 2012, pp. 330, [8]).

As in the regression problems, in the classification there are also rule-based models, where they construct a set of IF-THEN rules and predict specific classes. In classification there are also the Bayesian type models which generate two types of predictions, a regression-like, which is usually in a form of a probability, and a class membership prediction. For most of the cases, the focus is on the discrete prediction, although, the probability value contains very useful information. (Han and Kamber, 2012, pp. 335, [8]).

About the evaluation metrics, in the context of this work we focused on precision (10), which, as per Tharwat, 2018, [17], is the number of correct predicted samples, divided by all the samples of this specific class. If the correct predicted defined as True Positives (TP), and the not correct predicted False Positives (FP), the precision value is the one described in equation 10.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

Confusion matrices used in the classification experiments section. Confusion is a 2D matrix in which, the one side presents the actual values and the other the predicted. Every cell describes the correct predictions and the misclassifications.

Regarding the explainability of the results, it is not always easy and depends on the type and the model that used.

2.4 Neural networks

As extension of ML algorithms, comes the Deep Learning (DL) which includes artificial neural networks(ANN) and is a concept built over a powerful not linear regression technique, inspired from the mechanism of learning in biological organisms. Artificial neural networks contain neurons that are connected to each other just like the human nervous system contains cells. Each one of these neurons in the artificial neural network, represent a combination of some or all of the predictor variables.

The simplest neural network is the perceptron. Perceptron contains a single input layer with an output node. A set of weights is getting assigned in the

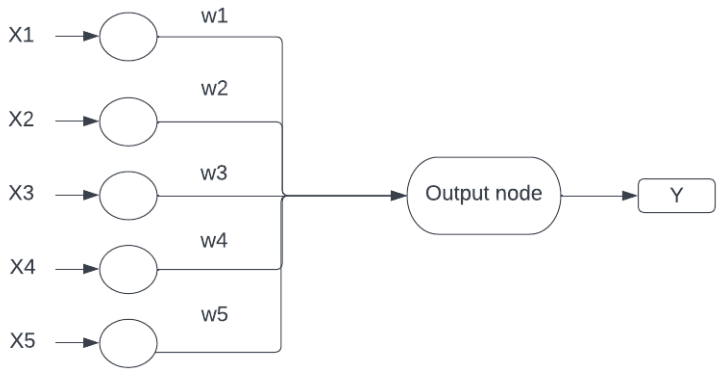

Figure 2: Perceptron basic architecture

input variables and then the output is getting calculated based on them. The basic architecture of Perceptron is shown in figure 2.

Each training instance is of the form (X, y), where each $X = [x_1, ..., x_d]$, contain d features, and y is the value we want to predict.

The input layer contains d nodes transmitting the d features with multilayer - weights $W = [W_1..W_d]$, to an output node. The output node calculates the linear function $W \cdot X = \sum_{i=1}^{d} w_j x_j$ and returns the result-prediction. (Aggarwal, 2018, pp. 5, [1])

In addition to perceptron's simple network, in DL, more complex architectures have been designed with perceptron like nodes. However because a combination of linear models would only be able to construct a bigger and more complex linear model, the activation function have been introduced at the end of each node - in order to break the linearity. These activation functions act as a regularizer and try to control the percentage of error that is being fed back to the network in order to update the weights appropriately. One of the most famous and most used ones is the Relu. (Kuhn and Johnson, 2013, pp. 141, [9])

2.5 Under-fit, over-fit and generalization

Bousquet et al., 2004, pp. 179, [4], describes the generalization error as the ability of models to be able to predict unseen data. Almost all ML models and DL architectures, are highly adaptable and able to describe complex relationships. Sometimes it is possible the model to not perform well, while some other times the model learns the structure of the training data so well, it can correctly predict every sample. The first happens because the model has not been trained enough and its getting called under-fit. The second happens because in addition of learning the general pattern in the data, the model learns the characteristic

of each sample's unique noise. This type of model is said to be over-fit and usually has poor accuracy when predicting a new sample. (Kuhn and Johnson, 2013, pp. 61, [9]).

Their is also the bias and variance. Bias can be defined as the accuracy of the predictions while variance is a type of error that occurs due to a model's sensitivity to small fluctuations in the training set. High variance cause the model to tend to over-fit, because it tries to learn the characteristics of each sample's unique noise. High bias would cause an algorithm to miss relevant relations between the input features and the target outputs which is related to the under-fitting which mentioned above.(Aggrawal, 2018, pp. 174, [1]).

Usually, a model's goal is to be able to perform well in at least a bigger subset, if not in the whole set, of the data that is trained on. In other words, is desirable the model to generalize well. For this reason it is necessary to have a balance between bias and variance.

2.6 Optimization problem

Optimization is a general term, and finds appliances in almost all fields. Optimization problem, is any problem that involves making a decision of choosing the best solution out of all feasible solutions. The task of decision making, during the optimization process, includes the choice which constitutes the best option. The measurement of goodness is described by an objective function. (Chong, 2013, pp. 15, [6]).

For example, one of the most famous optimization problems, is the knapsack. In this problem we have a bag in which we can fit at max a specific amount of quantity. We also have items that we want to fit in the bag. Each item has a specific quantity q_i and a price p_i. The goal is to fit in the bag the set of the available items, which maximizes the summary of the prices, without exceeding the maximum amount of quantity W which the bag can carry. The following equation, formalizes mathematically the previously described problem. The x_i represents the existence of an item in the bag and takes values 0 or 1. Our goal is to maximize the bags value, subject to not exceeding the maximum allowed quantity.

$$\begin{aligned} &\text{max. } \sum x_i p_i \\ &\text{s.t. } \sum x_i q_i \leq W, x_i \in \{0,1\} \end{aligned} \tag{4}$$

3 Related work

In this section we will present related work of ML in manufacturing, Interpretability of ML models and usages of ML in optimization problems.

3.1 ML in manufacturing

While the KBS requires the experts to pass the base knowledge, ML extracts patterns from data and constructs the rules.

Weichert et al., 2019 [20] expressed their opinion that the shortage of resources leads to increasing the acceptance of new approaches, like the ML, for the optimization of different processes like, energy, time, resource saving, waste minimization. The first challenge of every ML problem is the data gathering or in other words the construction of the dataset using which we will train the models. For this reason we can classify the data types in the following categories:

- qualitative vs. quantitative,
- time series vs. workpiece-related data,
- controllable vs. uncontrollable data,
- present vs. historical data,
- measured vs. simulated data,
- observable quantities vs. process state variables.

Weichert et al., 2019 [20] concluded, that because the employed models suffer from overfitting and lack of interpretability, it is necessary to review each step of the process in order to avoid making a problem specific solution and provide algorithms that generalize better. As well, Wuest et al., 2016 [21] focused in the advantages, the challenges and the use of ML in manufacturing processes. As advantages the authors present the ability of ML algorithms to solve NP hard problems and to handle high dimensional problems, as well as the ability of the algorithms to discover formerly unknown knowledge and to identify implicit relationships in datasets. The challenges that the organisations face, are almost always, the data collection, preprocessing and algorithm selection, but also there is a high need to be able to explain the model's prediction or in other words it is almost necessary, at some degree, the interpretation of the results.

From the previous presented papers, we conclude that the intepretability constitutes a hot topic in manufacturing.

3.2 ML interpretability

Based on Molnar, 2018, [13], interpretability is the degree to which the human can understand the cause of a decision. The higher the interpretability of a ML model, the easiest to explain the result of the output. interpretability is very useful for a ML model because it converts it from a black box, into a system in which we can:

- ensure that a prediction is not biased,
- ensure that sensitive information is protected,
- ensure that small changes to specific features does not change entirely the output of the model,
- being able to understand and explain the decision and so being able to know how to change the input in order to get a specific output,
- it's easier for humans to trust them.

Carvalho et al., 2019 [5] presented a survey on methods and metrics regarding the interpretability. The evolution of the interpretability's field started from the increase of data collection and the rise of ML usage. Authors point that even if the ML as a tool is powerfull, in the way that is being used, it lacks transparency, and this has triggered global awareness. Interpretability's research areas can be separated into the following three research areas:

- Data science, study of ML algorithms,
- Human science, understand human nature and try to give a human acceptable explanation of the algorithms,
- Human computer interaction, study interaction and improve the trust between the humans-users and the computer-models.

The main reason interpretability requires our attention is because, a single metric, as the accuracy, cannot describe entirely a model's success. Also it can help to extract more information regarding the decision - output of the model. In some cases we may not really care about the reason the model conclude to an output, however in most of real world scenarios the explanation is required. interpretability potentially can offer safety as we can explain a taken decision and we can prove the fairness and that it is biased free.

ML interpretability can be classified according to different criteria:

- Pre/In/After-model, based on which step the explanation tried to be given.
- Intrinsic or Post-hoc, is another way to classify whether the interpretability achieved through constraints imposed on the complexity of the ML model (intrinsic) or by applying methods that analyze the model after training (post hoc).
- Model specific or Model agnostic. Model specific means that we are studying model specific parameters, like the weight of a linear regression model, while model agnostic means that we are studying the features of the dataset.

- Results that each method produce, this includes: feature summary - statistic explanations for features, model internals - explanation output of all intrinsic models, data point - methods that return a data point to make a model interpret-able.

Regarding the interpretability's scope, the main goal is the transparency and the comprehensions of the model.

The easiest way to achieve interpretability is by using algorithms that produce interpretable models. Algorithms like these are linear regression, logistic regression, decision trees, etc. Rubin, 2019 [15] presented the problems of explainable ML and the challenges that comes with interpretability. He expressed the opinion that because ML started getting used very frequent in high stakes predictions which impact human lives, black box models should be replaced by interpretable ones. Apart from that, interpretability helps also in model's troubleshooting. As black box the author consider either a function that is too complicated for a human - like neural networks, or a function that is proprietary. Regarding explainable ML, he supports that:

- It provides explanations that are not faithful to what the original model computes, that's because the explanation model cannot have perfect fidelity with respect to the original model as if they were, they would be the same,
- Explanations often do not make sense or do not provide enough detail to understand what the black box is doing,
- Black box models makes harder database expansion and explanations can lead to an overly complicated decision pathway.

He believes that the governments should force organization to use interpret-able models and by recognizing that this is not easy, he presents the main algorithmic challenges, which are:

- Logical model construction, like decision trees,
- Optimal sparse scoring system construction,
- Define interpretability for specific domains and create methods accordingly.

3.3 Production planning

Regarding to some more related to production planning researches, Lubosch at al., 2018 [10] studied and suggested a different way of solving production scheduling problems by combining the Monte Carlo Tree Search algorithm and ML. In that way, they tried to get omit expert knowledge. In order for Monte carlo search algorithm to work, an environment with all possible states and the actions from each state has to be defined. The transmission from one state to another one is being done with the help of the selection function. The two

most frequent selection functions are the Upper Confidence Bound and the e-Greedy. The authors tested this theory using the production of microchips wafers dataset, and although the results were really promising, they couldn't guarantee that this approach would always find the optimal solution.

Gonzalez Rodriguez at al., 2019 [7] presented a research for Industry 4.0 focused on making an intelligent decision support system for production planning based on ML. They presented a methodology for solving a Closed-Loop Supply Chain (CLSC) management problem using fuzzy logic built on ML. In CLSC the final products are obtained by either processing raw materials or components, new or re-used ones. The basic terms of CLSC are:

- Productive availability (PA), as the estimated production capacity,
- Uncertainties in the quantities (SKUN),
- Errors in the stocks demanded by the consumers (SKD), nondeterminism in the returned stock due to logistic constraints.

The existence of uncertainties (SKUN) in the quantities is one of the main challenges in production planning and it makes it hard to define strict criteria when making decisions. Fuzzy logic dealt with this issue. Decision trees ML model were also used, and trained using the 10-fold cross-validation in order to reduce the over fitting possibility. The key problem when using a fuzzy inference system, is to define the distribution of the membership function. This is solved by extracting this information from regression trees. They divided their general methodology into two categories, in the data collection - preprocessing and then the regression tree train and FIS (fuzzy inference system) creation. Their idea was applied in an Industrial Hospital Laundry with 16 tons of daily production integrated in a CLSC. In numbers this means 4000 litres of fuel and 280.000 litres of water, daily. For the models validation, the authors used the MSE for the decision tree and for the FIS the squared error difference of the predicted and the actual value. The 80% of the predictions matched the decisions taken by the experts with an error lower than 5%. Based on the authors this means that this new designed system maybe cannot entirely plan the production, but it can be used for event recognition.

Usuga Cadavid et al., 2020 [18] reviewed 69 papers related to Production Planning and Control (PPC) and analyzed them based on four different categories

- Methods, techniques and tools,
- Data sources to implement ML in PPC,
- use cases analysis,
- new tools of Industry 4.0 usage.

They grouped applications by their existence and found that there is a lack of researches related to data acquisition and exploration and constant adaptation

of proposed model to the environment dynamics. Regarding the techniques and the models, Neural Networks, Q-Learning, Decision Trees, Clustering, Regression and Ensemble learning are most frequently used. Also most common ML types are the supervised and reinforcement learning. About the tools, Matlab, R, Python and RapidMiner are used, although this may not be representative for business as the papers have a more academic character. About the data sources, the authors mention that the organisation seems to focus more on collection from information systems rather than taking advantage of the IoT sources. The most addressed use cases were Smart Planning and Scheduling and Time estimation while inventory and distribution control and Smart design of products and processes is less.

Vitui and Chen, 2021 [19] presented a methodology for capacity planning prediction, named ML Assisted Capacity Planning (MLASP). Their paper focus on industrial environments where is critical to find out the capacity of the systems, like e-commerce websites. Capacity planning constitutes a critical activity and can impact the revenue of an industry. This process usually includes the tune of many parameters, and is being done after engineers perform many different tests and understand how these parameters affect the KPIs. However the size of the parameter's space makes this work challenging and costly. The MLASP uses ML models in order to define these system's KPIs. The paper's goal was to answer which is the accuracy of this architecture and then to compare it with the same models but trained with less examples. They applied their methodology into two large scale Erickson's enterprise systems, although because of the company's security policies they were no able to publish much of internal information and result and for this reason they worked also with the open source Apache Kafka. The methodology included three main parts

- Dataset generation - running of load tests,
- Data pre-processing - feature engineering and selection,
- Apply of ML techniques.

About the ML models, tree-based (Random forest, XGBoost), deep neural networks (MLP, CNN, LSTM) and traditional (linear regression) models used. For performance evaluation, many different metrics used but the authors focused mostly on Mean Absolute Percentage Error (MAPE). The MAPE values were low for XGBoost AND MLP neural network which may indicate that these models may be better at capturing the relationship between the configuration parameters and the system throughput. On the subset of tests experiments, the XGBoost achieved the best results compared to the other models.

3.4 Dissertation context

Inspired by the described related work, in the context of this dissertation, a research over a manufacturing optimization problem was done. The optimization problem is modeled as a knapsack-like and will be explained and formalized in

the next section. We will present a data driven tool that operates as a decision support alongside with an already developed and existing combinatorial algorithm which solves it, in order to minimize the waste of a paper industry given a set of constraints. In contrast with the research already conducted around the content, we approach the optimization problem as a black box and we focuse more on the connection between the input and the output by examining the interpretability of the proposed method.

4 Problem definition

This section contains an introduction into the paper industry, the definition of the problem.

4.1 Paper industry

Paper industry is the manufacturing sector that is associated with the production of any type of paper. Paper production usually starts with the production of large reels - also called jumbo through the factory's Primary machine. These jumbo reels are usually very large, both in length and width and rarely end up being the final product that will be shipped to the customers. There are many different routes that a jumbo reel can follow in order to be converted into a final product. In one of the most common, the jumbo get cut both in length and width through the winders. This is the point of production where if the right choices are made the wastage can be minimized to a great extent.

The primary machine usually converts raw material into paper. It can produce many different types of paper, depending on the final products of the factory. The type of paper is related to the recipe of the raw material which is fed to it. The production of each different type of paper, should be done based on the existing orders, and forecasts based on experience. This happens because it is very difficult to store this type of reel due to their size. It is also impossible to constantly switch from one type to another because this requires a lot of time which means a reduction in profit. For this reason, production planners try to fulfil the highest quantity of a specific type, during when the machine is set to produce it, in order to fulfil as many orders as they can and also, keep the number of alternations low. The production unit of a particular type of paper is called a run. The machine can produce a certain range of paper width (which is quite large) and this is related to its construction.

Winders is usually the type of machine that comes as a second step in production. As input take the reels produced by the primary machine and cut them into smaller ones both in width and length.

Run is the unit of measurement for producing a particular type of paper. A run contains orders which in turn contain the number of reels and the widths that a customer wants.

Example of runs:

- RunA,
    1. OrderA1,
        (a) 1 reel of 4000mm width,
        (b) 2 reels of 800mm width,
    2. OrderA2,
        (a) 3 reels of 3200mm width,

(b) 1 reel of 5000mm width,
(c) 6 reels of 700mm width,

3. OrderA3,
   (a) 15 reels of 600mm width,
   (b) 10 reel of 500mm width,

- RunB
  1. OrderB1,
     (a) 15 reel of 1500mm width,
     (b) 1 reels of 6000mm width,
     (c) 23 reels of 2450mm width,
     (d) 15 reels of 1230mm width,
  2. OrderB2,
     (a) ...
  3. ...

The **manufacturer** goal is to reduce the waste and therefore to produce these runs in the most efficient way. The **planner** is the person inside the manufacturing organisation, who defines which orders will be included in a run.

There are many different ways to produce the orders of a run. The goal is to perform the production using the most efficient way, or in other words, the one that reduces the most the waste. For example, supposing that a primary machine produces jumbos of 3600mm and using them needs to fulfill an order of 6 reels of 1000mm. These reels production can be done, in many different ways, like for example:

- get 1 reel of 1000mm from every jumbo of 3600mm (2600mm waste, 72%)
- get 2 reel of 1000mm from every jumbo of 3600mm (1600mm waste, 44%)
- get 3 reel of 1000mm from every jumbo of 3600mm (600mm waste, 16%)

For this case, the ideal production is to use 2 jumbos, and cut 3 reels from each one of them. However this is a very simple example, and most of the times the runs contain a big number of different widths and number of reels, which makes them hard to be planned from a human. Here comes Greycon, which using XTrim in order to solve the combinatorial optimization problem and returns the best plan for the set of orders that the planner chose under a specific run. This is being done, by providing the patterns to cut the jumbos for the winder machines. XTrim receives, a run - set of orders to be produced and provides the most efficient plan for the cutting. The planner decides which orders he wants to fulfill inside a specific run and asks from the XTrim to provide the plan - how to cut them. This can be a very time consuming process as for each change the planner performs, he has to manually deploy again the XTrim's

Figure 3: Jumbo reels waiting to be loaded

algorithm and wait to check whether the new produced plan has less or higher waste percentage.

The "solution" is the plan provided by XTrim. In figure 4, we can see a solution. Every row refers to a jumbo. The grey boxes represent ordered reels while the light blue the wasted part of the jumbo.

4.2 The problem

After the planner decides the orders to be included in a run, asks from the XTrim to provide the solution. The solution is the ideal plan for these orders, and is the one with the minimum waste. Although, this minimum waste sometimes may be more than a desirable upper bound value, and so the planner may have to edit the orders, by removing or adding new ones. After the edit of the run, the planner has to re-run the XTrim's algorithm and wait for the new solution to check whether is better or not. The new solution is not provided instantly, but requires some time to get calculated, based on the given parameters. Repeating these actions again and again during a work day, constitutes a time consuming

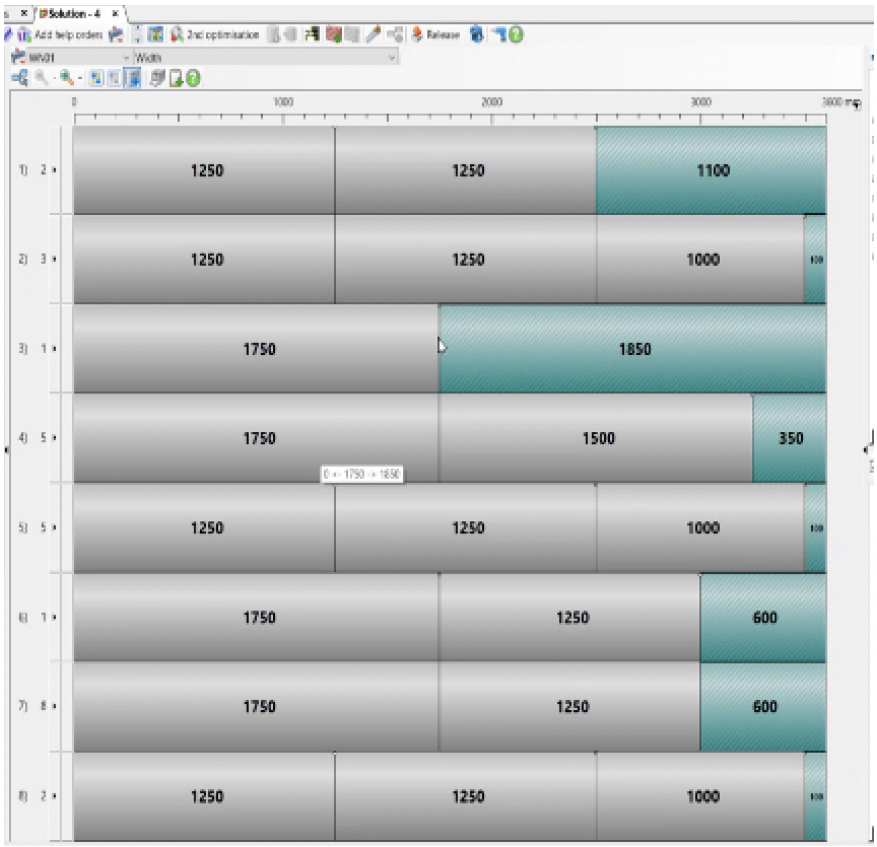

Figure 4: XTrim solution

process. Goal of this work, is to develop a model which will predict the total waste inside an acceptable space, without having to run the XTrim's algorithm, and to give back to the user, an instant feedback about the given orders. The feedback will be an anticipation about the potential waste. The model will act as an experienced planner who, may know in advance that a specific set of orders can end up in a good solution. After discussions with experts, the target in order for the tool to be valuable, is set to $MAE \leq 1$.

The problem can be modeled as knapsack-like optimization procedure. We are trying to find the least set of bags, which fits all the available items and which items belong to which bag. In that way we will achieve the minimum waste for the specific combination of input variables. For our case, the bag is the input jumbo's width, while the ordered reels are the items that we are trying to fit in.

Our problem can be mathematically formulated as follows. Given:

- a set of objects $O = o_i$, $0 < i < N1$, and w_s, $0 < s < N2$ each item's

weight/value in O, we want to find a partitioning P of O, creating K groups of $G_j, 0 < j <= K$, where $G_j \subset O$,

- the capacity of a machine is M,
- Each group G_j have a waste

$$waste(G_j) = M - \sum (w_l, w_l \in G_j) \tag{5}$$

and the overall waste can be formulated as

$$W(P) = \sum waste(G_j) \tag{6}$$

We search for a P that minimizes the following: Given that M is each bags/-machine capacity and w_i each items weight, we are trying to find K, which represents the number of bags, in which we can fit all the items.

Minimize W(P), K

Subject to

$$\sum (w_i) < M, w_i \in (G_j) \tag{7}$$

We can represent the assignment of a w_i to a G_j, through a membership variable m(w_i, G_j), where m equals to 1 when w_i has been assigned to G_j, otherwise m equals to 0. Searching P is equivalent to finding the m function.

Our motivation is to employ a data driven approach that approximates the total waste given a set of initial constraints referring to the number of items, number of rolls etc.

We want, given a set of orders S, to select a partitioning of the orders B(S), such that the sum of waste across all partitions S' generated by B(S) is minimum. Calculating the waste for a S' is very expensive. We create an estimation function W(S') which approximates the waste. W(S') is our machine learning model.

We focus on W(S') because if it is efficient enough, we can perform even exhaustive search to find a good enough solution for the B(S), in a fragment of the time we would need using the full, combinatorial solution.

By using the W(S') approximator we will be even able to find the best set of inputs, by checking and estimating the waste of all possible input combinations.

For example, supposing that we have three orders, the o1, o2, o3. We can have the following six possible partitions, [o1], [o2], [o3], [o1, o2], [o1, o3], [o2, o3], [o1, o2, o3]. Without our model, in order to find which of these partitions provide the best plan in terms of giving the min waste, we would have to solve the combinatorial problem seven times, which means seven multiplied by, on average, one minute. Using the approximator that we will try to employ we will be able to have an instant estimation about the total waste of all these partitions. This means that we will reduce the required time to solve the combinatorial problem from seven to one. This example consist of only three orders. In real

word we have on average more than twenty orders in each plan and so the advantage that we get is huge.

From ML perspective, the problem can be addressed both as regression and classification. In case of regression, we will try to predict the exact percentage of waste, while in the case of classification we will have to define classes and predict a specific class. For example two or more classes can be defined based on the total waste value (%) as:

- $Totalwaste(\%) \leq 3$, good waste, class 0,
- $Totalwaste(\%) > 3$, bad waste, class 1.

The classification approach might render easier the model's work, however a smaller error might be more beneficial. For example, suppose we have a run with total waste (%) of 2.9, a prediction of total waste (%) of 3.1, in regression means 0.2 error which is acceptable while in classification means miss classification.

5 Dataset

In this section we will present the data generation and datasets construction by exploring the metadata.

5.1 Dataset generator

For the train and evaluation process of the models, data from a dataset generator tool was used. This was an already existing project developed from Greycon, and it automates the process of: generate random runs and provide the XTrim algorithm's solution. The user defines a set of rules and the tool generates the runs based on then. For example, we can ask to generate N runs with:

- Order of type A,
  - 1 to 40 reels,
  - widths between 500-6500mm,
- Order of type B,
  - 1 to 5 reels,
  - widths between 100-700mm,

The rules that where utilized in order to construct the datasets for the experiments are shown in table 1. The artificial datasets were constructed in a way to mimic a real production situation. Each cell represents a category. For example, A represents the existence of 1-10 reels of 1000-2000 mm width in a run. The combinations of runs constructed for the dataset are the following: ACE, ACF, ADE, ADF, BCE, BDE, BDF.

For each one of these categories, we generated 50 runs, so the dataset contains in total 350 instances.

The Dataset generator generates the metadata and from them a dataset was constructed.
For the datasets construction it is necessary to focus on:

- Number of ordered reels per order,
- Width of ordered reels,
- Solution's total waste (%)

Following we see, how the orders table looks. Each row represents an order and contains:

- The Run in which the order belongs,
- The Order's name,
- The Number of reels. The number of ordered reels of a specific width from a customer

Dataset rules		
Widths/ Number of reels	1-10	11-40
1000-2000	A	B
2000-3500	C	D
3500-5000	E	F

Table 1: Dataset rules.

- Width (mm). The width in mm of the ordered reels

Apart from the order's related information, we need also the results for these runs after running the algorithm. This information includes:

- The run number,
- Total waste (%), the XTrim solution achieved.

The challenge with the dataset construction is how we will achieve this information to be represented in feature sets of stable size.

5.2 Statistical dataset

The objective is, given a set of orders, estimate the total waste percentage. This means that based on our metadata, we need to find a way to represents the orders in a vector of stable size. For example, for the table 2 we have with green the orders of the run 1, with yellow the orders of the run 2 and with blue the orders of the run 3. This means that we have:

- 4 rows, reference Run 1,
- 2 rows, reference Run 2,
- 3 rows, reference Run 3,

So the challenge is that we want somehow to depict the information of this variable number of orders into a specific number of features. In the first dataset attempt, each row represents a solution for a run. The information from the orders is presented using statistical values. These are:

- Number of orders, the number of different orders in the run,
- Number of distinct widths, the number of different widths in the run,
- Average width, the average width (mm),
- Max width, the maximum width (mm),
- Min width, the minimum width (mm),

Orders			
Run	Order	Number of reels	Width (mm)
1	1A	5	100
1	1B	12	100
1	1C	32	500
1	1D	17	1600
2	2A	19	700
2	2B	90	900
3	3A	43	500
3	3B	120	600
3	3C	4	1000

Solutions	
Run	Total waste (%)
1	3
2	5
3	1

Table 2: Orders classified by run.

- Deviation, the width's deviation,
- Weighted average, the average of multiplication of number of reels with the widths.
- Average number of reels, the average number of ordered reels.

For example for the previous presented tables the dataset looks as presented in table 3.

Statistical dataset									
Run	Number of order	Number of distinct widths	Average width(mm)	Max width(mm)	Min width(mm)	Deviation	Weighted average	Average number of reels	Total waste (%)
1	4	3	575	1600	100	1507500	123	16.5	3
2	2	2	800	1600	100	1207500	123	54.5	5
3	3	3	700	1000	500	140000	123	55.7	1

Table 3: Statistical dataset.

5.3 Extended dataset

There are multiple different ways that this problem could be approached. In effort to include more information in the dataset and inspired from natural language processing, we tried to find a way to map different widths with existed instances. If every unique width was treated alone, then this would mean that we would have a very big in feature size dataset. Every unique width would have to be a separate feature and this feature would represent the number of instances

Orders			
Run	Order	Number of reels	Width (mm)
1	1A	5	100
1	1B	12	100
1	1C	32	500
1	1D	17	1600
2	2A	19	700
2	2B	90	900
3	3A	43	500
3	3B	120	600
3	3C	4	1000

Solutions		
Run	Total waste (%)	Class
1	3	0
2	5	1
3	1	0

Table 4: Classification approach.

of this width in the run. The dataset would have to contain features from zero mm to jumbos width size mm, with a step of 10mm. Also we would need a lot more data, in order to cover all different possibilities or else, the models would end up to be biased from the widths of the training process. In order this to be avoided, but also to improve the contained information, in this second approach of the dataset, we extended the already existing information by following the idea of dividing the input jumbo in ten pieces, and extract the statistical values of the previous dataset for each of these pieces. This would mean that it will be easier for the model to construct rules like, "as bigger is the width of the reels which have width between $[0, jumboswidth/10]$, as harder it is to make a good solution". Also this way of modeling allows the categorization/quantification of the widths into small ones, slightly biggers, slightly b, etc..

5.4 Classification approach

As already presented and discussed in the previous section, the problem can be addressed also as a classification, as shown in table (4). For the classification's dataset both statistical and extended dataset used again and the Total waste (%) column replaced by the class.

6 Statistical dataset regression experiments

This section contains data pre-processing, visualization and the regression experiments for the statistical dataset. The follwoing section is structured a follows:

- Data visualization
- ML simple models evaluation
- Data preprocessing
- ML simple models tuning and interpretability
- NN experiments
- Generalization tests

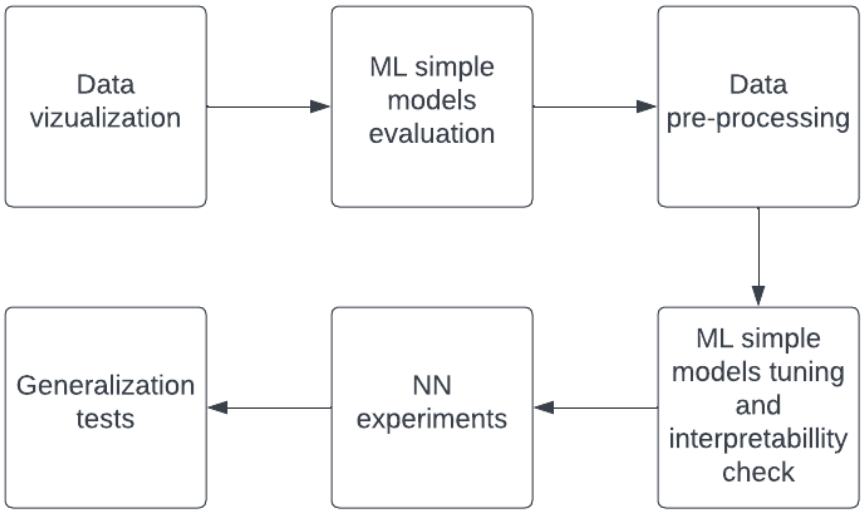

Figure 5: Experiments flow

At the end of this section, we will be able to answer:

- Do we need to pre-process the data? And which pre-process technique helps the models to perform better?
- Which features should be used?
- Can we achieve to perform a MAE of ≤ 1? Can we explain models predictions?
- Which is the best way to construct a base dataset (which will be able to define a model with better generalization capabilities)?

6.1 Data visualization

Starting from the statistical dataset, we will start by analyzing it and getting some insights about the features.

The dataset contains 350 runs. The most of the solutions total waste (%) belongs between $[0, 10]$. The model is necessary to be able to predict the solutions inside this space. Out of this space, the produced solution means that is already going to be bad, so it is not necessary to be able to predict an accurate value.

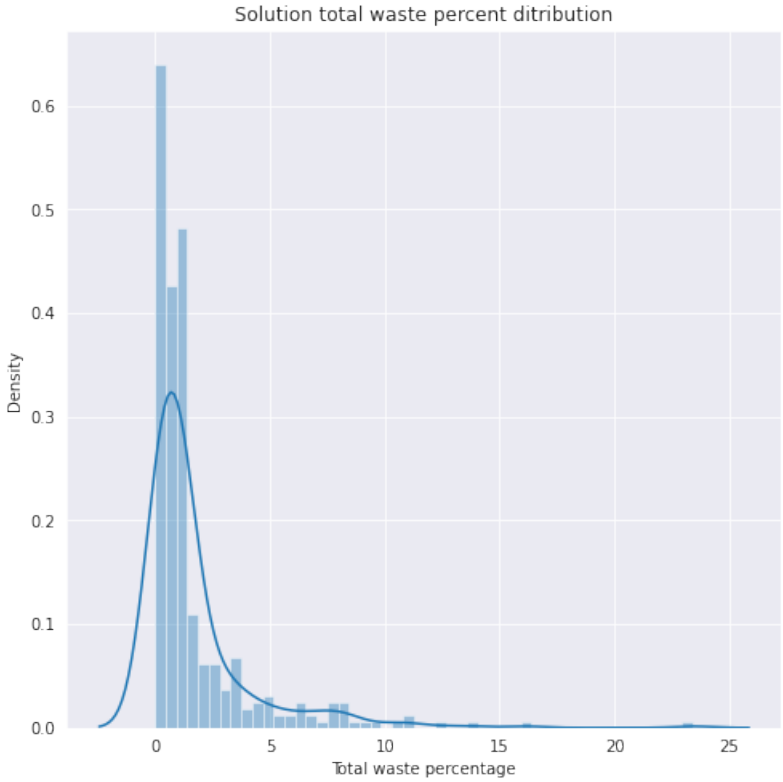

Figure 6: Total waste percentage distribution

The basic attributes of the features can be found in table 5.

After plotting the pairwise relationships between the variables, there is no any obvious observation that can be done.

The correlation matrix presents the correlation between the variables. The correlation values are between -1 and 1. Negative correlation means that when the value of the one variable increases the other one decrease while positive correlations means that both increase or decrease together. As closer to the bounds of -1, 1 the correlation value is, more correlated the variables are, and it

Describe features								
	Max width	Min width	Number of orders	Average width	Deviation	Average number of reels	Weighted average	Total waste (%)
count	350	350	350	350	350	350	350	350
mean	4784.28	685.14	125.21	2621.95	1123.79	961.45	2616.49	1.72
std	296.51	302.7228	73.23	505.29	200.41	548.37	510.07	2.62
min	3500	500	18	1316.66	566.14	124.27	1197.28	0.00
25%	4700	500	60	2241.05	999.40	477.06	2259.9619	0.31
50%	4900	600	114	2667.36	1121.62	883.49	2664.11	0.94
75%	5000	700	165	2987.91	1267.40	1281.28	2960.87	1.63
max	5000	2000	410	3750	1644.77	2581.50	3933.29	23.41

Table 5: Describe features.

Model evaluation without pre-processing		
Regressor	MAE	APT
Random forest	0.913	0.000457
XGBoost	0.946	0.000019
Decision tree	1.18	0.000036
Polynomial of 2nd degree	1.32	0.000002
Polynomial of 3rd degree	1.37	0.000003
Bayesian ridge	1.38	0.000359
Elastic net	1.39	0.000033
Linear	1.39	0.000040

Table 6: Model evaluation without pre-processing.

means that one of them contains all the required information for all. (Han and Kamber, 2012, pp. 55, [8]). For this reason we decided, because the average number of reels is highly correlated with the number of orders, not to include the values of the latter in the optimization procedure.

6.2 ML experiments

ML models evaluation section without pre-processing.

6.2.1 Train and test dataset split

For the models evaluation, the dataset split into train and test set. The train contained the 80% of statistical data while the test was the rest 20%. K-Fold cross-validation used as well. Bengio and Grandvalet, 2004, [3], described K-Fold cross-validation as an intensive train and test procedure using all the available examples. The algorithm works as follows: divide the data into K equal subsets, and perform K times train and test the models by leaving out each time a single different subset of the data from the training process to use it for testing.

6.2.2 Metrics

For the models evaluation we focused mostly on Mean absolute error (MAE). In some tables in this document you will see also that we include the Average Predict Time. The APT is calculated by measuring the time that takes to predict the test set and dividing it by their number.

6.2.3 ML models

The ML models which used from the experiments are:

- Linear regression
- Polynomial of 2nd degree regression
- Polynomial of 3rd degree regression
- Bayesian ridge regression
- Elastic net regression
- XGBoost regression
- Random forest regression
- Decision tree regression

The previous referred regressors can be separated into the ones that construct a function which describes the data and the ones that describe the data using rules. Random forest and Decision tree are originally classification models and when they are used in regression problems they describe the data with rules - if then statements - refer applied predicting modeling book.

6.2.4 Results

In table 6 we see the results of the model evaluation without pre-processing.

6.3 Data pre-processing

In this section data pre-processing techniques and more specifically two normalization techniques that will be discussed and presented. Models will be evaluated again for each different technique.

There are multiple ways that information can be represented. For example, numbers that represent mass depend on their unit. Their value shown in kilos would be a smaller number than their value shown in grammage. This can cause issues as the models are not able to recognize these differences and transform. Models, end up giving more "importance" to features that take bigger values. In order this to be avoided data can get standardized or normalized. Normalization reduces their range from $[-\infty, +\infty] \rightarrow [-1, +1]$ while standardization reduces their range from $[-\infty, +\infty] \rightarrow [0, +1]$. Normalizing the data attempts to give all attributes an equal weight.w (Han and Kamber, 2012, pp. 112, [8]).

Model evaluation with min max scaler		
Regressor	MAE	APT
Random forest	0.92	0.000125
XGBoost	0.94	0.000005
Polynomial of 2nd degree	1.05	0.000001
Decision tree	1.20	0.000002
Bayesian ridge	1.37	0.000002
Linear	1.39	0.000002
Elastic net	1.61	0.0000003
Polynomial of 3rd degree	1.71	0.00002

Table 7: Model evaluation with min-max scaler.

6.3.1 Min max scaler

Min-max scaler uses the min-max normalization which performs a linear transformation over the original data. Supposing that the min_A and max_A represent the minimum and maximum value of a feature A, this type of normalization maps a value v_i of a feature A to new_v_i in the range of [new_min_A, new_max_A] by using the formula shown in equation 8. (Han and Kamber, 2012, pp. 112, [8]).

$$new\_v_i = \frac{v_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A \tag{8}$$

This type of normalization preserves a relationship between the v_i and new_v_i. In case we cannot be sure about the range of future used data, this cannot by used, as in case the a v_i is outside of the original min_A and max_A range, will encounter an out of bounds exception. (Han and Kamber, 2012, pp. 112, [8]).

The results after this transformation got applied in the data are shown in the table 7.

6.3.2 Standard scaler

Standardization or z-score normalization is a different normalization technique. In this type of transformation, a feature A is normalized based on it's mean and standard deviation. The formula used for this transformation type is shown in equation 9.

$$new\_v_i = \frac{v_i - \overline{A}}{\sigma_A} \tag{9}$$

The results after this transformation got applied in the data are shown in the table 8.

Even if the results seems to be quite close experiments will continue using the stardard scaler due to the following two main reasons:

Model evaluation with standard scaler		
Regressor	MAE	APT
Random forest	0.91	0.000155
XGBoost	0.94	0.000013
Polynomial of 2nd degree	1.07	0.000002
Decision tree	1.17	0.000017
Elastic net	1.31	0.0000016
Bayesian ridge	1.38	0.000015
Linear	1.39	0.000021
Polynomial of 3rd degree	1.40	0.000003

Table 8: Model evaluation with standard scaler.

Model evaluation after tuning	
Regressor	MAE
Random forest	0.885
Elastic net	1.384
Bayesian ridge	1.375
Decision tree	1.053

Table 9: Model evaluation after tuning.

- Normalization on average seems like it helps models perform better when compared with without pre-processing schems.
- Min max scaler out of bounds exception. We cannot be sure that all the future values will be inside the bounds of the data using which we trained the models and we developed the scaler.

6.4 Model tuning

Many models have parameters which cannot get directly estimated from the data. This type of parameters are named tuning parameter because there is no analytical formula available to calculate an appropriate value. (Kuhn and Johnson, 2013, pp. 64, [9]). The tuning of the models is done by using extensive grid search, in which, a set of values for each parameter is defined and an extensive search on all possible combination of them, and comparison of the results is being conducted. In the experiments we compare the MAE of each combination after 10-fold cross-validation.

The results after tuning are shown in the table 9

6.4.1 Results discussion

The advantage of the regression models against classification but in general ML models against DL, is that the results can be more easily explained. Especially

Bayesian regression - statistical dataset weights	
Column name	weight
Weighted average width	1.176609
Min width	0.433434
Average reel width	0.046081
Max width	0.012132
Deviation of widths	-0.140826
Number of ordered reels	-0.498461

Table 10: Bayesian ridge statistical dataset weights

regression models because they construct a function after all, it is quite easy by extracting the weights of each feature given from the model to the function, to understand how each one of them affects the results. On the other hand, because as already have been referred, tree models have been used and tree models are using rules, weights cannot be extracted from them as they work in a different way. Although the rules can be extracted and applied. These models, classify the input features by their importance. There are many different ways, this importance can get calculated. For these experiments, used the gini index. Based on Menze et al., 2009, [11] the purpose of gini index is to describe a feature's importance as a ranking and represents a number which is calculated as the sum of the number of splits that includes it, divided by the number of samples, it splits. In other words it is an approximation of the entropy which pass from it.

Bayesian ridge achieved the best performance from the regression models with 1.375 MAE, by having constructed the function with the weights shown in table 10. This table can be translated to "The average multiplication of widths with number of reels and the minimum width increase while the deviation of the widths and the number of ordered reels decrease the total waste percentage."

Elastic net, achieved similar performance to Bayesian ridge and more specifically, achieved 1.384 MAE. The two dominant fields which increase the waste percentage, are the same as before. The strange is that this model has entirely ignored the deviation and the number of ordered reels. The weights are in table 11.

From the other hand, the best performed regression model, Random forest, has ordered the features with not as similar sequence. As has already been referred because of its a type of rule model, does not construct weights but extracts feature importance. 12.

Finally, decision tree regressor have achieved a slightly worst performance and interesting with an almost entirely different order of the features, table 13.

6.5 NN

In this section the steps followed and results of the NN experiments will be presented. The number of the samples is not ideal for using DL, however a

Elastic net regression - statistical dataset weights	
Column name	weight
Weighted average width	1.263152
Min width	0.586354
Max width	0.181213
Deviation of widths	0.000000
Number of ordered reels	0.000000
Average reel width	-0.144277

Table 11: Elastic net statistical dataset weights

Random forest - statistical dataset feature importance	
Column name	feature importance
Min width	0.402933
Weighted average	0.230533
Average number of reels	0.164966
Average width	0.076853
Deviation	0.075120
Maximum width	0.049615

Table 12: Random forest statistical dataset features importance

Decision tree - statistical dataset feature importance	
Column name	feature importance
Average reel width	0.541101
Max width	0.222852
Min width	0.100663
Weighted average width	0.061040
Number of ordered reels	0.054209
Deviation of widths	0.020135

Table 13: Decision tree statistical dataset features importance

effort was made.

6.5.1 Architecture

Because off the number of features is quite low (6), simple model architectures are used. All the architectures started with a dense layer of input size the same as the features (6) and end with a dense layer of size of one (1). In the internal part of the network we tried with one and two extra dense layers. In every dense layer the relu activation function used except from the last one in which we assessed both relu and linear activation function. As loss function again was used MAE. Chosen optimizer was the ADAM. All models trained in a maximum of 4000 epochs with batch size of 2 and used early stopping when the validation loss does not decrease for a maximum of 10 epochs. In case of early stopping the best weights are restored.

6.5.2 Results

The best performance achieved with two internal dense layers and relu activation in the last on as show in the image 7.

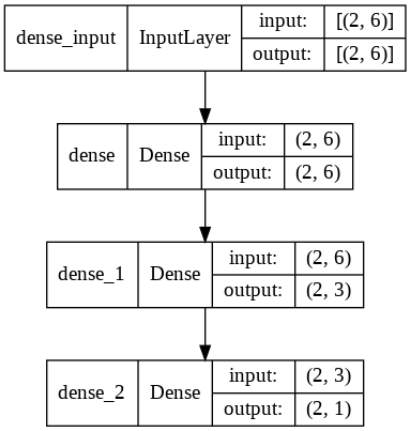

Figure 7: Statistical dataset neural network architecure

Models train stopped after 164 epochs and achieved a MAE in the validation set of 0.68. Even if the results of DL are quite better than the results of ML model, there are still reasons to use the seconds.

6.6 Generalization tests

From the previous experiments we conclude that only the Random forest regressor and DL can achieve the requested performance. In this section the generalization performance of them is tested and compared. As described in

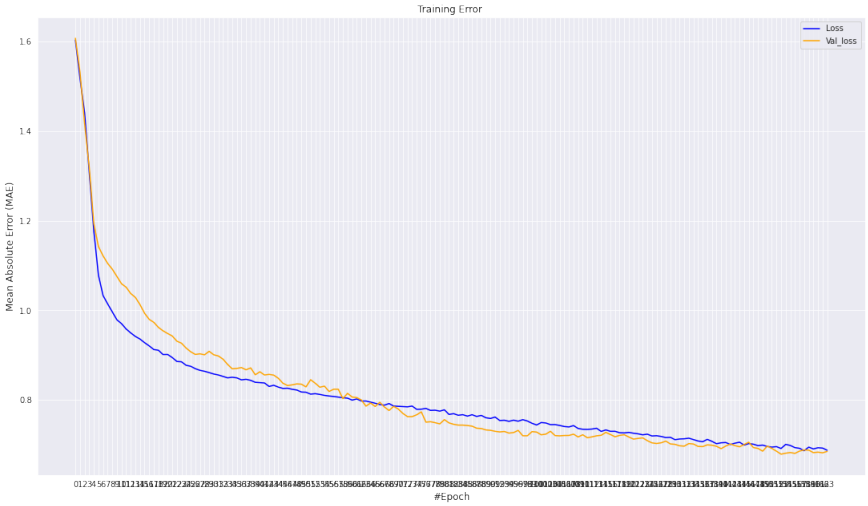

Figure 8: Loss diagrams

Generalization tests - statistical dataset		
Used for validation	Random forest regressor	DL
BDE	0.4428	0.5727
BCE	0.4445	0.6329
BDF	0.5219	0.6015
ADE	0.8123	1.3568
ACE	1.0825	2.3218
ACF	1.3695	2.8166
ADF	1.9445	3.9272

Table 14: Generalization test on statistical dataset without preprocessing

the problem's definition section the dataset contains runs of 7 different categories. Each of these categories will be used as a validation - test set on models trained with the other six. For the ML experiments, the model was trained using the whole dataset without including the runs of validation. For the DL experiments the model was trained with the 80% of the disjoint dataset, the rest 20% was used as a validation of the training process. In table 14 we see the results of these experiments without having applied any pre-processing in the data, while in the table 15 the standard scaler was constructed using the training data and was used for both the train and validation set. The method where scale the data using standard scaler seems to generalize slightly better. Also, it seems that the datasets that contain smaller number of reels, the width can be used as predictor for datasets with higher values, while the opposite is no true.

Generalization tests - statistical dataset - standard scaled		
Used for validation	Random forest regressor	DL
BDE	0.4380	0.4022
BCE	0.4460	0.4500
BDF	0.5212	0.4294
ADE	0.8130	1.0985
ACE	1.1063	1.1514
ACF	1.3688	1.4654
ADF	1.8804	3.7837

Table 15: Generalization test on statistical dataset with data standard scaled

6.7 Summary

Following back to the questions which set in the beginning of this section:

- Do we need to pre-process the data? And which pre-process technique helps the models to perform better?

  The experiments showed that data pre-processing helped the models to perform better. Normalization on average seems like it helps models perform better.

- Which features should be used?

  Correlation matrix showed high correlation between the average number of reels and the number of orders and for this reason the second one was excluded from the experiments.

- Can we achieve a MAE of ≤ 1? Can we explain models predictions?

  Random forest regressor and the presented NN architectures achieved the baseline set in the context of this work (MAE ≤ 1). Regarding the random forest regressor, it validates the findings of the related work presented in the previous sections. After tuning and using 10-Fold validation, we achieved a MAE of 0.885. The model ordered the input features as shown in table 12. Regarding NN, it achieved a MAE ≤ 1 and more specifically a MAE of 0.68. The architecture used to performed this, can be seen in image 7. Although the NN architecture performs better than the random forest regressor, it is harder to explain the predictions. So, about which model is suggested to be used, the answer depends on what we need. There is a trade off between better performance and interpretability. In case we want to provide suggestions and also to be able to explain specific predictions, it would be better to use the ML model while in case we care only about the accuracy it would be better to use the NN.

- Which is the best way to construct a base dataset?

  The experiments showed models trained on a dataset which is constructed from reels with smaller reels can achieve higher accuracy while the opposite

is not true. Furthermore the dataset based on the experiments should be normalized.

Model evaluation without pre-processing		
Regressor	MAE	APT
Random forest	0.93	0.000133
XGBoost	0.96	0.000007
Bayesian ridge	1.22	0.000034
Decision tree	1.24	0.000021
Elastic net	1.25	0.000034
Linear	1.26	0.000003
Polynomial of 2nd degree	5.10	0.000062
Polynomial of 3rd degree	2.01	0.000070

Table 16: Extended dataset - Model evaluation without pre-processing.

Model evaluation with min max scaler		
Regressor	MAE	APT
Random forest	0.93	0.000133
XGBoost	0.96	0.000007
Bayesian ridge	1.19	0.000002
Decision tree	1.25	0.000002
Linear	1.26	0.000003
Polynomial of 3rd degree	1.55	0.000070
Elastic net	1.61	0.000003
Polynomial of 2nd degree	1.82	0.000062

Table 17: Extended dataset - Model evaluation with min max scaler.

7 Extended statistical dataset regression experiments

In this section the results of the experiments using the extended statistical dataset will be shown and will be compared with the results of the statistical one. The flow which followed was the same. In this section mostly we will focus on the comparison of the simple statistical dataset with the extended one and we will answer the question which is better to use. Also an effort will be made by extracting the same number of features as we had in the previous section, using PCA.

7.1 Results and comparison

In table 16 we can see the results of the model evaluation without pre-processing, while in table 17 and in table 18 the results after having pre-processed the data with min max and standard scaler. Compared with the previous version of the dataset, the results does not seem to be as better in order to give a reason to use a more complex input and so a harder model to explain.

Model evaluation with standard scaler		
Regressor	MAE	APT
Random forest	0.92	0.000129
XGBoost	0.96	0.000008
Bayesian ridge	1.19	0.000002
Polynomial of 3rd degree	1.25	0.000089
Linear	1.26	0.000003
Decision tree	1.26	0.000002
Elastic net	1.27	0.000003
Polynomial of 2nd degree	1.77	0.000006

Table 18: Extended dataset - Model evaluation with standard scaler.

Random forest - statistical dataset feature importance	
Column name	feature importance
Average width	0.451670
Min width	0.017200
Weighted average	0.230533
Deviation	0.009526
Number of orders	0.004959
Maximum width	0.000582

Table 19: Extended dataset - Random forest statistical dataset features importance

After the models tuning, random forest regressor, achieves the best performance with MAE 0.897. The related position of the fields are not the same, so, it cannot be said that the extra information improved the already created models, but made new. The order and the importance can be found in table 19.

Regarding the deep learning experiments using this dataset, the model did not achieve similar results as using the statistical dataset. The reason which could explain this is that the number of training data was not enough in order to train the model to "learn" the extra weights.

About the generalization of this architecture, the results of the experiments can be seen in table 20. As before, the models trained with runs generated to have a smaller range of widths - reels seems like can predict quite good, runs with bigger widths - amount of reels, while the opposite is not true. Although, also it is worth to mention that in this case a mix of low and high number of reels performed better from just having trained the models with only the sets of low numbers.

Generalization tests - statistical dataset - standard scaled	
Used for validation	Random forest regressor
BDE	1.0978
ACF	1.0767
BDF	0.9911
ACE	0.9778
BCE	0.7985
ADE	0.8155
ADF	0.4942

Table 20: Generalization test on extended statistical dataset with data standard scaled

7.2 Principal component analysis

In this section, principal component analysis technique will be used in order to extract specific number of features out of the extended dataset. More specifically, in this section we tried to extract the same number of features as the statistical dataset has, and so to compare, whether the extra features of the extended dataset, can feed the models with extra information. Principal component analysis is a mathematical algorithm which reduces the dimensions of the data. This reduction is being done by identifying directions among the data, called principal components. Out of these components, new variables are getting generated, which are linear combination of the original ones. (Ringnér, 2008, [14]). Similarly as in the previous experiments, the data get first pre-processed by applying into them min max scaler or standard scaler and then tries are being done using different models and deep learning. Unfortunately with the PCA, models did not achieved to perform $MAE \leq 1$. Apart from this, using PCA would make it even harder to explain the results.

7.3 Summary

Summarising the results of this section, seems like the extra information of this dataset, didn't help the models to perform better than before. The most of the times, the results were similar, but there were cases in which the results was not as good as in the experiments of the previous section, like for example in the NN experiments. This could be related to the small amount of samples. Also, a try made, the same number of features as in the statistical dataset to be extracted from the extended, using the PCA technique, in order to check whether the extra information helped, however the results showed that did not. The answer to the question whether to utilize a simple statistical dataset or the extended, is to utilize the statistical, as using the extended did not translate to a better performance for the models. Also even for the cases in which the models performed the same, the simple statistical would be a better choice, because using it we construct simpler models and so predictions get to explained more easily.

8 Classification experiments

In this section the problem will be modeled and addressed as classification. As already have been mentioned, it is more correct for the specific case the problem to be treated as regression, however this can be a part of a bigger pipeline, in where first we check if a run is good or bad (classification's model prediction), and only in case it is good, we try to perform a more accurate prediction (regression's model prediction). So, in this section we will answer, whether we can address with success this problem as classification.

8.1 Problem definition

As presented in the problem definition section, there are two approaches that the problem can be addressed. Either as regression or as classification. In the previous two sections the results of the regression experiments presented, while now the flow and the results of the classification will. For the classification experiments, both datasets used as before. The classes made in that way in which they are specific, clear and discrete.

8.2 Experiments

The rules that classes made based on the total waste percentage, are the:

- $Totalwaste(\%) \leq 1$, class 0,
- $Totalwaste(\%) > 1$ and also $Totalwaste(\%) \leq 2$, class 1.
- $Totalwaste(\%) > 2$, class 2.

The distribution of the samples for each class can be seen in figure 9. The dominant class is the 0, with number of samples as almost the summary of the other two.
The classification models used are the following:

1. Logistic regression,
2. Gaussian Naive Bayes,
3. Random forest,
4. XGBoost,
5. K neighbors,
6. Decision tree,
7. Linear discriminant analysis,
8. SVM.

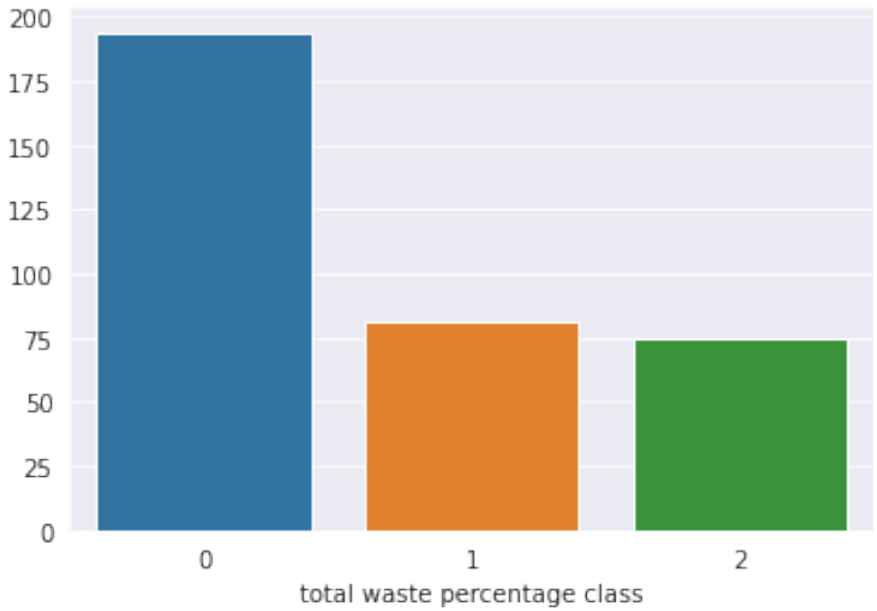

Figure 9: 1st classification approach distribution

Because of the number of samples in each class is different, models compared based on the weighted average precision value and then by their confusion matrix. As per Tharwat, 2018, [17], precision is the number of correct predicted samples, divided by all the samples of this specific class. If the correct predicted defined as True Positives (TP), and the not correct predicted False Positives (FP), the precision value is the one described in equation 10.

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

This value is being evaluated for every different class, and then the weighted average of them is evaluated.

Confusion matrix is a 2D matrix in which, the one side presents the actual values and the other the predicted. Every cell describes the correct predictions and the mis-classifications.

The results can be seen in table 21. The best performance achieved by K neighbors model, although in the confusion matrix of figure 10, it's clear that the model, have learned to separate class 0 from class 2, and has miss-classified almost all the class 1 to class 0. The model does not perform well.

A second attempt using the same dataset will be done, after fixing the imbalance of the classes by oversampling the training input using the synthetic minority oversampling technique (SMOTE), as shown in figure 11. Oversampling of minority classes helps the model to reduce the bias caused by the im-

Classification experiments - statistical dataset	
Model	Precision
K neighbors	0.65
Logistic regression	0.63
Decision tree	0.60
Random forest	0.56
XGBoost	0.56
Gaussian naive bayes	0.55
Linear disciminant analysis	0.54
SVM	0.46

Table 21: Classification experiments statistical dataset

Classification experiments - statistical dataset after oversampling	
Model	Precision
Random forest	0.59
XGBoost	0.63
K neighbors	0.56
Decision tree	0.54
Logistic regression	0.54
Gaussian naive bayes	0.48
Linear disciminant analysis	0.54
SVM	0.52

Table 22: Classification experiments statistical dataset after oversampling

balanced data. The results at these experiments are slightly better than before. The models still seems to not be able to recognize all the three classes. It appears that the oversampling does not guarantee better performance, since it only brings improvements to specific algorithms, as shown in the results table. 22.

Similar results achieved with the extended dataset and seems as like in the regression experiments that the extra information did not help the models to perform better.

8.3 Summary

Summarising the results of this section, seems like the models cannot predict with very high accuracy between 3 classes, even when the models are trained using similar number of samples for each class. Further tries and experiments should be done.]

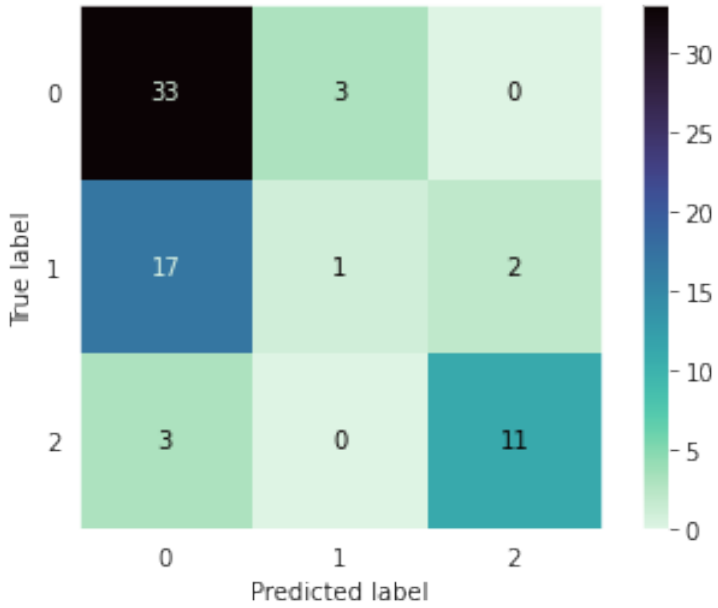

Figure 10: Statistical dataset - K neighbors confusion matrix

9 Conclusion

In this section a discussion about the results of the experiments will be conducted and also future work will be proposed.

9.1 Summary of contribution

Summarizing the work conducted in the context of this thesis we started from defining the problem continued with metadata generation and conclusively we constructed datasets that were utilized in the experiments. The problem was approached both as regression and classification. Also the interpretability of the regression models was discussed. The generalization capabilities of the proposed methods was also evaluated.

The produced and presented regression models can help organizations to save time and increase their productivity, but significantly reducing the time needed to select a production plan. Coupling the estimator with an automated parameter generator can allow a very quick search in the parameter space of a production plan to reduce the waste. This reduced the need for an expert who would suggest specific input parameters, thus allowing people with little domain knowledge to suggest promising production plans. The search process itself provides suggestions to reduce production waste.

Following back to the questions that were set in the introduction, the ques-

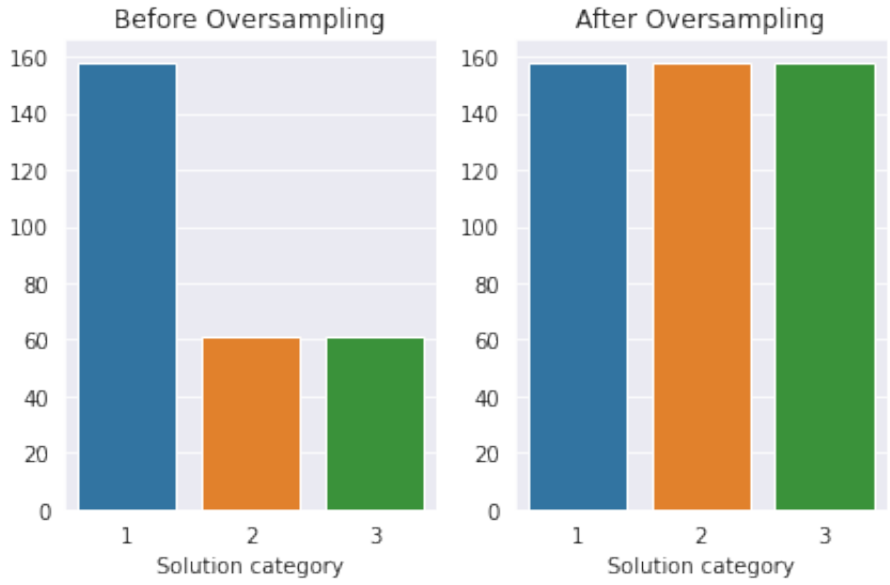

Figure 11: Statistical dataset classification approach distribution after oversampling

tions that answered are:

- Which is the best way to model a dataset in this knapsack-like problem. Does the extra information help the ML models perform better? Which samples can describe a bigger part of the input space?
- Which ML model performs the best and why?
- Can we explain the predictions?

9.1.1 Dataset

The experiments showed that using both the statistical and the extended statistical datasets helped the models to predict accurately the total waste. The performance was almost the same with the statistical dataset, that performed slightly better in some cases like in NN and in Random forest regressor experiments, while the extended statistical dataset performed better to some others, like in Bayesian ridge regressor. The extra information that was passed into the models, did not seemed to help them in terms of accuracy. Furthermore it had a negative impact on the interpretability of them. The correlated position of the common fields was not the same, meaning that the models constructed using the second dataset are not the same as the ones constructed with the first, fed with extra information. The features comprising the Statistical datasets seemed

enough (as shown by the experiments conducted) for the models to have an acceptable performance in terms of accuracy within statistical significance. The generalization of the models that achieved the goal was also also by taking advantage of the way the dataset was constructed (table 1). Out of the six groups of runs we were leaving one of them out on, and we were training the models with the other five. The experiments showed that the ones trained with the groups which contained reels with lower width values can construct predictor in comparison with the ones with greater width values. Especially when a model gets trained with a combination of big and small groups it performs better. Also the normalization of the data, helped the models to generalize better.

9.1.2 Which ML model performs the best and why?

Baseline ML models and custom NN architectures were both utilized in the context of this work. Both approaches seemed to pass the threshold of MAE that was set for the purposes of this work, as demonstrated in the experiments in section 4. Custom NN architectures performed better than ML algorithms, while the latter provided easier explainability - interpretability of the results. In case that it is not mandatory to focus on the explanation of the prediction, or we do not want to construct a system which will also give suggestions, the author of this work proposes that a Deep Learning model should be used, while in the other cases, simple ML models can also be utilized.

9.1.3 Can we explain the predictions

Interpretability by its self is a research subject. For the experiments of the context of this thesis, the interpretability only of the simple machine learning models was examined. The explanation of the prediction of the models, increase our confidence to use them, and help us to understand how the models work.

9.2 Future work

As future work, this thesis can be extended to:

- Focus on model's interpretability - explainability. Give suggestions for the input in order to reduce the total waste percentage.
- Construct a reinforcement learning agent which will use any of the already constructed models as a reward function. This agent will act as a planner.

References

[1] Charu C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. en. Cham: Springer International Publishing, 2018. ISBN: 978-3-319-94462-3 978-3-319-94463-0. DOI: 10.1007/978-3-319-94463-0. URL: http://link.springer.com/10.1007/978-3-319-94463-0 (visited on 02/20/2022).

[2] Adel Ahmed et al. "Knowledge-Based Systems Survey". en. In: 3.7 (2019), p. 22.

[3] Yoshua Bengio and Yves Grandvalet. "No Unbiased Estimator of the Variance of K-Fold Cross-Validation". en. In: (), p. 17.

[4] Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, eds. *Advanced lectures on machine learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003 [and] Tübingen, Germany, August 4-16, 2003: revised lectures*. en. Lecture notes in computer science, Lecture notes in artificial intelligence 3176. Meeting Name: Machine Learning Summer School OCLC: ocm56492380. Berlin ; New York: Springer, 2004. ISBN: 978-3-540-23122-6.

[5] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. "Machine Learning Interpretability: A Survey on Methods and Metrics". en. In: *Electronics* 8.8 (July 2019), p. 832. ISSN: 2079-9292. DOI: 10.3390/electronics8080832. URL: https://www.mdpi.com/2079-9292/8/8/832 (visited on 10/17/2021).

[6] Edwin K P Chong. "An Introduction to Optimization". en. In: (), p. 642.

[7] Germán González Rodríguez, Jose M. Gonzalez-Cava, and Juan Albino Méndez Pérez. "An intelligent decision support system for production planning based on machine learning". en. In: *Journal of Intelligent Manufacturing* 31.5 (June 2020), pp. 1257–1273. ISSN: 0956-5515, 1572-8145. DOI: 10.1007/s10845-019-01510-y. URL: http://link.springer.com/10.1007/s10845-019-01510-y (visited on 10/24/2021).

[8] Jiawei Han and Micheline Kamber. *Data mining: concepts and techniques*. 3rd ed. Burlington, MA: Elsevier, 2012. ISBN: 978-0-12-381479-1.

[9] Max Kuhn and Kjell Johnson. *Applied Predictive Modeling*. en. New York, NY: Springer New York, 2013. ISBN: 978-1-4614-6848-6 978-1-4614-6849-3. DOI: 10.1007/978-1-4614-6849-3. URL: http://link.springer.com/10.1007/978-1-4614-6849-3 (visited on 02/19/2022).

[10] Marco Lubosch, Martin Kunath, and Herwig Winkler. "Industrial scheduling with Monte Carlo tree search and machine learning". en. In: *Procedia CIRP* 72 (2018), pp. 1283–1287. ISSN: 22128271. DOI: 10.1016/j.procir.2018.03.171. URL: https://linkinghub.elsevier.com/retrieve/pii/S2212827118303299 (visited on 10/02/2021).

[11] Bjoern H Menze et al. “A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data”. en. In: *BMC Bioinformatics* 10.1 (Dec. 2009), p. 213. ISSN: 1471-2105. DOI: 10.1186/1471-2105-10-213. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-10-213 (visited on 01/30/2022).

[12] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. en. Second edition. Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2018. ISBN: 978-0-262-03940-6.

[13] Christoph Molnar. “Interpretable Machine Learning”. en. In: (), p. 431.

[14] Markus Ringnér. “What is principal component analysis?” en. In: *Nature Biotechnology* 26.3 (Mar. 2008), pp. 303–304. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt0308-303. URL: http://www.nature.com/articles/nbt0308-303 (visited on 02/12/2022).

[15] Cynthia Rudin. “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead”. en. In: *Nature Machine Intelligence* 1.5 (May 2019), pp. 206–215. ISSN: 2522-5839. DOI: 10.1038/s42256-019-0048-x. URL: http://www.nature.com/articles/s42256-019-0048-x (visited on 10/24/2021).

[16] Klaus Schwab. “The Fourth Industrial Revolution”. en. In: (), p. 172.

[17] Alaa Tharwat. “Classification assessment methods”. en. In: *Applied Computing and Informatics* 17.1 (Jan. 2021), pp. 168–192. ISSN: 2634-1964, 2210-8327. DOI: 10.1016/j.aci.2018.08.003. URL: https://www.emerald.com/insight/content/doi/10.1016/j.aci.2018.08.003/full/html (visited on 02/12/2022).

[18] Juan Pablo Usuga Cadavid et al. “Machine learning applied in production planning and control: a state-of-the-art in the era of industry 4.0”. en. In: *Journal of Intelligent Manufacturing* 31.6 (Aug. 2020), pp. 1531–1558. ISSN: 0956-5515, 1572-8145. DOI: 10.1007/s10845-019-01531-7. URL: http://link.springer.com/10.1007/s10845-019-01531-7 (visited on 10/31/2021).

[19] Arthur Vitui and Tse-Hsun Chen. “MLASP: Machine learning assisted capacity planning: An industrial experience report”. en. In: *Empirical Software Engineering* 26.5 (Sept. 2021), p. 87. ISSN: 1382-3256, 1573-7616. DOI: 10.1007/s10664-021-09994-0. URL: https://link.springer.com/10.1007/s10664-021-09994-0 (visited on 11/06/2021).

[20] Dorina Weichert et al. “A review of machine learning for the optimization of production processes”. ς. In: *The International Journal of Advanced Manufacturing Technology* 104.5-8 (Oct. 2019), pp. 1889–1902. ISSN: 0268-3768, 1433-3015. DOI: 10.1007/s00170-019-03988-5. URL: http://link.springer.com/10.1007/s00170-019-03988-5 (visited on 09/06/2021).

[21] Thorsten Wuest et al. “Machine learning in manufacturing: advantages, challenges, and applications”. en. In: *Production & Manufacturing Research* 4.1 (Jan. 2016), pp. 23–45. ISSN: 2169-3277. DOI: 10.1080/21693277.2016.1192517. URL: https://www.tandfonline.com/doi/full/10.1080/21693277.2016.1192517 (visited on 09/11/2021).