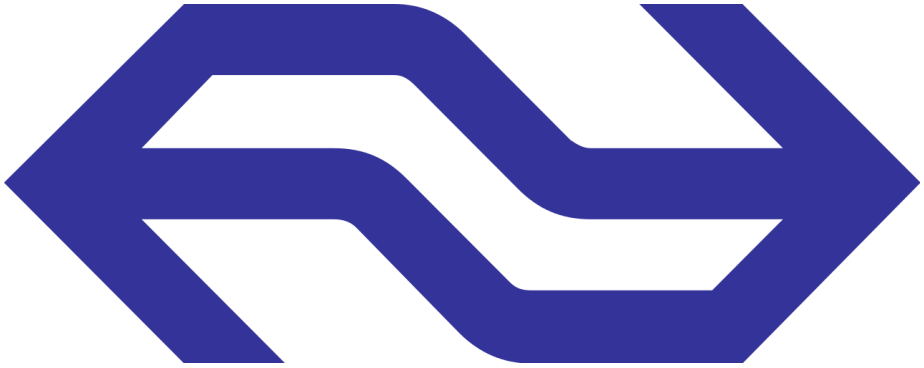
PROCESS IMPROVEMENT USING MACHINE LEARNING



UNIVERSITY OF TWENTE.

Daniël Marc Johannus van Horn, S2397838.

Dr. rer. nat. D. Braun, 1st supervisor

Ir. Mick Pas, Company supervisor

dr. E. Topan, 2nd supervisor

Foreword.

I want to thank Mick Pas and Daniel Braun for supervising me and giving guidance and constructive feedback during the entire process. Without them this thesis would not have been possible. I also want to thank Engin Topan for his feedback on the entire thesis, his help was invaluable in increasing the quality of this work. Lastly I want to thank the NS for providing me with this opportunity and wonderful experience working at the factory.

All sensitive production information has been multiplied by factor X to preserve the privacy and business interest of the NS

Management summary

The NS revises its wheelsets in the factory hall 12 in its repair complex in Haarlem. This complex is advanced in its data collection and logs almost all actions performed in the factory. The current average speed in Haarlem averages 150 wheelsets per week. The factory would like this to be increased to 180 to provide more operational slack and to accommodate future growth.

The non-linear flow through the factory system and its complexity make analysis causal analysis difficult. That is why machine learning has been chosen. This is because machine learning has the potential to find deeper patterns in the data and can potentially find connections that people cannot.

The factory itself is extremely modern having been built five years ago. Every step of the process is logged, providing a relatively data-rich environment. There are nineteen different sets of wheelsets, which can be discretized into seven distinct classes to reduce complexity. Each class has the same inherent characteristics. A combination of wheelsets in a particular input order is called a strain.

A KPI has been made to track the time in the system between the starting positions and the bottleneck. The data about each wheelset from 2022 and, partially, from 2021 is then transformed into the strains, using the KPI for each wheelset with a weighted average for all included wheelsets per strain.

Among regression and classification, the choice of regression was made because the factory has no classes about performance and numeric predictions are easier to rank. The algorithms are split up into two, the “standard” and “complex” algorithms. This distinction is made because the “standard” algorithms are likely to produce easier-to-interpret results but less predictive performance and the “complex” algorithms are the opposite.

The algorithms are run on the data and the best-performing algorithm is SMOreg, with 77.28% relative absolute error. This percentage shows that the model could not fit the data to an acceptable degree for predictive purposes. This means that, with the currently available data volume and measured attributes, that machine learning cannot influence the time in the system through input order combinations. The recommendations are then as follows. Firstly, improve how certain attributes are logged and available for data mining. Secondly, try to predict axle rejections before the axles enter the system as this can have a major impact on the system. Thirdly, do a waiting time analysis before each station in the factory to see where the highest average waiting time is. This is to see if a certain station requires extra employees, or needs to be run parallel to handle the required workload.

Contents

List of figures & tables

1. Introduction

This chapter explains the situation at NS technical Haarlem and its current problem. The action problem is established using the difference between the norm and reality, and this is then used to construct a problem cluster to find a core problem to tackle. The research question is established and the corresponding research design is explained. This is synthesized into the problem-solving approach. The action problem, the amount of wheelsets produced, is the driving factor. The use of machine learning is explained and the requirements are listed. These requirements for machine learning for the input process are then used as the basis for the research design.

NS technical Haarlem

NS Technical Haarlem is the factory in Haarlem which revises trains after about ten years of service or a million kilometers on the rails. With around 900 staff and several production halls, the complex is big and has many different operations going on at the same time. The wheelset revision in hall 12 revises the 19 distinct kinds of wheel sets which are used for the different trains currently used in the Netherlands. Each wheelset must pass through hall 12 for inspection and repair and when necessary, replacement. The factory in hall twelve is state of the art and was built five years ago. At every station, the beginning and end of every step are logged in the mechanical enterprise system (MES), which makes for a very data-rich environment. The factory has about 60 potential steps which a wheelset can go through. The wheelsets have a non-linear path through the factory, affecting the situation's performance and complexity. The configuration is the determining factor for what path a wheelset takes through the system. A wheelset with an electromotor will have to follow a different path than a wheelset without an electromotor but with braking plates. There are 19 different wheelsets in production in six distinct configurations.

Management problem

The factory hall in Haarlem was built five years ago with the assumption that the production of 180 wheelsets per week, with a yearly volume of 9000, was possible. The theoretical working time of each station was 90 minutes per action, 48 per day, and 240 a week, and can complete each task in less than 90 minutes. It is however unable to complete wheelsets per day. The current operational conditions produce 30 wheelsets completed per day right now. The problem of the factory does not seem to stem from the individual stations but the whole system. The factory’s design philosophy had lean as its core principle in mind with each station being able to complete each task in less than thirty minutes in theory. The initial factory had only one buffer station between each production stage in the process, which made it extremely vulnerable to malfunctions or bottlenecks. . After five years the factory has changed into something that fits operational needs better. This has been achieved by increasing buffer space before the paint shop, adding another crane for transport in a key location, and increasing buffer space in other key locations to reduce the threat of a system shutdown if one station cannot handle demand in the allotted time. The factory is currently able to implement smaller lean initiatives to increase worker productivity and remove inefficiencies, but it is unable to realize the potential of all the data being gathered at this hyper-modern factory. This has led to smaller, superficial problems being improved upon, but it has not been able to use the data to find deeper patterns in the performance of the factory. The current operations can keep up with the weekly demand of 150 but future demand from the factory will require 180 weekly per week and a different mix of wheelset types. This increase in production

volume has been mandated by the business side of the NS. This increase requires a deeper understanding of the system.

The complexity of the situation

The flow of wheelsets through the system itself is non-linear and dynamic. This means that wheelsets can skip certain steps if parts are rejected and each wheelset type has a distinct path through the system according to its characteristics. . However, some of these wheelset types have the same characteristics and therefore have the same path through the system. These skips and different paths result in bottlenecks and lower production numbers, This complexity makes it difficult to plan longer configurations of wheelsets. A combination of wheelsets of input in a specific order is called a strain. The system currently only analyzes weekly production numbers for benchmarking, not individual sets. This makes analyzing individual sets more difficult and increases the difficulty in analyzing the system.

Norm and reality

The norm is the desired state and the reality of the current situation. The difference between the norm and reality is the action problem described by Heerkens & van Winden (2017) is the discrepancy between the norm and reality as perceived by the problem owner. The NS currently has a norm of sixty wheelsets per week whilst the reality is fifty. The action problem, an increase of ten wheelsets per week, will be the center of the problem cluster which will follow below.

Problem cluster

Heerkens & van Winden (2017) state the four steps to be taken to identify the core problem. These are:

1. Make a problem cluster with all relevant problems to the action problem,
2. Follow the chain back to the root causes of the action problem,
3. Only use problems you can influence when selecting a potential core problem
4. If more than one core problem remains in the cluster, select the one which has the best cost-benefit ratio according to your thinking.

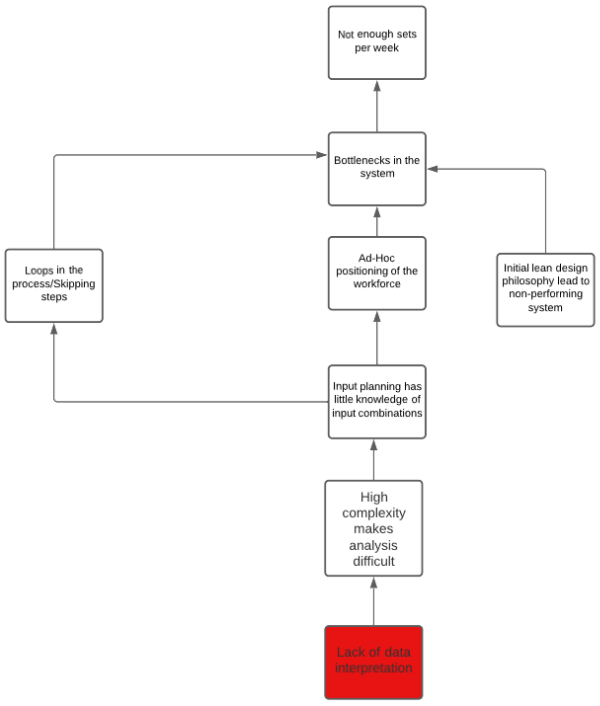


FIGURE 1, PROBLEM CLUSTER

Figure 1 shows the remaining core problems. These are the "Initial lean design philosophy leading to a non-optimally performing system" and "Lack of data interpretation". The factory has already been adjusted to cope with the initial over-lean design but there are areas where there might still be room for improvement. The problem with this is that rebuilding the factory is extremely expensive and can be very time-consuming getting the different systems to adjust to the new change. The option with the better cost-benefit outlook, shown in red, is the "Lack of data interpretation" The complexity of the dynamic, non-linear system makes the entire system difficult for people to analyze. The current human operational understanding is therefore also at its limits. This makes it an ideal candidate for machine learning. Machine learning has the potential to potentially find deeper patterns in the data that are difficult or impossible for people to find with standard statistical analysis.

Machine learning

Machine learning has the potential to find patterns in complex data that humans cannot find. The process in which this can have the most impact whilst having the least costs is the input process for how wheelsets combinations are loaded into the system. This raises questions as to how machine learning can be used in this context. Machine learning requires a few things to work. This is the input, the order in which wheelsets are loaded into the system. This is called a strain. The amount of wheelsets in a strain also matters. Larger predictions mean larger uncertainty. This means a balance between length and certainty needs to be found. The algorithms, each algorithm is in essence a different mathematical model that takes input, builds a model and, then gives output. The two candidates in this case are regression or classification. Classification and regression are the basic techniques of machine learning suited for this problem. And the output, is what the algorithms predict based on the historical input and method of processing. A KPI is needed as output. All of these things need to be defined and chosen in more detail to solve this lack of data interpretation. That is what the research design will do.

Research Design

Research questions

The main research question will be stated and then split up into smaller sub-questions that have to be answered before answering the main question.

Main research question:

"Can the historic data be used to predict the best input order for a given set of wheelsets through machine learning"

Sub-questions:

1. What length should the strains of wheelsets for prediction have?
2. What is the best way to measure the time in the system with a key performance indicator (KPI)?
3. What attributes and algorithms perform best for the algorithm(s) for regression and/or classification?
4. What algorithm(s) are best suited to predicting the performance of strains?
5. Does the sequence of wheelsets and their variables as input have an impact on the KPI?

Problem-solving approach

1. Analyze the current data structure and determine what is already available/measured. Also, make a KPI that gives a reliable measuring value about the throughput time, this is done through two interviews with the planner and discussions with the internal supervisor.

2. Decide on what algorithm should be used through a literature study: Classification OR Regression. Then decide the strain length. After that decide on which attributes can be used and which ones must be created.

3. Analyze the strains that have already appeared.

4. Run the algorithms for the results.

5. Process the results into conclusions that are of value to the factory.

Deliverables

- An analysis of which type of data mining algorithm(s) is best suited for the planning of wheelsets going into the system,
- The model with the best predictive score.
- Recommendations on how to improve the factory and to expand on using data mining for the performance of the factory.

2. Current situation

This chapter provides information about how the factory works and the configuration of the factory. The sub-question about the desired length of the strains is answered with information about the planning process and an interview with the planner for the process. The wheelset variety is explained and shown through a figure. Finally, the data quality and measuring rules are explained.

Description of the factory

The factory of hall 12 is concerned with the revision of wheelsets. The factory works with one daily shift that starts that start at 7:00 and ends at 15:30 with a break from 11:45 until 12:15. This results in eight hours worked per day by the industrial personnel. There are two 15-minute coffee breaks during the morning and evening shifts but these are not at regular times and cannot be accommodated in calculations. The factory works five days a week under normal operating conditions, with Friday often slightly undermanned due to contractual considerations from the NS. The factory was built five years ago to modernize the process. The old factory was able to produce 180 wheelsets per week. The new factory was built with the "lean" design philosophy in mind. That initial design was unfortunately too lean, this means that there were hardly any buffer places and that this caused bottlenecks in the system when wheelsets could not move further through the system. This did not fulfill the operational needs of the NS. The design of the factory has been improved through bottleneck analysis and lean methodologies in isolated parts of the factory to reconfigure the initial factory design to repair an acceptable amount of wheelsets. The average production in the year 2020 was only around 57* wheelsets per week. This was far from promised sixty wheelsets per week at capacity. This increased to an average of 120 per week in 2021[1]. 2022 has an average of 123[2] wheelsets per week. The current production goal is fifty per week and the desired improvement is to increase this to sixty.

[1] The average was calculated by taking the 2nd until 51st ISO weeks average.

[2] The average of the 2nd until the 42nd week were taken at the time of measurement.

The layout of the factory

The factory that revises wheelsets in Haarlem is placed in hall 12 of the industrial complex. The wheelsets are loaded in at the “Start” position, takt_180, into the process as can be seen in the Factory floorplan figure. The dismounting press is where anything that can be removed, is removed from the axle. This is also known as the “press-off”. The second part is the area where axles can be transported with an overhead crane toward the stations which are required for checking and repairing the axles. Axels can be removed from the line if faults are detected. The third part is where the paint shop sprays on the necessary conservation layers. The fourth area is the preparation for the “press on” and the actual “press on”. This process takes components from the part wall and prepares them to be pressed onto the repaired or new axel. After this is done the axle is tested to see if the fit is to the specified tolerances. The press-on can have failures. When this happens the wheelset is removed from the line and the next day the press-on is tried again. When the wheelset is ready it is moved to the fifth part of

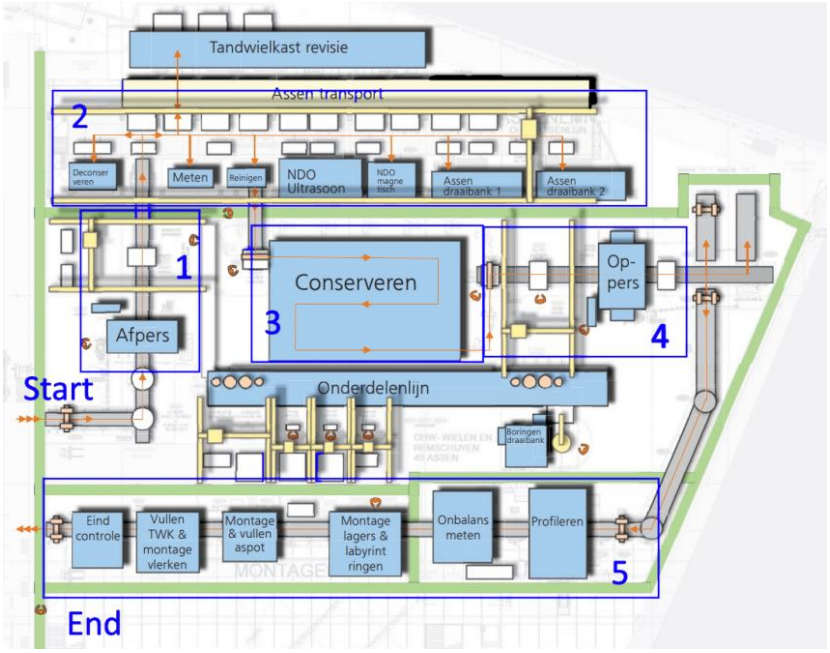


FIGURE 2, FACTORY FLOORPLAN (DERKINK, 2021)

the line where the last steps take place to see if the last measurements are within the acceptable range and the last parts are added to each wheelset. The second part of the factory is the workshop floor where engine, gearbox, and rubber revision takes place. These “takts” are described as having in essence an infinite capacity and are not the bottlenecks experienced by the rest of the factory by the MES product owner. The workforce of the factory is unevenly distributed during the day. Half of the takts are mostly done in the morning and the other half is then inversely done in the afternoon. This changes from day to day and operational requirements.

THE PATH THROUGH SECTOR 2

The second sector of the factory shows the complexity and non-linearity which defines the revision process. Figure 3 shows the two potential loops through the system and the problems that can arise from these loops. The first change from the norm, “A, Axle rejection” shows that a wheelset can skip five steps in the process if at 221 the axle is found to be defective. This leads to more peak demand at takt_260. The other small loop shown here is that of a gearbox. Sometimes only the gearbox needs to be

examined and revised. This means that the wheelset will skip the entirety of the other points in the line and create more peak demand at takt_260. An axle can be pulled out of sector two at any measuring point and then moved to 260 after the new axle is brought in. these two loops are not the only loops in the system but they showcase the complexity and possibility of creating bottlenecks.

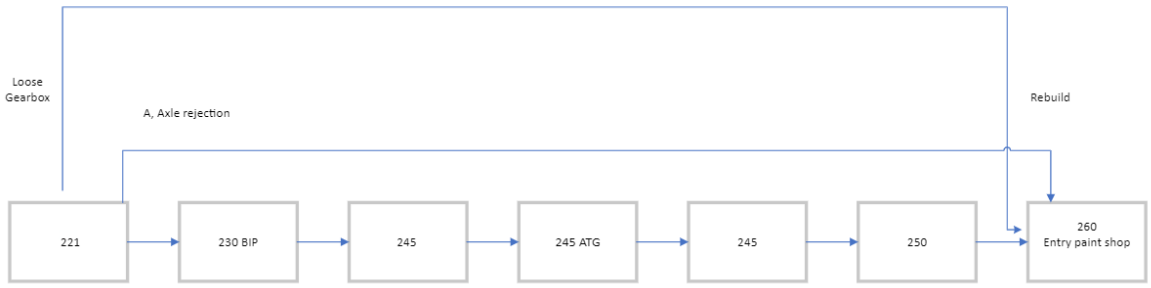


FIGURE 3, EXAMPLE FLOW

Wheelset description

The factory currently services 19 different kinds of wheelsets. These are from the different train models the NS operates. Each type is interchangeable for each train that uses a particular model. The factory mainly services trains which are the property of the NS. The wheelsets have three main styles as can be seen in the Example wheelsets types figure, (engine/Cardan, braking plates, braking discs, simple axle). The wheelsets are checked before going into the line with a visual inspection. If the axle looks good it does not need to get replaced and pulled out of the line. The same goes for the parts which are on the axle. This is all processed into a "material plan". Wheelsets that do not have to get pulled out of the line have a shorter path in the factory and most likely need to spend less time in the system. However, if it is found out during testing that E.G. the axle has damage that was not spotted or foreseen it is removed from the line and a new axle is machined to specifications. The material plan is therefore an expectation, not a given.

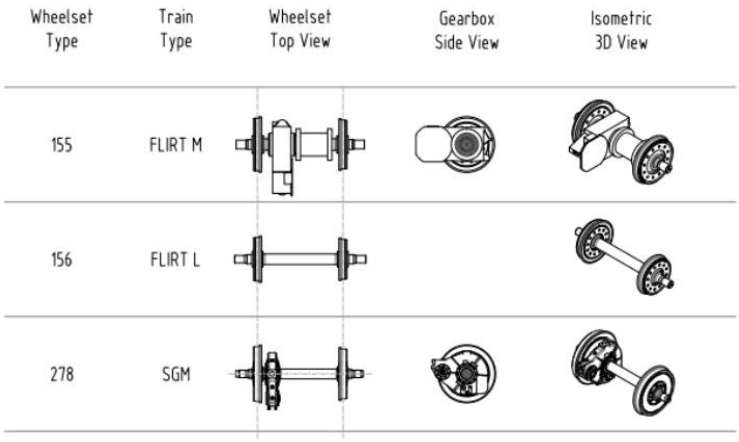


FIGURE 4, EXAMPLE WHEELSET TYPES (DERKINK,2021)

Description of the MES

The mechanical enterprise system (MES) functions as the middle part of the IT framework. This can be seen in Figure 5, IT architecture. The MES has the path of the wheelset at any time in memory. Each workstation has an operator who scans when a part arrives. The machine records when it's done working and lastly when the parts move on in the line. Parts cannot be moved to a new station without the authorization of the last station. This ensures that there is little to no missing data in the MES. The MES sends the data necessary for the reports of each wheelset to SAP which is where the source data is stored. The MES also sends information to Maximo, Maximo is the asset management system of the NS and needs to know what assets are where at all times.

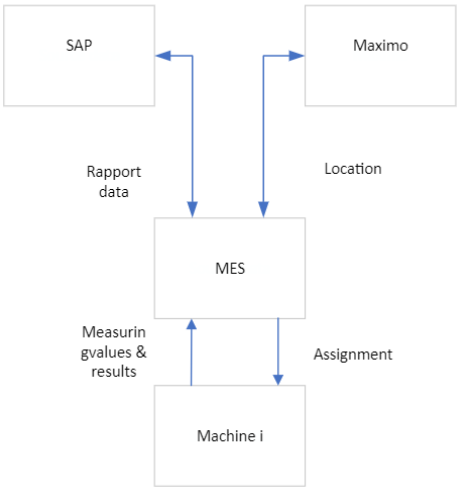


FIGURE 5, IT ARCHITECTURE

Quality & quantity of data

The data from 2020, 2021, and 2022 are available for Extract transform and load (ETL) and data mining. This data has been extracted from the MES and is converted into a usable excel file through a Python script provided by the NS. The current average production is around 123 wheelsets per week. This differs from the data from 2020 and 2021 as many improvements have been made to the factory. The choice between data volume and the quality of said data is dependent on 2021. The average production per week in 2020 was only around 57 wheelsets per week for the year, with many outliers which increase the average. This means the data from 2020 is not generalizable with much of the data from 2021 and 2022. The data collection will therefore begin when forty wheelsets have been produced in one week. Forty wheelsets balance the need for data volume and the need for generalized data. The first week in which 120 wheelsets have been produced is ISO week seven, in 2021. 123 wheelsets were achieved this week. The coding for this can be found in Appendix three.

The inactive time outside of working hours will be removed to give each unique wheelset a value according to the KPI measurements. This inactive time is composed of the 16 hours per day not worked and the 48 hours if the wheelsets stay one or more weekends. The available attributes can be found in the Figure Classes table and the selection will be explained in chapter 3.

The MES data that is extracted is what is known as "essential data" for the system. It should therefore never be blank unless intended. Many actions of the system are not possible if the required information has not been filled out or checked by an operator.

Length of strain

The hall which pre-processes wheelsets will be redesigned in the upcoming year. The most important change is that strains of wheelsets will have to be planned for that hall as well, this differs from the current situation in which the wheelsets can be chosen more ad-hoc for the factory as input in hall 12. The new situation has stricter boundaries for the planner, as the input into the preceding hall will also be the input into hall 12. The original process allowed the planner to choose what he wanted at takt_180 to enter into the process. The new process will have an extra length of nine wheelsets before takt_180 which the planner must consider. This places a larger emphasis on planning in the upcoming configuration.

These are the informal rules and important information that the planner uses in planning:

- Prior experience is used for some combinations of wheelsets, not doing 5 motor revisions in a row for example. This is done to avoid exceeding capacity on the motor revision workstations or other workstations,
- The planning is made every day, a few times a day with two to three wheelsets at a time,
- The planner uses his own experience when deciding on what to plan, there is no KPI for throughput currently in the factory.
- The required amount currently sits at 30 per day, 150 a week.

The planning priorities are as follows. There is a normal level of stock, the “norm”, which is what the factory wants to have on hand at all times. Then there is the minimal level, the “min”, which is what is minimally needed. The norm and the min can have a shortage, a shortage on the min is of course worse. Wheelsets that have a shortage on the min receive priority for planning and revision. This means that the set of wheelsets that have priority can be influenced but not the wheelsets already in the system.

The minimal length of strain is two. Single strains would test the attributes of each wheelset instead of the potential effect the configurations of strains have. The effect of the strain length can be tested by using the different lengths of strains and seeing what lengths perform best. The preferred length is derived from the current operational situation. The current required ten wheelsets per day can be split up into two groups of five wheelsets. This would reduce the amount of noise in the data, from a data mining standpoint, compared to ten wheelsets whilst still delivering value to the planning aspect of the factory. Five wheelsets are also easier to plan than other combinations adding up to ten wheelsets. The five wheelsets will be compared to two and three-string combinations to test the difference in performance.

Figure 6, Strain examples figure below shows a configuration of five wheelsets. Each letter is a representation of a wheelset type. The letters have been chosen at random to illustrate how the strains work. The combination of two A’s, one C, one D, and one E doesn’t change but their position as input

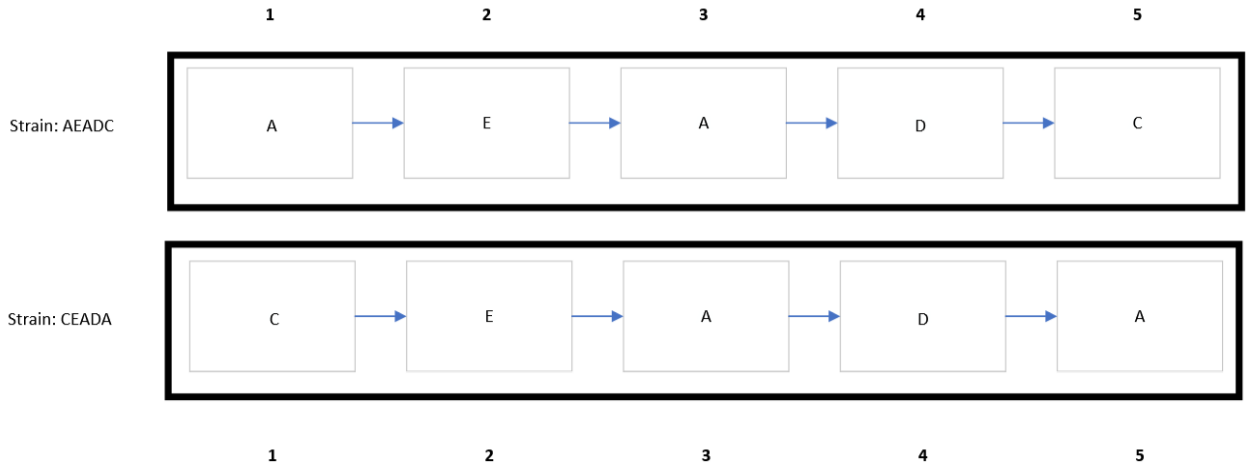


FIGURE 6, STRAIN EXAMPLES

into the system changes. Each unique combination of wheelsets is a strain that is possible to plan. The "1" is the first position in the strain and the first wheelset to go into the system, the "2" the second, etc.

Noise in the data

The quality of the data is influenced by noise caused by overtime, due to another project which required more wheelsets temporarily, in 2021 from weeks 24 until 27. These weeks are not removed from the data as the limited amount of data points makes every point important. Another factor for noise is that the factory slows down all production if the current supply outpaces the demand. This has happened 2 to 5 times in the last year but it has not been logged and therefore cannot be compensated or removed from the data. The holidays add noise to the data as they present days of lost production time. This makes the KPI slightly less reliable. Due to labor regulations, Fridays have less staff available for work, which translates into lost production time. It is not known how much production is lost each Friday and therefore not possible to compensate for this lost time. Some wheelsets only have a starting takt, and an ending takt(395). These wheelsets do not have a 336 takt, which means they cannot be measured according to the KPI. This is because MES takt 180 is the technical takt when wheelsets are entered into the system. The majority continue in the line but these do not as they only need an Axel bearing replacement. They don't enter the line until after the "press on", meaning they do not influence the part of the line researched and are deleted from the dataset. They also do not influence the wheelsets before and after. This makes it possible to make strains with wheelsets before and after this type of wheelset. The 15-minute break twice per day also influences the data but they can influence each wheelset the same, reducing the noise caused. The only non-generalizable noise is when some wheelsets have fifteen minutes more calculated but with working times of several days, this doesn't matter much.

Weekly average "Current situation"

The weekly average production for the "current situation" is calculated by taking the averages of each year except the first two and last two(so remove 1, 2, 52, 53). This is because these weeks can be overinflated when using ISO week numbers. 2022 has been measured from the third week until three weeks before the data stops, which is week 22. Weeks 3 until 19's average is measured.

3. Theoretical perspective

This chapter provides the theoretical background and argumentation of what algorithms are best suited to build a model that can predict the time in the system based on the configuration order and numeric attributes of the strains. The choice for a regression or classification model is explained and regression is chosen. The KPI for regression is made and the attributes which affect the KPI and different configurations of the wheelsets are explained. The algorithms suitable for testing are explained and split into two categories, “Standard” and “Complex”. The 19 different wheelsets are discretized into seven classes to reduce the complexity for the machine learning models.

KPI creation

The main problem identified in the first chapter of this thesis is that the data, and any potential underlying patterns, haven’t been analyzed by the factory. There is currently no measurement used by the NS as a KPI for individual wheelsets or combinations of wheelsets. This means that any KPI that is created should strive to help with gaining a deeper understanding of the system. The KPI will measure from “Takt_180” to “Takt_336”. Takt 180 is where the wheelsets are loaded into the system. Takt_336 is the ending of the press on the machine. The two ending Takt’s are chosen as they are the biggest bottlenecks in the system, measuring the start and then bottleneck should give an impression of how long individual or strains of wheelsets are in the system. The process is linear after 336, and the pace has proven to be around 15 wheelsets per day. More than enough for the currently desired 12 per day. The process also has around 12 buffer spots, which means that the time in systems variability does not come from this time in the system. The average time is measured for each of the wheelsets in the strain and the average of that is the KPI. Using the longest time of any of the wheelsets was considered but this would make the KPI very susceptible to outliers, whilst averages are less so. This KPI helps with measuring what attributes have an impact on the most important route in the system, the start to bottleneck route. An alternative KPI with “Takt_260” will also be used to do a comparison to see if the difference in chosen end station makes a difference in algorithm performance. 260 is chosen as it marks the beginning of the paint shop, after this, the wheelsets go into a linear process until the “Press-on” 336. The

Algorithms selection

Prediction vs Classification

With the KPI now created the question arises as to what algorithm(s) is best suited for the data of the performance of wheelsets The two techniques which are suitable for this are regression (numeric prediction) and classification. Regression makes a numeric prediction based on whatever algorithm is used on the dataset. The internal results of numeric prediction are easy to compare and don’t suffer from human bias about the reference point as classification does. Classification predicts which of the pre-determined classes an instance belongs to and gives this as the result. Both of these techniques use training data to predict new data. They use the attributes of the instances and their value to base their numeric or class prediction. The factory did not have a KPI before this research created one or a category measuring system in which strains or even single wheelsets are evaluated. This lack of categories makes prediction an easier-to-interpret choice. Creating categories is possible but that would introduce unnecessary bias about what range each class should have. These classes would have a set interval of time as defined by the KPI and could be labeled with e.g. “unsatisfactory”, “Currently satisfactory” and “Satisfactory for future demand” for example. Combined with this that numeric prediction makes more

sense in this scenario as it is more useful for the deliverable “what strains are the best in any given situation”. This makes prediction the clear favorite.

Clustering, as an approach, and neural networks as a technique are not considered for this research. Neural networks are known as difficult to interpret and clustering is not the best way of gaining an understanding of deeper patterns in the data. Clustering is also not suited to comparing strains of wheelsets. “Simple” Bayesian networks are based on the assumption of independence between attributes, which is an assumption that is not certain with this dataset. They also assume a normal distribution in the data which there is no basis for

The WEKA software has been chosen as it offers a wide variety of classical machine learning algorithms and also an open-source library. The advantage of WEKA is also that it does not require any coding to perform machine learning. As an IEM student, the focus of this thesis is on the application of data mining, not the mathematical details.

Separation of algorithms

The algorithms that will be used have been separated into two categories. The “standard” algorithms and the “complex” algorithms. This distinction is artificial and has been made because both algorithm types serve slightly different purposes in this research. The “Standard” algorithms provide more potential insights into the potential deeper patterns of the attributes and strains, which is done by using standard, more interpretable algorithms. The “complex” algorithms use less interpretable but more performance-focused algorithms afterward. This approach has the advantage of potentially providing interpretable models for human understanding with the “Standard” and performance-based models with the “complex” for the deliverable of “the model with the best predictive score”.

“Standard” algorithms

Decision tree (regression tree)

Decision trees are algorithms that split the data into different branches which end in end nodes which are known as leaves. The splitting is done based on attributes and the goal of each split is to divide the instances into groups that have the least intra-subset variation. This is done on numeric values. The specific tree used will be a regression tree with will predict a continuous numeric value. The advantage of decision trees is that they are easy to interpret with visual presentation and they do “not require any domain knowledge or parameter setting and therefore is appropriate for exploratory knowledge discover.” (Jiawei Han et al. 2012. p331). Furthermore, they are also known for their accuracy. (Jiawei Han et al. 2012. p357). The leaves are “pruned” after the construction of the tree based on certain parameters to see if pruning increases the precision of the tree.

Rules

Rules work slightly differently from decision trees. A rule-covering approach tries to find rules that reduce the input space by covering as many instances and then removing them once the rule is complete. This then happens again until the entire input space is covered. Rules are not the best algorithm for predicting. This is even more true with a numeric prediction as the boundaries of the rules have to result in hard numeric predictions instead of a categorical prediction, which makes it less precise. The rule approach does however produce relatively interpretable structures, like decision trees. This makes rule-covering approaches suitable for finding underlying patterns in the data but not suitable for the deliverable about the list of best configurations of strains.

Regression

Regression estimates how much each variable is responsible for the result and produces a linear formula that has the weights estimated earlier and the variables. It then fills in the attributes of new instants and gets a numeric prediction. This is not suited for the number of variables in the complex situation of the factory but it can provide insight into what variables have the largest weights, and therefore the most important in estimating the time of strains. Providing that the result has a reasonable amount of relative absolute error.

Model trees

Model trees differ from decision (regression) trees in that they do not use the average value of all the instances that reach a certain node but they have a local linear model at the end of each leaf node that predicts the value based on locally weighted regression.

"Complex" algorithms

SVM's

A support vector machine is an algorithm that splits the data according to the best-separated line with the widest margin between the boundary instances. It does this by finding the best line fit. The support vectors can be made in two-dimensional space but a mapping function via a kernel can be used if the data has more than two dimensions or does not fit in two dimensions to map into higher dimensions which might fit the data better. The advantages of SVMs are that they can be used for regression (prediction) or classification and "The complexity of the learned classifier is characterized by the number of support vectors rather than the dimensionality of the data. Hence, SCMs tend to be less prone to overfitting than some other methods" (Jiawei et al. 2012. p450). Coupling this with their generally high accuracy makes them a very good candidate for this problem. The disadvantages are that the parameters have to be finetuned with additional steps and that they can take a long time to train and that the solution can be difficult to interpret. SMOreg is the name WEKA uses for linear regression support vector machines.

"Complex" learners

Bagging (Bootstrap aggregating)

Bagging uses random sampling with replacement (bootstrapping) to train k models which are trained with the aforementioned random sampling. These models then take give their prediction and the average value will be the result. "The bagged classifier often has significantly greater accuracy than a single classifier derived from D, the original data" (Jiawei et al. 2012. p379)

Random forest

Random forest uses a random selection of attributes, with replacement, to form new data sets (bootstrapping). The algorithm then chooses a random subset of attributes to train the algorithms on. The advantage of random forest is that bootstrapping makes the new models less dependent on the original data set. The random feature selection then helps with reducing the correlation between the trees. The new models all take an equal vote and this result is the outcome of the algorithm. With numeric prediction, this is an average of the trees.

Additive regression

Additive regression works as a "stagewise additive model for numeric prediction" (H. Witten et al. 2011. 363). It does this by making a model and recording the errors. It then makes a new model that focuses

on improving these errors. This continues until a certain cutoff point is reached. The model predictions are added to each other to reduce the error. 10-fold cross-validation is useful here to reduce the overfitting of the model, which can happen as it builds on the same training data.

Hyper-parameter selection

Each machine learning algorithm has certain parameters which affect the performance. These can improve the performance over the standard settings of each algorithm. The algorithms with the best results will have their parameters tuned to see how their performance potentially increases with this change. The parameter selection will be done with 10-fold cross-validation.

AUTO-WEKA

AUTO-WEKA is an automatic algorithm that is used to automatically select the best possible algorithm and hyperparameters. It does this through "Bayesian optimization techniques that iteratively build models of the algorithms/hyperparameter landscape and leverage these models to identify new points in the space that deserve attention"(Thornton et al. 2013). The algorithm is essentially seeing all available algorithms as a parameter in the search space and the corresponding hyperparameters as other attributes in the search space. This algorithm is very computationally expensive for good results and is only used to have a reference point to see if the final results from the other models are close to what is possible according to the algorithm.

Attribute selection

The general principle that more data leads to better working algorithms does not necessarily apply to attributes in the same way. "Experiments with a decision tree learner (C4.5) have shown that adding to standard datasets a random binary attribute generated by tossing an unbiased coin impacts classification performance." (H. Witten et al. 2011. p307). This means that it is a good idea to choose a small number of attributes that have the biggest impact on the system. The available attributes can be seen in appendix COLUMNS REF. The choice of components is a major step for machine learning. It is essential "Because of the negative effects of irrelevant attributes on most machine learning schemes, it is common to precede learning with an attribute selection stage" (H. Witten et al. 2011. p308).

The attribute selection is done in two stages. The first stage is determining the three different sets of attributes. These are,

1. The nominal wheelsets positions (5 total),
2. The binary wheelset positions (35 total),
3. The aggregate number of defining attributes, engines, Cardan, braking plates, and braking discs.

Different combinations of these three attribute sets will be tested to see which combination works best and if there are any particular insights into what works best across different algorithms. T

Creation of the classes, and attributes

The attributes which have an effect on the time in the system between "Takt_180" and "Takt 336" are:

- Attached gearbox.
- Material plan.
- Attached Cardan.
- Braking plates.

- Braking disc(two or three).

The bearings and carriers are repaired in other parts of the line and have no impact on the KPI, and are therefore removed from consideration. The wheelsets are divided into classes that have identical attributes. Each class has the same path through the system and has the same repairs. This reduces the number of unnecessary attributes whilst keeping the differences that set them apart. Each class can be in one of the five positions of the strain. This is done with a binary attribute which is 0 if the class doesn't take for example the first spot of a strain and a 1 if it does. This creates 35 binary attributes as there are five spots which can be seven distinct classes. This will also be done with seven nominal attributes to see if this tests better than the five positions that are also defined with nominal attributes in five different variables, this is done to test the time series in different ways. The A class is split into two classes as the material plan can state that the attached gearbox needs repair. This predetermines that the wheelset will have to leave the line and this creates a significantly different path through the factory that warrants a split in class. Table 1, Classes table shows the distribution of classes. Each class has a large enough amount to not be discarded. Each strain made with these classes will be measured in the KPI. The coding for the classes can be found in appendix three.

Classes	**Distinguishable parts**	**Wheelsets**	**Amount**
- Class A*	- Attached gearbox, - 4 braking plates.	- 278, - 293, - 328, - 331.	- 701. - A1= 532. - A2= 169.
- Class B	- Attached Cardan.	- 315, - 324, - 325.	- 305.
- Class C	- Attached Cardan, - 4 braking plates.	- 155.	- 101.
- Class D	- 4 braking plates.	- 156, - 329, - 332.	- 301.
- Class E	- 3 Braking discs.	- 297, - 316, - 317, - 326, - 327.	- 1274.
- Class F	- 2 braking discs.	- 280, - 296, - 379.	- 310.

TABLE 1, CLASSES TABLE

4. Experiments & Results

This chapter tests the hypothesis “A model can be trained that predicts the time in the system based on the order of input and characteristics of each strain”. The methodology of calculating the KPI is then explained. The hypothesis is tested by looking at four different configurations of strains. These are, the initial five-string input combinations with the two and three-string combinations and the 260 KPI string. The attribute selection and hyperparameter selection and their impact on performance are then discussed. Two boxplots of the 260 and 336 KPI are shown to measure the internal and external variation of the 10 strains with the most support. This is to analyze the internal and external variation of the 10 most supported compositions. The effectiveness of machine learning in this context is explained and an answer to the hypothesis is given at the end.

Hypothesis

“The configuration and distinctive attributes can be used to train a machine learning model that can predict the KPI performance with at most 50% relative absolute error”

Experiment methodology

KPI calculation

The KPI is calculated by taking the time between the beginning of the takt_180 and the ending of takt 260 or 336. The excess time(non-operating hours) is then removed. This is done with these formulas in Excel.

```
=RC[-4]-RC[-5]-(2*RC[6])-(2/3)*(RC[12]-RC[6]*2)
```

Light blue: Ending takt.

Red: Starting takt.

Purple: Amount of weekends between the two dates.

Green: Amount of days between the two dates.

This calculation first takes the entire time in the system (blue minus red), then it removes the weekends from that(2 times the amount of weekends), then it removes the other inactive time(which is 2/3 day as 8 hours are worked per day). The weekend days are removed from the inactive time subtraction. This results in the end KPI. The KPI of each strain is calculated by taking the average of the strain. The data in excel is also formatted chronologically from the perspective of takt_180, this formats all the other data automatically into a time series. The source data from the NS was extracted as a CSV file and then transformed with a provided Python script into an excel file which was then used for the creation of classes, KPIs, and strains. The essential data for the factory specifically is:

- Starting takt(beginning time, chronologically ordered).
- Ending takt(ending time).
- Material plan(For the A1 or A2 distinction).
- Wheelset type(Determines class).

The classes are assigned according to the inherent characteristics of each wheelset and A1 or A2 is differentiated according to the treatment plan. The exact coding can be found in appendix three.

Experiments

Each of the experiments is run with the open-source WEKA package, 3.8.6 2022 version. Each of the experiments has been done with 10-fold cross-validation and each AUTO-WEKA algorithm has been run for an equal 120 minutes each. Each strain sequence has been made with the same amount of base wheelset data points. The algorithms used in the experiments are, linear regression[3] does not have an attribute selection filter or wrapper, this is the only deviation from the standard settings.

- Rules (M5).
- Model tree M5P.
- Linear regression.
- Regression tree (REPtree).
- Bagging(M5P tree).
- Random Forest.
- Additive regression(with linear regression).
- SMOreg.

Explanation of abbreviations

Comp = composition of strains.

Binary 35 = the binary 35 variables indicating the class place in each of the positions. This changes with the two-string and three-string into 14 binary variables and 21 binary variables.

Aggregate = these are the aggregates of the engine, cardan, braking plates, and braking discs.

Nominal = the places for the nominal class values.

Initial results (five-string)

Standard algorithms

The performance of the "standard" algorithms is over a 100% relative absolute error for all algorithms and configurations of attributes. This makes the corresponding models unusable for the prediction of unseen wheelsets.

Comp+ binary 35 + KPI,	rules (M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.1607	0.1606	0.1622	-0.0599
Mean absolute error	0.547	0.5472	0.5469	0.545
root mean squared error	0.9841	0.9841	0.9818	0.9458
relative absolute error	100.37%	100.40%	100.34%	100.00%
root relative squared error	104.05%	104.05%	103.81%	100.00%
Comp + binary 35 +aggregate + KPI	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.1596	0.16	0.143	-0.0637
Mean absolute error	0.5457	0.5467	0.4849	0.4813

[3] Linear regression standard algorithm without attribute pre-selection, this is also the case for additive regression.

root mean squared error	0.9846	0.9844	0.9437	0.9061
relative absolute error	100.30%	100.31%	100.76%	100.00%
root relative squared error	104.11%	104.08%	104.16%	100.00%
Comp + nominal place + aggregate	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.1623	0.1426	0.143	-0.0637
Mean absolute error	0.5463	0.4862	0.4849	0.4813
root mean squared error	0.983	0.9469	0.9437	0.9061
relative absolute error	100.23%	101.02%	100.76%	100.00%
root relative squared error	103.94%	104.51%	104.16%	100.00%

TABLE 2, 336 FIVE-STRING "STANDARD" ALGORITHMS

"Complex algorithms"

The best performance is with AUTO-WEKA, with additive regression and decision stump as a base algorithm. The relative absolute error is 87.11%. This result is better than the standard algorithms but also unusable for the prediction of unseen wheelsets.

Comp+ binary 35 + KPI,	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.1694	0.151	0.1622	0.2817
Mean absolute error	0.5301	0.4705	0.5469	0.4516
root mean squared error	0.9567	0.9196	0.9818	0.8999
relative absolute error	97.27%	97.76%	100.34%	93.82 %
root relative squared error	101.15%	101.50%	103.8113%	99.31 %
Comp + binary 35 +aggregate + KPI	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.143	0.2228	0.1622	0.2823
Mean absolute error	0.4859	0.4559	0.5469	0.4517
root mean squared error	0.946	0.897	0.9818	0.8998
relative absolute error	100.95%	94.71%	100.34%	93.85 %
root relative squared error	104.4089	99.00%	103.8113%	99.30 %
Comp + nominal place + aggregate	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.3251	0.2051	0.143	0.2823
Mean absolute error	0.4375	0.4569	0.4849	0.4517
root mean squared error	0.8653	0.9019	0.9437	0.8998

relative absolute error	90.89%	94.93%	100.76%	93.85 %
root relative squared error	96.50%	99.54%	104.16%	99.30 %

TABLE 3, 336 FIVE-STRING "COMPLEX" ALGORITHMS

AUTO-WEKA	Additive regression(decision stump)
Correlation	0.3964
Mean absolute error	0.4746
root mean squared error	0.868
relative absolute error	87.11%
root relative squared error	91.81%

TABLE 4, 336 FIVE-STRING AUTO-WEKA

Comparison with the two and three-string

The results of the five-string strains will be compared to a two-string and a three-string alternative. This is to see if the five-string is too long to fit a model on and if the methodology of testing strings makes sense in this context. If the two and three-string have a usable performance then the methodology of testing strings holds weight. If not then the length of the string is most likely not the determining factor for the KPI with the current attributes and volume of data points.

Standard algorithms

The three-string and two-string compare better performance-wise to the five-string. The algorithm with the best performance is the two-string Model tree M5P algorithm, with a performance of 84.75 % relative absolute error. This is still unusable for the prediction of strains of wheelsets.

THREE-STRING

Comp+ binary 35 + KPI,	rules (M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.2765	0.2768	0.1622	-0.0599
Mean absolute error	0.5424	0.5423	0.5469	0.545
root mean squared error	1.1064	1.1062	0.9818	0.9458
relative absolute error	89.13%	89.11%	100.34%	100.00%
root relative squared error	96.76%	96.74%	103.81%	100.00%
Comp + binary 35 +aggregate + KPI	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.2779	0.2778	0.2788	0.279
Mean absolute error	0.5419	0.5421	0.5408	0.5407
root mean squared error	1.1059	1.1059	1.1059	1.1058
relative absolute error	89.05%	89.08%	88.86%	88.86%
root relative squared error	96.72%	96.72%	96.71%	96.71%

Comp + nominal place + numeric, ag	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.2723	0.2727	0.2788	0.279
Mean absolute error	0.5433	0.543	0.5408	0.5407
root mean squared error	1.1086	1.1083	1.1059	1.1058
relative absolute error	89.23%	89.23%	88.86%	88.86%
root relative squared error	96.96%	96.93%	96.71%	96.71%

TABLE 5, 336 THREE-STRING STANDARD ALGORITHMS

TWO-STRING

Comp+ binary 35 + KPI,	rules (M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.3179	0.3179	0.3037	0.2844
Mean absolute error	0.59	0.59	0.589	0.614
root mean squared error	1.3463	1.3463	1.355	1.3519
relative absolute error	84.91%	84.91%	84.76%	88.36%
root relative squared error	99.67%	99.67%	100.31%	100.08%
Comp + binary 35 +aggregate + KPI	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.3186	0.3186	0.3037	0.2844
Mean absolute error	0.5905	0.5905	0.5889	0.614
root mean squared error	1.3459	1.3459	1.355	1.3519
relative absolute error	84.98%	84.98%	84.76%	88.36%
root relative squared error	99.63%	99.63%	100.31%	100.08%
Comp + nominal place + aggregate	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.3198	0.3198	0.3037	0.2844
Mean absolute error	0.5889	0.5889	0.5889	0.614
root mean squared error	1.3446	1.3446	1.355	1.3519
relative absolute error	84.75%	84.75%	84.76%	88.36%
root relative squared error	99.54%	99.54%	100.31%	100.08%

TABLE 6, 336 TWO-STRING STANDARD ALGORITHMS

"Complex algorithms"

The best-performing algorithm was the two-string WEKA-AUTO algorithm. The predictive performance was measured with a relative absolute error percentage of 82.65%. This is higher than the 84.30% standard configuration of the random forest which makes sense as AUTO-WEKA optimizes hyperparameters automatically. This result is still unusable however for useful predictions for the planning of input strains.

THREE-STRING

Comp+ binary 35 + KPI,	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.2735	0.151	0.2788	0.3514

Mean absolute error	0.54	0.4705	0.5408	0.5139
root mean squared error	1.1057	0.9196	1.1059	1.0863
relative absolute error	88.73%	97.76%	88.86%	84.45 %
root relative squared error	96.70%	101.50%	96.71%	95.00 %
Comp + binary 35 +aggregate + KPI	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOre g
Correlation	0.2748	0.3352	0.2788	0.3518
Mean absolute error	0.5399	0.5347	0.5408	0.5141
root mean squared error	1.1052	1.0794	1.1059	1.0862
relative absolute error	88.72%	87.86%	88.86%	84.49 %
root relative squared error	96.6565	94.40%	96.71%	94.99 %
Comp + nominal place + aggregate	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOre g
Correlation	0.2753	0.3043	0.2788	0.3518
Mean absolute error	0.5397	0.5353	0.5408	0.5141
root mean squared error	1.1048	1.0931	1.1059	1.0862
relative absolute error	88.69%	87.97%	88.86%	84.49 %
root relative squared error	96.62%	95.59%	96.71%	94.99 %

TABLE 7, 336 THREE-STRING COMPLEX ALGORITHMS

AUTO-WEKA	Random Forest
Correlation	0.5487
Mean absolute error	0.5201
root mean squared error	0.9576
relative absolute error	85.47%
root relative squared error	83.76%

TABLE 8, 336 THREE-STRING AUTO-WEKA

TWO-STRING

Comp+ binary 35 + KPI,	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOre g
Correlation	0.2953	0.3076	0.3037	0.3677
Mean absolute error	0.5878	0.5857	0.589	0.5387
root mean squared error	1.3337	1.3273	1.355	1.2787
relative absolute error	84.60%	84.30%	84.76%	77.52 %
root relative squared error	98.74%	98.26%	100.31%	94.66 %

Comp + binary 35 +aggregate + KPI	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.3045	0.3079	0.3037	0.3704
Mean absolute error	0.5869	0.5864	0.5889	0.5376
root mean squared error	1.3247	1.3282	1.355	1.277
relative absolute error	84.46%	84.39%	84.76%	77.37 %
root relative squared error	98.06%	98.33%	100.31%	94.54 %
Comp + nominal place + aggregate	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.3045	0.3082	0.3037	0.3711
Mean absolute error	0.5863	0.5855	0.5889	0.537
root mean squared error	1.3224	1.3277	1.355	1.2764
relative absolute error	84.38%	84.26%	84.76%	77.279
root relative squared error	97.90%	98.29%	100.31%	94.49 %

TABLE 9, 336 TWO-STRING COMPLEX ALGORITHMS

AUTO-WEKA	Random forest
Correlation	0.5104
Mean absolute error	0.5741
root mean squared error	1.1623
relative absolute error	82.65%
root relative squared error	86.00%

TABLE 10, 336 TWO-STRING AUTO-WEKA

Comparison with the 260 KPI

The 260 Takt with a five-string strain has been chosen as an end station to see if the change in what stations are measured changes the usability of the models. This is done to see if the way the system is measured has a major impact on performance and if the right endpoint for measurement has been chosen.

Standard algorithms

The performance of the "Standard" algorithms with over 100% relative absolute error does not significantly deviate from the 336 five-string strain. These models are therefore also unusable in the prediction of the KPI.

Comp+ binary 35 + KPI,	rules (M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.1407	0.1435	0.143	-0.0637
Mean absolute error	0.4872	0.4861	0.4849	0.4813
root mean squared error	0.9474	0.9462	0.9437	0.9061

relative absolute error	101.23%	101.00%	100.76%	100.00%
root relative squared error	104.56%	104.43%	104.16%	100.00%
Comp + binary 35 +aggregate + KPI	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.1415	0.143	0.143	-0.0637
Mean absolute error	0.4863	0.4859	0.4849	0.4813
root mean squared error	0.9469	0.946	0.9437	0.9061
relative absolute error	101.02%	100.95%	100.76%	100.00%
root relative squared error	104.51%	104.41%	104.16%	100.00%
Comp + nominal place + aggregate	Rules(M5)	Model tree M5P	Linear regression	Regression tree
Correlation	0.142	0.1426	0.143	-0.0637
Mean absolute error	0.4867	0.4862	0.4849	0.4813
root mean squared error	0.9476	0.9469	0.9437	0.9061
relative absolute error	101.13%	101.02%	100.76%	100.00%
root relative squared error	104.58%	104.51%	104.16%	100.00%

TABLE 11, 260 FIVE-STRING STANDARD ALGORITHMS.

Complex algorithms

The best-performing algorithm for the complex algorithms is the AUTO-WEKA, with a relative absolute error of 90.00%. The closest second is the SMOreg, with 93.82%. These results and accompanying models are both unusable for predicting the KPI times of unseen strains.

comp+ binary 35 + KPI,	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.151	0.2035	0.143	0.2817
Mean absolute error	0.4705	0.4584	0.4849	0.4516
root mean squared error	0.9196	0.9028	0.9437	0.8999
relative absolute error	97.76%	95.23%	100.76%	93.82 %
root relative squared error	101.50%	99.63%	104.16%	99.31 %
Comp + binary 35 +aggregate + KPI	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.1505	0.2228	0.143	0.2823
Mean absolute error	0.4107	0.4559	0.4849	0.4517
root mean squared error	0.9204	0.897	0.9437	0.8998
relative absolute error	97.79%	94.71%	100.76%	93.85 %
root relative squared error	101.58%	99.00%	104.16%	99.30 %

Comp + nominal place + aggregate	Bagging(M5P tree)	Random Forest	Additive regression(linear)	SMOreg
Correlation	0.1505	0.2051	0.1426	0.2823
Mean absolute error	0.4711	0.4569	0.4862	0.4517
root mean squared error	0.9202	0.9019	0.9469	0.8998
relative absolute error	97.87%	94.93%	101.02%	93.85 %
root relative squared error	101.56%	99.54%	104.51%	99.30 %

TABLE 12, 260 FIVE-STRING COMPLEX ALGORITHMS

AUTO-WEKA	
Correlation	0.0368
Mean absolute error	0.4329
root mean squared error	0.8494
relative absolute error	90.00%
root relative squared error	93.79%

TABLE 13, 260 FIVE-STRING AUTO-WEKA

AUTO-WEKA & hyperparameter tuning

AUTO-WEKA's best configuration with 120 minutes of processing time is with the 336 two-string with 82.65% as seen in the AUTO-WEKA comparison figure below. The performance of AUTO-WEKA was higher than that of the standard configuration with a score of 84.2706%. The result is however not usable in the current form as the algorithms were unable to fit a pattern or model to the historic data. Hyperparameter tuning has not resulted in performance enhancement great enough to make the current models usable in operational conditions. It does however show that the tuning of the hyperparameters does lead to a performance increase in the correlation coefficient of 0.2012 and 12.24 percentage points of root relative squared error in this particular case. AUTO-WEKA also shows that there are no distinctive patterns currently in the data that an algorithm can explore as it checks all available combinations instead of sticking to the ones I have chosen.

	AUTO-WEKA	*Standard configuration*
Correlation coefficient	0.5104	0.3092
Mean absolute error	0.5741	0.5856
Root mean square error	1.1623	1.327
Relative absolute error	82.65%	84.2706
Root relative squared error	86.00%	98.2408

TABLE 14, AUTO-WEKA COMPARISON

Boxplots

The 10 wheelsets with the highest support (Number of instances) have been put into a boxplot for the 336 five-string KPI and 260 five-string KPI. The Y axis is the KPI multiplied by time 24. The green lines, the median, are mostly within the 40 to 50 range for the 336 KPI. This means that the means of the different strains are quite similar. The strains show large internal deviation and quite similar behavior to other strains. These boxplots indicate that the order in which wheelsets are loaded into the system does not

have a major measurable effect on the time in the system. They also show major outliers which indicates that other variables have a major impact on the performance which are not taken into account.

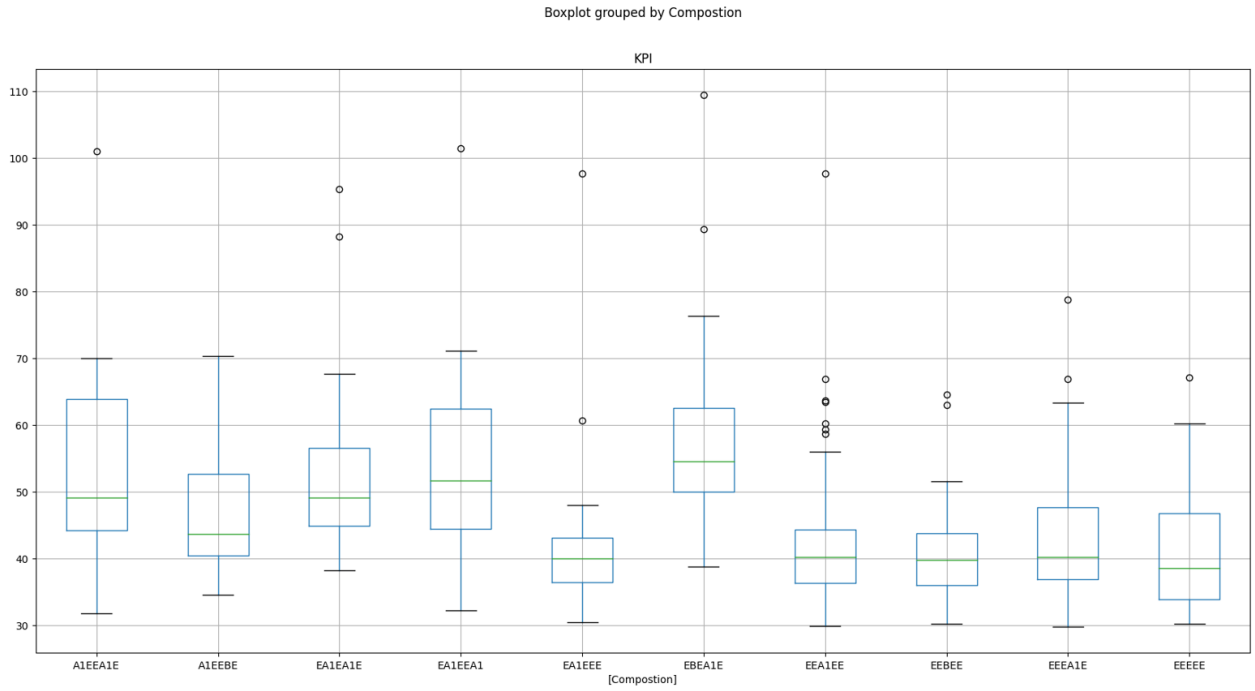


FIGURE 7, 336 KPI BOXPLOT

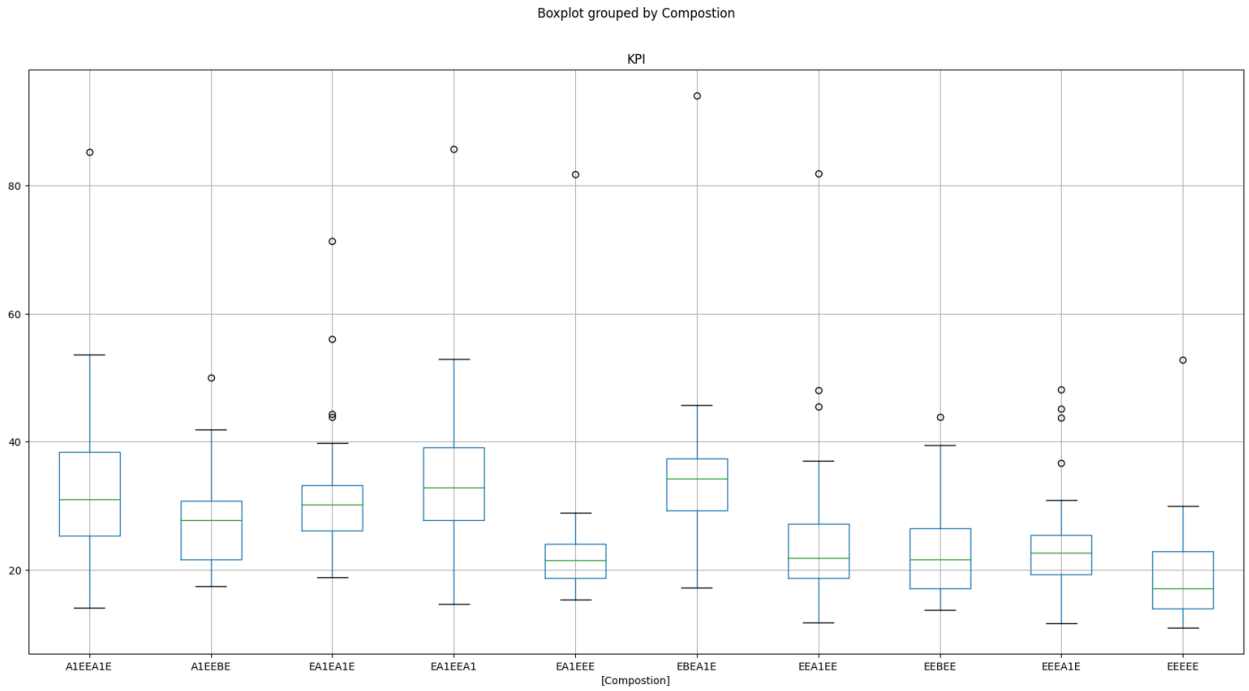


FIGURE 8, 260 KPI BOXPLOT

General conclusion

Every tested strain length and measuring point has resulted in models that don't fit the data for predictive performance in an acceptable manner. The lowest relative absolute error was found with SMOreg, at 77.28% with the two-string data. This model is fully described in appendix 1, the best-performing model. This error rate makes accurate predictions impossible and the absolute error rates of over 100% for the "Standard" algorithms make all their predictions unusable. Even AUTO-WEKA, which performs selection and hyperparameter tuning on its own was not significantly better than the other algorithms.

This means that the initial hypothesis "The configuration and distinctive attributes can be used to train a machine learning model that can predict the KPI performance with at most 50% relative absolute error" is rejected. The currently available data indicates that the input order at takt_180 cannot be used to train a model that is sufficient for the use of prediction. This is also somewhat supported by the boxplots, in Figures 7 and 8. The compositions with the same wheelsets but different input orders show almost the same median and quite similar internal variation. This indicates that the input order at takt_180 is currently not the deciding factor. The best-performing algorithms configurations are summarized in Figure nine and Figure 10 below.

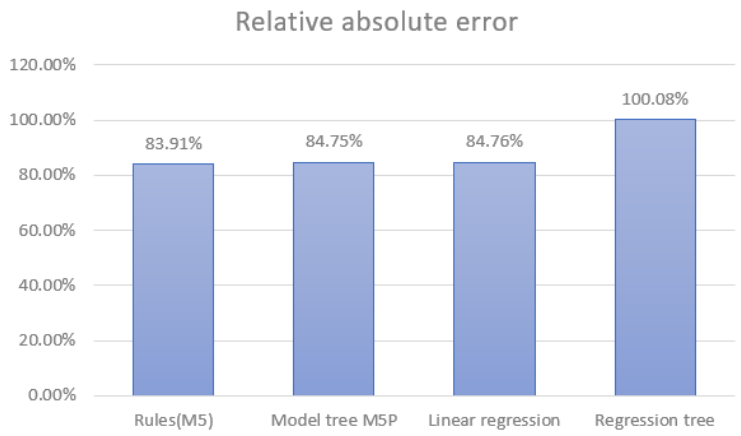


FIGURE 9, BEST PERFORMANCE STANDARD ALGORITHMS

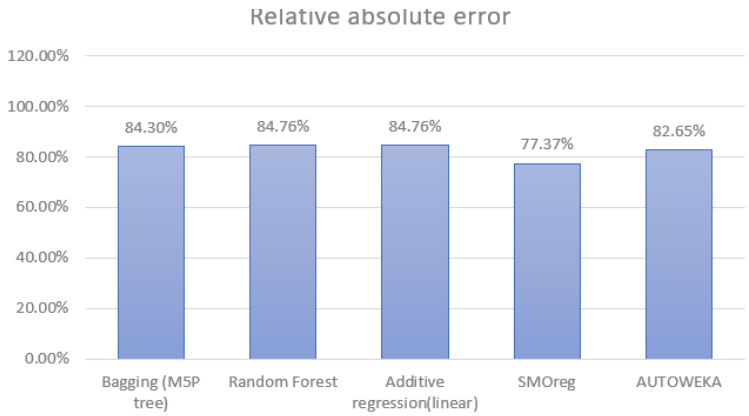


FIGURE 10, BEST PERFORMANCE COMPLEX ALGORITHMS

5. Conclusion

This chapter uses all the learned information from the previous chapters to provide key insights into how the process can be researched further for future optimizations. It does this through two distinct lenses. The first is the local optimizations and the second is the general machine learning variables. These recommendations are based on the conclusion that the input order for the system at takt_180 is not defining for the processing speed for strains of wheelsets. The implementation of this thesis is the setup for future research into the problem.

Recommendations

Local optimizations

The results from chapter four show that, with the current data available and limitations, it is not possible to train a model that can accurately predict the time in the system from the configuration of the input order for the strains. This means that improvement will have to be made on smaller parts of the system or single machines.

Factory improvements

Chapter four has shown that the input order into takt_180 does not have a major influence when tested up to 5 wheelsets in combinations. This means that the desired extra production capacity will have to be found or created in a different part of the system. A good approach to this would be to use the theory of constraints and do a local bottleneck analysis. This analysis can be done in two parts. The first part is the second sector, as seen in the figure “Factory floorplan” of the factory with the “line” with a corresponding analysis with the same data mining methodology as this research. This would determine if the input order has a significant impact on the time in the system at this potential bottleneck. Takt_221 would be the starting point and the beginning of takt_260 the ending time. These two are chosen as the beginning and endpoint of sector 2. The second part would be an analysis of each machine in the factory. The waiting time before each system would be measured and the machines with the highest waiting times would be selected for local optimizations. The distribution of waiting time can then be used to see what improvements can be made. The distribution type gives information about what the waiting time looks like and if there are options for improvement. A Pareto distribution would indicate that only some wheelsets have a large waiting time whilst a normal distribution would indicate problems with waiting times for all wheelsets. It might mean that there are machines in the factory that would need to work in less than 30 minutes per takt, if they are used more than others due to wheelsets skipping parts of the factory. The number of personnel at each bottleneck could be increased or new machines can be bought to perform the process in parallel.

Data mining improvements at the factory

Regression vs classification

The research performed in this thesis has used regression for building models on the data. This has been done because no prior classes had been established and numeric predictions have the advantage of easy comparison. However, some algorithms need classes to be able to work. An example of this is Bayesian networks, which require binary classes to build their models. Establishing classes alongside the current KPI allows for the use of most of the current algorithms currently available. This is not limited to the entire wheelset itself but also the individual parts of the factory.

Data mining for predictive purposes data mining

The algorithms and their uses as described in chapter three can be used to perform more research into numeric prediction for whatever process the NS thinks that historic data might provide insight into future behavior. There are a few processes which are in my opinion very suited to this.

Improving single machinery

The research performed has focused on a process that involves a great deal of non-linearity. This has been done because of the complexity of the process and the lack of deeper understanding. Machine learning is however also very suited to cases where the variables are easy to isolate. Every single machine and workstation is in essence a single isolated case where statistical analysis or machine learning could be used. The most interesting bottlenecks in the system can be isolated in terms of input variables and performance output.

Improving the material plan

The axle can be rejected at any time during the process in the factory. This means the initially assumed treatment plan is no longer valid and the path through the system radically changes. The current pre-processing stage before takt_180 only includes a visual inspection for the initial rejection of axles. This could be improved by taking measurements and seeing how they influence the rejection or non-rejection of the axles during their time in the system. A machine learning algorithm can then be trained to see if the classification “Rejection” or “No rejection” could be done within an acceptable error rate. The advantage of this is that this would be a single process with the corresponding variables, making analysis easier and less reliant on unknown connections that analyzing an entire system would be susceptible to. This would however potentially require an investment into machinery to measure these to-be-determined attributes. Examples of variables that could be used are,

- Time in service.
- Earlier revisions.
- Technical measurements that have a relationship with possible rejection.

Local variables

This research has used the available attributes and created classes of input. There are however several other attributes that require changes to the way the data is logged in the factory. These attributes include,

- The material plan, the current material plan gives some indication of what path through the system is required but some parts are currently not easily readable. The pre-rejection of axles is currently not logged in a manner conducive to data manipulation and should become part of the MES data making experimentation easier for all users. Expanding the way this attribute is logged would improve the validity of each class as a separate entity.
- What employee is working on each station? Each employee has a productivity level, logging this would give a better picture of how productive each employee is and how efficient each workstation can be.
- The planner’s ruleset, the planner currently does not have defined planning rules. This makes measuring the human impact on the planning procedures difficult and makes testing out different planning schemes very difficult. Using defined rulesets and logging this makes it possible to measure its impact more.

Limitations to this research

Impact of staff and planner

The planner's input and reaction to problems in the system have not been logged. This makes it more difficult to assess how his influence as a planner affected the system. The ad-hoc reactions of the workforce have also not been logged, making it impossible to measure their precise impact on KPI times for wheelsets. The impact of each workers productivity on the KPI has also not been measured, but this would be very time consuming and could get lost in the complexity of such a task.

Impact of the worst known strains has not been observed.

The planner has knowledge about the entire process and what strains definitely cause issues. These are therefore not planned. The problem with this is that the worst strains are not part of the dataset. This makes it more difficult to train a model if there is no clear pattern or distinction between good performing strains and bad performing strains.

Data limitations

Only 3000 data points of wheelsets where available for the training of the model. This is not a particularly large dataset and it could be that more data would increase the performance of the models. With the performance being at 80% relative absolute error however does mean that this is not very likely.

Noise and class purity

The holidays provided noise in the data increasing the KPIs for several wheelsets. The system is also very susceptible to outliers as removal from the line or repair of a gearbox can lead to a longer KPI or skipping large parts of the line resulting in a small KPI. The week number was also changed to 52 if the week number was 53. This was to reduce the large impact the 53rd week had on the data as a year only has 52,17 weeks. The pre-rejection of the axle was also not taken into account when making classes. Resulting in classes that could be separated better in the future according to their internal characteristics.

Appendix 1, Bibliography

Bibliography

Chris Thornton, F. h.-B. (2013). *Auto-WEKA: Combined Selection and Hyperparamter Optimization of Classification Algorithms.*

Derkink, T. (2021, 12 16). An insight int he planning of the maintenance process of wheel axles at Nederlandse Spoorwegen. University of Twente.

Ian H. Witten, E. F. (2011). *Data mining, Practical machine learning tools and techinques.* Elsevier.

Jiawei Han, M. K. (2012). *Data Mining Concepts and Techniques.* Elsevier.

Appendix 2, The best performing model

=== Run information ===

Scheme: weka.classifiers.functions.SMOreg -C 1.0 -N 0 -I "weka.classifiers.functions.supportVector.RegSMOImproved -T 0.001 -V -P 1.0E-12 -L 0.001 -W 1" -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007"

Relation: 335 2 string repair-weka.filters.unsupervised.attribute.NumericToBinary-R4-17-weka.filters.unsupervised.attribute.Remove-R4-17

Instances: 2948

Attributes: 8

Composition

1

2

Gearbox

Cardan

Breaking plates

Braking dicsc

KPI

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

SMOreg

weights (not support vectors):

+ 0.0015 * (normalized) Composition=A2F

- 0.0062 * (normalized) Composition=FA2

- 0.018 * (normalized) Composition=FB

- 0.004 * (normalized) Composition=BF

- 0.0056 * (normalized) Composition=BE

+ 0.0048 * (normalized) Composition=EE

- 0.0019 * (normalized) Composition=EA1

- 0.0005 * (normalized) Composition=A1F

- 0.0031 * (normalized) Composition=FA1

- 0.0018 * (normalized) Composition=A1E

+ 0.0059 * (normalized) Composition=EF

+ 0.0064 * (normalized) Composition=FF

+ 0.0076 * (normalized) Composition=A2E

+ 0.0005 * (normalized) Composition=EC

- 0.0202 * (normalized) Composition=CE

- 0.0161 * (normalized) Composition=EB

+ 0 * (normalized) Composition=BA2

- 0.0038 * (normalized) Composition=EA2

+ 0.0049 * (normalized) Composition=A1B

+ 0.0075 * (normalized) Composition=FE

- 0.0189 * (normalized) Composition=BA1

+ 0.0075 * (normalized) Composition=FD

- 0.0081 * (normalized) Composition=DA2

- 0.0051 * (normalized) Composition=BD

- 0.0003 * (normalized) Composition=DA1

- 0.0214 * (normalized) Composition=CF

+ 0.0037 * (normalized) Composition=DD

+ 0.0021 * (normalized) Composition=DE

- 0.0142 * (normalized) Composition=A1A1

+ 0.0037 * (normalized) Composition=A2A2

+ 0.0169 * (normalized) Composition=A1A2

+ 0.0028 * (normalized) Composition=A1D

+ 0.0072 * (normalized) Composition=A2D

- 0.0161 * (normalized) Composition=DB

+ 0.0044 * (normalized) Composition=ED

- 0.0029 * (normalized) Composition=A2A1

+ 0 * (normalized) Composition=BB

+ 0.0036 * (normalized) Composition=DF

- 0.0033 * (normalized) Composition=FC

- 0.036 * (normalized) Composition=CD

+ 0.0008 * (normalized) Composition=DC

- 0.0247 * (normalized) Composition=CC

+ 0 * (normalized) Composition=A2C

- 0.0256 * (normalized) Composition=A2B

+ 0 * (normalized) Composition=A1C

+ 0.0819 * (normalized) Composition=CB

+ 0.0328 * (normalized) Composition=BC

+ 0.0512 * (normalized) Composition=CA1

- 0.0086 * (normalized) 1=A2

- 0.0094 * (normalized) 1=F

- 0.0008 * (normalized) 1=B

- 0.0061 * (normalized) 1=E

+ 0.0082 * (normalized) 1=A1

+ 0.0309 * (normalized) 1=C

- 0.0142 * (normalized) 1=D

- 0.0084 * (normalized) 2=F

+ 0.0025 * (normalized) 2=A2

+ 0.011 * (normalized) 2=B

- 0.0056 * (normalized) 2=E

+ 0.0099 * (normalized) 2=A1

+ 0.0061 * (normalized) 2=C

- 0.0155 * (normalized) 2=D

+ 0.006 * (normalized) Gearbox

+ 0.0236 * (normalized) Cardan

+ 0.0097 * (normalized) Breaking plates

- 0.0118 * (normalized) Braking dicsc

+ 0.0317

Number of kernel evaluations: 299842105 (78.387% cached)

Time taken to build model: 115.71 seconds

=== Cross-validation ===

=== Summary ===

Correlation coefficient	0.3711
Mean absolute error	0.537
Root mean squared error	1.2764
Relative absolute error	77.279 %
Root relative squared error	94.4938 %
Total Number of Instances	2948

Appendix 3, Excel code & VBA code

The code described below has been adjusted in places to deal with the two-string or three-string strains. It cannot be transcribed verbatim to run with the data for the five-string strains. The logic however remains the same and only minimal adjustments are required to perform those transformations.

Remove unnecessary parts of the material plan,

The only characters influencing the A1 or A2 split are the last 4

```
Sub Right()

Dim i As Long

Dim j As Long

Dim Plan As String

Dim Death_Array(1 To 2)


Death_Array(1) = "Data 21"

Death_Array(2) = "Data 22"


Dim var As String

Dim output As String


For j = 1 To 2

   For i = 1 To 2500

      var = Worksheets(Death_Array(j)).Cells(i + 1, 5)
```

```
    Worksheets(Death_Array(j)).Cells(i + 1, 5) = Mid(var, 11, 5)
  Next i
Next j

End Sub
```

Assigning classes.

```
Sub Count_Classes()
Dim i As Long
Dim j As Long

Dim A As Long
Dim A1 As Long
Dim A2 As Long
Dim B As Long
Dim B1 As Long
Dim B2 As Long
Dim C As Long
Dim C1 As Long
Dim C2 As Long
Dim D As Long
Dim E As Long
Dim F As Long
Dim Count As Long
Dim Death_Array(1 To 2)

Death_Array(1) = "Data 21"
Death_Array(2) = "Data 22"
```

```
A = 0
A1 = 0
A2 = 0
B = 0
B1 = 0
B2 = 0
C = 0
C1 = 0
C2 = 0
D = 0
E = 0
F = 0

For j = 1 To 2
   For i = 1 To 2500
      If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 278 Then
         A = A + 1
         Call DeathToSmurfs1(i, Death_Array(j), A1)
      End If

      If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 293 Then
         A = A + 1
         Call DeathToSmurfs1(i, Death_Array(j), A1)
      End If

      If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 328 Then
         A = A + 1
         Call DeathToSmurfs1(i, Death_Array(j), A1)
      End If
```

```
 If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 331 Then
   A = A + 1
   Call DeathToSmurfs1(i, Death_Array(j), A1)
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 315 Then
   B = B + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "B"

   Call DeathToSmurfs2(i, Death_Array(j), B1)
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 324 Then
   B = B + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "B"
   Call DeathToSmurfs2(i, Death_Array(j), B1)

End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 325 Then
   B = B + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "B"
   Call DeathToSmurfs2(i, Death_Array(j), B1)

End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 155 Then
   C = C + 1
```

```
    Call DeathToSmurfs3(i, Death_Array(j), C1)
    Worksheets(Death_Array(j)).Cells(i + 1, 12) = "C"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 156 Then
    D = D + 1
    Worksheets(Death_Array(j)).Cells(i + 1, 12) = "D"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 329 Then
    D = D + 1
    Worksheets(Death_Array(j)).Cells(i + 1, 12) = "D"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 332 Then
    D = D + 1
    Worksheets(Death_Array(j)).Cells(i + 1, 12) = "D"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 297 Then
    E = E + 1
    Worksheets(Death_Array(j)).Cells(i + 1, 12) = "E"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 316 Then
    E = E + 1
    Worksheets(Death_Array(j)).Cells(i + 1, 12) = "E"
End If
```

```
If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 317 Then
   E = E + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "E"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 326 Then
   E = E + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "E"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 327 Then
   E = E + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "E"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 280 Then
   F = F + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "F"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 296 Then
   F = F + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "F"
End If

If Worksheets(Death_Array(j)).Cells(i + 1, 4) = 379 Then
   F = F + 1
   Worksheets(Death_Array(j)).Cells(i + 1, 12) = "F"
End If
```

```
    Next i
Next j

Worksheets("Results").Cells(2, 2) = A
Worksheets("Results").Cells(3, 2) = B
Worksheets("Results").Cells(4, 2) = C
Worksheets("Results").Cells(5, 2) = D
Worksheets("Results").Cells(6, 2) = E
Worksheets("Results").Cells(7, 2) = F


Worksheets("Results").Cells(2, 4) = A1
Worksheets("Results").Cells(2, 5) = Worksheets("Results").Cells(2, 2) - A1
Worksheets("Results").Cells(2, 6) = B1
Worksheets("Results").Cells(2, 7) = Worksheets("Results").Cells(3, 2) - B1
Worksheets("Results").Cells(2, 8) = C1
Worksheets("Results").Cells(2, 9) = Worksheets("Results").Cells(4, 2) - C1

Dim RowMagic As Long
i = 0
j = 0
For j = 1 To 2
    RowMagic = Worksheets(Death_Array(j)).UsedRange.Rows.Count
    For i = 1 To RowMagic - 1
        If Worksheets(Death_Array(j)).Cells(i + 1, 12) = "" Then
        Worksheets(Death_Array(j)).Cells(i + 1, 12) = "A2"
        End If
    Next i
```

```
Next j



End Sub
```

Extra code, checks if A1 needs to be assigned or assigns A2 if empty with different piece of code above

```
Function DeathToSmurfs1(i, j, A1)

Dim Death_Array(1 To 2)



Death_Array(1) = "Data 21"

Death_Array(2) = "Data 22"


   If Worksheets(j).Cells(i + 1, 5) = " M202" Then

      A1 = A1 + 1

      Worksheets(j).Cells(i + 1, 12) = "A1"


   End If


   If Worksheets(j).Cells(i + 1, 5) = " M204" Then

      A1 = A1 + 1

      Worksheets(j).Cells(i + 1, 12) = "A1"

   End If



' check the material plan for the i,j position, if revision then, if revision on attacheted gearbox or etc then count = count + 1

End Function
```

```
Function DeathToSmurfs2(i, j, B1)

   If Worksheets(j).Cells(i + 1, 5) = " M202" Then
      B1 = B1 + 1
   End If

   If Worksheets(j).Cells(i + 1, 5) = " M204" Then
      B1 = B1 + 1
   End If

' check the material plan for the i,j position, if revision then, if revision on attacheted gearbox or etc then count = count + 1

End Function

Function DeathToSmurfs3(i, j, C1)

   If Worksheets(j).Cells(i + 1, 5) = " M202" Then
      C1 = C1 + 1
   End If

   If Worksheets(j).Cells(i + 1, 5) = " M204" Then
      C1 = C1 + 1
   End If

' check the material plan for the i,j position, if revision then, if revision on attacheted gearbox or etc then count = count + 1

End Function
```

Assign numeric attributes,

```
Sub Numeric()

Dim i As Long
```

```
Dim j As Long



For j = 2 To 2985
   For i = 1 To 4
      Worksheets("Voorlopig").Cells(j, 44 + i) = ""

   Next i
Next j

For j = 1 To 2985
   For i = 1 To 5

     If Worksheets("Voorlopig").Cells(j, 2 + i) = "A1" Then
     Worksheets("Voorlopig").Cells(j, 45) = Worksheets("Voorlopig").Cells(j, 45) + 1
     Worksheets("Voorlopig").Cells(j, 47) = Worksheets("Voorlopig").Cells(j, 47) + 4
     End If

     If Worksheets("Voorlopig").Cells(j, 2 + i) = "A2" Then
     Worksheets("Voorlopig").Cells(j, 45) = Worksheets("Voorlopig").Cells(j, 45) + 1
     Worksheets("Voorlopig").Cells(j, 47) = Worksheets("Voorlopig").Cells(j, 47) + 4
     End If

     If Worksheets("Voorlopig").Cells(j, 2 + i) = "B" Then
     Worksheets("Voorlopig").Cells(j, 46) = Worksheets("Voorlopig").Cells(j, 46) + 1
     End If

     If Worksheets("Voorlopig").Cells(j, 2 + i) = "C" Then
     Worksheets("Voorlopig").Cells(j, 46) = Worksheets("Voorlopig").Cells(j, 46) + 1
```

```
    Worksheets("Voorlopig").Cells(j, 47) = Worksheets("Voorlopig").Cells(j, 47) + 4
    End If

    If Worksheets("Voorlopig").Cells(j, 2 + i) = "D" Then
    Worksheets("Voorlopig").Cells(j, 47) = Worksheets("Voorlopig").Cells(j, 47) + 4
    End If

    If Worksheets("Voorlopig").Cells(j, 2 + i) = "E" Then
    Worksheets("Voorlopig").Cells(j, 48) = Worksheets("Voorlopig").Cells(j, 48) + 3
    End If

    If Worksheets("Voorlopig").Cells(j, 2 + i) = "F" Then
    Worksheets("Voorlopig").Cells(j, 48) = Worksheets("Voorlopig").Cells(j, 48) + 2
    End If

  Next i
Next j

For j = 2 To 2985
  For i = 1 To 4
    If Worksheets("Voorlopig").Cells(j, 44 + i) = "" Then
    Worksheets("Voorlopig").Cells(j, 44 + i) = 0
    End If

  Next i
Next i
End Sub
```

Binary and numeric attributes

```
Sub Strain_2()
```

```
Dim i As Long
Dim j As Long

For i = 2 To 3199
   Worksheets("String 5 long").Cells(i, 46) = (Worksheets("Data 5 long").Cells(i, 1) + Worksheets("Data 5 long").Cells(i + 1, 1)) * 0.5
Next i

For i = 2 To 3199
   Worksheets("String 5 long").Cells(i, 2) = Worksheets("Data 5 long").Cells(i, 2)
   Worksheets("String 5 long").Cells(i, 3) = Worksheets("Data 5 long").Cells(i + 1, 2)
Next i

End Sub

Sub Aggregates()

For j = 2 To 2991
   For i = 1 To 4
      Worksheets("String 5 long").Cells(j, 41 + i) = ""

   Next i
Next j
```

```
For j = 1 To 2991
   For i = 1 To 2

      If Worksheets("String 5 long").Cells(j, 1 + i) = "A1" Then
      Worksheets("String 5 long").Cells(j, 42) = Worksheets("String 5 long").Cells(j, 42) + 1
      Worksheets("String 5 long").Cells(j, 44) = Worksheets("String 5 long").Cells(j, 44) + 4
      End If

      If Worksheets("String 5 long").Cells(j, 1 + i) = "A2" Then
      Worksheets("String 5 long").Cells(j, 42) = Worksheets("String 5 long").Cells(j, 42) + 1
      Worksheets("String 5 long").Cells(j, 44) = Worksheets("String 5 long").Cells(j, 44) + 4
      End If

      If Worksheets("String 5 long").Cells(j, 1 + i) = "B" Then
      Worksheets("String 5 long").Cells(j, 43) = Worksheets("String 5 long").Cells(j, 43) + 1
      End If

      If Worksheets("String 5 long").Cells(j, 1 + i) = "C" Then
      Worksheets("String 5 long").Cells(j, 43) = Worksheets("String 5 long").Cells(j, 43) + 1
      Worksheets("String 5 long").Cells(j, 44) = Worksheets("String 5 long").Cells(j, 44) + 4
      End If

      If Worksheets("String 5 long").Cells(j, 1 + i) = "D" Then
      Worksheets("String 5 long").Cells(j, 44) = Worksheets("String 5 long").Cells(j, 44) + 4
      End If

      If Worksheets("String 5 long").Cells(j, 1 + i) = "E" Then
      Worksheets("String 5 long").Cells(j, 45) = Worksheets("String 5 long").Cells(j, 45) + 3
      End If
```

```
      If Worksheets("String 5 long").Cells(j, 1 + i) = "F" Then
      Worksheets("String 5 long").Cells(j, 45) = Worksheets("String 5 long").Cells(j, 45) + 2
      End If

   Next i
Next j

For j = 2 To 2991
   For i = 1 To 4
      If Worksheets("String 5 long").Cells(j, 41 + i) = "" Then
      Worksheets("String 5 long").Cells(j, 41 + i) = 0
      End If

   Next i
Next j

End Sub

Sub Binarized()
Dim i As Long
Dim j As Long



For i = 1 To 2990
   For j = 1 To 2
      If Worksheets("String 5 long").Cells(i + 1, j + 1) = "A1" Then
            Worksheets("String 5 long").Cells(i + 1, 7 * j - 3) = 1
```

```
End If

If Worksheets("String 5 long").Cells(i + 1, j + 1) = "A2" Then
   Worksheets("String 5 long").Cells(i + 1, 7 * j - 2) = 1

End If

If Worksheets("String 5 long").Cells(i + 1, j + 1) = "B" Then
      Worksheets("String 5 long").Cells(i + 1, j * 7 + -1) = 1

End If

If Worksheets("String 5 long").Cells(i + 1, j + 1) = "C" Then
      Worksheets("String 5 long").Cells(i + 1, j * 7 + 0) = 1
End If

If Worksheets("String 5 long").Cells(i + 1, j + 1) = "D" Then
      Worksheets("String 5 long").Cells(i + 1, j * 7 + -1) = 1
End If

If Worksheets("String 5 long").Cells(i + 1, j + 1) = "E" Then
      Worksheets("String 5 long").Cells(i + 1, j * 7 + -2) = 1
End If
If Worksheets("String 5 long").Cells(i + 1, j + 1) = "F" Then
      Worksheets("String 5 long").Cells(i + 1, j * 7 + 3) = 1
End If
```

```
    Next j
Next i


For i = 2 To 2991
    For j = 4 To 25
        If Worksheets("String 5 long").Cells(i, j) = "" Then
        Worksheets("String 5 long").Cells(i, j) = 0
        End If
    Next j

Next i



End Sub
```

Appendix #3

Coding for weekly totals, this code has been changed throughout the research but the 4th column is the ISO week number and the 5th is the week number. Each data entry was the finishing time of each wheelset at the bottleneck 336.

```
Sub count()
Dim i As Long
Dim j As Long

For i = 1 To 53
    Worksheets("Results 2").Cells(2, i) = 0
Next i

For i = 1 To 53
    Worksheets("Results 2").Cells(8, i) = 0
```

```
Next i


For j = 1 To 2200
   For i = 1 To 53
      If Worksheets("2020").Cells(j, 4) = i Then
         Worksheets("Results 2").Cells(2, i) = Worksheets("Results 2").Cells(2, i) + 1
      End If
   Next i
Next j


For j = 1 To 2200
   For i = 1 To 53
      If Worksheets("2020").Cells(j, 5) = i Then
         Worksheets("Results 2").Cells(8, i) = Worksheets("Results 2").Cells(2, i) + 1
      End If
   Next i
Next j




End Sub, and end of thesis.
```