Eindhoven University of Technology

MASTER

Data driven approach for generating insights into sprint planning

van Puijenbroek, Jan A.

Award date:
2024

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Technische Universiteit
Eindhoven
University of Technology

Operations management and logistics
Information systems

Data driven approach for generating insights into sprint planning

Master Thesis

Jan van Puijenbroek

Supervisors:

R. Eshuis
I. Grau
L. Genga

Eindhoven, February 2024

Abstract

This research addresses multiple ways to gather useful insights into agile sprint planning. The main goal is to understand what is causing sprint performance to be poor and thus to improve performance. Planners have to make a planning which consists of a certain amount of work items. The challenge for sprint planners is to make a sprint planning that does the most amount of work with the least amount of time. To understand this, the individual work items are assessed in multiple ways. These techniques are: visualisation, process mining and (causal) association rule mining. When these insights into the work items are generated, a way of improving the entire sprint planning is proposed. This method is based on the concept 'causal decision making'. This way, both individual work items and the entire sprint are addressed. These methods will help understand what causes poor sprint performance and what can be done in the future to get better performance.

Preface

This master thesis is the last project that I will complete as a student at the TU/e. My journey at the TU/e started as a pre-master student mechanical engineering. Unfortunately, I did not make the cut. After some consideration, I decided that I was not going to give up that easily and started a pre-master in a whole different field of study. In the end I am very happy with my decision and think it has turned out to be even better than I could have imagined.

During my time at TU/e I have met some great people of which I am very grateful for. These people were not only very helpful but also made time at TU/e more enjoyable. My family, friends and girlfriend have also been a tremendous support throughout my (pre-) master and therefore I am very thankful.

I want to thank Rik Eshuis for being my supervisor and helping me finishing my graduation project. I appreciate the freedom I have received together with the feedback which has been really helpful. I also want to thank Isel Grau and Laura Genga to be my second and third supervisors.

Finally, I would like to thank Thierry Bennis and Arco van der Velde to grant me the opportunity to graduate at Blis digital. They gave me a lot of freedom to shape the project but were also very helpful when needed. Like Thierry and Arco, rest of the employees at Blis digital have been incredibly helpful.

I see my time at the TU/e as a very positive time in my life and I am glad to have learned as much as I have during this period.

Jan van Puijenbroek, February 2024

Executive summary

Introduction

In this report, research has been done in order to gather better insights into the relationship between sprint data, time data and sprint performance. A sprint is a work iteration which is usually between 2 and 4 weeks long. These work iterations are often used in software development. The reason for this, is that it is rather hard to predict precisely how long an employee is going to take for certain tasks. This is a problem that is not only facing one company in particular but many companies will have the same issue.

The company at which research has been done is Blis Digital. This company develops "system critical" software for customers. In order to do this, managers commit resources to projects to add value for the customer. Blis works according to the agile methodology. There are multiple methods that are classified as agile. One main similarity between the methods is short work iterations. For these work iterations, a sensible estimation has to be made. This is rather hard since not all tasks and people are similar. This is where the main problem lies. In essence, the company wants a better insight into what causes a sprint planning to be wrong. Also, the company is interested to see whether it is possible to predict if a sprint planning is likely to be right or wrong upfront.

Throughout the years, a vast amount of data has been generated with information about past projects and sprints. Although this data is available for managers, the data is not used to its fullest extent. The data from past projects could potentially play a bigger role into future projects. This can be done if business insights of past data can be extracted. This can help the managers to make better informed decisions.

The essence of the problem is that software development is hard to manage. The changing traditional software development methodologies, traditional roles, distributed teams, dynamic customer requirements, high feedback loop, a large number of deployments, less documentation, multiple sprints, automation of DevOps and Agile setting, etc are making projects complicated to manage.

Research questions

From above, the following main research question has been formulated as:

How to improve sprint planning performance through data driven insights?

In order to answer this research question, the following sub research questions were defined:

- How is the current development process performing? Where is the most need for improvement?
- What causes poor sprint planning performance? What role does work item meaning play in this?
- How can the potential causal relations be used to improve future planning performance?
- How can the solution be generalised for other companies?

In order to answer the main research question, multiple steps have been taken. These steps include, data visualisation, process mining, Natural language processing, (Causal) association rule mining and causal decision making.

Research method

The research method selected for this project is CRISP-DM which is the industry standard for data mining. The research method describes the following stages:

- **Business understanding** At this stage, the business objectives are determined. The current situation is assessed and the goals are set in order to determine what a succesful project should accomplish.
- **Data understanding** This stage focuses on gathering and exploration of the available data.
- **Data preparation** When the data was gathered, it has to be cleaned, transformed and created.
- **Modeling** Multiple models have been proposed to answer the research questions such as process mining, (causal) rule mining and causal decision making.
- **Evaluation** The evaluation stage is meant to evaluate whether or not the goals have been met and if the research is generalizable.
- **Deployment** At last, the solution was applied inside a dashboard for planners to use.

Data

There were two main data sources that are available for this project. At first there is Azure devops, which includes all software related data. The second is Simplicate which houses all hour registration data. For convenience reasons, the hour registration data is also stored in a SQL database. For this project, the SQL server is used to gather data about hour registration.

When all data from these two sources were gathered, all the hours were matched with work items to see how many hours were spent on each work item. The data visualisation part gave more insight on how variables were distributed and how variables related to one and other.

Process mining

In order to gather more insight into the process performance, process mining has been applied. With the techniques belonging to process mining, the target was to get a better idea of which specific processes are performing well and which are not. The most occurring path is:

$$New \Rightarrow Active \Rightarrow Resolved \Rightarrow Closed$$

Other paths can be shorter but might be incomplete, these incomplete paths can have a shorter duration. It is important to see performance as a path that resolves the issue whilst not taking too long.

Some differences between User stories and Bugs is that bugs never visit states 'Done' and 'Committed'. Another difference is that User stories are more likely to go from 'closed' to 'active' which means that the work item was not finished in the first instance.

The best process flow of this process is also very similar to the most occurring process flow. This flow is determined by several aspects. In general, the good process flow goes through all stages without rework. With process conformance, it is possible to see which traces align with best process flow. This gives a measurement to assess how well a certain work item is performing.

The distribution of the events per day is interesting. The most work seems to be done on Monday and Tuesday. Wednesday is average, Thursday is a little less than average and Friday is the least productive day.

Natural language processing

In order to get better insights on what the meaning is of work items, natural language processing is applied. The idea is that other analysis methods can interpret the meaning of the title of a work item. When a certain meaning is discovered, hopefully a link can be made on which user stories or bugs are performing poorly.

The proposed method NLP method has shown good results for this particular use case. At first, the text preparation for the text embedding was very intuitive since no additional text cleaning or pre-processing had to be done except for language translation. When the embeddings were made, the reduction and clustering algorithm made very sensible clusters which indeed cluster user story titles based on topic.

(Causal) rule mining

For the association rule mining, the apriori algorithm is used. This is a data mining tool which returns frequently occurring patterns. These patterns are measured in support and confidence. Support meaning, the percentage of the pattern occurring in the entire data set. While confidence is the percentage of occurrences in the subset of where the antecedent occurs. Later, the Causal association rule mining algorithm is used to see if there are causal relations withing the data.

At first, it is clear that there are definitely a lot of relations to be found within the data; causal and non-causal. It is safe to say that there are almost no causal relations between work item meaning and performance. It would have been interesting to see which topics within the development process were performing better or worse. Fortunately there are some correlations found for these relations. The main take on part is that, even though there are barely any causal relations found, NLP does give better results. The reason for this is that it acts as a confounding variable. This makes sure that relations found within the data relate to the same meaning. When these confounding NLP variables are not taken into account, the algorithm makes claims about totally different types of user stories which makes the claims less relevant.

Causal decision making

Machine learning models are often used to predict causal effects. The causal effect is estimated and the best intervention is applied to the planning. The desired outcome of this machine learning algorithm is the performance of a sprint. It is desired that the input consists of user stories and bugs and the outcome the total sprint duration. This result can later be used by planners to make a better planning. For this machine learning problem, a sequential algorithm is proposed. The reason for the sequential machine learning is because, all the work items together add up to the total sprint duration. The structure of the machine learning algorithm is thus a many to one structure where multiple work items are linked to a single output. This output being the total sprint duration. This prediction algorithm can predict sprint duration with a precision of 2 days. When the algorithm has made a prediction on how long a sprint takes, it is going to calculate what the best course of action is. This course of action is essentially calculating which and how many work items should be added or removed.

Conclusion

The main goal of this research was to get more insight into sprint planning performance. This goal has been satisfied by using multiple tools including, data mining, (causal) association rules mining and deep learning. The data and process mining part was useful for determining past performance and causes for bad performance. The machine learning part assisted mostly in causal decision making for future sprint plannings. This means that planners now have an idea of what has caused work items to perform poorly in previous sprints and now know what intervention is necessary to improve future sprints.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this report, research has been done in order to gather better insights into the relationship between sprint data, time data and sprint performance. A sprint is a work iteration which is usually between 2 and 4 weeks long. These work iterations are often used in software development. It is rather hard to predict precisely how long an employee is going to take for certain tasks. This is a problem that is not only facing one company in particular but many companies, mostly in software development, will have the same issue.

The company in question is Blis digital. This is an upcoming software development company located in Barendrecht. Blis was founded in 2006 and has grown to around 60 employees since then. Blis started out as a marketing and software company. The marketing side has split off and has continued as a subsidiary called B.made. The core business is software development for businesses. Blis comes up with 'mission critical' software and data solutions for companies. Mission critical software means that the software is essential to the business. Since 2022, Blis has expanded business into Lithuania. The company called Hiper has become part of Blis and specialises in maintenance of software. The combination of the software development of Blis and software maintenance by Hiper is deemed to be a good competitive advantage. The reason for this is that, this way, the company can ensure to provide a more complete solution to the customer where other companies can not. Furthermore, wages in Lithuania are slightly lower. As mentioned above, Software development companies like Blis develop 'system critical' software for customers. In order to do this, managers commit resources to projects to add value for the customer. To plan these projects, many software development teams (including Blis' software development teams) work according to the agile methodology [Al-Saqqa et al., 2020]. There are multiple methods that are classified as agile. One main similarity between the methods is short work iterations. These iterations are often called sprints and usually are between 1 and 4 weeks long. Within these sprints, multiple tasks will be assigned to team members. The team leader expects the team member to finish these tasks within the length of the sprint. In order for this to happen, a sensible planning has to be made. This is rather hard since not all sprints, tasks and people are similar. This is where the main problem lies. In essence, the company wants a better insight into what causes a sprint planning to take longer than expected. Also, the company is interested to see whether it is possible to predict if a sprint planning is likely to be right or wrong upfront. Optimising sprint planning is an issue that has been studied before in various ways [Boschetti et al., 2014] [Golfarelli et al., 2012] [Jansi and Rajeswari, 2015]. None of these past studies have studied the data driven aspect of optimisation.

Throughout the years, a vast amount of data has been generated with information about past projects and sprints. Although this data is available for managers, the data is not used to its fullest extent. The data from past projects could potentially play a bigger role into future projects. This can be done if business insights of past data can be extracted. This can help the managers to make better informed decisions.

A considerable amount of data from sprints is text. In order to process this type of data, natural language processing techniques are suitable [Chowdhary and Chowdhary, 2020] [Evermann et al., 2017].

The advantage of natural language processing is that it can be used to represent a corpus such that a computer can understand it. This can then be used to find relationships with other data variables.

Correlation, association, and causality analysis are useful tools to determine relationships, spot patterns and detect abnormalities and irregularities in data [Rezaee et al., 2020]. These types of analysis can be useful for this research to determine which variables occur together more often than others but also which variable causes the other. This could potentially give insight in what causes sprints to perform poorly. Correlation analysis focuses on the changes in two variables, regardless of the effects of other variables. Association analyses examines the relationship between two variables while holding the effects of other related variables constant. In the study of the causation, researchers are interested in the effect of variable X on variable Y. In other words, whether or not one variable causes another variable to change. [Rezaee et al., 2020]

In the past, natural language processing was mostly useful for predictive tasks. Because advancements in recent years, this is beginning to expand into other research fields. The area of interdisciplinary research at the convergence of causal inference and language processing is starting to emerge [Feder et al., 2022b]. This means that causal relations from text can be inferred.

An addition to these data analysis techniques is process mining [Erdem et al., 2018]. Process mining makes use of event log data in order to provide insights in processes and also in data, organizational and even social structures. The main advantage to this technique is that it focuses on business processes unlike the other techniques [van der Aalst, 2012].

Even with a vast amount of data available, it can be a challenge to find meaningful relations in the data. The reason for this, is that there are many aspects that influence performance. Human behaviour, for example, can influence performance significantly but can barely be measured. Another human aspect can be the interpretation of results. Techniques as linguistic summarization can help to cope with this problem[Yager et al., 1991]. Another human factor is "algorithm aversion" or "algorithm appreciation". This is the negative or positive behaviour towards algorithmic advice compared to human advice.[Hou and Jung, 2021].

After correlations, causation and processes were defined, deep learning algorithms were used to predict which sprints are likely to comply with the estimated duration. For process prediction, recurrent neural networks are used in accordance with natural language processing [Evermann et al., 2017]. Apart from natural language, sprint context can be used to make even more accurate predictions [Hong et al., 2009]. Recurrent neural networks lend themselves to be used for sequential data, this is convenient for processes since these have sequential elements to them. A challenge to ordinary recurrent neural networks is fading and exploding, which happens to larger sequences. This can be solved by using long short term memory networks.

1.1 Problem statement

Software development is hard to manage. The changing traditional software development methodologies, traditional roles, distributed teams, dynamic customer requirements, high feedback loops, a large number of deployments, less documentation, multiple sprints, automation of DevOps and Agile setting, etc. are making projects complicated to manage [Angara et al., 2020].

A study on software performance [Woodside et al., 2007] mentions that half of information technology executives experience performance problems with at least 20% of the applications they deployed. In the field of software performance engineering, one tries to address these issues and optimise performance.

At this point, there is no methodology or application that shows what causes sprints to be late. Certainly not in combination with natural language processing techniques. So far, models like greedy, Lagrangian optimisation and mathematical optimisation models have been used to improve sprints [Boschetti et al., 2014] [Jansi and Rajeswari, 2015] [Golfarelli et al., 2013]. These approaches have certain benefits but do not take in to account company specific information. This company specific information can be experience of certain employees or task content for example. Another aspects which the previous improvements have not taken into account, is the estimated time. The previous solutions only take into account the planning with estimated times and does not account variability in these times.

Another approach that has been widely successful is machine learning (ML). The challenge with these types of solutions is interpretation [Moraffah et al., 2020]. Many of ML approaches act as a so called 'black box' in which it is not clear how solutions are generated. This can result in a weak understanding of the actual problem. Another issue is that these ML approaches often predict solutions only on correlation instead of causation. This can result in sub-optimal and sometimes even dangerous outcomes [Richens et al., 2020].

This results in the problem statement being:

Sprint plannings are generally hard to predict which may cause sprints to perform poorly.

1.2 Purpose of research

The purpose of this research is to explore the potential of integrating causality, natural language processing (NLP), and machine learning techniques to improve agile methodologies in software development. The study aims to investigate how these tools can be used to better understand the causal relationships that impact performance, as well as extract valuable insights from large amounts of unstructured data.

The research will focus on the following objectives:

- To review the literature on causality, NLP, and machine learning in the context of agile methodologies, and identify gaps in the existing research.
- To develop a framework for integrating causality, NLP, and machine learning techniques into agile methodologies, and evaluate the effectiveness of this framework in improving performance.
- To investigate the impact of using causality, NLP, and machine learning techniques on key performance metrics in an agile environment, such as cycle time, quality, and lateness.
- To explore the potential of using predictive modeling to identify potential issues before they occur, and investigate how this approach can be integrated into agile methodologies.
- To evaluate the feasibility and scalability of using causality, NLP, and machine learning techniques in an agile environment, and identify any challenges or limitations that may arise.

The research will be conducted using a mixed-methods approach, involving both qualitative and quantitative data collection and analysis. Qualitative methods, such as interviews and focus groups, will be used to gather insights from software development teams and stakeholders, while quantitative methods, such as statistical analysis, will be used to evaluate the effectiveness of the proposed framework.

The findings of this research will contribute to the development of a more comprehensive understanding of how causality, NLP, and machine learning techniques can be integrated into agile methodologies to improve performance. The study will provide practical recommendations for software development teams and organizations looking to leverage these tools to optimize their agile processes and deliver high-quality products to their customers.

1.3 Research questions

In order to solve for the problem statement, 5 research questions were formulated. These research questions apply to Blis specifically. The research questions are as follows:

- **Main RQ :** How to improve sprint planning performance through data driven insights?

  This is the main question that is going to be answered in the research. As mentioned before, planning performance is the degree of which the sprint finishes in time.

- **Sub RQ 1: How is the current development process performing? Where is the most need for improvement?**

  The performance of the current process is important to asses in order to get a first insight into where the exact problem lies. It will be assessed on how many of the sprints are not on time. Another aspect is to get an insight into which sprints are more likely to perform poorly. This will generate a first general insight into the problem solution.

- **Sub RQ 2: What causes poor sprint planning performance? What role does work item meaning play in this?**

  Causal relations imply more than correlations. With this question, it will become clear which aspects cause sprints to be late. Also meaning of text in combination with causality techniques will be interesting to investigate.

- **Sub RQ 3: How can the potential causal relations be used to improve future planning performance?**

  With this question, it can be proven whether or not the future planning performance can be optimised. This will solve the problem Blis is facing.

- **Sub RQ 4: How can the solution be generalised for other companies?**

  This research mainly focuses on a case study for one company. It would be interesting to argue whether or not this research could be expanded into other, similar companies.

In order to answer the research questions, CRISP-DM is applied. This stands for: The cross industry standard process for data mining. This method has six phases:

- Business understanding
- Data understanding
- Data preparation
- Modeling
- Evaluation
- Deployment

1.4 Justification

In this section, the justification of the research is discussed. This means that, in this section, it is justified why the research is important and why it should be done. It could for example have a social impact for certain people or perhaps an economic impact. There can be many justifications on why the research should be done. Furthermore, there is a distinction between the justification from business perspective and scientific perspective. Business perspective might focus more on operational and economic benefits while the scientific perspective focuses more on adding or improving on current research.

1.4.1 Business perspective

Utilising multiple techniques such as causality, natural language processing and machine learning techniques can bring promising results for improving agile methodologies in software development. By integrating these tools, this research aims to help organizations understand the causal relationships that impact their performance, as well as extract insights from vast amounts of unstructured data.

The design of agile methodologies enable software development teams to be more responsive to changing requirements, improve collaboration and communication without sacrificing quality. However, the success of a project relies on the ability to track and measure performance. This is where the proposed methodologies can play a significant role.

Causality helps an organization to identify the causal relationships among different variables and outcomes. This allows companies to make better informed decisions in the future. By using causal inference for instance, organizations can learn which factors are most closely associated with positive outcomes and leverage this to improve performance.

Natural language processing can extract insights from large amounts of unstructured data. This can help organizations help by processing customer feedback, product reviews and social media posts for example. This in turn can help understanding customer needs and preferences as well as identify potential performance issues that need to be addressed.

Machine learning techniques can help organizations predict potential issues before they occur. Historical data can be used to identify patterns, these patterns are used by machine learning algorithms to predict future trends and help organizations to address issues before they have negative impact.

Integrating causality, NLP and machine learning methods can provide valuable insights for software development teams. It can be used as a tool to monitor and improve performance. This will result in an even better performance and more informed decisions.

1.4.2 Scientific perspective

One way of justifying this research is arguing that this research has not been done before. Though improving sprint planning has been tried multiple times in scientific papers [Boschetti et al., 2014] [Jansi and Rajeswari, 2015] [Golfarelli et al., 2013], there are still improvement opportunities. While past approaches have the same goal, none of these approaches are similar to the proposed approach in this research. One improvement aspect is; using more specified data which differ for every project, user story or task. There are papers which have touched upon this issue [Malgonde and Chari, 2019]. This approach takes 5 user story specific data attributes into account. Jet this approach does not take into account natural language processing, causality or a combination. Another improvement is the fact that [Malgonde and Chari, 2019] makes use of a black box machine learning model which does not give insights into why certain things might be happening. Other approaches [Lucassen et al., 2015][Raharjana et al., 2021] have included natural language processing with success. Furthermore, studies have researched causality in agile sprint planning [Noreika and Gudas, 2023]. In other papers, this combination of NLP and causality is researched [Feder et al., 2022a] but not in combination with agile sprint performance.

In general, research into this topic can lead into new insights and advancements in the field of software development, project management, NLP and causality. This research can help identify underlying causes of project delays and can be a starting point into researching these underlying causes even further in the future. Furthermore the research can provide empirical evidence through the use of scientific methods to support or reject the effectiveness of using NLP and causality for optimization purposes in sprint planning. This evidence can then be used to develop models that can be further tested and refined, leading to more accurate predictions and better decision-making.

The data used in this research is data created by an actual company. Many studies on this topic have been done on public data or data from student software development teams in universities. The fact that this study is going to be done on business data can expand the current research field into more practical applications. This can help to verify theories based on public data sets.

1.5 Thesis structure

The thesis is structured as follows. A literature review which addresses all relevant concepts of the agile development, data/process mining, causality and machine learning is given in chapter 2. The research method conducted during this researched is discussed in chapter 3. The business and data understanding is addressed in chapter 4. Process mining and how it has been applied to this research is shown in chapter 5. Chapter 6 is about the natural language processing of work item titles. The results of (causal) rule mining methods are presented in chapter 7. Chapter 8 presents a machine learning model for making decisions for future sprints. The results are evaluated in chapter 9 based. And finally, the conclusions limitations and future work of the research is discussed in chapter 10.

Chapter 2

literature review

In this section, the general framework of previous research will be further elaborated. Furthermore, gaps and problems in current literature will be identified. To conclude this section, a general statement on how to deal with these factors will be established.

According to a recently published paper[Al-Saqqa et al., 2020]; "agile is a wide umbrella of software development beliefs". It is said to be a conceptual framework for software engineering which starts with the planning phase and iteratively progresses through the life-cycle of the project. Under the agile "umbrella", multiple methods exist[Abrahamsson et al., 2017]. These methods are:

Extreme Programming, Scrum, Crystal family of methodologies, Feature Driven Development, The Rational Unified, Dynamic Systems Development Method, Adaptive Software Development, Open Source Software development, Agile Modeling, Pragmatic Programming.

These methods claim to place more emphasis on people, interaction, working software, customer collaboration, and change, rather than on processes, tools, contracts and plans. Of course there are some major differences withing the different methods, but there are also some aspects that occur in every method. The aspects that make the mentioned methods agile are the following:

- **Incremental:** small software releases, with rapid cycles
- **Cooperative:** customer and developers working constantly together with close communication
- **Straightforward:** the method itself is easy to learn and to modify, well documented
- **Adaptive:** able to make last moment changes

Every method has its own advantages and disadvantages, there is no optimal methodology for all types of projects [Al-Saqqa et al., 2020]. Selecting the best agile methodology to be used in project development must be done carefully based on the variabilities. It can also occur that agile is not suitable for a certain type of project, in that case traditional methods are preferred. This can happen in organizations with a large number of teams, employees and projects. Agile is most suitable for organizations with a small number employees.

In the beginning of agile, extreme programming was the most popular method. However, since 2004 the use of extreme programming has declined. Nowadays, scrum is leading the development trends and plays a very influential role. Scrum is a more general approach which focuses on project management. In the future, agile implementations in big data and cloud environments are likely to occur more frequently [Al-Saqqa et al., 2020].

2.1 Related work

Earlier in this report, it is mentioned that optimising sprint planning has been done before in scientific literature. This section is going to elaborate upon these previous approaches.

2.1.1 Agile & Process mining

Process mining is a very brought field of research and has also found its way into agile methodologies [Erdem et al., 2018]. The main uses of process mining in agile is process discovery. It is useful to determine which processes are actually executed in organisations. The way this is done is by using the available log data on the state changes. The state changes are used in process discovery to see which states the work item has visited. Another use case of process mining is process conformance checking. After process discovery, the processes are checked whether or not these conform with a predefined process model. The final application of process mining in agile is process enhancement where actual executed processes are analysed to find out bottlenecks and delays.

2.1.2 Mathematical optimization

A very popular approach in operations research is optimization modelling. This approach has also been applied to sprint planning optimization [Golfarelli et al., 2013]. This approach states an objective function z which maximizes the integral of the cumulative utilities, where each cumulative utility is the sum of the utilities achieved by the first k sprints. Furthermore, there are constraints to which the solution has to comply with. An example of a constraint can be the maximum amount of story points in a sprint. Story points are the estimated workload of a certain user story. The paper identifies the following improvements upon the current model :

- Allowing different development speeds for different sprints due to a variable team composition
- Modeling different team capabilities (e.g., design, implement, test) so that, in each sprint, the team will be able to deliver a different number of story points for each capability
- Extending the model to support multiple teams working on the same project
- Implementing a structured approach to utility definition and measure its impact on the accuracy of the estimates and consequently on the effectiveness of plans.

Besides these points, this paper does not take into account the variability of user stories and other aspects like developer experience etc.

2.1.3 Lagrangian optimization

A very similar approach to mathematical optimization is discussed in another paper [Boschetti et al., 2014]. In this paper, a similar optimization model is solved, but in a different way. Lagrangian relaxation is used to solve for the most optimal sprint planning. Lagrangian relaxation is a method where a constraint is 'relaxed'. This way, the result of the optimization problem can have a solution which would be infeasible in the original problem. To make sure that the solution stays within the feasible region, the maximisation (or minimization) function is 'punished' by infeasible solutions and rewarded by feasible ones. The reason Lagrangian relaxation is used in this case is because of computational benefits. It is said that the conventional problem takes too much time to be effective for operational use. The computational results show that this heuristic is able to find good quality solutions which were erected much quicker than the original optimization problem. The average gap between the Lagrangian solution and the optimal solution is 0.3 percent and the maximum gap was 1.8 percent. Improvements to this paper are similar to [Golfarelli et al., 2013]. This paper also does not take into account developer specific skills for example.

2.1.4 Greedy heuristic

The greedy heuristic is also a popular method in operations research. In recent research, this has also been applied to sprint planning optimization [Jansi and Rajeswari, 2015]. Greedy optimization focuses on optimising a solution step-by-step based on the most attractive solution. The algorithm takes into account what step can provide the most benefit with the least downsides. Every decision the algorithm makes, has a ratio on which step provides the most profit for the least downside. The reason this optimization model is chosen is similar to previous mentioned reasons. The reason being; the low time efficiency of integer linear programming solvers. In the proposed approach, planning problem are chunked into sub-problem and solved separately in iterations. From the iteration of sub-problem, best feasible solution is retrieved at each step. The computational result using greedy heuristic approach will be able to attain a feasible and optimal solution with very quick time constraints. This approach can be improved by identifying the risk and designing a plan in a real time approach. Then by identifying the skill of each expert, sprint plan can be designed to obtain a further better feasible value as a result.

2.1.5 Genetic algorithm

Genetic algorithms can be used to solve a large variety of problems, this includes sprint planning optimization. In a recent published paper, genetic optimization is applied on sprint planning optimization [Kumar et al., 2014]. Genetic algorithms are based on the idea of natural selection. These algorithms maintain a population of chromosomes. Each chromosome represents a solution. These solutions may or may not be optimal. Three operations called selection, crossover and mutation are applied on the population every generation. The process repeats itself until a acceptable solution is found or the maximum number of generations is reached. The parameters are, the population size, mutation probability, crossover probability and tournament size. The population is evaluated against a fitness function. The fitness function determines the score of a gene and thus how good the solution is. The genes with the highest score are selected to reproduce through crossover and mutation and create the next generation. Two parents are selected at random which create two children which form the new generation (along with the other children). This proposed approach has been tested with several population sizes, crossover rates and mutation rate. The result over 100 generation was that algorithm generated optimal or close to optimal solutions

2.1.6 Machine learning

Machine learning has been rising in popularity in recent years. Optimising sprint planning with AI methods has thus also a field of interest. In recent research [Salehi, 2022], the role of artificial intelligence methods has been investigated. From this research it can be said that the combination between artificial intelligence brings advantages but also many complexities. It is stated that AI is not likely to replace human planners in the near future, but merely automate repetitive tasks [Salehi, 2022]. This can speed up processes and reduce human error. Furthermore, AI can help understand and adopt agile methodologies within organizations. This can help organizations increase productivity and performance.

Machine learning encompasses multiple different methods. In a recent study [Malgonde and Chari, 2019], an ensemble of multiple methods has been proposed. An interesting insight from this study is that none of the predictive algorithms consistently outperformed others.

2.2 State of the art

This section is going to elaborate upon the current state of the art of the current available technology. While section 2.1 has focused more on previous research done on this topic, this section is going to elaborate on the current state of knowledge and suggest directions for future research.

2.2.1 Process mining

Process mining is a sub-field of data science, focusing on the analysis of event data generated during the execution of (business) processes [Berti et al., 2019]. Process mining aims to discover, monitor and improve real processes by extracting knowledge from event logs readily available in today's information systems [van der Aalst, 2012]. Over the last decade there has been a significant growth of (event) data and process mining techniques have improved significantly. This results in management trends related to process improvement using these techniques.

Event logs can be used to conduct three types of process mining:

- **Process discovery** takes an event log and produces a model without using any prior information. This is the most prominent process mining technique. For many organizations it is surprising that real processes are revealed from data. Despite the success of process discovery, there are still many challenges. Real processes are more 'spaghetti-like' than people like to think. It is still quite difficult to capture the complex reality in a model[van der Aalst, 2010].
- **Conformance** is the practice where the mined process is compared with an excising process model of the same process. This can be useful to check if the real process conforms with the model and vice versa. There are a couple of ways to determine conformance[Syring et al., 2019]. The first measure is recall and aims to quantify the fraction of observed behavior that is allowed by the model.The second measure is precision, which aims to quantify the fraction of behavior allowed by the model that was actually observed. The last measure is generalization, which is a challenging concept to define. Generalization needs to reason about behavior that was not observed in the event log and establish its relation to the model. Generalization aims to quantify the probability that new unseen cases will fit the model. The reason this is an interesting measure is because it shows how the model is over-fitted. It can work very well for seen data but perform rather poorly on unseen data.
- **Enhancement** is the idea of extending of improving an existing process model using the information about the actual process. One could use timestamps for example to extend the model to see what the bottlenecks, service levels and throughput times are.

In general, process mining is a new technology which enables evidence-based process analysis [van der Aalst, 2012]. Nevertheless, there are still challenges and most organisations are not yet aware of the potential of process mining.

2.2.2 Causality

Determining and measuring the independent, true impact of specific phenomena is known as causal inference [Holland, 1986]. A key difference between causal inference and association or correlation is that causal inference describes the response of an effect variable when the cause of that effect variable is altered. Association merely shows correlation and does not take into account underlying causal mechanisms. The basic idea of the theory of inferred causation is described in [Pearl, 2009].

[Pearl, 2009] presents how a graph can represent a causal model. A graph where the nodes are variables and the arrows entering any node are the immediate causes can be used as a representation of a causal model.

Causal decision making is an important part of business. Causality is increasingly based on statistical models and machine learning algorithms. This makes it easier for businesses to algorithmically target offers, incentives and recommendations to affect consumer behavior. Recent

research [Fernández-Loría and Provost, 2022] highlights the difference of causal decision making and causal effect estimation. The article states that accurate cause effect estimation is not necessary for accurate causal decision making. The experience is that this phenomenon is not well understood by practitioners and researchers.

The fundamental problem in many predictive tasks is determining whether an intervention or treatment will have a certain effect. For example, when attempting to stop customers from leaving, the goal is not to target the customers most likely to leave, but instead to target the customers who will stay when a certain treatment is applied. In advertising, the advertiser does not simply want to target ads to people who will purchase after seeing an ad. The advertiser wants to target people who will buy something who would not have bought anything if they did would not have seen the advertisement. Generally speaking, instead of predicting the likelihood of a certain outcome, it is often preferred to predict whether a particular case can be improved by means of an intervention or treatment. This type of decision making is part of the problem of treatment assignment [Manski, 2004]. This problem focuses on assigning the treatment associated with the most beneficial outcome. Treatment assignment policies may be estimated from data using statistical modeling, allowing decision makers to map individuals to the best treatment. There are multiple proposed policies across different fields like econometrics [Bhattacharya and Dupas, 2012], data mining [Radcliffe and Surry, 2011] and machine learning in the contextual bandit setting [Li et al., 2010].

One popular approach for causal decision making is making use of machine learning models to estimate causal treatment effects at the individual level, after which models make intervention decisions automatically [Olaya et al., 2020b]. When a company has limited advertising budget, causal effect models can predict expected effect of a promotion on each potential customer and thus target those customers more effectively. As mentioned earlier, causal decision making and causal effect estimation are not the same.

2.2.3 Natural language processing

Natural Language Processing (NLP) is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things [Chowdhary and Chowdhary, 2020]. NLP began in the 1950s as the intersection of artificial intelligence and linguistics. NLP was originally distinct from text information retrieval (IR), which employs highly scalable statistics [Nadkarni et al., 2011].

Classical approaches to natural language processing include [Indurkhya and Damerau, 2010]:

- **Text preprocessing** is used to prepare the text for further analysis. The first step is usually extraction. This means the file content is tokenised into individual words. The second step is to remove stop words. These words are deemed not to add meaning to the text. Then stemming can be applied in order to find the stem or root of a word. The purpose of stemming is to remove various suffixes, to reduce the number of words, to have accurately matching stems, to save time and memory space. [Vijayarani et al., 2015]

- **Lexical analysis** or lemmatisation is used in different ways depending on the task of the natural language processing system. In machine translation, the lexical semantics of word strings can be accessed via the lemma dictionary. In transfer models, it can be used as part of the source language linguistic analysis to yield the morphosyntactic representation of strings that can occupy certain positions in syntactic trees, the result of syntactic analyses. This requires that lemmas are furnished not only with semantic but also with morphosyntactic information. [Hippisley, 2010]

- **Syntactic parsing** determines the sentences structure in one way or another. In NLP approaches based on generative linguistics, this is generally taken to involve the determining of the syntactic or grammatical structure of each sentence. [Indurkhya and Damerau, 2010]

- **Semantic analysis** focuses on identifying the underlying syntactic structure of a sequence of words is only one step in determining the meaning of a sentence; it provides a structured object that is more amenable to further manipulation and subsequent interpretation. [Indurkhya and Damerau, 2010]

- **Natural language generation** focuses on generating text. Subjects that come to mind are: machine translation, summarization, simplification of complex texts or automatic spelling, grammar and text correction.

Recently, huge improvements in speech recognition have taken place. Talking to one's phone is becoming a common activity and web search engines are increasingly successful in understanding complex search requests [Hirschberg and Manning, 2015]. ChatGPT is the latest language model that uses deep learning to generate text in a human-like language format. Like other autoregressive language models, chat GPT predicts the next text element in a sequence of words based on a task given in natural language. Language models like chat GPT belong to a group known as 'transformers', which are capable of predicting text through a mechanism called 'attention'. Attention builds upon mechanisms used by earlier language models which were sequential in nature which consider all prior inputs in order to determine an output. [Nath et al., 2022]

2.2.4 Machine learning

Machine learning is the study of algorithms and statistical models that computer systems use to perform a specific task without being explicitly programmed. These algorithms are used for various purposes like data mining, image processing, predictive analytics, etc.. The main advantage of using machine learning is that, once an algorithm learns what to do with data, it can do its work automatically. [Mahesh, 2020] Machine learning can either be supervised or unsupervised. If there is a small amount of data which is clearly labelled, supervised learning is preferred. Unsupervised learning, generally, is better for larger data sets.

There are different kinds of machine learning. One can distinguish between the following different kinds [Ghahramani, 2004].

Supervised learning is a method where the machine is given a desired sequence of outputs. The goal is to learn to produce the correct output given a new input. This output can either be a class or a real number.

Reinforcement learning is a method in which the machine interacts with an environment by producing actions. These actions affect the state of the environment. This then results in the machine receiving some rewards or punishments. The goal of the machine is to learn how to act such that the future rewards are maximised.

The last type of machine learning is **unsupervised learning**. The machine receives inputs but does not obtain target outputs or rewards from an environment. The goal is to build representations of the input that can be used for future predictions. Unsupervised learning can be thought of as finding patterns in the data.

In order to make good predictions, the available data has to be of good quality but also the hyper parameters should be optimized. For this task, there are multiple algorithms available [Bergstra et al., 2011]. The most basic approach would be to select parameters randomly and asses the results, this is called random search [Bergstra and Bengio, 2012]. In this approach, a spectrum is given in which the random results are to be selected. A similar approach is grid search, the difference is that the parameters are not selected at random. The parameters are pre-defined and results are calculated for every combination [Pontes et al., 2016]. These algorithms can be time consuming for large sets of parameters. Bayesian optimisation is a well-established strategy for optimisation of hyper parameters [Snoek et al., 2015]. This approach relies on the construction of a probabilistic model that defines a distribution. This makes it possible to seek the optimum of a function of interest.

2.2.5 NLP & Process mining

Process modeling has been adopted by many organizations in various context. The process mining techniques extracts information from systems event log. In many cases, the process is incorporated by human activities which will not be present in logs. Recent literature [de AR Goncalves et al., 2009] presents an approach that explores the techniques associated with text mining and natural language interpretation for automatic generation of process models. The paper presents a refinement of a collaborative method for designing a process model. It assumes that telling a story is natural to people and thus a good way of capturing knowledge about activities. It is not easy to extract information from text and it can be rather time consuming. These are the main reasons the paper suggests automating this process using text mining techniques. Challenges like the variety of ways that workflow elements can be described and domain-specific concepts are still challenges that need to be taken into account in the future. This can be solved with an external glossary, dictionary or ontology. Future work on this topic includes performing a case study on a real scenario within an organization.

2.2.6 NLP & causality

A recent study has shown that it is possible to extract cause-effect relations from requirements [Fischbach et al., 2020]. Requirements often describe behavior by causal relations (if A, then B). Causal relations embedded in requirements are extracted to derive test cases automatically and to reason about dependencies between requirements. Existing methods were not capable of extracting these types of relations but using Tree Recursive Neural Networks, great performance can be achieved. Another study [Doan et al., 2018] also extracts causal relations from text (from tweets). In this case, a rule set template including causal verbs, clausal verb phrases, and clausal noun phrases is created. For example, a tweet containing “A caused B” has “caused” as a clausal verb, or “A result to B” has “result to” as a clausal verb phrase. However, another study [Li and Mao, 2019] suggests that defining rules requires extensive manual work. Instead of a rule based approach, the paper suggests a machine-learning-based method. current approaches either rely on sophisticated feature engineering which is error-prone or rely on large amount of labeled data which is impractical for causal relation extraction problems. To address these issues, a knowledge-oriented Convolutional neural network is suggested. The convolutional filters in knowledge-oriented channel are automatically generated from lexical knowledge bases such as WordNet and FrameNet. The paper proposes filter selection and clustering techniques to reduce dimensionality and improve the performance of K-CNN. Furthermore, additional semantic features that are useful for identifying causal relations are created. Three datasets have been used to evaluate the ability of K-CNN to effectively extract causal relation from texts, and the model outperforms current state-of-art models for relation extraction.

Previous mentioned approaches are effective at extracting relations within text. Another study [Liu et al., 2021] investigates causality and contributory factors of pipeline incidents by employing natural language processing and text mining techniques. In this paper, two methods of text analytics: K-means clustering and co-occurrence network, are employed to infer latent causality of incidents.

2.2.7 Causality & process mining

The study of process mining builds directly on relations. Even though various improvements have been made last decade, there are weaknesses of these relationships. The reason being is that event sequence as described in classical event logs differs from event causation. In a recent study [Waibel et al., 2022], the problem of representing the causal structure of process-related event data is addressed. A new approach called causal process mining is developed to improve upon current methods. This approach makes use of relational databases instead of event data. This means that relational data can be transformed into a causal event graph. The result of this paper is evaluated and compared different techniques which both have been applied to a case study. The result of

the research is that of all relationships, the proposed approach captures the differences between causal and non-causal relationships.

2.2.8 Causality & machine learning

The interest in causality by the machine learning community's has significantly increased in recent years [Schölkopf, 2022]. The reason for this is that causality can lead to more invariant or robust models. Researchers in many fields have acknowledged that machine learning models are considered to be black-box. This means that the user does not know how the machine learning model works and why it makes certain decision. Even when the mathematics is clearly defined, the models still lack an explicit declarative knowledge representation. This makes it difficult to identify the underlying explanatory structure [Hair Jr and Sarstedt, 2021]. To address this problem, computer scientists have demanded that machine learning methods should also provide an explanation of the process that is understandable for humans [Holzinger et al., 2019]. Unfortunately, the ability to formulate an understandable line of reasoning has been a major limitation for current machine learning approaches [Pearl, 2018]. Explainable AI would help increase acceptance among decision makers and also bring insights into potential causal mechanisms. Another issue is the inability to connect cause and effects which [Pearl, 2019] considers to be a 'necessary ingredient for achieving human-level intelligence". [Pearl, 2009] [Schölkopf, 2022] argue that the the obstacles mentioned above can be overcome when traditional ML is enriched with causal modelling tools, which is called Causal Machine learning. Recently, studies [Bozorgi et al., 2020] [Bozorgi, 2021] [Shoush and Dumas, 2022] have tried to support business decision-making by applying uplift modelling. This is a group of techniques used to estimate the incremental effect of certain actions [Gutierrez and Gérardy, 2017]. Another field of study, which is called 'causal decision making', can also be useful but is differs from causal effect estimation in the following three ways [Fernández-Loría and Provost, 2022]:

- Causal decision making optimizes for accurate "treatment assignment" rather than for accurate effect-size estimation.
- Confounding does not have the same effect on causal decision making as it does on causal effect estimation. It may be just as good or even better to learn with confounded data as with unconfounded data.
- Causal statistical modeling may not be necessary at all to support causal decision making because a proxy target for statistical modeling might do as well or better.

2.3 Gaps & problems

As can be seen in this chapter, sprint planning optimization has been attempted numerous times. In this subsection, the potential for new approaches is going to be discussed.

Up to this point there is limited understanding on causal relationships in Agile contexts. Causality involves understanding the causal relationships between multiple factors such as; user stories, team composition and project outcomes. However, the research addressing the challenges of inferring causality in the Agile environment is lacking. Developing methodologies and models specifically purposed for causal analysis in the Agile environment is a significant research gap.

Natural language processing can be of great use in extracting information from textual data from Agile sprint planning. However, most excising NLP models are used to determine correlations instead of causation. Research efforts are needed to explore how NLP techniques can be used to erect causal relationships.

Agile methodologies emphasize the importance of human collaboration and communication. Many research suggests that human aspects are not taken into account. While NLP can partly automate human aspects of Agile sprint planning, it is crucial to consider more human-centric factors which may impact decision making. These factors being experience, team composition etc.. Research should explore how the human-centered aspect can be taken into account. Furthermore research should investigate how human decision making can be supported and enhanced.

Addressing these research gaps and problems can significantly advance the various fields concerning Agile sprint planning. By developing methodologies, models, and techniques, the research can contribute to more accurate, efficient, and human-centric agile planning processes.

Chapter 3

Research method

In this chapter, the methodology and necessary materials are going to be discussed. In the end of chapter 1, research questions were formulated. The literature review allows us to acquire all the necessary knowledge to establish a methodology to answer these research questions. The goal of this chapter is go sketch an outline of the intended stages of the research.

CRISP-DM is the de-facto standard and an industry-independent process model for applying data mining projects [Schröer et al., 2021]. This method has the following six stages:

1. **Business Understanding:**

   - **Determine business objectives**
     It is important to understand, from a business perspective, what the business really wants to accomplish.
   - **Assess situation**
     Determine resources availability and project requirements.
   - **Determine goals**
     In addition to defining the business objectives, it is important what a successful project should accomplish.
   - **Produce project plan**
     Select technologies and tools and define detailed plans for each project phase.

2. **Data Understanding**

   - **Data collection**
     At first, the data will have to be collected. Multiple data sources are available. Most sprint data is collected by azure devops. Azure DevOps is a Microsoft tool which provides a complete end-to-end toolchain for building, testing, and deploying software. It includes various features such as: version control, agile planning, continuous integration and delivery, and automated testing. With Azure DevOps, development teams can collaborate and manage their work more efficiently, allowing them to deliver high-quality software faster. The tool integrates third-party applications and services, making it a versatile solution for software development and project management. In azure devops, one can find which sprints belong to a certain project and which user stories belong to certain sprints. Along with this information, the time when a certain user story was initiated and when one was finished can also be found. This data can be extracted with a python API for azure devops. The second data source is called Simplicate. Simplicate is a business management solution, designed for small to mid-sized businesses. It offers different types of tools to help users manage various aspects of their business. These

aspects include: project management, CRM, time tracking, invoicing, and financial reporting. The software is designed to be user-friendly and customizable to fit the needs of different businesses. The main relevant data from Simplicate is going to be the time tracking data.

- **Data exploration**

  A deeper understanding of the available data is going to be obtained during this phase. The data is explored in order to investigate which data is available and what the quality of the data is. The data exploration phase is guided by the problem definition and literature review. The focus of this phase is to determine which data is suitable for the chosen data-driven strategy. At first, general statistics are going to be extracted to get a general idea of how the data is distributed. Subsequently, sprint performance in analysed to see if deviations or abnormalities occur. This can give a general insight into which aspects cause a sprint to be late. The data exploration can already give a slight insight into how the process is performing, this means the basis is established for answering sub RQ 1.

3. **Data Preparation**

   In order for the analysis to perform well, the data has to be prepared properly. Using incorrect, skewed or too noisy data can affect the quality of the model negatively. The data has to be prepared such that the performance will be optimal. In order to prepare the data, the following steps will have to be taken: Data integration, data cleaning, data transformation, feature selection and data reduction.

   - **Data cleaning**

     A large part of data cleaning is removing data points that are not complete or do not make sense. Removing typos and such can also be a part of this process. It could also occur that data is added to the data set with estimated values. In this phase, it can also be useful to asses whether or not the data set is skewed and resolve this. Most likely, the data is going to be skewed since most sprints are going to be on time

   - **Natural language processing**

     In order to answer sub RQ 2, the causal relation between sprint context and performance will be analysed. At first the natural language is going to be processed. The natural language in the data describes the content of the user story or bug fix. The general approach of NLP is described in subsection 2.2.3, in this section a more detailed approach is proposed.

     To start the NLP procedure, the text has to be pre-processed. This is done with sentence segmentation and word tokenization. Sentence segmentation is not necessary because every user story already is a single sentence. Tokenization is necessary in order to evaluate the words separately.

     Next up is stemming or lemmatization. Stemming is most likely more suitable since this method does not rely on a dictionary. The dictionary of lemmatization does not include IT specific words. Lemmatization can also have benefits so both options should be investigated. Words that do not give extra meaning to the phrase can be removed.

     This process is called stop-word removal. Stop words are in a pre defined list which are removed from the sentences.

     In order to describe the meaning of a sentence, the sentence can be represented as a vector. There are multiple ways of doing this. In order to take the context into account, a paper about sense2vec [Trask et al., 2015] describes how part of speech is taken into account. This means that it matters whether a word is a noun, verb or adjective. For example, apple can be a noun which means that it is a fruit. Apple can also be a proper noun which means that it is the brand apple. Other approaches like

bag-of-words do not take these types of relations into consideration. When vectors of sentences are formed, the vectors can be clustered using unsupervised machine learning techniques. This means that all vectors corresponding to a certain cluster have similar meaning. Another way of determining similar meaning is cosine similarity which can also be investigated.

- **Feature construction and selection**
  At this point, some features have already been constructed. These features are natural language oriented or process oriented. An example of a feature created from process perspective can be whether or not a task has taken longer than average. Other features can be created and selected based on which features contain most useful information. Features without relevant information can be left out of the further analysis. According to [Kumar and Minz, 2014], data mining algorithms speed up and accuracy improves when redundant or noisy data is left out. This means that before these analysis' take place, it can be beneficial to implement (partial) feature selection.

.

4. **Modeling**

   In this part, various models are built with different modeling techniques. First, multiple methods are selected that are deemed necessary. These methods are implemented in models which are then assessed.

   - **Process mining**
     During this phase, process mining is going to be used in order to gather more insights into the process performance. This step will help answer sub RQ 1 by determining what the process is and how the process is behaving. Models like CMMN, DMN and goal modeling can be used to give a visual representation of how the process behaves.

     User stories can have multiple states. Which states, how many states and the duration can provide a vast amount of information. The log data from the state transitions can be imported in the Fluxicon Disco environment to analyse it. This environment contains a discovery algorithm which is an updated version of Christian W. Gunthers Fuzzy Miner process discovery algorithm [Rozinat,]. To asses the exact number of hours put into each process step, the time warp function in Disco can be used. This function takes into account business working hours, which can be set manually (between 9:00 and 18:00 for example). With this information, the algorithm calculates the amount working hours there are in between process steps. When the process is modeled, a process model can be made with signavio for instance. In order to determine the performance and answer sub RQ 1, multiple performance measures can be used. An example of measuring performance can be conformance. Conformance takes into account if the process conforms with the desired process. The question that arises is: what is the desired process? This will have to be analysed during the research. Examples of the desired process can be a process with a path that has the least amount of steps or a path without a step backwards. Other performance measures can be throughput time.

   - **Causality**
     When the meaning of each user story is determined, statistical causal methods can determine the causal relationship with performance. Causality is a complex concept which is based on answering the question 'why' and 'how'. Causal inference is an important branch of causal analysis. Machine learning and casual inference are two techniques that developed separately, however, recent studies suggest that these techniques intersect and provide an interesting new approach called causal machine learning [Zhao and Liu, 2023]. This can be done with the causal ML package in python which provides a suite of uplift modeling and causal inference methods using machine learning

algorithms. These algorithms allows users to estimate the conditional average treatment effect (CATE) or individual treatment effect (ITE) from experimental or observational data. Another useful tool could be causal association rules. There is a method available [R.eshuis, 2023] which combines Causal Analysis with Causal Association Rules. This method is also applied to improve business processes so this is also likely to be applicable to this case. In order to find out which approach is most suitable, multiple approaches can be applied after which the most suitable approach will be selected.

- **Causal decision making**
  In order to answer sub RQ 3, causal intervention is going to be applied. Causal decision making differs from causal effect estimation [Fernández-Loría and Provost, 2022]. Causal decision making focuses more on whether a particular individual's outcome cam be improved by means of an intervention. For example, when attempting to stop customers from leaving the store, the goal is not to target customers which are most likely to leave. The target is to target customers who would have left if they did not receive the treatment.

  One type of causal decision making is an instance of the problem of treatment assignment [Manski, 2004]. Each possible course of action corresponds to a different treatment an ideally, each individual is assigned to a treatment with the most beneficial outcome. Treatment assignment can be estimated from data using statistical modeling, this allows decision makers to decide the best treatment according to their characteristics.

  Another approach for causal decision making is to used machine-learned models to estimate causal effects at the individual level. This model can then be used to estimate intervention decisions automatically [Olaya et al., 2020a]. These causal effect models can be used to predict the expected effect of sprint performance on each individual aspect of a sprint and subsequently target the aspects that predict the largest effect.

5. **Evaluation**

   Do the models meet the business success criteria? Which one(s) should be approved for the business? It is also important to review the process. Did the work accomplish what it was trying to accomplish? was anything overlooked? When these steps are taken, the next steps can be determined. These steps can be wheter to proceed deployment, iterate further or initiate new projects.

   Another part of the evaluation is generalizability. In order to answer sub RQ 4, the following will have to be taken into account:

   - **Research Design**: Examine the research design used in the study. Strong research designs enhance the likelihood of generalizability.
   - **Contextual factors** such as the specific agile practices employed, the maturity level of the organizations involved, or the specific techniques used.
   - **Variables and causal relations**: Are the variables used available in other companies and are they likely to have similar causal relations?
   - **Practical implications**: Check whether it is practically possible for other companies to use this methodology.
   - **Expert opinion**: Discuss with a practitioner of agile methodologies whether or not the research can be generalized to other companies.

6. **Deployment**

   Finally the solution is deployed inside an app for planners to use.

Chapter 4

Business and data understanding

In order to get the best results, a better understanding of the business and the available data is gathered. First, the business understanding part is addressed. After the business goals are set, the data can be gathered and visualised.

4.1 Business understanding

This phase involves understanding the project's objectives, defining the current situation, and determining the goals.

4.1.1 business objective

This section is going to determine what the business really wants to accomplish. In essence, Blis is a software company that provides software solutions to their customers. But what Blis is really trying to accomplish is to help other business to get more value from their businesses through digitalisation. This is done in the following ways:

- Data driven business. Help other companies to gather more information from their data.
- Business process automation. Automate business processes to reduce cost.
- Software modernization. Update outdated systems such that these work more efficiently.
- Software research. Determining what software solutions are necessary for the customer.
- Software management. Helping other businesses managing their software.

It can be said that Blis does not focus on the end consumer. Blis creates value through delivering software to other businesses. The value is created through optimizing, automating and managing software these businesses use to generate revenue. The business' main objective is thus: creating value for other companies through digitalisation. As mentioned before, managing these projects are cut down into sprints. When a sprint is late or not all work items are finished within the sprint, the customer is not satisfied. When there are too little items in a sprint, the project is not moving fast enough. Therefore it is necessary to optimize these sprints in order to get as much value for the customer as possible without losing customer satisfaction.

4.1.2 Current situation

In order to manage projects, projects are cut down into work items. These work items are allocated to sprints. The sprints are filled by a planner which decides how many work items are going to be done per sprint and by whom. The sprints are mainly planned based on skill and experience.

The available resources for this project are the databases. There are two main databases that are going to be used in this project. At first, the Devops database which harnesses all data about sprint planning and work items. The second data source is Simplicate which contains data about hour registration on projects.

Risks associated with this project include:

- Incomplete data. When there is a large amount of data missing that should have been available.
- Wrong data entries. Data entries are always relying on people and people can always make mistakes. Wrong data entries can lead to wrong results.
- Insufficient amount of data. The success of certain techniques also relies on the amount of data. When there is not enough data, it is often hard to recognise patterns that make sense.
- Lacking skill resources. Whenever there is not enough knowledge within the company to help overcome hurdles that might arise along the way.
- Lacking performance. The solution is not performing as intended.

4.1.3 Data mining goals

Apart from answering the research questions, the goals of this project is to make sure the planner is going to make better informed decisions. It should be clear what has caused certain sprints to perform poorly in the past and what can be done to make them perform more optimally. Other data mining goals include:

- Gather data from the available data sources.
- Merge, visualise and create data.
- Determine patterns within the data.
- Create decision support system for sprint planning.

4.2 Data source: Devops

In order to get useful results. Good data has to be gathered, cleaned and in this case also merged. There are two main data sources that are available for this project. At first there is Azure devops, which includes all software related data.

For the Devops data source, multiple projects will be analysed. For these projects, all work items will be pulled from the database. The following attributes are available for the work items:

- **Work item ID:** Every work item has a unique id number.
- **Title:** A string which describes the work item.
- **Work item type:** Task, User story, Bug, Feature or Epic.
- **Comment count:** This is the amount of comments developers have left on work items.
- **story points:** Amount of story points.
- **Created by:** Whoever created the work item.
- **Resolved by:** Whoever resolved the work item.
- **Project:** To which project the work item belongs to
- **Iteration:** Sprint to which the work item belongs.
- **Priority:** Some work items have priority over another and thus will have a higher priority level.
- **Parent:** This shows the id of the parent work item
- **Children:** This shows a list of id's of the child work items

The next step is gathering the event log data. The event log data frame has slightly different attributes. The main difference is to the first data frame is that, this log data has all historical activities of the work items. So while the first data frame has one data point for every work item, this data frame can have multiple data points for every work item. The attributes per log item are as follows:

- **Work item ID:** Every work item has a unique id number.
- **Title:** A string which describes the work item.
- **Comment count:** This is the amount of comments developers have left on work items.
- **Created by:** Whoever created the work item.
- **Activity:** Type of work, mostly backend or frontend developent.
- **Project:** To which project the work item belongs to
- **Iteration:** Sprint to which the work item belongs.
- **State:** In which state the work item is at a certain time.
- **Change date:** Date where work item is changed.
- **Changed by:** Whoever changed the work item.

As can be seen, there is not yet information about how long a sprint takes. For this, a separate data frame is created. For this data frame, the start date and end date have been subtracted from one and other with only workdays being taken into consideration.

4.3 Data source: SQL / Simplicate

For the second data source, there are 18 data attributes available. For the sake of this project, only a couple will be exploited. The following data is available:

- **Employee Name:** Employee name.
- **Activity:** Activity name, mainly
- **Description:** Small description of activity.
- **Hours:** Amount of hours written on certain activity.
- **Date:** Date on which the activity has taken place.
- **Organisation:** The organisation the project is done for.

4.4 Data Merge

When merging these data sources, it is necessary to know which hours are committed on certain work items. The first problem is that hours are not registered on work item id's but on company names. An algorithm has to be made that matches hours and work items based on different aspects. In this subsection it is going to be made clear which how an algorithm make sensible matches.

The first attribute on which matches can be made is Employee. This is an attribute which occurs in both data sources. The names of employees are identical in both sources so no extra changes will have to be made.

The second aspect on which can be merged, is 'Project' from the devops side and 'Organisation' on the Simplicate side. The organisation name and project name are often very similar. Mostly the only difference is 'B.V.' at the end or something similar. In order to match these names, fuzzy merge is used. This algorithm matches most similar items in two separate lists. Using this technique, knowing which project belongs to a certain organisation can be automated. Only in one instance the algorithm fails to make the right match. In this case, the match has to be made manually.

The third attribute on which can be merged, is 'Activity' on the devops side and 'Activity' on the Simplicate side. On the devops side, the activities are mainly frontend and backend development. On the simplicate side there are multiple activities including frontend, backend, app development and user interface etc. With these different attributes, fuzzy merge also has to be applied.

When a merge would be applied purely on nearest date, hours would be matched to wrong people, wrong projects and wrong activities. Thus making a lot of miss matches. When all attributes are matched at the same time, there are matches that should have been made but are not. One example of why this happens is, because there are more activities on the simplicate side.

To address these problems, multiple merges are going to take place to make sure all matches are made properly. The hours are going to be matched with the log data. The reason for this is because, every time a work item changes, it is registered in the log data. This change means that an employee has changed something regarding that work item. The first merge is going to match based on all three attributes: Employee/Changed by, Project/Organisation and Activity/Activity. This merges matches items which have nearest dates to one and other. The maximum date difference of this merge is set on one week. The reason for this is that, if all the criteria are met, it is still highly likely that the hours are meant to go to the work item. The reason that it can be one week apart is that some employees admit to not saving work for an entire week.

The second merge is going to try to merge the rest of the items and hours that have not been merged yet. This merge is based on two attributes: Employee/Changed by, Project/Organisation. The merge is also being done on the closest date with a maximum of 3 days. The reason that

the maximum time difference is set on 3 days is that it is less certain that the match is actually correct since there the match only filters on two attributes. Matches that are made in this part that have not been made before are matches that are registered as "frontend' on the devops side and 'app development' on the simplicate side for example.

The last merge is going to match all work items and work hours that have not been matched yet. This is done on only a single common attribute: Employee/Changed by. Because there is only one attribute that has to be satisfied, the time difference is set on a maximum of one hour. The reason this is not larger is because the certainty of the match actually being true is very small when the only similarity is the employee.

4.5 Data creation

Now that the data is collected and merged, other insights can be created from the two data sets. The most important one is a performance based column. The difficulty of determining performance is, that it does not always matter how long a work item takes (unless it takes longer than the sprint duration). Some work items take a lot of time but do not necessarily perform poorly. One could say that a work item performs poorly when it takes more hours than the planner has predicted. Unfortunately, there is no data available on how many hours were predicted per work item. A way of predicting the general workload of a work item (mostly user stories) is story points. The first performance measure is measured as the amount of hours spend on a work item divided by the story points.

Another variable creation is the total active time of a work item. This can also give an indication on how much work is put into a work item. This is not the exact figure of how many hours are spend on a work item but can certainly give more insight into performance. The reason being is that, there are is a chance that mismatches made in matching hours to work items. This is a different approach to the same statistic such that the problem can be seen from multiple perspectives.

4.6 Data visualisation

To get a general idea of the data, the descriptive statistics are shown in Table 4.1:

Table 4.1: Descriptive statistics

Attribute	**Occurance**
Total work items	15801
Total events	187186
Total sprints	184
Total employees	45
Total states	15

To get a better understanding of the data, the most important factors of the data are visualised. At first, the type distribution of the work items is displayed in figureFigure 4.1.

Table 4.2: Sprint performance

	Total sprints	**Average sprint duration**	**Succes rate**
Project 1	30	11.45 days	79.3 %
Project 2	38	9.68 days	89.4%
Project 3	48	9.31 days	91.5 %
Project 4	25	9.48	84.0 %

In Table 4.2 the sprint duration metrics are given. A sprint is deemed successful if it takes less than two workweeks (10 workdays). As can be seen, even with some work items being moved to other sprints, the amount of sprints taking less than 10 days is in the worst case only 79 percent.

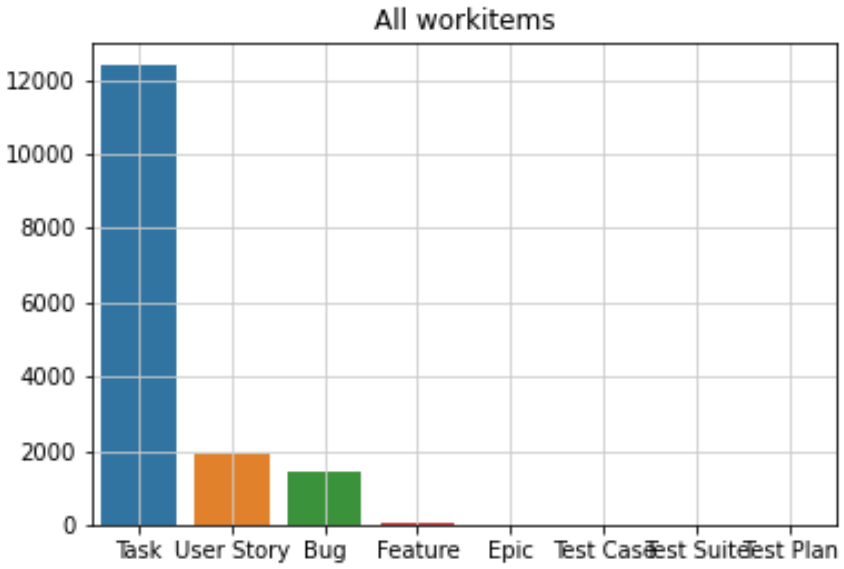
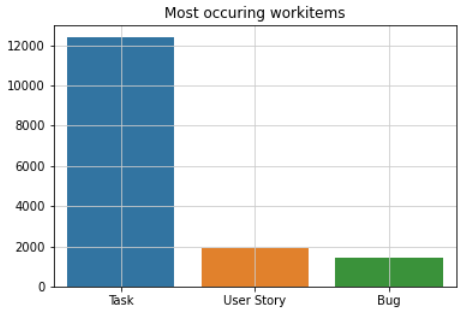
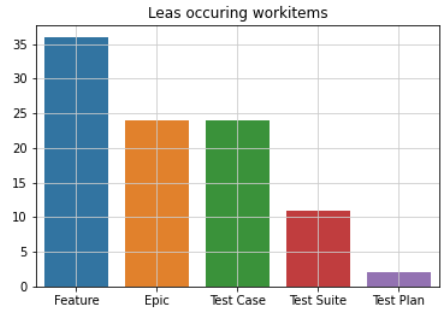


Figure 4.1: Types of work items

Immediately, it becomes clear that there are many more tasks than other work item types. The amount of user stories and bugs are about the same. This distribution makes sense because tasks are part of user stories and user stories are part of features. Features on their part are part of epics. In Figure 4.2, it is visualised more clearly how many of each work item type there are.



(a) Frequent



(b) Infrequent

Figure 4.2: Separated work items into occurance

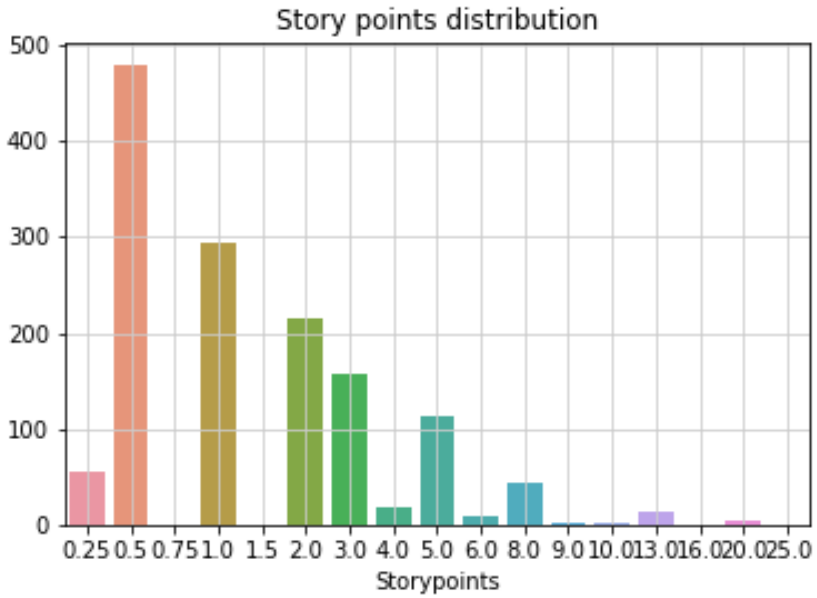


Figure 4.3: Story points

In Figure 4.3 the distribution of the amount of story points appointed to work items is visualised. Most work items that have story points appointed to them have 0.5 story points. The company tries to give Story points to work items according to the Fibonacci sequence. This sequence starts with one and adds the last two numbers in the sequence to compute the next. This results in the sequence: 1,1,2,3,5,8,13 which corresponds with the most occurring amount of story points given to a work item.

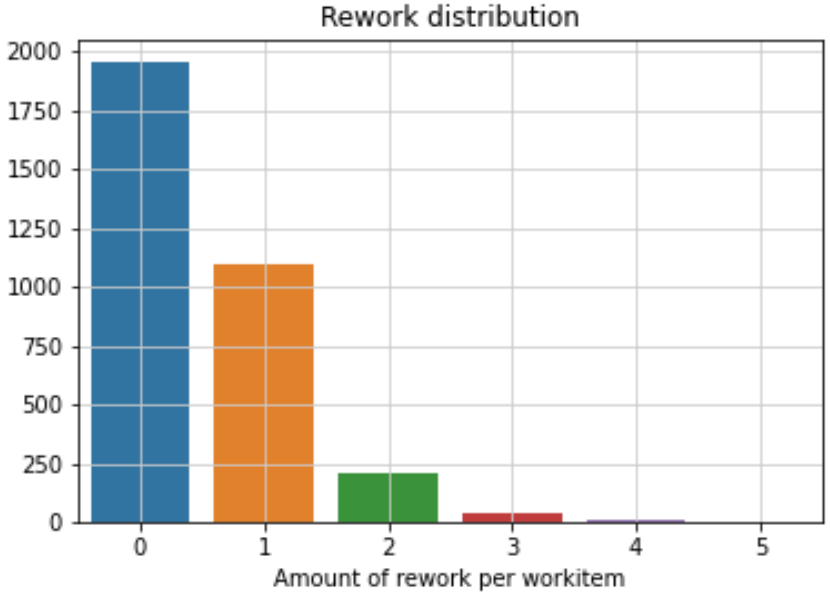
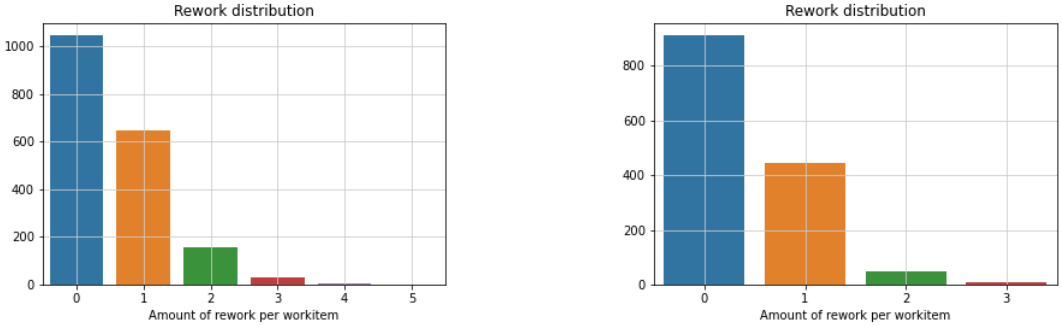


Figure 4.4: Rework bugs and user stories

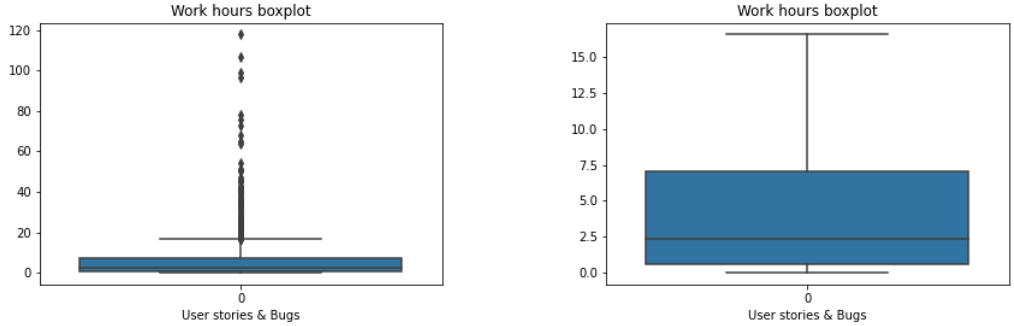
In Figure 4.4 the amount of rework is visualised for user stories and bugs. Whenever a work item becomes active twice or more, it is considered to have had rework. This means that the initial finished work item was not good or needed to be moved to another sprint. It is remarkable to see that one third of every user story or bug has rework done to them. In Figure 4.5 the difference in rework for work items and bugs can be seen. Not only do bugs have rework done more frequently but also more rework per work item.



(a) Rework user stories

(b) Rework bugs

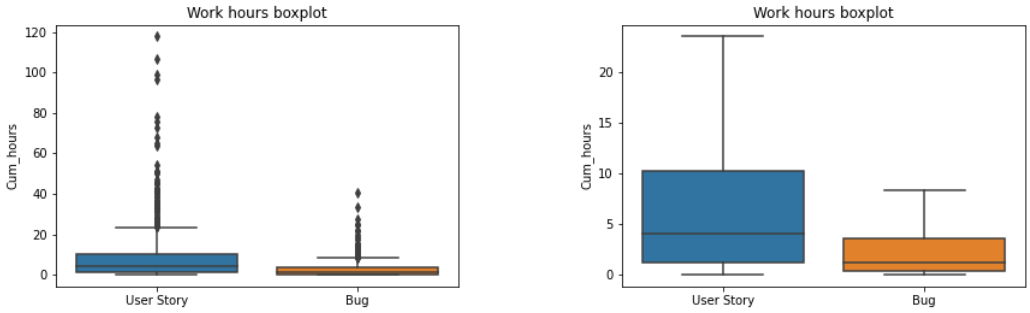
Figure 4.5: Separated rework



(a) Work hours with outliers

(b) Rework without outliers

Figure 4.6: Work hours user stories and bugs



(a) Work hours split with outliers

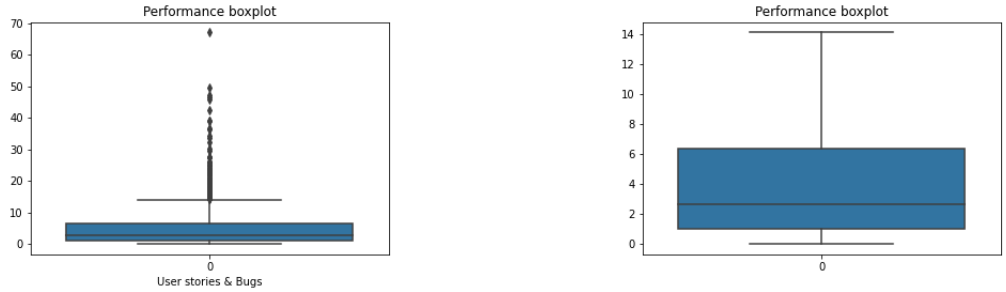
(b) Work hours split without outliers

Figure 4.7: Separated work hours with and without outliers

In Figure 4.6 one can see the work hours from simplicate which are assigned to user stories. On average, a bug or user story takes 2,5 hours to complete. The time it takes is skewed and has quite a lot of outliers. Figure 4.7 shows that bugs take less time on average than user stories. The distribution of bugs also seems to be less skewed.

Next come the performance measurement statistics. Remember that performance is measured in hours divided by story points. This means that a higher performance actually means a worse performance.

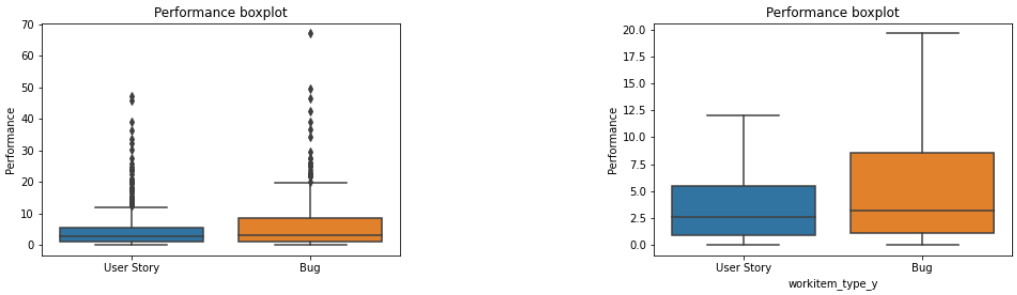
Figure 4.8 shows that the performance is skewed. Interesting is that; while the hours spent on user stories is higher, the average performance is lower (thus better). This means that the hourly prediction for bugs is less likely to be accurate than the user story prediction. It makes sense that bugs are a little harder to predict because a bug is a mistake in the code. It is often unknown what the cause of the bug is and it tedious to find out.



(a) Performance with outliers

(b) Performance without outliers

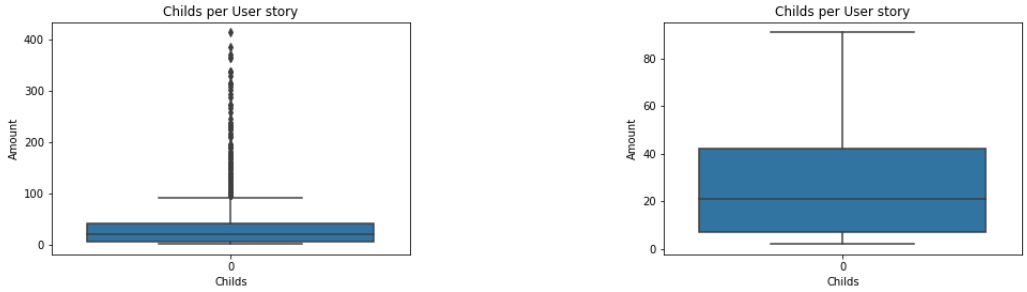
Figure 4.8: Performance



(a) Separated performance with outliers

(b) Separated performance without outliers

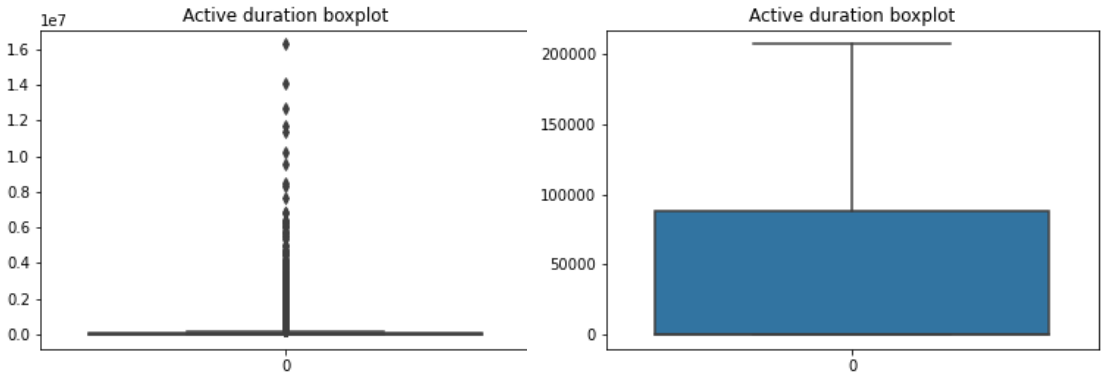
Figure 4.9: Separated performance



(a) children with outliers (b) children without outliers

Figure 4.10: Amount of children per user story

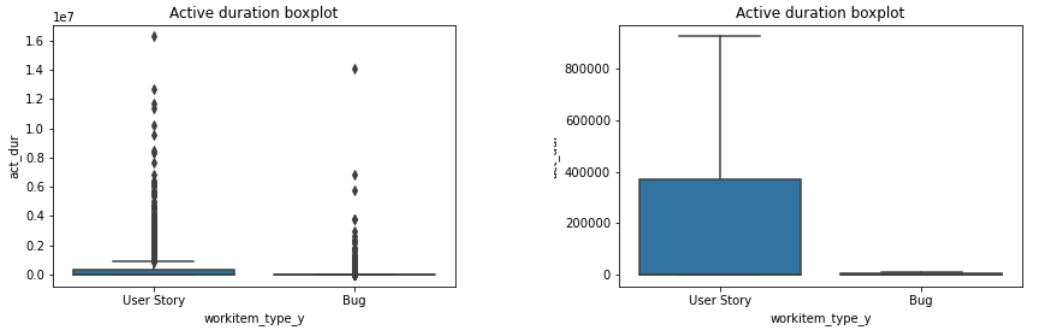
In Figure 4.10, the amount of tasks per User story are shown with and without outliers. The most amount of task a user story has is 400 and the average is 20.



(a) Active duration with outliers (b) Active duration without outliers

Figure 4.11: Active duration

Figure 4.11 shows how long a work item has been active. This means that the state of the work item is set on 'active' which means that it is being taken care of. In Figure 4.11a it is shown that the distribution is incredibly skewed. The work items that take incredibly long are most likely poor performers. Another cause of this large duration can be that people forget to make the item non-active while the item is being transferred to another sprint. Figure 4.12 shows the difference between bugs and user stories with respect to the active duration. It is clear from this picture that the active duration of bugs is significantly less.

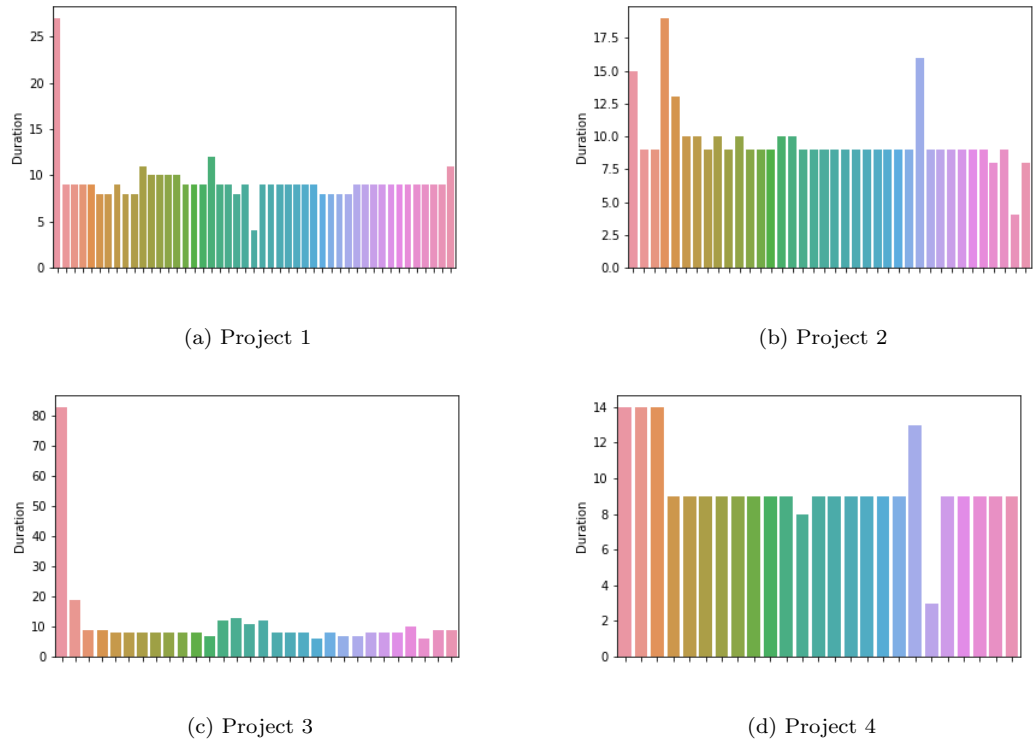


(a) Separated active duration with outliers

(b) Separated active duration without outliers

Figure 4.12: Separated active duration

Figure 4.13 shows all sprint durations per project. The project names are anonymous because they often include a company name. As can be seen, the first sprints are often longer than anticipated. Another factor to take into account is that, at the end of a sprint, some work items can be skipped and moved to the next sprint. This also means that the sprint is 'too late'. This is going to be taken into account in next chapters. For this visualisation, only sprints taking longer than 10 days are taken into account.



(a) Project 1

(b) Project 2

(c) Project 3

(d) Project 4

Figure 4.13: Sprint duration

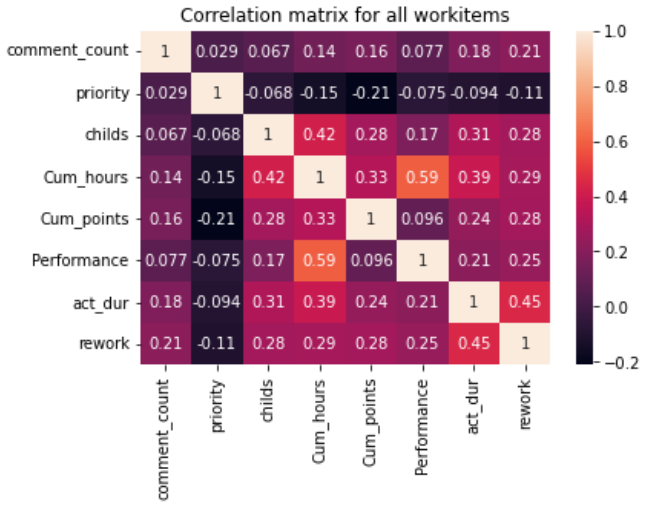
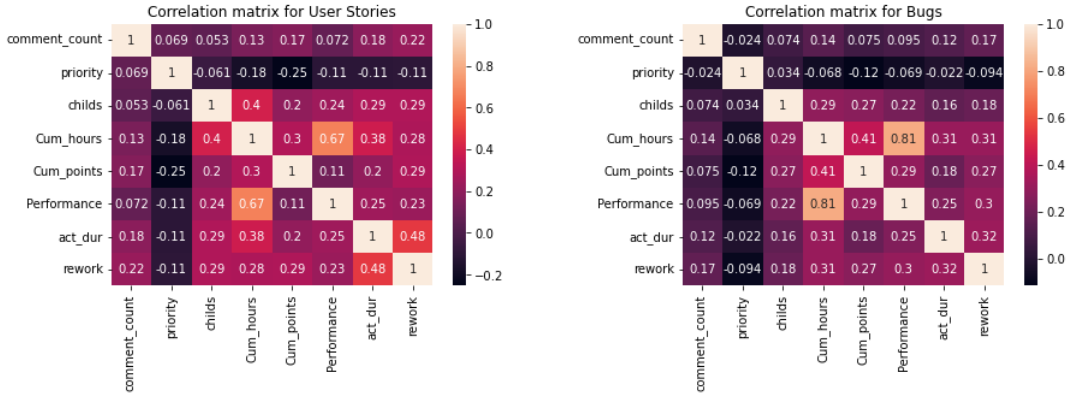


Figure 4.14: Correlation matrix

In Figure 4.14 all correlations between the important variables are shown. The highest correlation occurs between performance and cumulative hours. This makes sense because performance is cumulative hours divided by story points. It is therefore strange to see that the cumulative points do not have a high correlation with performance. In figure Figure 4.15b it shows there is an even higher correlation between cumulative hours and cumulative points. This shows that these work items often get too little story points. Furthermore Figure 4.14 shows that there is quite a strong correlation between cumulative hours and the amount of children a work item has. This also makes sense because the more tasks a user story has, the longer it will take. This also goes for active duration and children. Then there is a rather strong correlation between active hours and cumulative hours. This also makes sense because, the longer an item is active, the more likely it is that hours are spent on that work item. Another interesting insight is that there is a rather high correlation with rework and active duration. This also really makes sense since rework means that a work item has. Variables like comment count do barely have any correlation with any other variable. It would be expected that this variable would have more correlation with other variables. Another variable that does not have a lot of correlation with other variables is priority. This is an even stranger one because the priority would likely mean that it has to be done quicker than others and thus have less active duration.



(a) Correlation matrix user stories

(b) Correlation matrix user bugs

Figure 4.15: Separated correlation matrix

4.6.1 Conclusion

The data gathering is an essential part of the project and, if done correctly, sets a good basis for the rest of the project. What is surprising is that not all user stories have story points. Before gathering the data, it was deemed that every user story would have a certain amount of story points in order to indicate how long it would approximately take. Whether the story points are useful will have to be seen in a later stage of the project.

When merging the hours to the work items, it became apparent that almost all hours were matched with the work item type: 'Task'. This makes sense because this is the actual work item that developers put work into. The other hours have mainly been matched with bugs. All the hours that have been matched with tasks are have been added up together to see how many total hours are spend on the parent. The parent usually being the user story. This is robust because when there is a mismatch (which is almost unavoidable on this scale) other tasks might not have a miss match or will be miss matched with tasks of the same user story thus making the total hours of the user story less vulnerable to miss matches.

The data visualisation part gave more insight on how variables were distributed and how variables related to one and other. The data visualisation also gave perspective on the differences of bugs and user stories. Bugs do not take as much time as user stories (usually) but seem harder to predict.

This section partly answers sub research question one: 'How is the current development process performing? where is the most need for improvement? What can be seen from the results in this chapter is that, quite a lot of work items have rework. This has a high correlation with active duration and with performance. This means that work items with more rework often take longer. Rework is therefore an aspect where there is room for improvement. Another aspect is the amount of tasks a work item has. When a work item has a lot of children, it has a relatively high correlation with cumulative hours, duration and rework. Meaning that it is probably wise to separate large work items into smaller ones in order to get better performance.

Chapter 5

Process mining

In this section more insight into the process performance will be given. This relates to the first research question: 'How is the current development process performing? where is the most need for improvement?

In order to gather more insight into the process performance, process mining has been applied. With the techniques belonging to process mining, the target was to get a better idea of which specific processes are performing well and which are not.

In [Erdem et al., 2018] the opportunities of process mining in agile software development are discussed. Most of what is discussed in this paper will be applied in this research as well. In addition to what is stated in [Erdem et al., 2018], the trace fitness will also be used in other analysis to determine what causes poor fitness. The benefit of this approach is that the process aspect will be included into the other aspects of this research.

Whenever a work item changes, a record is saved to the log data. As described in [Erdem et al., 2018], only state changes will be seen as events.

5.1 Data preparation

For this specific analysis, the data has to be prepared. At first, for the process discovery, all log data is removed where the state does not change. The reason for this is that an employee can change code which makes a change in the log data. This change does not mean that anything meaningful has happened for process discovery and returns a much clearer image.

The second thing is that only user stories and bugs are going to be assessed. The reason for this is that: lower level work items (Tasks) often are not registered correctly. Of the user stories and bugs that are going to be assessed, only user stories that have been closed are going to be assessed. It is not desired to look at unfinished processes. This will give a biased view of how many processes are performing poorly.

5.2 Process discovery

From the log data gathered in chapter 4, the process can be visually represented. Figure 5.1 provides a process map of all possible traces. In the process map of Figure 5.1, 2439 total traces are shown with 247 variants. The darker the squares are, the more visited the states are. This means that the state 'Resolved' is one of the most visited states together with 'New','Active' and 'Closed'. It can be seen that the following trace is most likely one of the most occurring traces. This is most likely a process that is performing well:

$$'New' \Rightarrow' Active' \Rightarrow' Resolved' \Rightarrow' Closed'$$

What also can be seen is that relatively many traces go from 'new' to 'closed' or from 'Active' to 'Closed'. These are processes that are likely to not perform well or at least not fit in a single sprint. When the work item does not fit in a sprint, it might be set on closed for a while until it can be resolved in another sprint. At first, one might think that this has to do with the fact that there are too many work items in a sprint. But whilst talking to stakeholders, it became apparent that the customer does not know what he/she wants all the time. It can be the case that a customer suddenly decides that another work item is prioritised over another or is not necessary anymore at all.

In Figure 5.2, two process maps have been made for Bugs and User stories separately. The first difference is that, in Figure 5.2a, the activities 'Done' and 'Committed' do not occur. While in Figure 5.2b, they do occur. These activities seem only to occur for bugs. Another difference is that; User stories generally have more rework. Meaning that User stories are more likely to go from 'closed' back to 'active' for example. Perhaps this is because bugs are more urgent to get fixed and are less likely to be moved to a future sprint. Another reason could be that Bugs are easier to address and thus are more likely to have less rework.

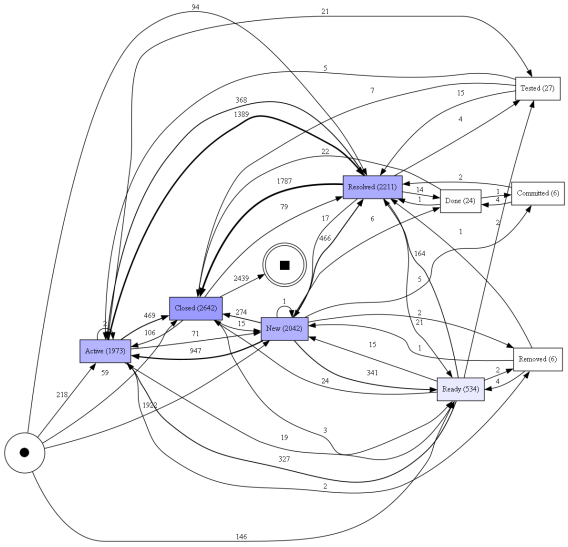
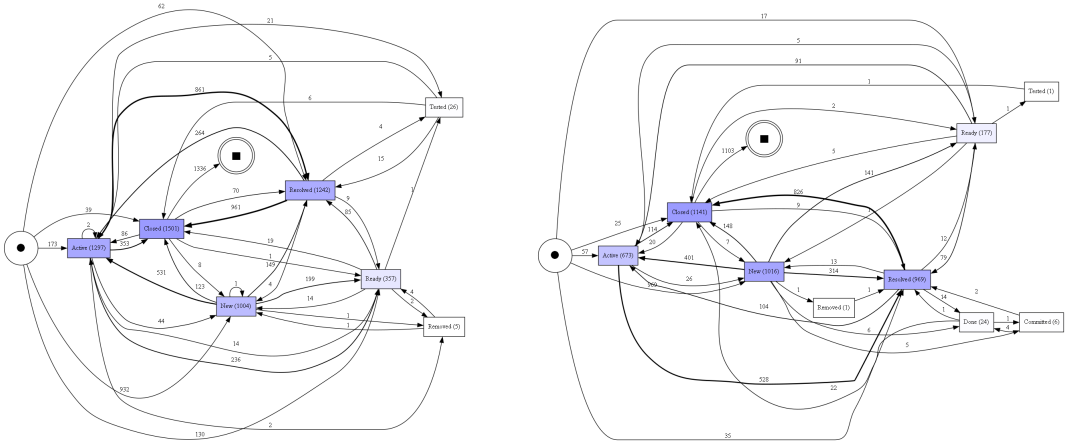


Figure 5.1: Process discovery map



(a) User stories

(b) Bugs

Figure 5.2: Separated traces

5.3 Performance

Apart from the process flow, the duration of traces might also be an insightful measurement of performance. The average trace duration is as long as 10 days. The reason this number is thins large is because some traces are moved to sprints that take place weeks later. The median time of a trace, which is 4,1 days, gives a more realistic time span. Still this is a lot longer than expected. The reason why this number is still larger than the sprint duration, is because the sprint duration only includes the time between 'active' and 'resolved'. The work item is created long before the sprint starts. Looking at the total trace length does not completely mean that the trace is performing poorly.

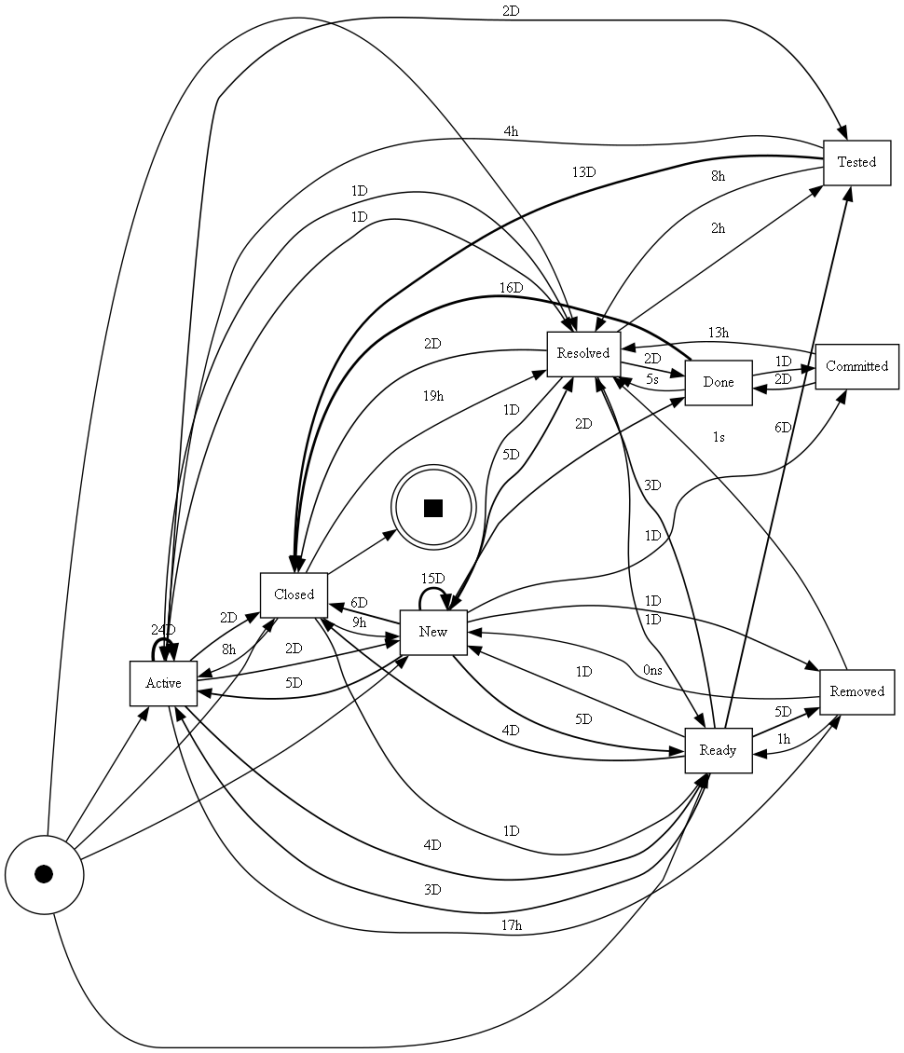
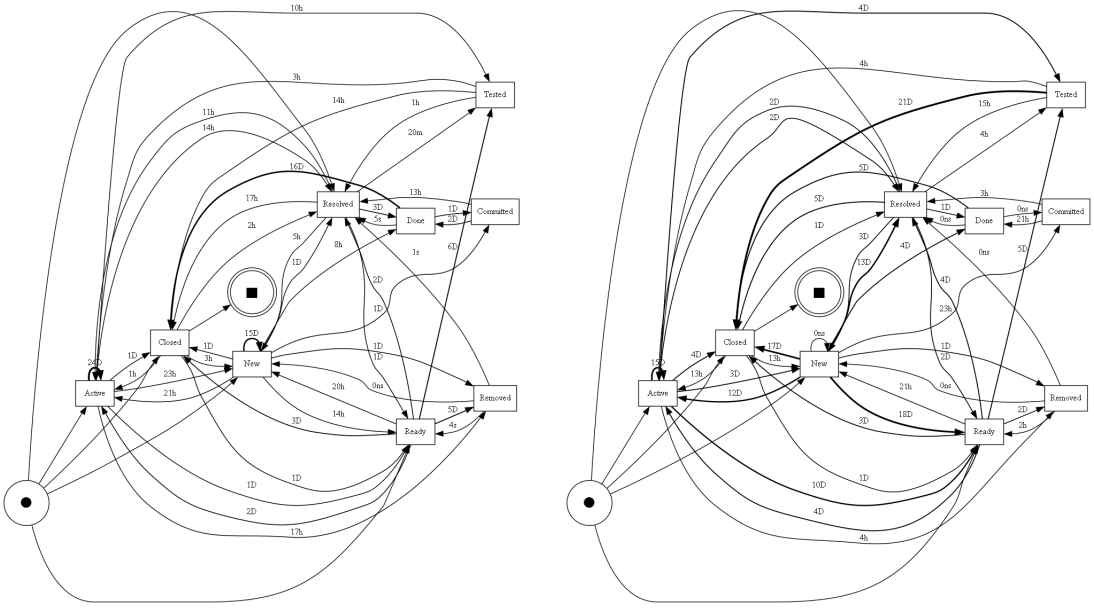


Figure 5.3: Duration map

In Figure 5.3, an additional perspective into the process is visualised. This perspective is the average time until a transition of activities takes place. For instance, on average, a work item is 18 days new before it becomes active. This makes sense since planners typically try to plan more than one sprint duration ahead, which would be at least 14 days. The goal at this point, according to stakeholders, is to plan further ahead than the current 18 days. In Figure 5.4, additional information is given on the duration. In Figure 5.4a and Figure 5.4b, the median duration and standard deviation are given respectively.

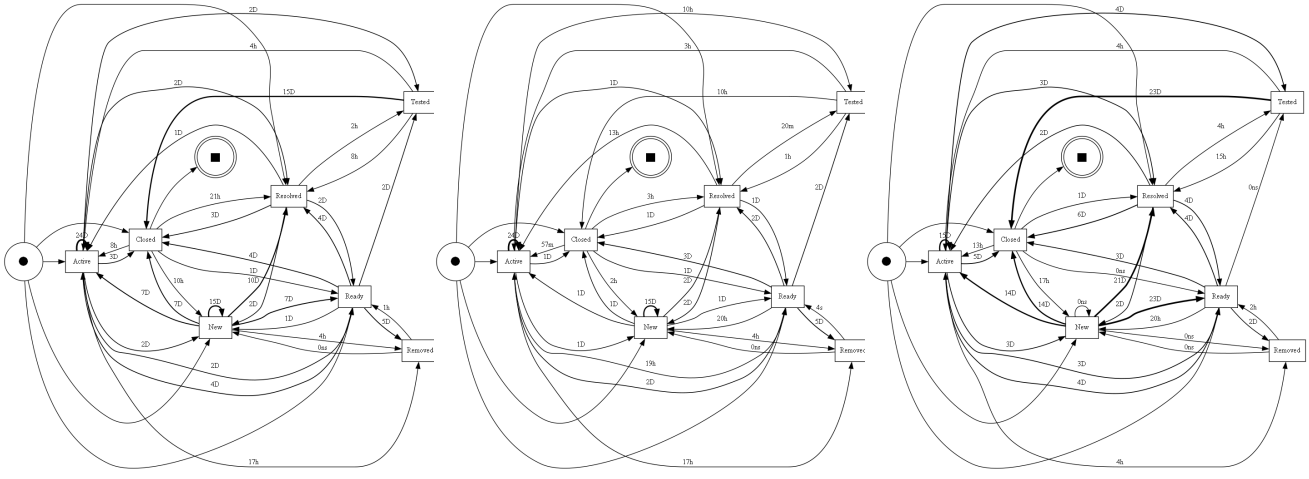
To resolve an active item, takes 1 day on average wit a standard deviation of 2 days. This means that there is a lot of variation on how long it takes for a work item to be resolved. The median time to resolve an active item is only 14 hours. So most work items are resolved within a day but a relatively small number of work items takes really long to resolve. These work items are most likely poor performers and should be looked into.



(a) Median (b) Standard deviation

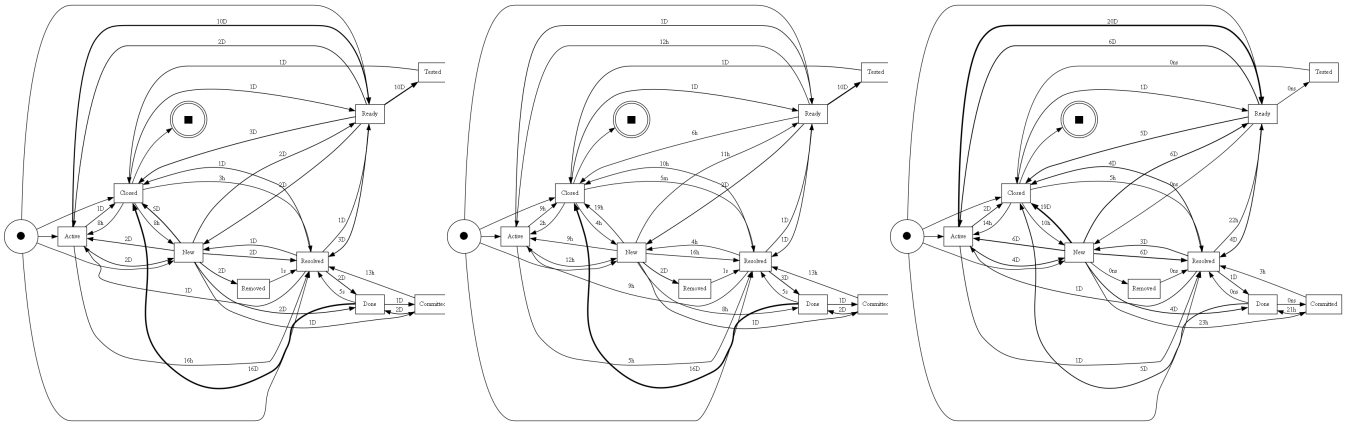
Figure 5.4: Additional information trace duration

In the following part, the User stories and Bugs are viewed separately in order to see differences between the two in performance:



(a) Mean duration (b) Median duration (c) Standard deviation

Figure 5.5: User story performance



(a) Mean duration (b) Median duration (c) Standard deviation

Figure 5.6: Bug performance

In chapter 4 the conclusion was made that the active duration of bugs had more standard deviation than User stories. This can still be seen in Figure 5.6 but clearly shows a different perspective. Figure 5.6c shows that the higher standard deviation mainly comes from traces that go from 'Active' to 'Ready'. In Figure 5.2b it can be seen that cases in which 'Active' goes to 'Ready' barely occurs. This thus gives a wrong perception of what is actually happening. The standard deviation from a User story going from 'Active' to 'Resolved is 3 days while a bug's standard deviation is only 1 day. This shows that, in most cases, bugs even have a smaller deviation than user stories.

After assessing the traces, it is necessary to determine which traces perform well and which do not. To assess what traces are good and which ones are bad, the log is being filtered on aspects which make the traces perform worse. Now it is important to take into consideration the fact that the trace does not only perform well whenever a traces takes a short amount of time. It is important that, during the trace, the actual problem is resolved. For instance, if the log is filtered on shortest duration, traces with only 'closed' will come by. This is most likely a trace that has not been addressed properly and thus does not have good performance even though is probably has a very short duration. To make a sensible filter of the good performing traces, the following steps are taken:

- **Filtering on starting state** makes sure that every trace starts with the desired state. In this case, the desired start activity is 'new'. The reason it is important to make sure all good traces start with 'new', is because this is the state that traces are supposed to start in when they are made. When a trace does not start on new, it might be because data is missing and therefore the wrong duration is measured.

- **Filtering on ending state** makes sure that every trace ends in a certain state. In this case, the ending state is 'closed'. The reason it is important that these filtered traces end in closed, is because the work item might not be finished when it does not end in closed. This would also mean that a wrong duration is measured for these work items.

- **Filtering on rework** makes sure that none of the states are visited twice. This is important because rework has a negative correlation with active duration according to Figure 4.14.

- **Filtering on visited state** like 'Removed' which means that the work item has been removed at some point and are not necessary anymore. Therefore traces which have visited 'Removed' are not deemed to be good traces. On the other hand, the states 'Resolved' and 'Active' are necessary in order to resolve the work item. For this reason, the traces that have visited these states are going to be good traces.

- **Filtering on performance** makes sure that only traces with a certain duration are selected. In this case, the total active duration is taken into account. The reason the total duration is not taken into account is because; it matters less how long it takes for a work item to move from new to active for example.

When all filters have been applied to the log data, the good performing traces are filtered out. These good traces will have addressed the actual work item while also having a relatively low duration. In Table 5.1, the different performance measures are given for each type of trace.

Table 5.1: Trace performance

	Average	**Median**
All traces	10 days	4,1 days
Good traces	7.3 days	2.1 days
Bad traces	17.1 days	10.35 days

As can be seen in Table 5.1, the average and median duration of the good traces is significantly shorter than the average and median duration of all traces. The median duration is almost twice as high as the good traces. In order to visualise the good traces in Figure 5.7 there is a process map of what the good traces are supposed to look like.

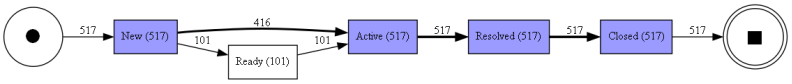


Figure 5.7: Good traces flow

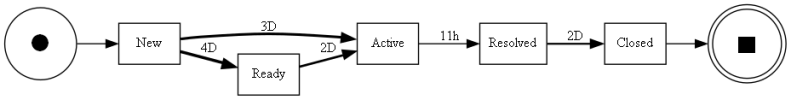


Figure 5.8: Good traces duration

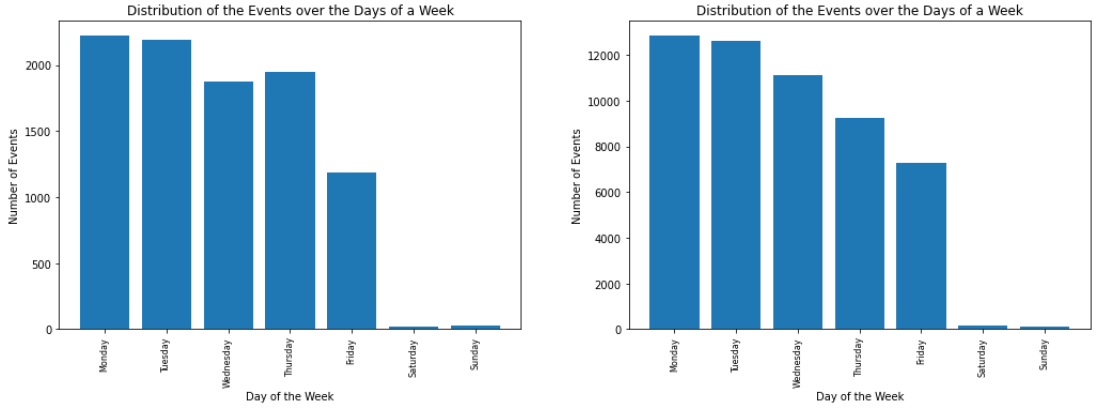
Another approach could be: filtering the entire log on trace duration. All traces filtered on a duration between 1 and 10 days have an average duration of 6.2 days which is significantly less than average. But when the entire log is filtered on these variants, the average time goes up to 9.1 days. This average is not better than the average of the good traces. The same goes for the median, which is 4.2.

5.4 Conformation

In order to determine whether or not a certain trace is fit, the fitness is assessed. This is done with the process conformance. This means that the fitness is the percentage of which the trace conforms with Figure 5.7. The reason this is done, is because this is one way of determining whether or not the process is performing in a certain way. Hopefully, in the following chapters it is possible to connect the performance with other variables. This will give a better insight in what causes bad performing traces.

5.5 Workdays

Something that struck interest during investigating the log data is the distribution of events among the workdays. In Figure 5.9 the distribution of events is given. The difference between the two graphs is the following: Figure 5.9a shows the events where the state is changed. This subset is also the subset on which all other analysis have been done in this chapter. Figure 6.2b shows all events, even when the state has not changed. This subset of the log data has not been shown so far. The reason why this graph is showed, is because most sprints are ended on Monday or Tuesday. Therefore, it could make sense that most state changes take place on these days because these are the days that the task is due. To gather more insights on the overall events, both of the subsets are shown.



(a) State change

(b) All events

Figure 5.9: Event distribution

One noticeable aspect is that, in both graphs, Monday and Tuesday are the busiest days. These are the days that probably do not fit any more work in them. Wednesday is the most average day in both graphs. The days where more work can be done are Thursday and Friday. On Thursday, relatively many work item states are changed while there are less overall events. The day that has the least events and state changes overall is Friday. This might also have to do that not everyone works on Friday. Although Friday is the day that some people do not work, this can not account for half the amount of events compared to Monday and Tuesday.

5.6 Conclusion

In this section, the individual cases are assessed with process mining. During this analysis, the first research question is addressed. After the analysis was completed, the following conclusions were be made:

- The most occurring path is:

  $$'New' \Rightarrow' Active' \Rightarrow' Resolved' \Rightarrow' Closed'$$

  Other paths can be shorter but might be incomplete, these incomplete paths can have a shorter duration. It is important to see performance as a path that resolves the issue whilst not taking too long. When a work item goes from 'new' to 'closed' it is highly likely that the work item has not been resolved and the actual performance is low even though the duration is short.

- Some differences between User stories and Bugs is that bugs never visit states 'Done' and 'Committed'. Another difference is that User stories are more likely to go from 'closed' to 'active' which means that the work item was not finished in the first instance. Of the 1141 times that bugs visit 'Closed' 20 times (1.8 percent) it has gone back to active. While from the 1501 times of User stories that have been closed 86 cases go back to active (5.7 percent). This means that User stories are three times more likely to have this type of rework in comparison with bugs. Going from 'Active' to 'Resolved' is also worse for User stories than for bugs. 78 percent of bugs go from 'Active' to 'Resolved' compared to 66.3 percent for User stories. In an earlier analysis, it seemed like bugs had more standard deviation in duration. While this is true, high standard deviation only happens in certain traced that do not occur very often.

- The best process flow of this process is given in Figure 5.7 and Figure 5.8. This flow is determined by several aspects. In general, the good process flow goes through all stages without rework. With process conformance, it is possible to see which traces align with best process flow. This gives a measurement to assess how well a certain work item is performing. Hopefully a relationship can be derived between other variables and process performance.

- The distribution of the events per day is interesting. The most work seems to be done on Monday and Tuesday. Wednesday is average, Thursday is a little less than average and Friday is the least productive day.

In relation to the the first sub research question; 'How is the current development process performing? where is the most need for improvement? The following can be said. Currently, a lot of the work items do not go through the ideal path. The ideal path has significantly less duration than other paths. This makes that the current process is not performing as good as it could be. In order to quantify this, a fitness is created based on process alignment. This shows how each individual trace is performing with respect to the process and thus also which traces could perform better. The most room for improvement was thought to be bugs, but when looking at this analysis, it became apparent that this was not the case. The higher standard deviation in active duration for bugs was only occurring with a small amount of cases. Furthermore on Fridays and on Thursdays, there is room for improvement these are the days that are least productive.

Chapter 6

Natural language processing

Doing an NLP analysis itself does not answer any of the sub research questions. The reason this is done anyway is because of research question 2. This research question is: What causes poor sprint planning performance? What role does work item meaning play in this?. Therefore the meaning of each work item is going to be determined. By doing this analysis, it will be possible to determine whether or not certain work item meanings cause poor performance. The idea is that other analysis methods can interpret the meaning of the title of a user story or bug. When a certain meaning is discovered, hopefully a link can be made on which user stories or bugs are performing poorly. An interesting result would be that, whenever a certain meaning occurs, the performance is poor. The method used to define what topic a certain phrase belongs to is also known as topic modeling which is described in [Marinov, 2021]. The way this paper describes how raw text data is transformed into clustered sentences by meaning. To transform this data, the pipeline in Figure 6.1 has been used. The reason each of these steps is optimal is going to be discussed separately.

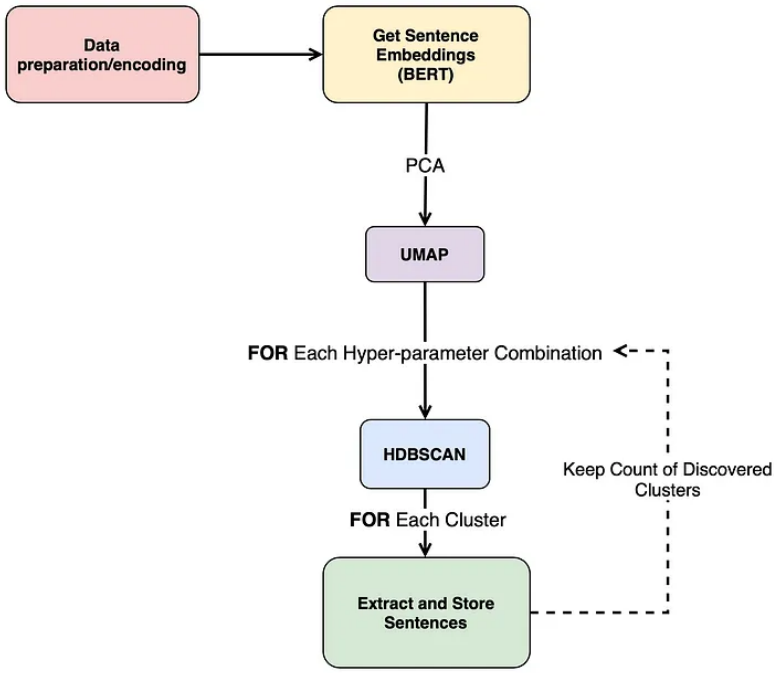


Figure 6.1: Pipeline

According to [Marinov, 2021], there are certain advantages and disadvantages of this approach of topic modeling. The advantages that are named are:

- No need for preprocessing: such as stop-word or specific POS-tag removal
- No need for hyper-parameter tuning or optimization
- Handles sentences of varying degrees
- Output is easier to interpret, not a collection of single word
- Able to handle small datasets

Unfortunately there are also disadvantages to this method, which are:

- Rather slow compared to other methods
- Large overlaps between some of the discovered clusters
- Lacks useful additional functionalities

6.1 Text preparation

Some work items have a dutch title and some have got an English title. This problem is resolved by translating every work item title into English. The reason for choosing the English language is that pre-trained language models are often trained on English sentences. There are some pre-trained dutch models but these are often less developed. The translation model used is google translator. The reason google translator is chosen is because it is a very popular and verified translator.

6.2 Text embedding

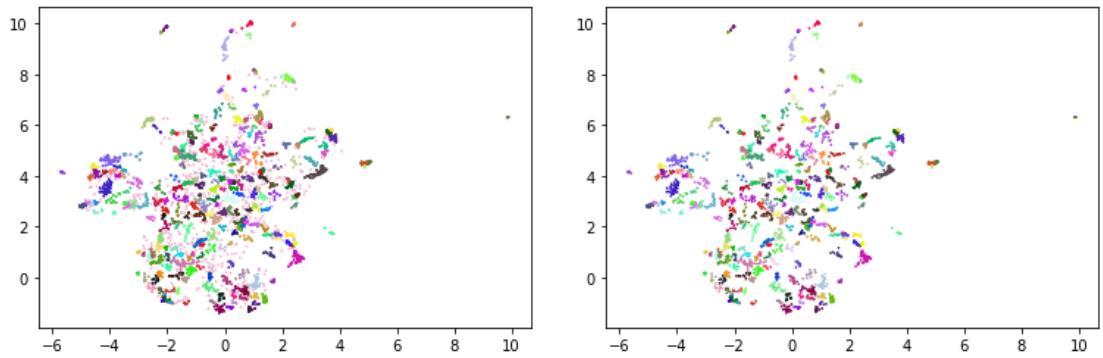
This brings us to the next step in this process; creating a vector representation for every work item. The bag-of-words approach is probably not suitable for this problem. When work items are translated from dutch to English, the translation might be slightly different than what people in the company normally use in the work items. This causes vectors not to align even though they have similar meaning. With pre-trained models this problem does not occur. One approach could be, embedding each word separately and take the average vector of all the words in the sentence. This approach does not take into take the complete sentence meaning into account. For this reason, sentence embedding is used. This approach takes the entire sentence into account and transforms it into a vector. The model used is; 'Bert sentence encoder'. This encoder creates a vector with a length of 512 values for every work item title. The reason Bert was chosen over other over other language models is because it is one of the most used and verified models available [Xie, 2022].

6.3 Clustering

In order to discover which meanings have a certain impact, the meaning of every title has to be determined. In order to do this, the vectors of every title are going to be clustered. This means that an algorithm is going to recognise whether vectors have similar length and direction and thus meaning.

The first problem that arises is the fact that most clustering algorithms do not work well with sparsely distributed data. Vectors with 512 dimensions that represent text are known to be very sparsely distributed. To compensate for this problem, the data has to be reduced. This means that the 512 dimensions are going to be reduced to 2 dimensions. The method used in this instance is Umap, which means: Uniform Manifold Approximation and Projection. This method has quickly grown in popularity as a dimensionality reduction technique. UMAP is much faster and more scalable than most other techniques [PAI,]. Another advantage is that UMAP is good at preserving the global structure of the data. According to [Allaoui et al., 2020], UMAP is a very suitable choice for dimension reduction for clustering purposes.

The next step is to cluster the reduced data. For this specific application, it is preferable to use an algorithm that does not require a specific number of clusters upfront. The second preference is that the algorithm can perform well when clustering noisy data. Density-based algorithms can be a good option as they do not require the number of clusters up front and do not take the cluster shape into account. A popular approach is hdbscan which means; Hierarchical Density-Based Spatial Clustering of Applications with Noise. This is a density based cluster which is robust to variable-density clusters [Marinov, 2021]. The only parameter required for the cluster algorithm is the minimum size of a cluster. There are two main parameters that can be set, min cluster size and min samples. The goal is to set these parameters such that there are not too many clusters and not too many outliers. Because outliers do not get appointed to a cluster. When no min samples are chosen, the min samples are set to be equal to the min cluster size. Making the min samples smaller than the min cluster size, it is more likely that outliers are going to be appointed to certain clusters. The amount of clusters with a minimum cluster size of five is 371. The amount of outliers is 16 percent which is deemed to be reasonable. The result of the clustering is shown in Figure 6.2a and in Figure 6.2.



(a) Clusters with outliers

(b) All events

Figure 6.2: Topic clusters

When looking at Figure 6.2a it is clearly visible that the clustering algorithm is density based. The points with higher density indeed get clustered together and those with less density are considered outliers. Just for the general idea, in section B.1 there are 3 clusters given. This shows that, indeed every cluster has the same topic. The first cluster shown in section B.1 has the topic of software updates. The second cluster shown has the topic renovate and the last cluster has the topic"technical debt. This shows that the clustering algorithm has done what it was supposed to do.

6.4 Performance

Now all the necessary steps have been taken in order to cluster all work items by topic, the next step is the performance. The most common way of determining performance for clustering algorithms is the silhouette score. According to [Dorfer, 2022] the silhouette score does not perform well on density based clustering algorithms which is used to cluster in this particular example. For density based clustering algorithms like hdbscan, the paper suggest to use DBCV (Density based clustering validation) Which is much more suitable for for HDB-scan performance measuring. This performance measurement will return a score between -1 and 1. -1 means that the clustering algorithm is performing incredibly poor and 1 means that it is performing incredibly well. The performance measurement for this specific clustering outcome is: 0.73. This is definitely on the good side of the spectrum and therefore acceptable.

6.5 Conclusion

The proposed method described in [Marinov, 2021] has shown good results for this particular use case. At first, the text preparation for the text embedding was very intuitive since no additional text cleaning or pre-processing had to be done except for language translation. When the embeddings were made, the reduction and clustering algorithm made very sensible clusters which indeed cluster work item titles based on topic and also had a reasonably good performance. The result of this effort is that every work item has an extra variable which determines the topic of the title. In the next chapter the analysis on what topics are going to perform better than others is going to take place.

Chapter 7

Causal rule mining

In this section, the second research question is adressed: What causes poor sprint planning performance? What role does work item meaning play in this? In order to do this, two association rule mining algorithms are used.

At first, the apriori algorithm is used. This is a data mining tool which returns frequently occurring patterns. These patterns are measured in support and confidence. Support meaning, the percentage of the pattern occurring in the entire data set. While confidence is the percentage of occurrences in the subset of where the antecedent occurs. Later, the Causal association rule mining algorithm is used to see if there are causal relations withing the data.

7.1 Data preparation

For this specific analysis, the data has to be transformed into a different format. The data has to be transformed into binary data such that the algorithm can use it. To do this, the categorical data is one-hot encoded and the other data is transformed by binning. The variables; performance (hours divided by story points), fitness(determined by process mining) are going to be separated in bins. The amount of bins per variable is three bins, namely: low, medium and high. So for the performance variable, three variables are made: performance_low , performance_medium, performance_high and the same goes for fitness. These bins are made to be of equal size.

7.2 Apriori

When running the algorithm to search for rules with at least a support of 0.15 and a minimum lift of 1, 184 rules were found. In this section, a couple of association rules are going to be highlighted. This will give a better insight in how variables relate to each other. When running the algorithm, one rule in particular seems interesting.

$$(active\_short, points\_none) \rightarrow (rework\_infrequent, performance\_none)$$

With a support of 16 percent and a confidence of 100 percent. The reason this rule exists is because some work items have never been active and have never had any points appointed to them. These work items have only been created and never used. For this reason, these work items will not be analysed and will be filtered out of the data set.

$$1.(active\_long, points\_many) \rightarrow (type\_User\_Story)$$

$$2.(active\_short) \rightarrow (fitness\_good)$$

$$3.(active\_short) \rightarrow (type\_bug)$$

$$4.(rework\_frequent) \rightarrow (children\_large)$$

$$5.(children\_small) \rightarrow (fitness\_good)$$

$$6.(active\_long) \rightarrow (fitness\_medium)$$

$$7.(rework\_frequent) \rightarrow (active\_long)$$

Above, a few rules have been picked to evaluate. For instance, the first rule shows that work items that take long and have many points are often user stories. This is interesting to see because the confidence is almost 100 percent as can be seen in Table 7.1. The second rule shows that, when a work item is active for a short amount of time, it indeed means that the fitness is good most of the time. This is interesting to see and proves that fitness is related to the duration. The third rule states that most bugs are active for a short amount of time which is also what was found in chapter 4. The fourth rule states that, when a work item has to be reworked frequently, it often means that it has a lot of tasks. Rule five states that if there are little tasks, the fitness is often good. Then rule 6 states that, if a work item takes long, the fitness is medium. The last rule is something that is also seen in chapter 4 when a work item has frequent rework, the duration is often longer.

Table 7.1: Association metrics

	antecedent support	**consequent support**	**support**	**confidence**	**lift**	**leverage**	**conviction**	**zhangs metric**
1	0.156	0.626	0.153	0.976	1.558	0.054	15.770	0.425
2	0.333	0.504	0.202	0.608	1.205	0.034	1.264	0.255
3	0.333	0.373	0.193	0.579	1.549	0.068	1.487	0.532
4	0.333	0.446	0.220	0.659	1.476	0.071	1.625	0.48
5	0.255	0.504	0.167	0.6588	1.306	0.039	1.452	0.314
6	0.333	0.394	0.163	0.489	1.242	0.031	1.187	0.292
7	0.333	0.333	0.150	0.452	1.355	0.039	1.216	0.393

All found rules can be seen in Appendix C.

7.2.1 NLP relations

The association rules mined so far never show any relations with NLP variables. The reason for this is that there are more than 200 meanings and thus NLP variables are very unlikely to pass the 0.2 support mark. It would be interesting to see which meanings have a high confidence with other variables. Therefore, in the next part of this section, the 0.2 support mark is lowered and filtered for NLP variables. Three meanings are highlighted to show that there are indeed some relations:

NLP variable 21 has a high confidence with children_many and also a high confidence with infrequent rework. This is counter intuitive because usually work items with a larger amount of children have more rework done. NLP variable 21 represents a topic that has everything to do with structure like: improve structure or restructuring.

NLP variable 67 have a high confidence with good fitness and a short active duration. These meanings are impact analysis related. An impact analysis is an analysis which determines the impact of a certain change.

NLP variable 177 has a high confidence with many children, and also good fitness. This NLP variable has to do a lot with saving, data and logging,

All of these nlp variable related rules have little support but all of them have a confidence higher than 75 percent. This means that these meanings do not occur very often but when they occur, they are very likely to have these relations.

7.3 Causal association rules

The next step into gathering even better insights into the data with causal analysis. For this, the same data prepossessing is used as in section 7.1. The method used in this section is causal association rule mining. This method can mine relations which cause certain target variables. The addition of this analysis in comparison to section 7.2 is that this analysis determines causal relationships instead of correlations.

7.3.1 Duration

The first analysis is done on active duration. This simply means that the algorithm will calculate what causes long active duration:

Table 7.2: Long active duration

Rule	Odds ratio
['childs_large'] —>Z	2.1219512195121952
['Resolved_by_Person1'] —>Z	1.540983606557377
['rework_None'] —>Z	3.5
['Resolved_by_Person2'] —>Z	8.0
['typeUser Story'] —>Z	2.8333333333333335
['points_many', 'rework_Medium'] —>Z	2.3636363636363638
['Performance_good', 'rework_Medium'] —>Z	3.6666666666666665
['fitnessmedium', 'priority_2.0', 'rework_Medium'] —>Z	3.25
['priority_2.0', 'fitnessbad', 'rework_Medium'] —>Z	2.1818181818181817
['fitnessmedium', 'hours_none', 'rework_Medium'] —>Z	3.0
['fitnessmedium', 'Performance_none', 'rework_Medium'] —>Z	6.5
['fitnessmedium', 'hours_high', 'Performance_bad'] —>Z	2.6666666666666665

The results are somewhat similar to the results of section 7.2. For example, many children causes the user story to have a longer active duration. Another result that has been seen before is, the fact that user stories often take longer than bugs. Even though these results have been found in an earlier section, it is still valuable because this analysis validates the results.

This analysis also shows results that have not been seen before like certain people resolving a work item causing long active duration. This is interesting because this would mean that these people generally performs less well than others. Then there is: 'rework none' —> Z which seems counter intuitive. Furthermore, medium rework and fitness medium both occur often in combination with other variables to cause long active duration.

The next analysis is about what causes short active duration. The results are:

Table 7.3: Short active duration causes

Rule	Odds ratio
['typeBug'] —>Z	1.76
['childs_small'] —>Z	1.829268292682927
['points_few', 'fitnessgood'] —>Z	1.72
['hours_low', 'points_few'] —>Z	2.0
['hours_low', 'Performance_none'] —>Z	3.625
['hours_low', 'childs_medium'] —>Z	2.0
['points_medium', 'Resolved_by_Person3', 'hours_medium'] —>Z	10.0
['Performance_none', 'priority_2.0', 'Resolved_by_Person4'] —>Z	5.0

These rules also validate the results from section 7.2 like bugs that generally take less time and a small amount of children also takes less time. Hours low also occurs multiple times in combination with other variables.

7.3.2 Rework

The next analysis is going to be on rework. These rules will show what causes rework.

Table 7.4: Rework causes

Rule	Odds ratio
['fitnessbad'] —>Z	3.6363636363636362
['act_dur_long'] —>Z	2.2
['points_medium', 'hours_none'] —>Z	2.2222222222222223
['priority_2.0', 'Resolved_by_Person3'] —>Z	8.0
['typeUser Story', 'priority_2.0', 'childs_medium'] —>Z	2.8333333333333335
['points_many', 'typeUser Story', 'nlp_-1'] —>Z	2.8
['points_medium', 'priority_2.0', 'childs_large'] —>Z	2.5714285714285716
['Performance_bad', 'childs_large', 'Resolved_by_Person3'] —>Z	5.5
['childs_medium', 'hours_none', 'typeUser Story', 'Resolved_by_Person1'] —>Z	8.0

As can be seen in rule 1 and 2: bad fitness and a long active duration will cause rework. These are not surprising but still interesting to see. Furthermore, priority 2 in combination with other variables causes rework. This could make sense because priority 2 means lower priority and therefore might be finished in a later sprint instead of others. The sixth rule states that user stories with a lot of points and -1 for the NLP variable also cause rework. When the NLP variable is -1 it is an outlier and therefore has a meaning that does not occur very often. It could make sense that these types of work items might be harder to solve within the given time and therefore need rework.

The next set of rules is mined based on the target: no rework. With other words, these rules describe what variable causes no rework.

Table 7.5: No rework causes

Rule	Odds ratio
['fitnessbad'] —>Z	13.0
['act_dur_long', 'childs_large'] —>Z	6.0

The first makes a lot of sense since one of the variables on which fitness is based on is rework. The second rule on the other hand is more surprising. It could be possible that the planner notices

that the work item is rather large and therefore gives the developer more time such there is no need for rework.

7.3.3 Performance

In this section the focus is going to be on performance. Remember that performance is hours divided by story points. This means that a high performance measurement means that the performance is bad. The reason this means that the performance is bad is because there are a lot of hours spent on few story points. Essentially saying that the initial estimate of the work item duration was wrong and that there were more hours spend than planned. The rules found that will cause bad performance are given below:

Table 7.6: Causes of bad performance

Rule	Odds ratio
['points_few'] —>Z	12.75
['hours_high'] —>Z	40.0
['typeBug', 'hours_medium'] —>Z	2.4166666666666665
['points_medium', 'act_dur_medium'] —>Z	2.625
['hours_medium', 'childs_small'] —>Z	2.625
['points_medium', 'childs_large'] —>Z	2.4444444444444446
['points_medium', 'fitnessbad'] —>Z	4.333333333333333
['Resolved_by_Person1', 'hours_medium'] —>Z	1.7894736842105263
['fitnessgood', 'hours_medium'] —>Z	1.8235294117647058
['priority_2.0', 'act_dur_short', 'hours_medium'] —>Z	2.857142857142857
['fitnessmedium', 'priority_2.0', 'hours_medium'] —>Z	2.8

The first and the second rule are both part of how performance is defined and therefore it is not surprising that these have very high odds ratios. On the other hand, it is interesting to see that 'hour_high' causes a lot more bad performance than 'points_few'.

More interesting to see is that bugs with medium hours tend to have poor performance. These work items often have a low amount of story points and therefore it also makes sense that a medium amount of hours already makes the performance poor.

Medium story points and medium active duration apparently also causes bad performance. This is rather counter intuitive since one would expect that having medium active duration would also have a medium amount of hours spent and therefore medium performance. When looking at subsection 7.3.1 and subsection 7.3.5 it is clear that active duration and work hours do not necessarily cause one and other.

Table 7.7: Causes of good performance

Rule	Odds ratio
['hours_medium'] —>Z	2.08
['points_many'] —>Z	4.428571428571429
['hours_low'] —>Z	11.5
['typeUser Story', 'act_dur_medium'] —>Z	2.3333333333333335
['priority_2.0', 'typeBug', 'points_few'] —>Z	4.0
['typeBug', 'points_few', 'fitnessgood'] —>Z	3.6666666666666665

The first three rules are not surprising because these variables are used to define performance. Yet these variables do not cause as much good performance as similar variables cause bad performance in Table 7.6. Bugs with few points and either low priority or good fitness seem to have rather good performance. This while user stories with medium active duration have better performance.

7.3.4 Fitness

This analysis is going to be on fitness. It is interesting to see if there is anything that causes good fitness because fitness, in this case, is a combination of good performance while also having no rework and having resolved the work item.

Table 7.8: Causes of good fitness

Rule	Odds ratio
['Resolved_by_Person5'] —>Z	2.5
['Resolved_by_Person6'] —>Z	3.1818181818181817
['childs_small'] —>Z	1.3493975903614457
['typeBug'] —>Z	1.4
['Performance_medium', 'Resolved_by_Person3'] —>Z	5.0
['Resolved_by_Person3', 'childs_medium'] —>Z	4.333333333333333
['Resolved_by_Person3', 'hours_medium'] —>Z	4.5
['act_dur_medium', 'childs_medium'] —>Z	1.4583333333333333
['points_few', 'act_dur_short'] —>Z	2.0
['Performance_medium', 'points_medium'] —>Z	2.3333333333333335
['Resolved_by_Person1', 'priority_1.0'] —>Z	2.1538461538461537
['act_dur_short', 'Resolved_by_Person1'] —>Z	1.7435897435897436
['childs_medium', 'priority_1.0'] —>Z	3.0
['Resolved_by_Person1', 'Performance_bad', 'childs_medium'] —>Z	8.0
['points_few', 'Resolved_by_Person1', 'nlp_-1'] —>Z	8.0
['hours_none', 'act_dur_medium', 'priority_1.0'] —>Z	8.0
['Performance_medium', 'act_dur_short', 'hours_medium'] —>Z	3.25
['Performance_none', 'act_dur_short', 'priority_1.0'] —>Z	3.6666666666666665
['hours_medium', 'act_dur_short', 'childs_medium'] —>Z	2.2
['typeUser Story', 'act_dur_short', 'hours_medium'] —>Z	2.2222222222222223

The first two rules already tell us that there are certain people who cause a work item to have good fitness. This means that these people handle their work items better than others. The third and fourth rule state that a small amount of children cause good fitness and also bugs generally have better fitness. Furthermore there is are a lot of 'Resolved_by' variables, mostly in combination with other favourable variables which might imply that certain people are better at certain specific things.

Table 7.9: Causes of bad fitness

Rule	Odds ratio
['childs_large'] —>Z	1.6551724137931034
['Performance_bad'] —>Z	1.7894736842105263
['rework_Medium'] —>Z	2.966666666666667
['Resolved_by_Person7'] —>Z	2.4285714285714284
['rework_None'] —>Z	17.0

It makes sense that bad fitness is caused by a large amount of children, bad performance and rework. One person seems to be causing bad fitness as well. The most striking is that no rework seems to cause bad fitness. The reason for this is that some traces go from active to closed without having any work done. These work items have no rework but also poor fitness because these have not gone through all stages that define good fitness.

7.3.5 Work hours

The last analysis is going to be on work hours. These are the work hours employees put into Simplicate. This is interesting because the actual amount of hours directly correlates with the cost of a project.

Table 7.10: Causes of many work hours

Rule	Odds ratio
['Performance_bad'] —>Z	65.0
['Performance_medium'] —>Z	3.1333333333333333
['points_many'] —>Z	3.375
['act_dur_long', 'priority_1.0'] —>Z	5.5
['act_dur_long', 'typeUser Story'] —>Z	2.625
['fitnessmedium', 'typeUser Story', 'childs_large'] —>Z	3.0
['act_dur_long', 'priority_2.0', 'rework_Medium'] —>Z	5.0

As expected, performance is a major cause in higher work hours. A work item with a lot of points also causes more hours to be spent on a work item. Strangely, active duration only causes more work hours when a priority level is assigned or when it is a user story.

7.4 Conclusion

At first, it is clear that there are definitely a lot of relations to be found within the data causal and non-causal. The second sub research question: Sub RQ 2: What causes poor sprint planning performance? What role does work item meaning play in this? is answered in this section, and it is safe to say that there are almost no causal relations between work item meaning and performance. It would have been interesting to see which topics within the development process were performing better or worse. Fortunately there are some correlations found for these relations.

Even though there are barely any causal relations found with topics, topics do give better results. The reason for this is that the algorithm makes pairs in order to make causal rules. These pairs are constructed as follows: Given an association rule $p \rightarrow z$ and a set of controlled variables C, a pair of records match if one contains value p, the other does not, and both have the same value for C.

For example, assume that C = (A, B, D) is the controlled variable set for association rule $p \rightarrow z$, then records (P = 1, A = 1, B = 0, D = 1) and (P = 0, A = 1, B = 0, D = 1) form a matched pair [Li et al., 2013].

This makes sure that relations found within the data relate to the same context. When these confounding NLP variables are not taken into account, the algorithm makes claims about totally different topics in user stories which would make the claims less relevant.

Chapter 8

Causal decision making

In order to answer research question 3: How can the potential causal relations be used to improve future planning performance? Causal decision making is applied. According to [Fernández-Loría and Provost, 2022], causal decision making means using machine learning models to predict causal effects. The causal effect is estimated and the best intervention is applied. The desired outcome of this machine learning algorithm in this case is the performance of a sprint. It is desired that the input consists of user stories and bugs and the outcome the total sprint duration. This result can later be used by planners to make a better planning. For this machine learning problem, a sequential algorithm is proposed. The reason for the sequential machine learning is because, all the work items together add up to the total sprint duration. The structure of the machine learning algorithm is thus a many to one structure where multiple work items are linked to a single output. This output being the total sprint duration. In Figure 8.1 the many to one approach is visualised. The columns of the input data are the different work item attributes, while the rows are the work items themselves.

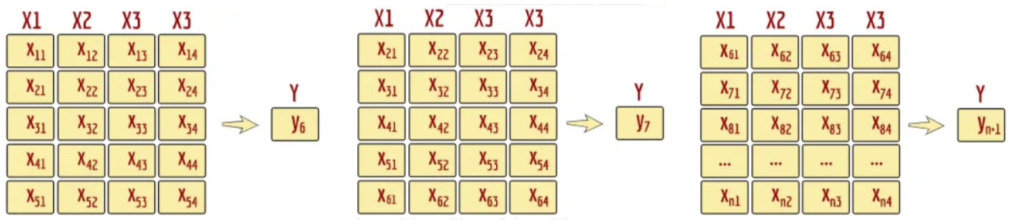


Figure 8.1: Many to one

The type of sequential layers used in the model is the long-short-term-memory. The reason for this is that these layers, as the name suggest, can recognise long and short term patterns. Normal layers are also added to the model and the last layer is a layer with one dense. This makes sure that one output the algorithm returns a single value for each sprint.

8.1 Data processing

At first, it is important to select the data which is going to be used in the algorithm. In chapter 4 and chapter 7 the relations between data points have been discussed. It is important to realise the algorithm should only predict on data which is not known by the predictor at the time the prediction is made. For these reasons the following data attributes are chosen:

- Work item type
- Resolved by
- Priority

- Amount of children
- Amount of story points
- NLP variables

In order to get the wanted result, the data has to be processed. The first thing that comes to mind is the following problem: Let's say that a sprint has 10 user-stories planned. It might be the case that some of these work items are not finished within the sprint or not even started in that sprint because there was not enough time. When an algorithm tries to learn these inputs and outputs, it is learning the wrong pattern. The algorithm is learning a duration for work items which have not taken place. For this reason, when a work-item has been active in a sprint for less than 10 percent of the total active duration, the work item is not included in the learning process.

For other cases where a work item has been active for more than 10 percent, a different solution is necessary. Let's give an example for a work item that has been active in one sprint for 40 percent and 60 in another. For the 40 percent sprint, 60 percent of amount of work hours appointed to that work item is added to the sprint duration. In the other sprint, it would be 40 percent of the appointed hours. The last step in this process is to divide the extra appointed hours among the amount of people who were working on the sprint. This gives a more realistic view of the extra sprint duration. The reason for this is, that if two people both have a work item that takes 4 hours longer than the sprint the sprint would not last 8 hours longer but only 4. Now the machine learning algorithm learns how long the sprint would have taken if all work items were finished within the sprint.

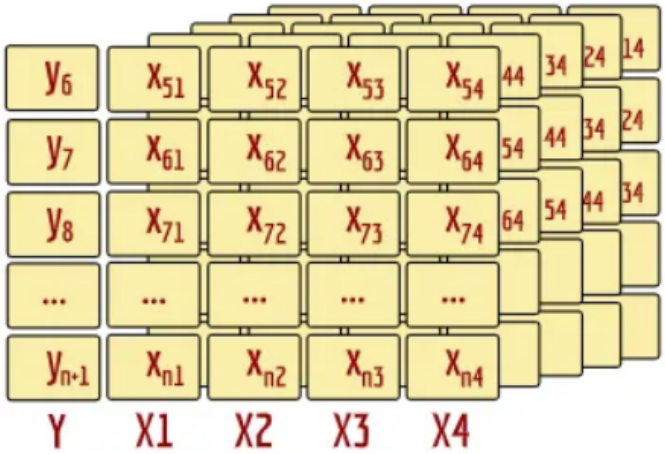


Figure 8.2: Input data format

The next data processing is the format of which the data has to be fed into the algorithm. The data format is visualised in Figure 8.2. This is a three dimensional data structure of a fixed size. This fixed size is a problem for the way sprints are formed. These sprints do not always include the same amount of work items. In order to address this problem, padding and masking is used. Padding is a technique to make sure every instance is of similar length. This is done by filling the sprint with null values up to the maximum amount of work items addressed to a sprint. The maximum amount of work items addressed to a sprint is 43. In order to make sure the algorithm does not take in these null values as actual values, masking is applied. Masking is a way to identify and ignore all padded values during processing of the data.

8.2 Data split

The data is going to be split into three different data sets. Namely training, validation and testing. The main reason for splitting the data three ways is to prevent over fitting.

The train set is the set to train the model and make the model learn the patterns. In each epoch, the same training data is put into the neural network architecture. The validation set is the set of data that is used to validate the model performance during training. While the model is trained on the training set, the model evaluates how it performs on the validation set after every epoch. The test set is a separate set of data which is not used during training. This test data provides an unbiased evaluation of the final model performance. In Figure 8.3, the process is visualised.

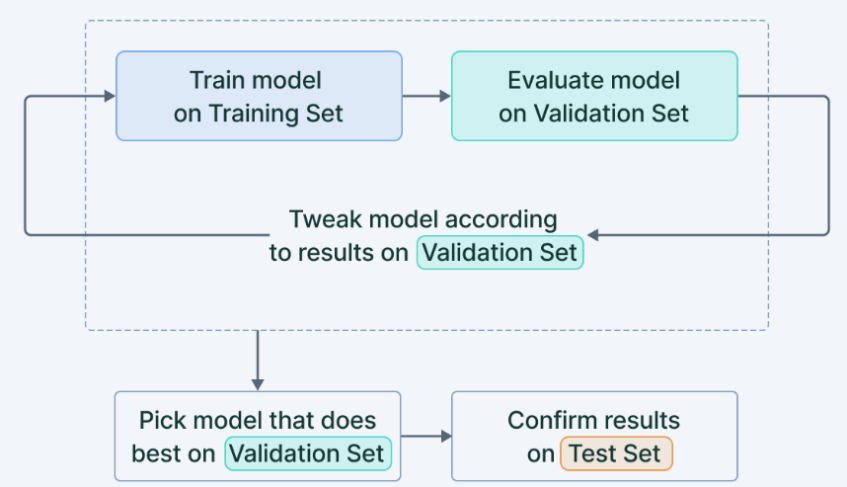


Figure 8.3: Training

The data is split in the following way:

- Training set: 80
- Validation set: 10
- Test set : 10

While splitting the data into different sets, the sprints have to be taken into consideration. The set can not be split exactly within these percentages because it could be that half a sprint falls into one set and the other half into the other. This would result in the problem of trying to predict the full sprint duration based on only half the data available on that sprint. For this reason, complete sprints are put into these split data sets.

8.3 Hyper parameter optimization

In order to get the best results, the hyper parameters are optimised. This is done with Bayesian optimization. The optimizable parameters are shown in Table 8.1:

As can be seen, the hyper parameter optimisation also determines the structure of the machine learning model. The amount of layers and the size of the layers are crucial for a good result.Between these layers, dropout layers are added. These dropout layers have a certain probability of blocking certain connections between layers. This results in less over fitting of the model. Furthermore, a patience of 20 is used while optimising. This means that, when the model does not improve for 20 epochs, the optimisation stops and the next optimisation attempt starts. This is done such that

Table 8.1: Hyper parameters

Parameter	**Value**
Amount of LSTM layers	1 to 4
LSTM layer size (for every layer)	16 to 1024 with steps of 32
Activation of LSTM layers (for every layer)	relu, softmax or sigmoid
Dropout percentage between LSTM layers (for every layer)	0 - 0.05 - 0.1 - 0.2 - 0.3 - 0.4
Amount of dense layers	0 to 6
Dense layer size (for every layer)	32 to 1024 with steps of 32
Activation of dense layers (for every layer)	relu, softmax or sigmoid
Dropout percentage between dense layers (for every layer)	0 - 0.05 - 0.1 - 0.2 - 0.3 - 0.4
Optimizer	Adam, SGD or RMS prop
Learning rate	0.1 - 0.01 - 0.001 - 0.0001 - 0.00001

the optimisation procedure takes less time and more attempts can be done in a shorter time span. The maximum amount of epochs during optimization is 150. The optimal parameters after 500 optimisation iterations can be seen in Table 8.2.

Table 8.2: Best hyper parameters

Hyperparameter	Best Value
units	32
activation_start	sigmoid
num_LSTM_layers	1
num_dense_layers	3
activation_end	relu
Optimizer	Adam
learning_rate	0.01
dense_LSTM0	64
activation_lstm0	relu
dropout_LSTM0	0.4
dense_0	384
activation_dense0	sigmoid
dropout_0	0.4

Hyperparameter	Best Value
dense_1	64
activation_dense1	sigmoid
dropout_1	0.3
dense_2	384
activation_dense2	relu
dropout_2	0
dense_3	384
activation_dense3	relu
dropout_3	0.1
dense_LSTM1	512
activation_lstm1	relu
dropout_LSTM1	0.2

This optimal machine learning algorithm has a validation mean squared error of 2.72 and for the test data a mean squared error of 4.32 days. The test data is a set of unseen data by the algorithm and therefore likely the performance that the algorithm will have on other unseen data. This performance translates to a mean absolute error of 2.07 days. This result is reasonably good but will get better when more training data becomes available in the future.

8.4 Decision making

As mentioned before, causal decision making is not about causal effect measurement [Fernández-Loría and Provost, 2022]. It is about appointing the most effective treatment to a certain case. This does not even have take into account confounding variables etc. The proposed way to determine the most optimal intervention is discussed in this section. The best outcome is not the intervention that has the most effect in this case. The best outcome is the closest outcome to the desired outcome. So even though another intervention has a lot more effect on the total duration, it might not be the most optimal intervention.

At first, the planner makes a sprint planning. The machine learning algorithm is going to determine what the expected duration is for this planning. This is either shorter or longer than

desired. According to this information, the rest of the algorithm is going to predict which work items are most optimal to add or remove in order to get as close to the desired duration as possible.

8.4.1 Early planning

Let us say there are ten work items that could potentially be added. The algorithm is going to predict the outcome for adding one work item for all ten possible variations. The best outcome is selected and the next step is removing all combinations when 2 work items are added. When choosing 2 items from 10, there are 45 combinations possible. For all these combinations, the best outcome is again selected. If this outcome is worse than the first, the first will be deemed to be the most efficient intervention. If not, three work items are going to be added. This process continues until the most optimal intervention is found.

8.4.2 Late planning

The late planning case is similar to the early planning case. When a planning of 10 work item is late, the machine learning model is going to calculate what the effect is of removing one work item for every possible combination. The solution closest to the desired sprint length is deemed the best of the first iteration. Then, the second iteration is going to remove two work items for every possible combination and calculate the best solution. This goes on until the best solution of the iteration is not better than the previous.

8.5 Result

The result of this effort is an algorithm that makes decisions based on the total duration of the sprint. These decisions are shown, discussed and compared to the causal rules. To do this, decisions are made on the test dataset of one project. The test data of this project has 6 sprints in them which are shown in Appendix D. Two examples are given in Table 8.3 and Table 8.4 one where items are added and another where items are removed. Green highlights items being added and red highlights items being removed.

Table 8.3: Decision making sprint 3

Work item type	Story points	Resolved by	priority	children	nlp
User Story	8.0	Person1	1.0	161	148
User Story	3.0	Person1	none	49	170
Bug	1.0	Person3	1.0	28	193
Bug	2.0	Person1	2.0	14	29
Bug	1.0	Person5	3.0	14	-1
Bug	0.5	Person7	1.0	21	182
User Story	3.0	Person1	2.0	28	169
User Story	2.0	Person3	2.0	70	140
User Story	5.0	Person7	none	28	165
Bug	1.0	Person3	2.0	42	211
User Story	5.0	Person7	2.0	28	23
Bug	5.0	Person5	2.0	14	243
User Story	2.0	Person1	2.0	168	9
Bug	0.5	Person3	1.0	21	199
Bug	1.0	Person3	1.0	14	129

The results from section 7.3 showed that a large amount of children in work items often showed large active duration. One might expect thus that work items in Table 8.3 with a large amount of children would be removed. The opposite seems to be true and two work items with small amount of children are removed. The same goes for Table 8.4 where work items with a rather small amount

Table 8.4: Decision making sprint 4

Work item type	Story points	Resolved by	priority	children	nlp
User Story	3.0	Person5	1.0	49	129
User Story	2.0	Person5	3.0	119	-1
User Story	3.0	Person6	2.0	42	132
User Story	3.0	Person1	2.0	49	132
User Story	2.0	Person5	1.0	70	122
User Story	1.0	Person1	3.0	245	15
User Story	13.0	Person5	2.0	98	178
User Story	1.0	Person6	none	21	129
User Story	2.0	Person1	2.0	315	15
Bug	0.5	Person1	2.0	7	152
User Story	1.0	Person1	none	7	29
User Story	0.5	Person1	none	35	5

were added. This pattern can also be recognised in the other sprint decisions made in Appendix D. This would mean that the most optimal intervention is not necessarily the intervention that has the most total effect.

8.6 Application

The optimal planning intervention has been put into a dash application. For the sprint data in section 8.5, previous sprints have been used to make predictions. For this application it is necessary to use actual sprint data for future sprints. For this reason, the application communicates with azure devops to get access to work items that have not been addressed yet. The planner chooses work items that have to be finished in the upcoming sprint. The application shows how long the expected duration is and shows which intervention is most optimal just like shown in Table 8.3 and Table 8.4.

8.7 Conclusion

In this chapter it can be seen how an optimal prediction model can be made for the total duration of sprint planning. The accuracy of this sprint planning prediction an MSE of 4.32 days which translates to an absolute error of 2.08 days. This is not tremendously good but will get better when more data is available. This data has only been trained on 4 projects and the results will definitely get better when more projects become available. Also, it can be seen that the most error comes from a small amounts of sprints that have taken a lot longer than average. This means that most sprints are predicted pretty accurately. This also suggests that there are other factors that are not taken into account which do have effect on the total sprint duration.

The proposed solution for the sprint planning optimisation seems to be working well. The actual performance can only be assessed when the planners have been working with it for a while.

The research question: How can the potential causal relations be used to improve future planning performance? Is answered as follows. It is possible to use causal relations to improve future planning performance by using causal decision making. This method predicts the best intervention for the current proposed sprint planning.

Chapter 9

Evaluation

In this section, the solution is evaluated. Blis planners and decision makers will tell how they think the solution will help them in the future. Furthermore, the generalizability to other companies is going to be assessed.

9.1 Company

According to Blis experts, the planning tool is useful to get a general idea of how long the sprint is going to take and what can be done to make the planning more optimal. The main benefits seen are:

- Having a better optimised planning. Having a better optimised planning will make better use of resources. Also, the customer satisfaction will increase when sprints are performing better.
- Less time needed to make an optimal planning. Less thought has to be put in and therefore the planning process is quicker.
- Less expertise needed to make a planning. Usually, a planner needs experience. With this tool, it is easier for beginners to make a sensible planning.

What could potentially be added in their eyes is a prediction tool that also predicts how many hours are spent per work item. The reason for this is that, it could help to estimate the total hours (and therefore cost) of a project beforehand. The prediction model that has been developed can indeed not predict the total amount of hours. But it could potentially calculate the total amount of weeks necessary to finish a certain project. It could calculate the optimal sprint planning for all weeks of the project and therefore determine the total project duration.

9.2 Generalizability

In order to determine the generalizability of the project, the following aspects are taken into account:

- **Research Design** is based on CRISP-DM and is widely used in the industry. Therefore it is highly likely that this type of research design is also applicable to other companies.
- **Contextual factors** like the type of agile practice could play a role in generalizability. Blis uses a scrum type of agile and therefore it is highly likely that other companies using agile scrum methodologies are able to reproduce the same kind of results. Companies that do not use scrum are less likely to have similar results. The reason for this is that scrum often uses sprints while other methods do not. Another factor is that companies vary in size

significantly and that larger companies have more budget to arrange certain work related aspects better. On the other hand larger companies might have more management issues than smaller companies. Therefore it can be said that this study is more generalizable to companies of equal size.

- **Variables** used in this research are very likely to be found in companies that also use Azure devops. These variables could be interpreted slightly differently per company or perhaps these companies do not make use of these variables. The variables from Simplicate are less likely to occur with other companies since this is not a specific program for software development. The likelihood of a company also having the combination of azure devops and Simplicate is rather small. On the other hand, similar data from other data sources could be utilised instead of Simplicate data. In other words: in order to be generalizable, companies do not necessarily need Simplicate but could also have a different hour registration.
- **Causal relations** found in the data, such as User stories taking longer than bugs, will most likely translate to other companies. There are some causal relations found that are very company specific such as: 'Resolved by person X' causing 'longer duration'. This person works at this company and therefore nothing can be said about these types of relations with regards to other companies.
- **Expert opinions** from blis agile practitioners are that it is highly likely that this research is generalizable to other companies. Many other companies use agile sprint planning along with azure devops. These companies will likely experience the same challenges.

Chapter 10

Conclusions

The main goal of this research was to get more insight into sprint planning performance. This goal has been satisfied by using multiple tools including, process mining, (causal) association rules mining and deep learning. The data and process mining part was useful for determining past performance and causes for bad performance. The machine learning part assisted mostly in causal decision making for future sprint plannings. Both of these methods were insightful and contributed to more insight into sprint planning performance. During this process the following research questions were answered:

How is the current development process performing? Where is the most need for improvement?

This question was partly answered by data visualisation. This was the first step into understanding how the process was behaving. The second step was process mining, this part also helped by understanding which specific work items were not performing well and deriving a metric for how well the process aspect of this work item was performing. This gave a better insight into which work items specifically were not performing well and thus where there was room for improvement. Not only on work item level there was room for improvement, also, certain days of the week were performing better.

What causes poor sprint planning performance? What role does work item meaning play in this?

What can be seen from the results in chapter 7 is that there are many causal relations within the data. These show how different measurements of performance are caused by other variables. Unfortunately there have only been minimal causal relations been found between the sprint context and sprint planning performance. Only some correlation based relations were found. This does not mean that part of the research has no use. The reason being is that, for the causal analysis, the NLP variables acted as controlled variables. This did in fact give more meaning to the causal claims. Some other interesting causal claims where when certain people resolve work items this can have causal relations. This again shows that agile is a very human centered way of planning and that the human aspect is very important in agile methodologies.

How can the potential causal relations be used to improve future planning performance?

At first, the causal claims can be used to take into consideration while making a planning. These causal claims can also be used to select variables to predict future sprint planning on. The last way causality can be used is by estimating intervention with causal decision making. which tries to fit the best intervention to a certain case.

How can the solution be generalised for other companies?

Agile is a very common way of working for software development. Therefore it is highly likely that these types of analysis can also be practiced at other companies using agile methodologies.

10.1 Limitations & future work

The section describes the limitations of this research and future work for this research. The limitations of this research are listed below:

- Agile is a rather human centered methodology. This human aspect is very hard to account for because it is very hard to predict. Sometimes the reason why someone is not performing well is because they had a bad day.
- The prediction algorithm predict on past data. When a new coworker joins the company or when a new project is launched, there is no data available yet. This means that, in these cases, a lot less precise predictions are made.
- The causal decision making algorithm does only takes a part of the user preference in account. There is no solution in the algorithm jet that does not allow to add or remove certain work items but it could definitely be an opportunity for the future.
- When there are a lot of items to choose from, it can take a rather long time to calculate all possibilities. This makes the result less useful for large decision tasks.

Research is never done and there are a couple improvements possible for future work.

Future work on this topic could include prediction based on the total amount of hours spend on each work item. This would give a different view on the duration and could be insightful.

The decision algorithm is only partly taking preference into account. This preference could be included into the algorithm in a more suitable way.

The solution is only based on one sprint. When a work item is moved to the next sprint, the next sprint might be performing poorly. An addition could be made to predict an optimal planning for multiple sprints.

Bibliography

[PAI,] 45

[Abrahamsson et al., 2017] Abrahamsson, P., Salo, O., Ronkainen, J., and Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439*. 7

[Al-Saqqa et al., 2020] Al-Saqqa, S., Sawalha, S., and AbdelNabi, H. (2020). Agile software development: Methodologies and trends. *International Journal of Interactive Mobile Technologies*, 14(11). 1, 7

[Allaoui et al., 2020] Allaoui, M., Kherfi, M. L., and Cheriet, A. (2020). Considerably improving clustering algorithms using umap dimensionality reduction technique: A comparative study. In *International conference on image and signal processing*, pages 317–325. Springer. 45

[Angara et al., 2020] Angara, J., Prasad, S., and Sridevi, G. (2020). Devops project management tools for sprint planning, estimation and execution maturity. *Cybernetics and Information Technologies*, 20(2):79–92. 3

[Bergstra et al., 2011] Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24. 12

[Bergstra and Bengio, 2012] Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2). 12

[Berti et al., 2019] Berti, A., Van Zelst, S. J., and van der Aalst, W. (2019). Process mining for python (pm4py): bridging the gap between process-and data science. *arXiv preprint arXiv:1905.06169*. 10

[Bhattacharya and Dupas, 2012] Bhattacharya, D. and Dupas, P. (2012). Inferring welfare maximizing treatment assignment under budget constraints. *Journal of Econometrics*, 167(1):168–196. 11

[Boschetti et al., 2014] Boschetti, M. A., Golfarelli, M., Rizzi, S., and Turricchia, E. (2014). A lagrangian heuristic for sprint planning in agile software development. *Computers & Operations Research*, 43:116–128. 1, 3, 6, 8

[Bozorgi, 2021] Bozorgi, Z. D. (2021). A causal approach to prescriptive process monitoring. In *BPM (PhD/Demos)*, pages 62–66. 14

[Bozorgi et al., 2020] Bozorgi, Z. D., Teinemaa, I., Dumas, M., La Rosa, M., and Polyvyanyy, A. (2020). Process mining meets causal machine learning: Discovering causal rules from event logs. In *2020 2nd International Conference on Process Mining (ICPM)*, pages 129–136. IEEE. 14

[Chowdhary and Chowdhary, 2020] Chowdhary, K. and Chowdhary, K. (2020). Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649. 1, 11

[de AR Goncalves et al., 2009] de AR Goncalves, J. C., Santoro, F. M., and Baiao, F. A. (2009). Business process mining from group stories. In *2009 13th International Conference on Computer Supported Cooperative Work in Design*, pages 161–166. IEEE. 13

[Doan et al., 2018] Doan, S., Yang, E. W., Tilak, S., and Torii, M. (2018). Using natural language processing to extract health-related causality from twitter messages. In *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)*, pages 84–85. IEEE. 13

[Dorfer, 2022] Dorfer, T. A. (2022). How to evaluate clustering performance without ground truth labels. 46

[Erdem et al., 2018] Erdem, S., Demirörs, O., and Rabhi, F. (2018). Systematic mapping study on process mining in agile software development. In Stamelos, I., O'Connor, R. V., Rout, T., and Dorling, A., editors, *Software Process Improvement and Capability Determination*, pages 289–299, Cham. Springer International Publishing. 2, 8, 35

[Evermann et al., 2017] Evermann, J., Rehse, J.-R., and Fettke, P. (2017). Predicting process behaviour using deep learning. *Decision Support Systems*, 100:129–140. 1, 2

[Feder et al., 2022a] Feder, A., Keith, K. A., Manzoor, E., Pryzant, R., Sridhar, D., Wood-Doughty, Z., Eisenstein, J., Grimmer, J., Reichart, R., Roberts, M. E., et al. (2022a). Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Transactions of the Association for Computational Linguistics*, 10:1138–1158. 6

[Feder et al., 2022b] Feder, A., Keith, K. A., Manzoor, E., Pryzant, R., Sridhar, D., Wood-Doughty, Z., Eisenstein, J., Grimmer, J., Reichart, R., Roberts, M. E., Stewart, B. M., Veitch, V., and Yang, D. (2022b). Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Transactions of the Association for Computational Linguistics*, 10:1138–1158. 2

[Fernández-Loría and Provost, 2022] Fernández-Loría, C. and Provost, F. (2022). Causal decision making and causal effect estimation are not the same… and why it matters. *INFORMS Journal on Data Science*, 1(1):4–16. 11, 14, 20, 55, 58

[Fischbach et al., 2020] Fischbach, J., Hauptmann, B., Konwitschny, L., Spies, D., and Vogelsang, A. (2020). Towards causality extraction from requirements. In *2020 IEEE 28th International Requirements Engineering Conference (RE)*, pages 388–393. IEEE. 13

[Ghahramani, 2004] Ghahramani, Z. (2004). Unsupervised learning. *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*, pages 72–112. 12

[Golfarelli et al., 2012] Golfarelli, M., Rizzi, S., and Turricchia, E. (2012). Sprint planning optimization in agile data warehouse design. In *DaWaK*, pages 30–41. Springer. 1

[Golfarelli et al., 2013] Golfarelli, M., Rizzi, S., and Turricchia, E. (2013). Multi-sprint planning and smooth replanning: An optimization model. *Journal of systems and software*, 86(9):2357–2370. 3, 6, 8

[Gutierrez and Gérardy, 2017] Gutierrez, P. and Gérardy, J.-Y. (2017). Causal inference and uplift modelling: A review of the literature. In *International conference on predictive applications and APIs*, pages 1–13. PMLR. 14

[Hair Jr and Sarstedt, 2021] Hair Jr, J. F. and Sarstedt, M. (2021). Data, measurement, and causal inferences in machine learning: opportunities and challenges for marketing. *Journal of Marketing Theory and Practice*, 29(1):65–77. 14

[Hippisley, 2010] Hippisley, A. R. (2010). Lexical analysis. 11

[Hirschberg and Manning, 2015] Hirschberg, J. and Manning, C. D. (2015). Advances in natural language processing. *Science*, 349(6245):261–266. 12

[Holland, 1986] Holland, P. W. (1986). Statistics and causal inference. *Journal of the American statistical Association*, 81(396):945–960. 10

[Holzinger et al., 2019] Holzinger, A., Langs, G., Denk, H., Zatloukal, K., and Müller, H. (2019). Causability and explainability of artificial intelligence in medicine. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(4):e1312. 14

[Hong et al., 2009] Hong, J.-y., Suh, E.-h., and Kim, S.-J. (2009). Context-aware systems: A literature review and classification. *Expert Systems with applications*, 36(4):8509–8522. 2

[Hou and Jung, 2021] Hou, Y. T.-Y. and Jung, M. F. (2021). Who is the expert? reconciling algorithm aversion and algorithm appreciation in ai-supported decision making. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW2):1–25. 2

[Indurkhya and Damerau, 2010] Indurkhya, N. and Damerau, F. J. (2010). *Handbook of natural language processing*. Chapman and Hall/CRC. 11, 12

[Jansi and Rajeswari, 2015] Jansi, S. and Rajeswari, K. (2015). A greedy heuristic approach for sprint planning in agile software development. *International Journal for Trends in Engineering & Technology*, 3(1):18–21. 1, 3, 6, 9

[Kumar et al., 2014] Kumar, A., Nagar, R., and Baghel, A. S. (2014). A genetic algorithm approach to release planning in agile environment. In *2014 International Conference on Information Systems and Computer Networks (ISCON)*, pages 118–122. IEEE. 9

[Kumar and Minz, 2014] Kumar, V. and Minz, S. (2014). Feature selection: a literature review. *SmartCR*, 4(3):211–229. 19

[Li et al., 2013] Li, J., Le, T. D., Liu, L., Liu, J., Jin, Z., and Sun, B. (2013). Mining causal association rules. In *2013 IEEE 13th International Conference on Data Mining Workshops*, pages 114–123. IEEE. 53

[Li et al., 2010] Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. 11

[Li and Mao, 2019] Li, P. and Mao, K. (2019). Knowledge-oriented convolutional neural network for causal relation extraction from natural language texts. *Expert Systems with Applications*, 115:512–523. 13

[Liu et al., 2021] Liu, G., Boyd, M., Yu, M., Halim, S. Z., and Quddus, N. (2021). Identifying causality and contributory factors of pipeline incidents by employing natural language processing and text mining techniques. *Process Safety and Environmental Protection*, 152:37–46. 13

[Lucassen et al., 2015] Lucassen, G., Dalpiaz, F., Van Der Werf, J. M. E., and Brinkkemper, S. (2015). Forging high-quality user stories: towards a discipline for agile requirements. In *2015 IEEE 23rd international requirements engineering conference (RE)*, pages 126–135. IEEE. 6

[Mahesh, 2020] Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9:381–386. 12

[Malgonde and Chari, 2019] Malgonde, O. and Chari, K. (2019). An ensemble-based model for predicting agile software development effort. *Empirical Software Engineering*, 24:1017–1055. 6, 9

[Manski, 2004] Manski, C. F. (2004). Statistical treatment rules for heterogeneous populations. *Econometrica*, 72(4):1221–1246. 11, 20

[Marinov, 2021] Marinov, B. (2021). Inter-class clustering of text data using dimensionality reduction and bert. 43, 44, 45, 46

[Moraffah et al., 2020] Moraffah, R., Karami, M., Guo, R., Raglin, A., and Liu, H. (2020). Causal interpretability for machine learning-problems, methods and evaluation. *ACM SIGKDD Explorations Newsletter*, 22(1):18–33. 3

[Nadkarni et al., 2011] Nadkarni, P. M., Ohno-Machado, L., and Chapman, W. W. (2011). Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551. 11

[Nath et al., 2022] Nath, S., Marie, A., Ellershaw, S., Korot, E., and Keane, P. A. (2022). New meaning for nlp: the trials and tribulations of natural language processing with gpt-3 in ophthalmology. *British Journal of Ophthalmology*, 106(7):889–892. 12

[Noreika and Gudas, 2023] Noreika, K. and Gudas, S. (2023). Causal knowledge modelling for agile development of enterprise application systems. *Informatica*, 34(1):121–146. 6

[Olaya et al., 2020a] Olaya, D., Coussement, K., and Verbeke, W. (2020a). A survey and benchmarking study of multitreatment uplift modeling. *Data Mining and Knowledge Discovery*, 34:273–308. 20

[Olaya et al., 2020b] Olaya, D., Vásquez, J., Maldonado, S., Miranda, J., and Verbeke, W. (2020b). Uplift modeling for preventing student dropout in higher education. *Decision support systems*, 134:113320. 11

[Pearl, 2009] Pearl, J. (2009). *Causality*. Cambridge university press. 10, 14

[Pearl, 2018] Pearl, J. (2018). Theoretical impediments to machine learning with seven sparks from the causal revolution. *arXiv preprint arXiv:1801.04016*. 14

[Pearl, 2019] Pearl, J. (2019). The seven tools of causal inference, with reflections on machine learning. *Communications of the ACM*, 62(3):54–60. 14

[Pontes et al., 2016] Pontes, F. J., Amorim, G., Balestrassi, P. P., Paiva, A., and Ferreira, J. R. (2016). Design of experiments and focused grid search for neural network parameter optimization. *Neurocomputing*, 186:22–34. 12

[Radcliffe and Surry, 2011] Radcliffe, N. J. and Surry, P. D. (2011). Real-world uplift modelling with significance-based uplift trees. *White Paper TR-2011-1, Stochastic Solutions*, pages 1–33. 11

[Raharjana et al., 2021] Raharjana, I. K., Siahaan, D., and Fatichah, C. (2021). User stories and natural language processing: A systematic literature review. *IEEE Access*, 9:53811–53826. 6

[R.eshuis, 2023] R.eshuis (2023). GitHub - heshuis/CA2RM: Causal Analysis with Causal Association Rule Method — github.com. `https://github.com/heshuis/CA2RM`. [Accessed 11-May-2023]. 20

[Rezaee et al., 2020] Rezaee, Z., Aliabadi, S., Dorestani, A., and Rezaee, N. J. (2020). Application of time series models in business research: correlation, association, causation. *Sustainability*, 12(12):4833. 2

[Richens et al., 2020] Richens, J. G., Lee, C. M., and Johri, S. (2020). Improving the accuracy of medical diagnosis with causal machine learning. *Nature communications*, 11(1):3923. 3

[Rozinat,] Rozinat, A. Disco tour. 19

[Salehi, 2022] Salehi, F. (2022). Role of artificial intelligence on agile planning and organizational performance in the ict industry. *Asian Journal of Economics, Finance and Management*, pages 581–586. 9

[Schölkopf, 2022] Schölkopf, B. (2022). Causality for machine learning. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pages 765–804. 14

[Schröer et al., 2021] Schröer, C., Kruse, F., and Gómez, J. M. (2021). A systematic literature review on applying crisp-dm process model. *Procedia Computer Science*, 181:526–534. 17

[Shoush and Dumas, 2022] Shoush, M. and Dumas, M. (2022). Prescriptive process monitoring under resource constraints: a causal inference approach. In *Process Mining Workshops: ICPM 2021 International Workshops, Eindhoven, The Netherlands, October 31–November 4, 2021, Revised Selected Papers*, pages 180–193. Springer. 14

[Snoek et al., 2015] Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M., Prabhat, M., and Adams, R. (2015). Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pages 2171–2180. PMLR. 12

[Syring et al., 2019] Syring, A. F., Tax, N., and van der Aalst, W. M. (2019). Evaluating conformance measures in process mining using conformance propositions. *Transactions on Petri Nets and Other Models of Concurrency XIV*, pages 192–221. 10

[Trask et al., 2015] Trask, A., Michalak, P., and Liu, J. (2015). sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. *arXiv preprint arXiv:1511.06388*. 18

[van der Aalst, 2012] van der Aalst, W. (2012). Process mining. *ACM Transactions on Management Information Systems*, 3(2):1–17. 2, 10

[van der Aalst, 2010] van der Aalst, W. M. (2010). Process discovery: Capturing the invisible. *IEEE Computational Intelligence Magazine*, 5(1):28–41. 10

[Vijayarani et al., 2015] Vijayarani, S., Ilamathi, M. J., Nithya, M., et al. (2015). Preprocessing techniques for text mining-an overview. *International Journal of Computer Science & Communication Networks*, 5(1):7–16. 11

[Waibel et al., 2022] Waibel, P., Pfahlsberger, L., Revoredo, K., and Mendling, J. (2022). Causal process mining from relational databases with domain knowledge. *arXiv preprint arXiv:2202.08314*. 13

[Woodside et al., 2007] Woodside, M., Franks, G., and Petriu, D. C. (2007). The future of software performance engineering. In *Future of Software Engineering (FOSE'07)*, pages 171–187. IEEE. 3

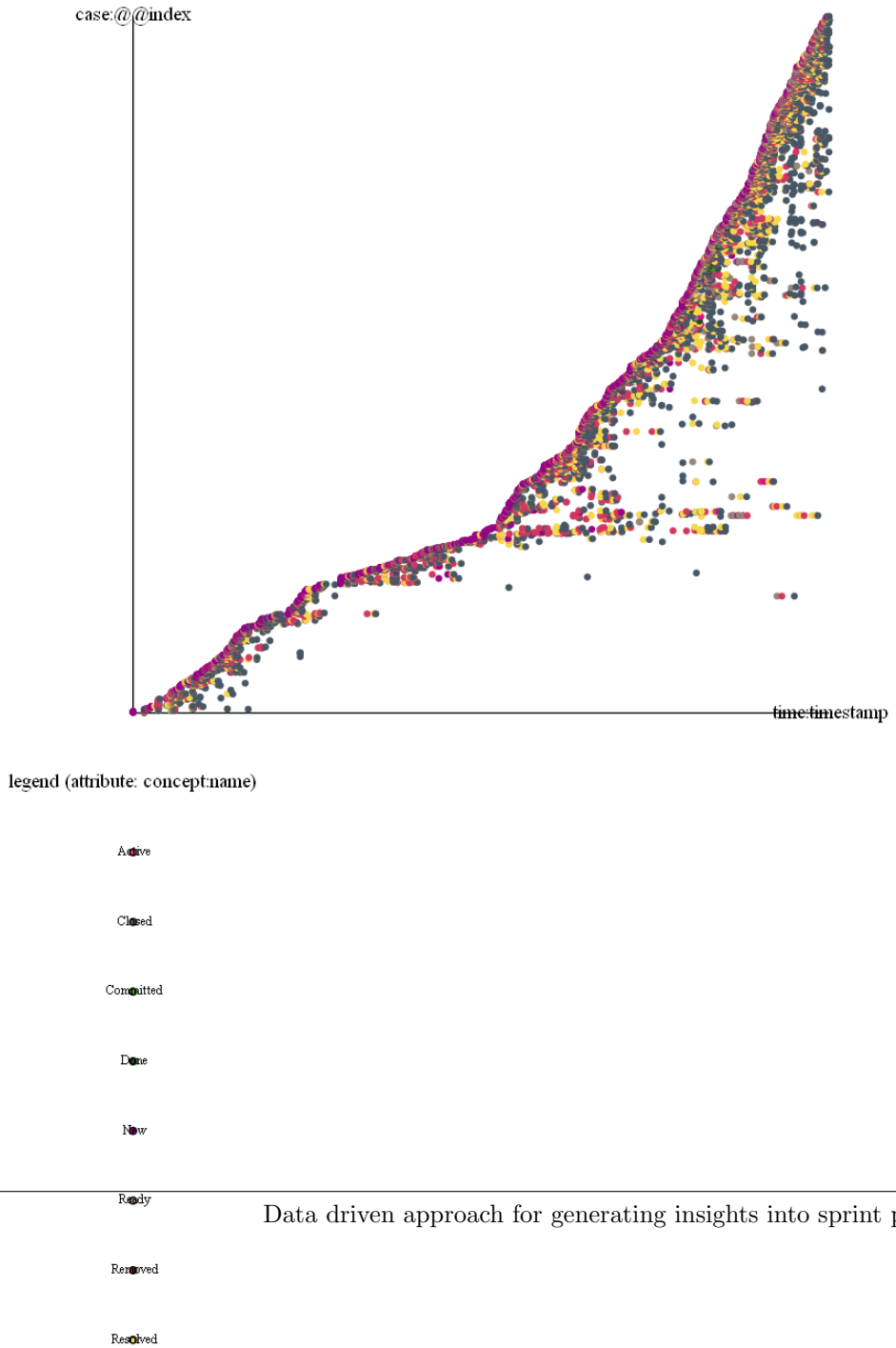
[Xie, 2022] Xie, J. (2022). Top pre-trained models for sentence embedding. 44

[Yager et al., 1991] Yager, R. R., Ford, K. M., and Cañas, A. J. (1991). An approach to the linguistic summarization of data. In *Uncertainty in Knowledge Bases: 3rd International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU'90 Paris, France, July 2–6, 1990 Proceedings 3*, pages 456–468. Springer. 2

[Zhao and Liu, 2023] Zhao, Y. and Liu, Q. (2023). Causal ml: Python package for causal inference machine learning. *SoftwareX*, 21:101294. 19

Appendix A

Process mining



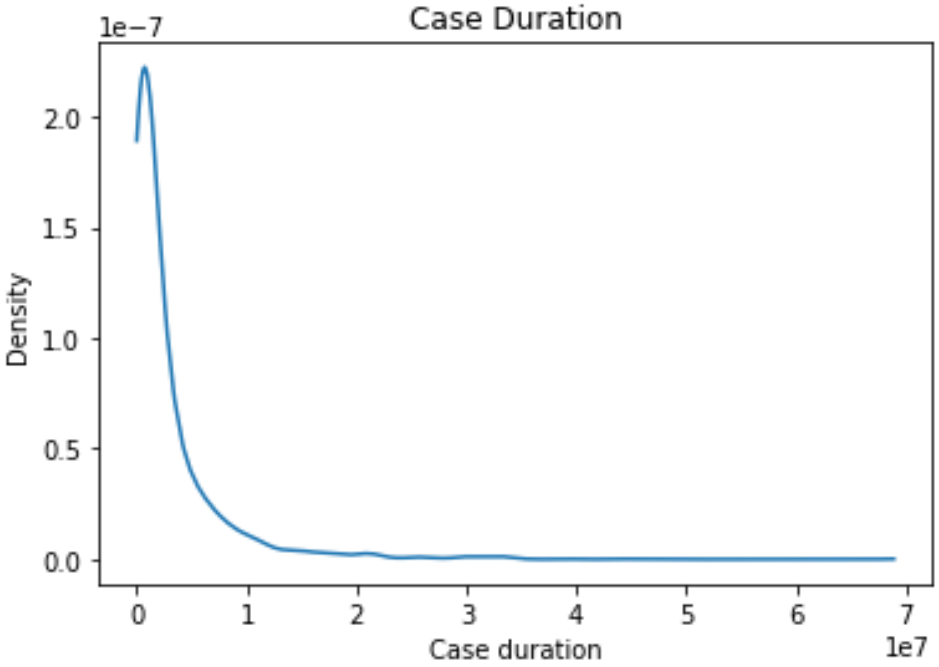


Figure A.2: Story

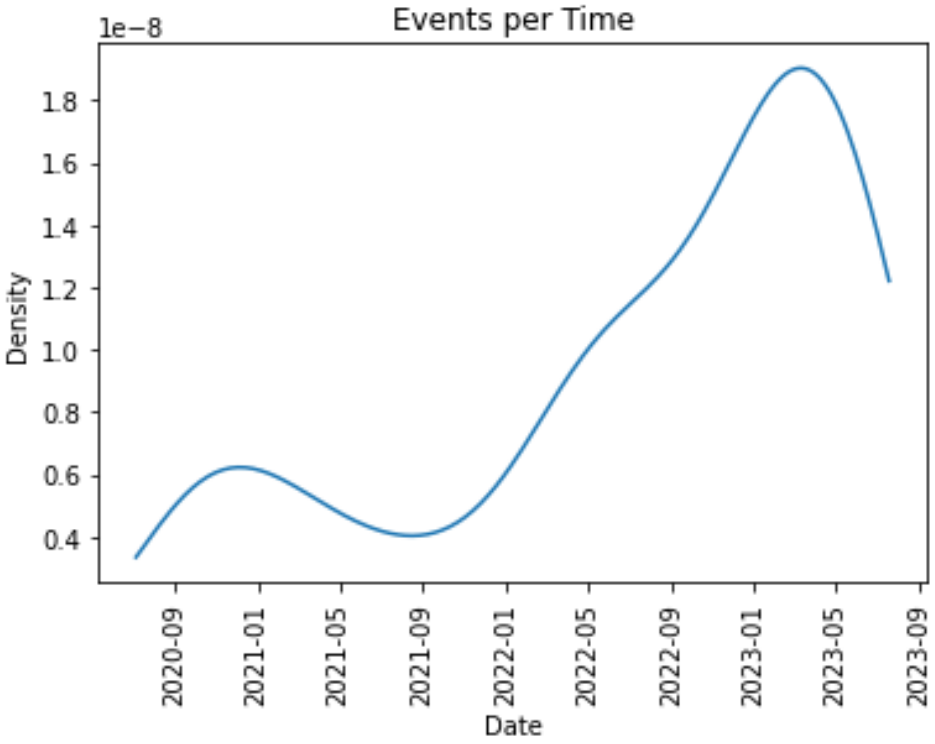


Figure A.3: Story

Appendix B

Clusters

B.1 Cluster examples

Table B.1: cluster 1

Cluster	Title
1	Software Updates 23.19
1	Software Updates Sprint 23.20
1	Style update
1	Software Updates 23.22
1	Software Updates 23.18
1	Software Updates 23.17
1	Third-party software update augustus 2021
1	Software Updates 23.16
1	Third-party software updates
1	Software Updates Sprint 23.13
1	Software Updates Sprint 23.12
1	Software updates sprint 22.14
1	Software updates Sprint 22.4
1	Software updates Sprint 22.3
1	Software Updates 23.15

Table B.2: clusters 2 and 3

Cluster	Title
2	Sprint 27: Renovate, SonarCloud, Mendbolt
2	Sprint 28: Renovate, SonarCloud, Mendbolt
2	Sprint 26: Renovate, SonarCloud, Mendbolt
2	Sprint 8: Renovate, SonarCloud, MendBolt
2	Sprint 24: Renovate, SonarCloud, Mendbolt
2	Sprint 10: Renovate, SonarCloud, MendBolt
2	Sprint 25: Renovate, SonarCloud, Mendbolt
2	Sprint 12: Renovate, SonarCloud, MendBolt
2	Sprint 13: Renovate, SonarCloud, MendBolt
2	Sprint 14: Renovate, SonarCloud, MendBolt
2	Sprint 16: Renovate, SonarCloud, Mendbolt
2	Sprint 17: Renovate, SonarCloud, Mendbolt
2	Sprint 15: Renovate, SonarCloud, MendBolt
2	Sprint 19: Renovate, SonarCloud, Mendbolt
2	Sprint 20: Renovate, SonarCloud, Mendbolt
2	Sprint 9: Renovate, SonarCloud, MendBolt
2	Sprint 21: Renovate, SonarCloud, Mendbolt
2	Sprint 22: Renovate, SonarCloud, Mendbolt
2	Sprint 23: Renovate, SonarCloud, Mendbolt
2	Sprint 18: Renovate, SonarCloud, Mendbolt
3	Technical debt sprint 22.12
3	Technical debt sprint 22.9
3	Technical debt sprint 22.10
3	Technical debt sprint 22.11
3	Technical Debt Sprint 22.19
3	Technical debt sprint 22.14
3	Technical debt sprint 22.15
3	Technical debt sprint 22.16
3	Technical debt sprint 22.17
3	Technical Debt Sprint 22.18
3	Technical debt - Service maintenance sprint 23.11 by updating Renovate PRs with .NET packages
3	Technical debt sprint 22.13
3	Technical debt hardcoded Currency values in views
3	Technical debt - missing translations in Localize sprint 23.13
3	Print out debt service / IRR

Appendix C

Association rules

Table C.1: Association rules 1

antecedents	consequents	support	confidence
['points_many', 'children_large']	['typeUser Story']	0.21	0.99
['act_dur_long', 'points_many']	['typeUser Story']	0.15	0.98
['points_many']	['typeUser Story']	0.26	0.97
['act_dur_long', 'children_large']	['typeUser Story']	0.23	0.92
['priority_2.0', 'rework_infrequent']	['Performance_none']	0.15	0.91
['priority_2.0', 'children_large']	['typeUser Story']	0.19	0.87
['children_large']	['typeUser Story']	0.39	0.87
['priority_2.0', 'act_dur_long']	['typeUser Story']	0.15	0.87
['act_dur_long']	['typeUser Story']	0.29	0.87
['fitnessmedium', 'children_large']	['typeUser Story']	0.18	0.86
['Performance_none', 'Resolved_by_Person 1']	['priority_2.0']	0.17	0.85
['rework_medium', 'Resolved_by_Person 1']	['fitnessgood']	0.16	0.84
['rework_frequent', 'children_large']	['typeUser Story']	0.18	0.83
['typeUser Story', 'rework_frequent']	['children_large']	0.18	0.82
['typeUser Story', 'points_many']	['children_large']	0.21	0.81
['rework_medium', 'priority_2.0']	['fitnessgood']	0.18	0.81
['rework_medium']	['fitnessgood']	0.27	0.8
['points_many']	['children_large']	0.22	0.8
['points_many']	['typeUser Story', 'children_large']	0.21	0.79
['typeUser Story', 'act_dur_long']	['children_large']	0.23	0.78
['typeUser Story', 'Resolved_by_Person 1']	['priority_2.0']	0.18	0.77
['children_small']	['Performance_none']	0.2	0.77
['act_dur_long']	['children_large']	0.25	0.74
['Resolved_by_Person 1']	['priority_2.0']	0.27	0.74
['fitnessgood', 'Resolved_by_Person 1']	['rework_medium']	0.16	0.73
['points_medium']	['typeUser Story']	0.16	0.73
['rework_infrequent']	['Performance_none']	0.24	0.72
['Performance_none', 'fitnessmedium']	['priority_2.0']	0.15	0.72
['fitnessmedium']	['typeUser Story']	0.28	0.71
['children_small']	['priority_2.0']	0.18	0.71
['fitnessgood', 'Resolved_by_Person 1']	['priority_2.0']	0.15	0.71
['priority_2.0', 'typeBug']	['Performance_none']	0.15	0.68
['typeUser Story', 'Performance_none']	['priority_2.0']	0.2	0.68
['act_dur_long']	['typeUser Story', 'children_large']	0.23	0.68
['priority_2.0', 'Resolved_by_Person 1']	['typeUser Story']	0.18	0.68
['priority_2.0', 'fitnessmedium']	['typeUser Story']	0.16	0.67
['rework_frequent']	['typeUser Story']	0.22	0.67
['Performance_none']	['priority_2.0']	0.36	0.67

Table C.2: Association rules 2

antecedents	consequents	support	confidence
['act_dur_short']	['Performance_none']	0.22	0.66
['rework_frequent']	['children_large']	0.22	0.66
['rework_medium', 'fitnessgood']	['priority_2.0']	0.18	0.66
['children_small']	['fitnessgood']	0.17	0.66
['Performance_none', 'fitnessgood']	['priority_2.0']	0.17	0.65
['rework_infrequent']	['typeUser Story']	0.22	0.65
['rework_medium']	['priority_2.0']	0.22	0.65
['Performance_none', 'typeBug']	['priority_2.0']	0.15	0.65
['typeUser Story', 'fitnessmedium']	['children_large']	0.18	0.65
['typeBug']	['fitnessgood']	0.24	0.65
['priority_2.0', 'fitnessmedium']	['Performance_none']	0.15	0.65
['Resolved_by_Person 1']	['typeUser Story']	0.24	0.65
['priority_2.0', 'Resolved_by_Person 1']	['Performance_none']	0.17	0.64
['Performance_none', 'rework_infrequent']	['priority_2.0']	0.15	0.63
['priority_2.0']	['Performance_none']	0.36	0.63
['act_dur_short']	['priority_2.0']	0.21	0.63
['typeBug']	['Performance_none']	0.23	0.62
['typeUser Story']	['children_large']	0.39	0.62
['priority_2.0', 'fitnessgood']	['rework_medium']	0.18	0.61
['act_dur_short']	['fitnessgood']	0.2	0.61
['fitnessmedium']	['priority_2.0']	0.24	0.6
['typeUser Story', 'priority_2.0']	['Performance_none']	0.2	0.59
['typeBug']	['priority_2.0']	0.22	0.59
['rework_medium', 'fitnessgood']	['Resolved_by_Person 1']	0.16	0.59
['typeUser Story', 'children_large']	['act_dur_long']	0.23	0.58
['typeUser Story', 'points_many']	['act_dur_long']	0.15	0.58
['act_dur_short']	['typeBug']	0.19	0.58
['Resolved_by_Person 1']	['fitnessgood']	0.21	0.58
['typeUser Story', 'fitnessgood']	['priority_2.0']	0.15	0.58
['children_medium']	['fitnessgood']	0.17	0.58
['points_many']	['act_dur_long']	0.16	0.58
['fitnessgood']	['priority_2.0']	0.29	0.57
['children_medium']	['typeBug']	0.17	0.56
['points_many']	['typeUser Story', 'act_dur_long']	0.15	0.56
['rework_medium']	['Resolved_by_Person 1']	0.19	0.56
['act_dur_medium']	['fitnessgood']	0.19	0.56
['priority_2.0', 'Resolved_by_Person 1']	['fitnessgood']	0.15	0.56
['Resolved_by_Person 1']	['Performance_none']	0.2	0.56
['typeUser Story', 'priority_2.0']	['children_large']	0.19	0.56
['children_large']	['act_dur_long']	0.25	0.55
['typeUser Story', 'children_large']	['points_many']	0.21	0.55
['rework_frequent']	['typeUser Story', 'children_large']	0.18	0.55
['fitnessmedium']	['Performance_none']	0.21	0.54
['fitnessmedium']	['children_large']	0.21	0.54
['typeUser Story', 'priority_2.0']	['Resolved_by_Person 1']	0.18	0.53
['typeUser Story', 'act_dur_long']	['points_many']	0.15	0.53
['priority_2.0', 'fitnessgood']	['Resolved_by_Person 1']	0.15	0.53

Table C.3: Association rules 3

antecedents	consequents	support	confidence
['fitnessgood']	['rework_medium']	0.27	0.53
['rework_medium']	['priority_2.0', 'fitnessgood']	0.18	0.53
['rework_infrequent']	['fitnessmedium']	0.17	0.52
['typeBug']	['act_dur_short']	0.19	0.52
['children_large']	['typeUser Story', 'act_dur_long']	0.23	0.51
['priority_2.0']	['fitnessgood']	0.29	0.51
['Resolved_by_Person 1']	['rework_medium']	0.19	0.51
['Resolved_by_Person 1']	['typeUser Story', 'priority_2.0']	0.18	0.5
['children_large']	['rework_frequent']	0.22	0.49
['act_dur_long']	['fitnessmedium']	0.16	0.49
['rework_frequent']	['fitnessmedium']	0.16	0.49
['Performance_none', 'priority_2.0']	['Resolved_by_Person 1']	0.17	0.49
['children_large']	['points_many']	0.22	0.49
['fitnessgood']	['typeBug']	0.24	0.48
['priority_2.0']	['Resolved_by_Person 1']	0.27	0.48
['children_large']	['typeUser Story', 'points_many']	0.21	0.48
['children_large']	['fitnessmedium']	0.21	0.47
['Resolved_by_Person 1']	['Performance_none', 'priority_2.0']	0.17	0.47
['act_dur_long']	['points_many']	0.16	0.47
['rework_medium']	['fitnessgood', 'Resolved_by_Person 1']	0.16	0.47
['typeUser Story', 'children_large']	['fitnessmedium']	0.18	0.47
['typeUser Story', 'children_large']	['rework_frequent']	0.18	0.47
['fitnessmedium']	['typeUser Story', 'children_large']	0.18	0.46
['typeUser Story']	['act_dur_long']	0.29	0.46
['typeUser Story', 'priority_2.0']	['fitnessmedium']	0.16	0.46
['act_dur_long']	['typeUser Story', 'points_many']	0.15	0.46
['act_dur_long']	['typeUser Story', 'priority_2.0']	0.15	0.46
['rework_infrequent']	['Performance_none', 'priority_2.0']	0.15	0.46
['Performance_none']	['rework_infrequent']	0.24	0.45
['rework_frequent']	['act_dur_long']	0.15	0.45
['act_dur_long']	['rework_frequent']	0.15	0.45
['typeBug']	['children_medium']	0.17	0.45
['typeUser Story']	['fitnessmedium']	0.28	0.45
['typeUser Story', 'priority_2.0']	['act_dur_long']	0.15	0.45
['fitnessmedium']	['rework_infrequent']	0.17	0.44
['Performance_none']	['typeBug']	0.23	0.44
['Performance_none', 'priority_2.0']	['rework_infrequent']	0.15	0.43
['Performance_none', 'priority_2.0']	['fitnessmedium']	0.15	0.43
['Performance_none', 'priority_2.0']	['typeBug']	0.15	0.43
['children_large']	['typeUser Story', 'priority_2.0']	0.19	0.43
['Resolved_by_Person 1']	['rework_medium', 'fitnessgood']	0.16	0.43
['fitnessgood']	['Resolved_by_Person 1']	0.21	0.42
['typeUser Story']	['points_many']	0.26	0.42
['priority_2.0']	['fitnessmedium']	0.24	0.42
['Performance_none']	['act_dur_short']	0.22	0.42
['fitnessmedium']	['act_dur_long']	0.16	0.41

Table C.4: Association rules 4

antecedents	consequents	support	confidence
['fitnessmedium']	['rework_frequent']	0.16	0.41
['Resolved_by_Person 1']	['priority_2.0', 'fitnessgood']	0.15	0.41
['children_large']	['typeUser Story', 'fitnessmedium']	0.18	0.41
['children_large']	['typeUser Story', 'rework_frequent']	0.18	0.41
['typeBug']	['Performance_none', 'priority_2.0']	0.15	0.41
['fitnessgood']	['act_dur_short']	0.2	0.4
['Performance_none']	['fitnessmedium']	0.21	0.4
['fitnessmedium']	['typeUser Story', 'priority_2.0']	0.16	0.4
['priority_2.0']	['typeBug']	0.22	0.39
['fitnessmedium']	['Performance_none', 'priority_2.0']	0.15	0.39
['priority_2.0']	['rework_medium']	0.22	0.38
['Performance_none']	['Resolved_by_Person 1']	0.2	0.38
['Performance_none']	['typeUser Story', 'priority_2.0']	0.2	0.38
['typeUser Story']	['Resolved_by_Person 1']	0.24	0.38
['priority_2.0']	['act_dur_short']	0.21	0.37
['Performance_none']	['children_small']	0.2	0.37
['fitnessgood']	['act_dur_medium']	0.19	0.37
['typeUser Story']	['act_dur_long', 'children_large']	0.23	0.36
['priority_2.0']	['typeUser Story', 'Performance_none']	0.2	0.36
['typeUser Story']	['rework_frequent']	0.22	0.36
['fitnessgood']	['rework_medium', 'priority_2.0']	0.18	0.35
['typeUser Story']	['rework_infrequent']	0.22	0.35
['fitnessgood']	['children_medium']	0.17	0.34
['typeUser Story']	['points_many', 'children_large']	0.21	0.34
['Performance_none']	['children_medium']	0.18	0.34
['fitnessgood']	['children_small']	0.17	0.33
['Performance_none']	['priority_2.0', 'fitnessgood']	0.17	0.33
['Performance_none']	['priority_2.0', 'Resolved_by_Person 1']	0.17	0.33
['priority_2.0']	['typeUser Story', 'Resolved_by_Person 1']	0.18	0.32
['priority_2.0']	['children_small']	0.18	0.32
['priority_2.0']	['rework_medium', 'fitnessgood']	0.18	0.31
['priority_2.0']	['Performance_none', 'fitnessgood']	0.17	0.31
['fitnessgood']	['rework_medium', 'Resolved_by_Person 1']	0.16	0.31
['priority_2.0']	['Performance_none', 'Resolved_by_Person 1']	0.17	0.31
['typeUser Story']	['priority_2.0', 'children_large']	0.19	0.3
['fitnessgood']	['priority_2.0', 'Resolved_by_Person 1']	0.15	0.3
['typeUser Story']	['fitnessmedium', 'children_large']	0.18	0.29
['typeUser Story']	['priority_2.0', 'Resolved_by_Person 1']	0.18	0.29
['typeUser Story']	['rework_frequent', 'children_large']	0.18	0.29
['Performance_none']	['priority_2.0', 'rework_infrequent']	0.15	0.29
['Performance_none']	['priority_2.0', 'fitnessmedium']	0.15	0.29
['Performance_none']	['priority_2.0', 'typeBug']	0.15	0.28
['priority_2.0']	['Performance_none', 'fitnessmedium']	0.15	0.27
['priority_2.0']	['Performance_none', 'rework_infrequent']	0.15	0.27
['priority_2.0']	['Performance_none', 'typeBug']	0.15	0.27
['priority_2.0']	['typeUser Story', 'fitnessgood']	0.15	0.27
['priority_2.0']	['fitnessgood', 'Resolved_by_Person 1']	0.15	0.27
['typeUser Story']	['points_medium']	0.16	0.26
['typeUser Story']	['priority_2.0', 'fitnessmedium']	0.16	0.25

Appendix D

Decision making results

Table D.1: Decision making sprint 1

Work item type	Story points	Resolved by	priority	children	nlp
User Story	3.0	Person5	1.0	49	129
User Story	1.0	Person6	1.0	77	129
User Story	2.0	Person9	1.0	28	-1
User Story	8.0	Person6	1.0	63	129
User Story	2.0	Person6	1.0	28	122
User Story	2.0	Person10	1.0	21	72
Bug	0.5	Person6	2.0	49	231
User Story	3.0	Person5	1.0	49	148
User Story	2.0	Person5	2.0	42	148
User Story	2.0	Person5	2.0	63	148
User Story	2.0	Person3	1.0	28	148
User Story	13.0	Person5	2.0	98	178
Bug	none	Person6	none	14	200
Bug	none	Person5	none	28	191
Bug	none	Person3	none	14	207
Bug	none	Person3	none	14	217
Bug	none	Person3	none	14	-1
Bug	none	Person3	none	14	200
Bug	none	Person6	none	7	189
Bug	none	Person3	none	21	188

Table D.2: Decision making sprint 2

Work item type	Story points	Resolved by	priority	children	nlp
User Story	5.0	Person1	1.0	224	146
User Story	2.0	Person1	2.0	35	169
Bug	0.5	Person7	2.0	35	231
User Story	5.0	Person7	none	133	201
User Story	1.0	Person1	none	7	29
User Story	0.5	Person1	none	35	5
User Story	3.0	Person3	none	161	144
User Story	2.0	Person8	2.0	28	144
User Story	2.0	Person8	2.0	28	143
User Story	2.0	Person8	none	14	155
User Story	1.0	Person1	2.0	224	9
Bug	0.5	Person7	2.0	14	233
Bug	0.5	Person1	2.0	28	140

Table D.3: Decision making sprint 3

Work item type	Story points	Resolved by	priority	children	nlp
User Story	8.0	Person1	1.0	161	148
User Story	3.0	Person1	none	49	170
Bug	1.0	Person3	1.0	28	193
Bug	2.0	Person1	2.0	14	29
Bug	1.0	Person5	3.0	14	-1
Bug	0.5	Person7	1.0	21	182
User Story	3.0	Person1	2.0	28	169
User Story	2.0	Person3	2.0	70	140
User Story	5.0	Person7	none	28	165
Bug	1.0	Person3	2.0	42	211
User Story	5.0	Person7	2.0	28	23
Bug	5.0	Person5	2.0	14	243
User Story	2.0	Person1	2.0	168	9
Bug	0.5	Person3	1.0	21	199
Bug	1.0	Person3	1.0	14	129

Table D.4: Decision making sprint 4

Work item type	Story points	Resolved by	priority	children	nlp
User Story	3.0	Person5	1.0	49	129
User Story	2.0	Person5	3.0	119	-1
User Story	3.0	Person6	2.0	42	132
User Story	3.0	Person1	2.0	49	132
User Story	2.0	Person5	1.0	70	122
User Story	1.0	Person1	3.0	245	15
User Story	13.0	Person5	2.0	98	178
User Story	1.0	Person6	none	21	129
User Story	2.0	Person1	2.0	315	15
Bug	0.5	Person1	2.0	7	152
User Story	1.0	Person1	none	7	29
User Story	0.5	Person1	none	35	5

Table D.5: Decision making sprint 5

Work item type	**Story points**	**Resolved by**	**priority**	**children**	**nlp**
Bug	0.5	Person3	2.0	14	-1
User Story	8.0	Person1	1.0	336	179
User Story	3.0	Person3	2.0	161	144
User Story	5.0	Person1	2.0	63	139
User Story	5.0	Person3	2.0	42	139
Bug	0.5	Person7	3.0	14	200
User Story	3.0	Person7	1.0	56	35
User Story	2.0	Person1	2.0	301	9
Bug	0.5	Person1	2.0	7	165
Bug	0.5	Person1	2.0	14	191
Bug	1.0	Person3	1.0	28	193
Bug	0.5	Person5	1.0	14	-1
User Story	2.0	Person5	2.0	42	148
Bug	2.0	Person1	2.0	14	29

Table D.6: Decision making sprint 6

work item type	**Story_points**	**Resolved_by**	**priority**	**children**	nlp
User Story	1.0	Person5	none	21	132
User Story	2.0	Person5	none	42	170
Bug	1.0	Person3	1.0	35	201
User Story	2.0	Person3	none	28	177
User Story	2.0	Person5	none	21	173
Bug	0.5	Person5	none	14	132
User Story	13.0	Person3	1.0	98	154
User Story	3.0	Person3	1.0	42	154
Bug	2.0	Person1	2.0	21	182
User Story	2.0	Person5	2.0	154	9
User Story	1.0	Person5	2.0	35	10
Bug	0.5	Person5	1.0	21	233
User story	2.0	Person1	none	14	29