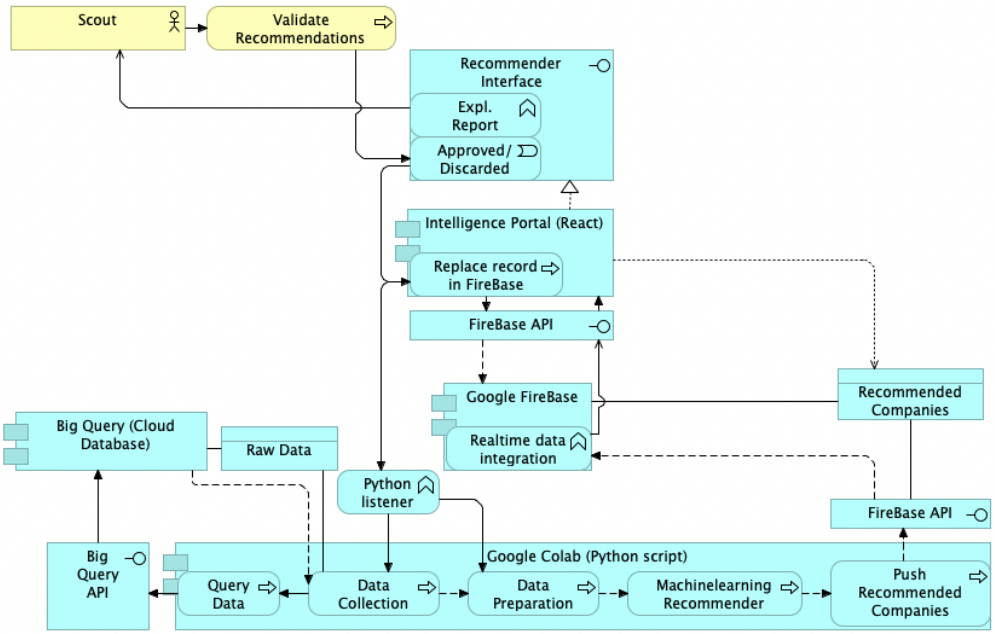
UNIVERSITY OF TWENTE.

UNKNOWN
BEYOND KNOWN



A hybrid recommender-system for startup scouting.

Design Science Research at the behest of Unknown Group

Master's thesis in Business Information Technology

KEES BOSCH

EEMCS MSc Business Information Technology

UNIVERSITY OF TWENTE

Enschede, Netherlands, May 5, 2022

www.utwente.nl

MASTER'S THESIS 2022

A hybrid recommender-system for startup scouting.

A Design Science Research at the behest of Unknown Group

Kees Tjalle Dries Bosch

UNIVERSITY
OF TWENTE.

EEMCS
MSc Business Information Technology
UNIVERSITY OF TWENTE
Enschede, Netherlands 2022

A hybrid recommender-system for startup scouting.
A Design Science Research at the behest of Unknown Group
KEES TJALLE DRIES BOSCH

Supervisor: Fons Wijnhoven, University of Twente
Supervisor: Maurice Verkeulen, University of Twente
Supervisor: Giels Brouwer, Unknown Group

Master's Thesis 2022
EEMCS Business Information Technology

University of Twente
Enschede, Netherlands
Telephone +31 638326955

Cover: The application view architecture of the recommender-system in this thesis.

Typeset in LaTeX
Utrecht, Netherlands 2022

Implementing a hybrid recommender-system to assist startup scouting in order to improve reliability and efficiency.
A Design Science Research at the behest of Unknown Group
Kees Tjalle Dries Bosch
EEMCS
University of Twente

Abstract

This design science research study investigates whether problems in the startup scouting process of Unknown Group can be solved by implementing a hybrid recommender-system, hereby improving efficiency and reliability of this scouting process. With use of a literature study, a design theory for a hybrid recommender-system for startup scouting was established. This design theory was translated to requirements that were supplemented by the direct stakeholders. These requirements were converted into design suggestions, which were assembled into an application architecture for the development of a prototype. The success for this solution to the problem was tested using the mean Average Precision and the Matthews-Correlation Coefficient as well as the inclusion of the requirements. Though the synthetic data testing seemed very promising, the recommender-system failed to provide a sufficient recommendation with real data. This confirmed the hypothesis that the data was too inconsistent and incomplete for proper data driven decision making, a major issue that should be a the top of the agenda for Unknown Group.

Keywords: Hybrid, Recommender-system, scouting process, startups, venture capital, design science research methodology.

Acknowledgements

I am very satisfied with the course of this thesis. The scheduled meetings every three weeks with my supervisors from both Unknown Group and the University of Twente have helped me to stay on track and scoping the project. Keeping the scope narrow was the biggest challenge as there were numerous interesting possibilities for data driven insights in the business processes of Unknown Group.

Looking back on this thesis project, I am amazed how much I have learned. I had to teach myself how to code in React and BigQuery SQL and working on behalf of Unknown Group has forced me to structure my coding much better. This research also required continuous interaction with stakeholders from many different departments within the organization which required a well thought out and structured approach. Participating in the tech team of Unknown Group has offered me a unique opportunity to have a better understanding of what my future occupation will look like. The domain that they are operating in really draws me as numerous interesting startups fly by every day. I have also found that my expertise can add a lot of value to the current tech team because they are in need of someone who can extract valuable quantitative information from the data. It is therefore that I am glad to be offered the opportunity to start working at Unknown Group.

I wish to express great thanks to Fons Wijnhoven and Maurice van Keulen, who have guided me during this research on behalf of the University of Twente. I also want to mark my appreciations to Giels Brouwer(CTO), who supervised my project at Unknown Group, as well as to the rest of the tech team (David Schrooten and Erwin Retel), who were very helpful in explaining the current platform and for suggestions for the development of the recommender-system. Also involved in this design science project were the scouting leads; Evangelos Kontos and Noah Paul, and the head of Scouting; Jasper Warmenhoven. Thank you for the discussions about the goals and requirement and for your input in the evaluation of the recommender-system. Next, I wish to thank Jurgen Nieuwenhuijsen, the COO of Unknown Group, who helped me setting the goals and requirements and explaining the motivation for this project. I also would like to thank Noortje van Heijst, for being my company buddy and for reading though and commenting on my thesis. Last, appreciate the help of my family; Lotje Bosch, Robbert Bosch and Willemijn van Tongeren for the support and also reading and commenting my thesis.

Kees Bosch, Utrecht, 2022

List of Acronyms

Below is the list of acronyms used throughout this thesis listed in alphabetical order:

DSRM	Design Science Research Methodology
F	Functional Requirement
HITL	Human-in-the-loop
II	Iterative Imputer
IP	Intellectual Property
IR	Information Retrieval
IS	Information System
KNN	K-Nearest Neighbor
mAP	mean Average Precision
MCC	Matthews-Correlation Coefficient
ML	Machine-Learning
N	Non-functional requirement
RQ	Research Question

Nomenclature

Below is the nomenclature of parameters, and variables that have been used throughout this thesis.

Parameters

tp	True Positives
fp	False Positives
tn	True Negatives
tn	False Negatives

Variables

Q	Number of tests
q	Sample test

Contents

List of Figures

List of Tables

1

INTRODUCTION

1.1 Unknown Group

"Unknown is an early stage venture capital and business development firm. A Venture Engine supporting founders and industry leaders to successfully bring innovations to the market"[1]. The company is a merger between a corporate scouting firm and a venture capital firm. Unknown strives to make impact by: (i) Connecting Beautiful Ventures[2] to corporates. (ii) Investing in- and incubating these startups their-selves. (iii) Organizing events that act as a steppingstone for promising early stage startups. Unknown has the following branches:

- *Investing,* where they act as a venture capital firm, maintaining their own portfolio of startups. They also act as an incubator to accelerate the growth of these startups.
- *Scouting,* where corporate clients knock on the door for expertise in startup scouting. Unknown Group possess a tight relation network with high value startups and scale-ups all over the world. Corporate clients outsource the quest for investing and partnering opportunities to Unknown Group.
- *Events,* where Unknown organizes activities for startups for growing traction. The apple of their eye being 'Get in the ring', where startups literally step in the boxing ring to challenge each other by pitching their company.

Recently, Unknown Group started a tech department to create the 'Intelligence Portal'(i.o.w. 'Scouting Portal'), for more efficient startup scouting for its investment and scouting branch. This platform acts as a business process supporting portal, that can be used to search through different startup databases by applying hard filters such as *Industry, Foundation year and Region*. The platform creates startup and founder profiles that can be added to lists and exported to 'SalesForce'[3], so the scouting team can start a 'scouting campaign'. The platform currently acts as a tool to collect and sort all the available data and facilitate easy export and synchronization of this information.

In future plans, the platform should be further developed to incorporate or assist even more business activities, for example:

1. Incorporating ongoing dialogues with the founders of the startups to make sure the data records are more complete and up-to-date.
2. Making the scouting process less complex by shifting the campaigns and other functionalities from 'Sales'Force' to the scouting portal.

[1] www.unknowngroup.com

[2] Startups that do good for the world.

[3] 'An integrated CRM platform that gives all your departments a single, shared view of every customer.' https://www.salesforce.com

3. Automating the standard information in the insight reports[4] to reduce workload extensively.

Even tough these opportunities remain on the agenda, the focal point of this thesis is to make the scouting process of Unknown more data driven.

1.1.1 Matching Cycle

Unknowns' startup scouting branch consists of a continuous matching cycle as is presented in 1.1.

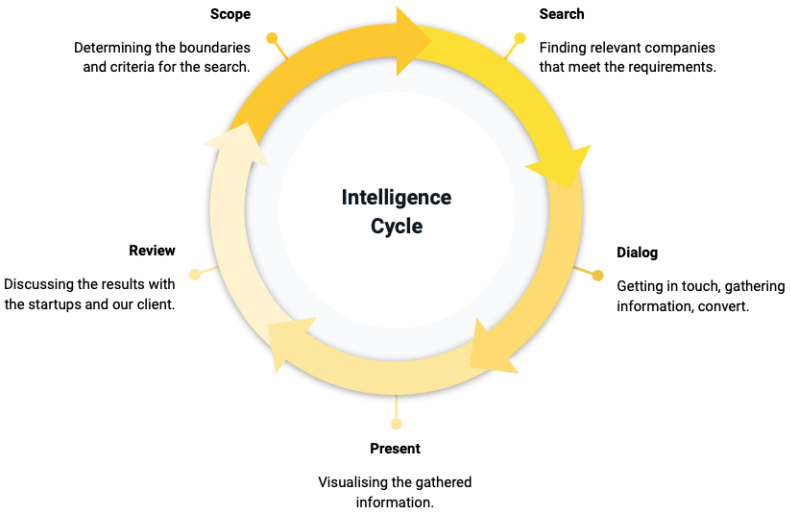


Figure 1.1: Continuous Matching cycle.

Together with the client, the search objective starts with setting the scope, which first remains quite wide. This scope is used to set 'filters' for the search query through the database. For example; "All companies in new energy". As a result, the amount of companies is reduced. The next step is to retrieve more data about the remaining companies in order to evaluate the results and create a selection that can be presented to the client. This data enrichment can be achieved with outreach to the founders directly or through internet research. The resulting selection will then be reviewed and depending on the desires of the client, the scope will be refined, hereby restarting the cycle.
A schematic presentation of reducing the startup pool is shown in Figure 1.2.

1.2 Motivation

In the last years, Unknown Group has grown rapidly, However, due to inefficiency in various business processes, maintaining fast growth by expanding in workforce with hiring new employees will no longer be a solution. Especially in the scouting process, the efficiency in various links in the chain could be significantly improved.
Additionally, because public data holders like Crunchbase[5], SpokeIntel[6], Owler[7] provide more data

[4]The product Unknown offers to their clients in which all valuable startups are extensively analyzed
[5]https://www.crunchbase.com/
[6]https://www.spokeintel.com/
[7]https://www.owler.com/

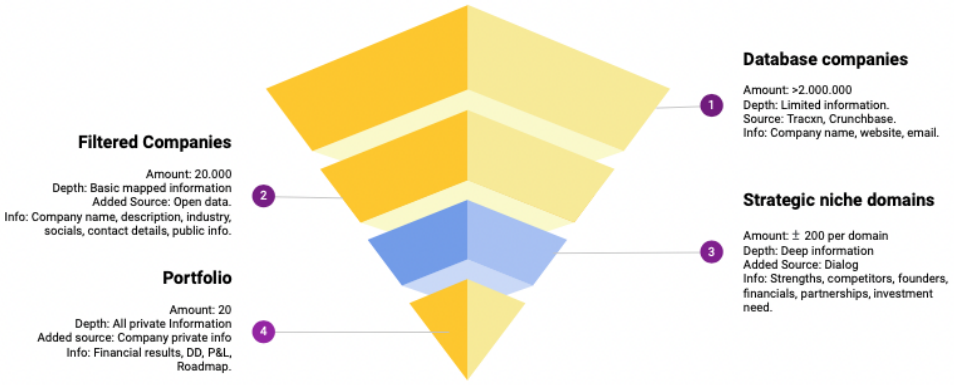


Figure 1.2: Startup Data pyramid.

about startups and their fund-raising records and because technology enables rapid computation and advanced analytical power, it becomes more feasible to develop data oriented analytical approaches for startup validation[62]. To maintain the competitive advantage, Unknown Group desires to innovate her business processes to increase the quality of her service. Furthermore, considering the time intensive startup evaluation process; Reorganization of this process, by incorporating data driven decision making, could be a solution.

Direct stakeholders that come into picture for this project are; (i) The board of Unknown Group, who is the commissioner for the project. (ii) The Scouting Department, who will be the user of the eventual solution. (iii) The tech team (in particular me), who is the executor of the project. Indirect stakeholders are; (i) The sales team, that can sell the improved services of Unknown group and wants to maintain unique value. (ii) The Clients, that will drive the fast growth and will benefit from increased quality. (iii) The competitors who drive the need to maintain the competitive advantage. A schematic representation of the stakeholders, drivers and the goals is shown in Figure 1.3

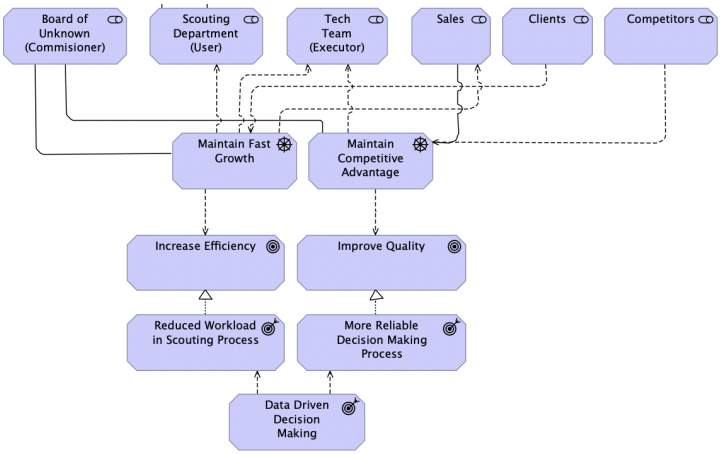


Figure 1.3: Stakeholders, Drivers and Goals for data driven decision making.

Inefficiency and time intensive labor are caused by various problems in the scouting process. The matching cycle from Figure 1.1 can also be visualized as a linear pipeline as shown in Figure 1.4. The amount of startups is gradually reduced, till they can be gathered into a final insight report and presented to

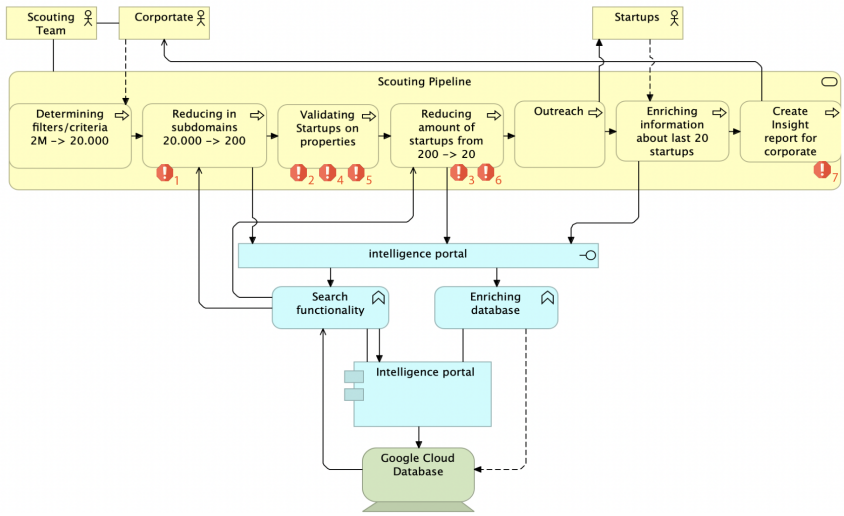


Figure 1.4: Current Scouting Process with Identified Problems.

the client. First, the search query is defined, hereby determining the hard filters for the primary search. Examples of these filters are; Industry, foundation year, region of foundation. By applying these filters, the amount of startups is heavily reduced; from 2 million to about 20.000 startups. *For example in the industry 'Energy', between 1990 and 2021 and in Europe.* The next step is to further specify the search quest to a strategic niche domain, for example *New Energy*. The amount of startups is then reduced from 20.000 to only 200. Then, from these startups which do all fit the desired characteristics, there must be determined which startups are the most valuable. This selection process is very time consuming and requires a lot of human labor[8]. This evaluation is based on detail level, where determinants as 'the team', 'the technology', 'their traction' and 'the competitors' are decisive[9]. Once the 200 companies have been narrowed down to about 20 high value startups, Unknown Group enriches the data of the startups by doing web-research and more important 'outreach' to the founders of the startups. In this process the founders of the startups are asked to supply a lot of additional information, which will be used to create the insight report that can be presented to the client. Additionally, the data will be stored in Unknown Groups' database to anticipate for future search objectives in the same domain.

As can be seen in the figure, this process suffers from eight problems that have bee identified during meetings with the the COO, the head of Scouting and the CTO of Unknown Group.

	Problems
1	Too much data.
2	Unreliable scouting process due to biases.
3	Inconsistent and incomplete data.
4	Scouting team is changing fast (many interns and a steep learning curve).
5	Lack of technical skills and knowledge.
6	Hard to rank different companies.
7	Time consuming. Not directly presentable to clients.

Table 1.1: Current Scouting Process with Identified Problems.

[8]Based on an interview with the Head of Scouting
[9]Based on the on-boarding course for scouts at Unknown Group

1. At first, because the database contains over 2 million startups, the scouting team is forced to roughly cut down the amount of startups. This is a logic process because the client will have some hard requirements that the startups must meet. However, this can cause promising startups to slip through these selection criteria.
2. Because the selection procedure is conducted based on human decision making, it is prone to various biases. As a result, high value startups could be overlooked or on the contrary, less value startups could be over rated.
3. The startup database contains high inconsistency in the data records of the startups. A lot of startups only contain name and foundation year.
4. Because the scouting team consists of a lot of temporary workers, the occupation of the team changes rapidly. This is a problem, because the learning curve for effective scouting is rather steep.
5. Even tough these scouts eventually know how to evaluate companies based on the standard scouting procedure, they often lack technical affinity. They therefore do not always recognize outstanding innovations.
6. Ranking startups is sometimes comparing apples to oranges as they are in a completely different domain, exploiting in completely different technologies.
7. The biggest problem is that the whole process is very time consuming. The scouting process is the most labor intensive process at Unknown group[10]. Manually filtering and evaluating all the startups takes months. As a result, the feedback loop with the client can sometimes be too long. It is desirable to have a solution that enables instant demonstration.

1.3 Scope

Unknown operates at a wide spectre of activities and because it is too extensive to target all these activities in this thesis, it is important to determine the scope. This thesis strives to make the scouting process more data driven. Although both investment as corporate scouting include a scouting process, this research will focus on the corporate scouting process. Though, findings from this research can probably be expanded to the investment branch.
Each step in the traditional startup investment process[26] includes decision making activities[53], the pre-investment screening phase appears the biggest driver for returns. Therefore, applying a data-driven decision system in this phase, is a point of interest. The pre-investment screening phase consists of the deal sourcing step (Figure 1.5) and the deal selection step which could both benefit from a more quantitative data driven decision method[53].

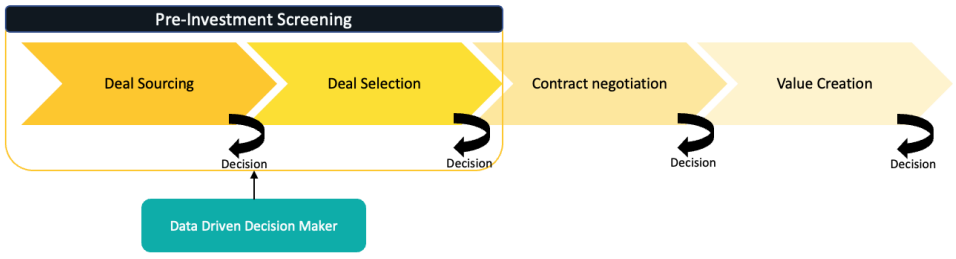


Figure 1.5: Pre-Investment phase and potential for data driven decision maker.

[10] According to the head of scouting

Subsequently, most problems from the scouting process (2,4,5,3,6 in Figure 1.4) have been identified in business processes that belong to this *deal selection* step: Validating the startups and cutting down the 200 startup that fit the filter criteria, to 20 valuable startups. Therefore, these two business processes make up the scope of this research. This means that the system will work with data in one specific strategic niche domain (Figure 1.6). Because Unknown group possesses significantly more data in certain focus areas, there is chosen to focus on the niche market 'New Energy', which are "companies that could contribute to a lower-carbon energy system"[11]. Hereby, the first two steps of the scouting pipeline have already happened, resulting in a remaining amount of about 200 startups. Later, once the solution has proven its functioning, it can be expanded to cover more parts of the scouting process.

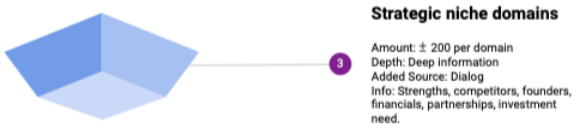


Figure 1.6: Strategic niche domains from the Startup data pyramid.

1.4 Proposed Solution

Together with the CTO of Unknown group, it is decided to explore the possibilities of implementing recommendation-decision support in the scouting process. In Figure 1.7, the current business process is alternated by incorporating the proposed solution: A recommender-system that enables data driven decision making. The first application process in this solution, has already been executed as the domain for the search has been determined upfront. The system will then validate startups based on their properties and reduce the amount of startups to a selection of 20 startups.

1.5 Thesis Structure

In the background research in chapter 2, different types of recommender systems, how to build and validate these systems and the challenges in implementing recommender-systems in this specific domain were investigated. This resulted in a design theory for a hybrid recommender-system for startup scouting. Once background knowledge is obtained, this thesis aims to turn the design theory in an artifact by holding on to the design science research methodology(DSRM). In chapter 3, the research question and sub-questions are established and the outline for the DSRM is explained. Chapter four aims to translate the design theory to requirements. It also adds requirements from the field, by interviewing the direct stakeholders. Furthermore these requirements are made measurable and goals are set for the solution. In chapter 5, the requirements are converted into design options. Additionally, constructs and classification metrics for the recommender system are selected and combined into a datamodel. As a result, a final design is proposed and compared with the design theory. In chapter 6, the conceptual design is turned into an application architecture, that will be used to create a prototype, which functioning will be tested in chapter 7. In this chapter, a five stage test is executed to measure the value of the solution. The

[11] https://www.shell.com/energy-and-innovation/new-energies/shell-ventures/new-energy-challenge.html

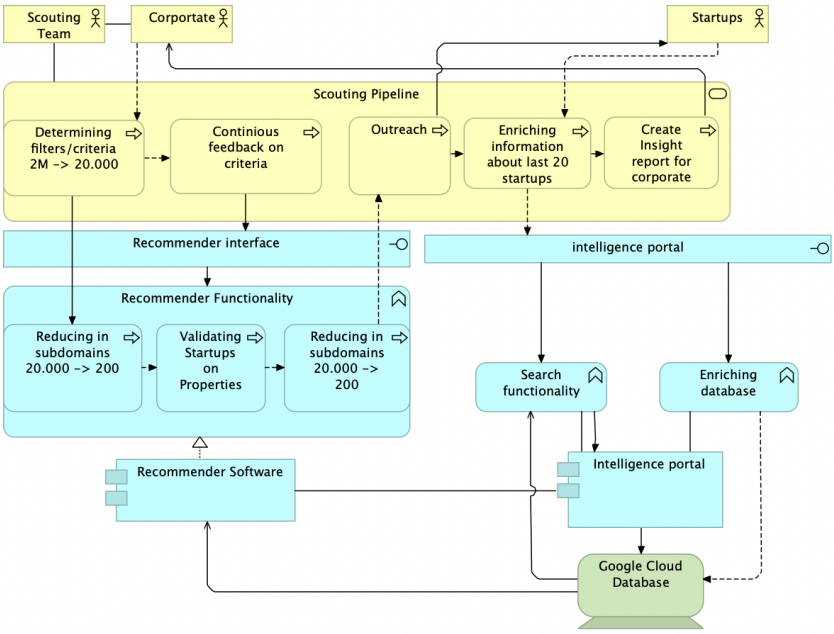


Figure 1.7: Proposed solution to overcome problems in the scouting process.

results from the testing will be discussed in chapter 8, leading to a conclusion on the research questions, problems and goals in chapter 9. In succession, this thesis will end with some future recommendations in section 10.

2

BACKGROUND LITERATURE

Because Unknown Group wants to explore the possibilities of implementing data driven decision making with use of recommender-systems, this chapter involves research in different types of recommender-systems, how these can be built and how these can implemented in the venture capital market. This chapter also includes an investigation on how to measure success of recommender systems. The full literature review[9] can be found in the document that is submitted along with this thesis. At the end of this chapter, a design theory for building a recommender system in the venture capital section is proposed. This framework will be the basis for the design and development of the recommender-system for the case of Unknown Group.

2.1 Recommender Systems

2.1.1 Types of Recommender Systems

Literature distinguishes between collaborative filtering, content-based filtering[5], knowledge-based recommendations and case-based recommender-systems.

Collaborative recommenders, which Arazy classifies as a social recommender-system, use associations with other users, based on the extent to which they share similar preferences. This is a more sophisticated method that depends on item ratings from other users and makes recommendations based on these rating scores[25]. Such a recommender-system is able to identify latent behavioral patterns that cannot be achieved with modeling metadata[63]. However, these types of recommenders are prone to suffer from the cold-start problem. A phenomenon that exists when social information about an item is sparse because it is new and has not been involved in any interactions with users[63],[33]. Secondly, such an algorithm could be hard to explain as it is based on associations and networks that are not easy to track down[30].

The content-based recommender-system uses similarities in the products and user profiles for personalized recommendations[5]. Content-based filtering is a query-based information retrieval system [25]. Such recommenders are resistant against the cold-start problem but are strongly depended on the quality of the metadata that is used to construct the system[63]. They are therefore often less accurate than collaborative filtering methods. Furthermore, content-based recommenders typically include newly added items (cold-start problem), but they fail to include recommendations that are outside the comfort zone of the user (myopia problem)[35].

Where a lot of literature confirm these to types of recommender-systems[50][45][31][2] Zibriczky[63] adds another two types of recommender-systems; Knowledge-based recommender-systems and case-

based recommender-systems.

Knowledge-Based Recommender-Systems (KBRS)[11], recommend items with the use of metadata. This metadata consists of user history and properties and relations of other available items (for example co-occurrence of items). A major advantage is that these type of recommendations are based on domain-knowledge and constraints that can be adjusted to the users preferences[63]. However, they do require more labor as they should be built up and maintained.

A fourth type of recommender-system that is mentioned in literature[63], is the Case-based recommender-system (CBRS) [36][52]. Such a recommender-system produces solutions to the problem, based on previous similar cases. They tend to fit better in financial domains than collaborative alternatives. However, such a method requires a lot of data, which is a problem for Unknown Group, as they have only just started collecting historical data. Therefore I will not consider case-based recommenders in this specific research.

Within recommender-systems one can distinguish between three 'knowledge sources'[13]: social (other users), individual(the user himself) and content(which are the items). These data sources correspond to the different types of recommender-systems. Figure 2.1 visualizes which of the recommender-types use each of the particular knowledge sources and highlight the overlap of the knowledge-based recommender with the content-based recommender-systems.

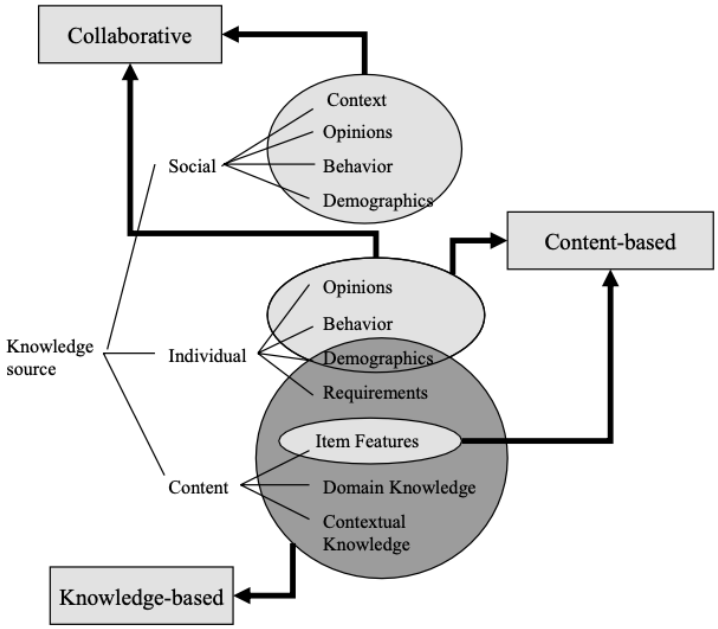


Figure 2.1: Knowledge sources and recommendation types by Burke and Razemani[13].

Combining multiple of these recommenders is called "Hybrid Recommender-Systems" [63][2][20][22]. These systems combine multiple decision support mechanisms[12], which often leads to enhanced advantages and reduced weaknesses hereby improving the recommendation accuracy[2][20][22][34][37]. Such methods are therefore more precise but the implementation can be rather complex[63].

The way a system is implemented and the extend to which it should replace human behavior is very case-specific. A two year long study by Grønsund and Aanestad.[27] investigated how human and algorithm augment each other. Many research[10][57][13][63] agrees upon that in the venture financing market, technology should augment human processes rather than replace them because of high risk involved. By applying technology to assist human processes, new human tasks emerge, like auditing and altering the algorithm[27]. Traditional manual work of classifying the data is now re-purposed to act as a baseline in validating the findings of the algorithmic approach[27]. Because human action was now synchronized with the process of the algorithm, this human-in-the-loop configuration resulted in a feedback mechanism that improved flexibility, increased accuracy and emend timeliness.[27]. A well implemented HITL facilitates learning for both user and machine, so they both grow knowledge about the domain[59]. Grønsund and Aanestad conclude with arguing that it is important to make deliberate choices regarding the division of labor between system and the humans. With a well-implemented human-in-the-loop configuration, sufficient performance of the algorithm with respect to the requirements, can be ensured. This is the result of learning through a feedback loop, enabling ongoing adaptions, extensions and improvement.

2.1.2 Building Recommender Systems

The different types of recommenders require different approaches.
The content-based part of the recommender could be built with a supervised machine-learning approach[54][1].
With a k-Nearest Neighbor clustering method[21][2], a top N recommendation could be delivered. However, this has only proved to work for one determinant. Literature[54] claimed that it is worth to add more constructs to such a system in order to improve the accuracy[54][33].
For the collaborative side of the recommender, the extended theory driven design framework for social recommenders can be used. In this framework, homophily[39], tie-strength[24][28], [56] and trustworthiness[38][5] are the added determinants that predict the willingness in accepting the recommenders advice[5]. These psychological constructs could be added in a model next to constructs like, among others; historical investment records of other venture capitalists and co-occurrences of startups in the portfolio of other venture capitalists[62][10]. Similar to calculations in the content-based filtering, a machine-learning clustering method can be used to deliver a top-N recommendation from all of this information[10].

2.2 Domain Analysis

The pre-investment screening phase of a venture capitalist (the scope of this thesis) typically involves various human biases. In this phase, data is too vast for simply shifting through all the possible options[57],[10] and because the current investing process of the venture capitalist sector is prone to various biases [18] (the over-confidentiality bias[53], the halo-effect[53], the representation bias[53], confirmation bias[53],[46], information overload[46] and local bias[46][17]), implementing a data-rich recommender-system could increase efficiency and reliability in this process[26][6][61][62].

As recommender-systems come in various options and the method of application of such system heavily depends on the characteristics of a market[54], the different options of recommender-systems are

[1] A method in which the input and output are known and thus the model can be calculated using machine-learning.
[2] A method in which the closest points in space are expected to be most similar.

analyzed based on the 6 aspects that characterize the venture capital market[13][29]. These aspects; *heterogeneity, churn, interaction style, preference stability, risk and scrutability*, are explained below in respective order. Additionally, *data quality and availability*, which are frequently mentioned in other literature[10] [57][62][54], are elaborated upon as well.

The more heterogeneous a data set becomes, the harder it gets to implement a recommender-system[13]. On the other hand, the more diverse a portfolio is, the more risk is reduced[62]. Contrary, some investors tend to prefer investing in startups that complement to each other. Furthermore, startups can be hetero- or homogeneous in a lot of different dimensions such as industry, maturity, popularity, geographical location etc[54]. To include all these social and content-based metrics, a hybrid approach seems to be an interesting option. However, the threat exists that the dataset becomes to diverse for accurate recommendation.

The startup market is ever emerging with the newest technologies and other ideas[13]. This involves both novel items that should be added, but also important novel information that relates to existing items such as rapid internal development of a startup or reorganization in the business structure[10]. Collaborative recommender-systems already include a mechanism to adapt to this item information change, as they are adaptive to change in portfolio of other investors. However, the other types of recommenders do not have built in resilience techniques. Continuously updating the recommender system with the newest items and its weights and values is therefore required.

A recommender-system tries to build a profile of the preferences of the user-item relation. The information for the user preferences can be obtained using either explicit or implicit user input[13]. Meaning actively providing information or requiring information by monitoring user-behaviour respectively. When collecting user-item data, one must be aware of the filter bubble[41], that exists when no new out of the comfort zone items are recommended[13]. This could be prevented with user-user collaborative filtering. By comparing the users profile to profiles of other venture capitalists, the possibility exists to find novel startups that are not particularly bound to the known preferences of the user. A hybrid recommender-system like this could be a solution to scarcity in explicit information from the user.

The venture capitalist market is often exposed to rapid change. This is caused by new items but also change in preferences of the user[13]. Emerging technology or global events such as the corona pandemic or climate change could influence the investors interests[8]. Although collaborative filtering can detect change by the behavior of others, change in preferences of one investor does not necessarily imply change in preference of the other. By letting the user specify their preferences through the human-in-the-loop approach, this problem can be solved[13].

Investing in startups involves a lot of risk[13][57]. This requires to combine the decision support mechanism with expert knowledge, rather than a autonomous application. [10][57][13].This can be achieved with a HITL mechanism. Furthermore, the psychology constructs of Arazy et al.[5] (section 2.1.2) could considered in increasing the users trust and willingness to accept the recommendation.

To be able to justify investment choices to their employer[13][62], investors should be able to ground their decisions. This must be incorporated in a recommender-system as well [40], hereby also increas-

ing the likelihood of accepting the advice[13]. However, collaborative filtering models are often hard to explain, as they are built on associations and network, rather than clear constructs[13]. It is therefore important to provide enough item-information for the scout to evaluate the company during the HITL process.

Even though the quantitative data in the venture capitalist sector is extensive[57][10], Zong et al.[62] mention that there exists a high scarcity in qualitative data of venture capitalists and this results in a cold-start problem, making it hard to derive useful recommendations by using existing IR techniques[54]. Boytcheva and Tagarev[10] agree with this statement and add that specifically for smaller companies, having the highest growth potential, the available information is sparse. As a result, automation in startup selection should be very sophisticated because it must withstand to incomplete information about promising companies.
Additionally, as venture capitalist targets startups in an early stage, their data quality is very inconsistent. A lot of databases do not posses the latest information and databases as 'Crunchbase'[3] and 'Tracxn'[4] rely on the startups' effort and reliability for providing the most recent and correct information[62], resulting in a lot of unreliable and missing data. Because of this reason, there was chosen to first implement data driven decision support in a specific strategic niche domain, as the data in these domains is more consistent and accurate.

2.3 Recommenders vs Domain Characteristics

The characteristics of the venture capital domain and how to interplay on these characteristics using the different types of recommender-systems and the human-in-the-loop mechanism, is summarized in table 2.1. Using a hybrid recommender-system with a HITL approach, all domain problems could be covered. Because the data processing methods of the different recommender-types are similar, the possibility of combining the model should be further investigated.

2.4 Validation Metrics for Recommender-Systems

This section investigates how to validate startups using a hybrid recommender system. From this section we will learn what kind of decision mechanism is suitable for this recommendation problem and which corresponding metrics should be used for performance testing.
Schröder and Thiele[47] stipulate the importance of setting the goal of the recommender system before the determination of the validation metrics. Additionally, they state that for setting the validation metrics, the domain of the recommender system also is an determining factor. They distinguish between four different classes, namely (i) Predictive accuracy metrics, (ii) Classification accuracy metrics, (iii) Rank-accuracy metrics, (iv) Non-accuracy metrics. Schröder and Thiele[47] provide a list of questions that should help to identify the goal of the recommender system and choose a proper validation metric. These questions are answered in appendix B. According to these guidelines, we are dealing with a classification problem and thus require classification accuracy metrics, that typically try to measure the successful decision making capacity of the recommender-system. In this technique, the amount of correct and incorrect classifications for relevant and irrelevant items is measured. Schröder and Thiele[47]

[3] https://www.crunchbase.com
[4] https://tracxn.com

			Recommender Systems (2.1)				
			Collaborative-Recommender	Content-Based Recommender	Knowledge-Based Recommender	Hybrid Recommender	+Human in the loop
Domain Analysis (2.2)	Domain Problems	Heterogeneity				X	
		Churn	X			X	X
		Interaction Style	X			X	X
		Preference Stability				X	X
		Risk		X	X	X	X
	Biases	Scrutability		X	X	X	X
		Over Confidentiality/ availability	X	X	X	X	
		Halo-effect	X	X	X	X	
		Representative	X	X	X	X	
		Confirmation	X	X	X	X	X
		Home/local	X	X	X	X	
		Horizon	N.A.	N.A.	N.A.	N.A.	N.A.
	Data properties	Affinity	N.A.	N.A.	N.A.	N.A.	N.A.
		Availability	X	X	X	X	
		Quality					X
		Consistency				X	

Table 2.1: Summarizing table venture capital domain problems and solutions by recommender-mechanisms(section 2.2 and 2.1).

state that this type of validation is useful when the goal of the recommender is to convince users to make a certain decision. Which is true in this particular case.

In classification models can be distinguished between linear classification and non-linear classification, depending on the distribution of the data as represented in Figure 2.2. With machine learning classifi-

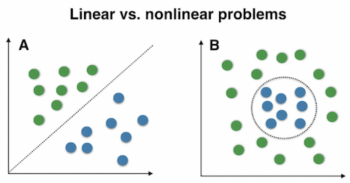


Figure 2.2: Linear and non-Linear Classification Problems [44].

cation, a line is drawn to split the data points into two or more classes. In linear classification, this can be achieved by drawing a straight line (or plane in a multidimensional problem). However, if the data points are clustered but the data cannot be split by a straight line (displayed in the right picture in Figure 2.2), non-linear regression is required [44]. In a Gitlab article, Jerm Watt[58] describes how linear and non-linear weight optimization can be executed for two-class classification problems.

The data model for this thesis is a non-linear two-class classification problem because the constructs

'knowledge', 'content' and 'collaborative' all depend on various other parameters making them non-linear parametric functions[58].

The formula for a non-linear classifier is shown below:

$$model(x,\Theta) = w_0 + f_1(x)w_1 + f_2(x)w_2 + \cdots + f_B(x)w_B$$

Where f_1, ... f_B etc are non-linear parametric or non parametric functions. In short, this can be written as

$$model(x,\Theta) = f^T w$$

The decision boundary consists of all inputs where $f^T w = 0$ and predictions are made with

$$y = sign(f^T w)$$

Optimizing the weights of the model can be achieved with minimizing the cost function:

$$g(w) = \frac{1}{p}\sum_{p=1}^{P} \log\left(1 + e^{y_p f_p^T w}\right)$$

The performance of such a classification model can be measured with the classification accuracy metrics. For this method the common basic information retrieval (IR) metrics are used[47], which can be summarized using a confusion matrix table (of which an example is displayed in Figure 2.3).

		True Class	
		Positive	False
Predicted Class	Positive	TP	FP
	False	TN	FN

Figure 2.3: Confusion Matrix.

Schröder and Thiele[47] discuss the *Matthews-Correlation Coefficient (MCC)* that combines this markedness and informedness. This metric relies on the geometric mean and the formula is given in formula 4.2. The range of the Matthews Correlation Coefficient is $[-1,1]$. This metric is used because it takes into consideration both the false positives as false negatives and is therefore not prone to biases, unlike basic IR metrics. The MCC returns a value of '0' when the recommender system has as much fails as it has successes. When it has more fails than successes, the value shifts towards '-1' and the more successes the higher the score. Therefore the algorithm can be validated and improved by using this metric. The higher the score, the better the performance of the recommender system.

$$MatthewsCorrelation = \frac{(tp*tn) - (fp*fn)}{\sqrt{(tp+fn)*(fp+tn)*(tp+fp)*(fn+tn)}} \tag{2.1}$$

$$= \sqrt{informedness * markedness} \tag{2.2}$$

MAP(mean Average Precision) is another metric for unbiased analysis of the confusion matrix, of which the formula[5] is given below (with Q being the number of tests). This metric indicates the rate of recommendations being correct.

[5]https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52

$$mAP = \frac{\sum_{q=1}^{Q} AveragePrecision(q)}{Q} \tag{2.3}$$

Eventhough the article of Schröder and Thiele stems from 2011, the MCC is still used[6] and frequently cited[4][15]. This makes this metric valuable for this thesis as it can be used to evaluate the recommender-system without bias.

2.5 Design Theory

Problems such as; continuous accretion of novel information, rapid changing preferences, and high risk of accepting recommendations, ask for a hybrid approach for implementing a recommender-system. Hereby, the downsides of both collaborative and content-based filtering are minimized. As is shown in the design theory in Figure 2.4, a hybrid recommender system for startup scouting can be created by combining content-based, knowledge and social constructs in one hybrid recommender-system and using machine learning non-linear classification techniques to produce a top-N recommendation. This recommendation requires a human-in-the-loop(HITL) validation before getting accepted, in which the investor's willingness to accept is based on the explanation of the collaborative recommendations using the three constructs for social recommenders[5]. Last, via the HITL-approach, preferences of the user can be adapted through an interface, hereby providing user-item information for personalized recommendations. The recommender-system can be validated and improved with the mAP and MCC metrics as described by Schröder and Tiele[47]. This design theory should make the scouting process more efficient and reliable. Reliable, because the selection of startups is now not only biased to the preferences of the user, as the recommendation also will include startups that are outside of the filter-bubble. Furthermore it solves most other biases from the traditional investing process. More efficient, because the vast amount of data can now be processed in a shorter amount of time and therefore be reduced to a smaller and more relevant selection, hereby partly automating the first step(deal sourcing) of the traditional investment process.

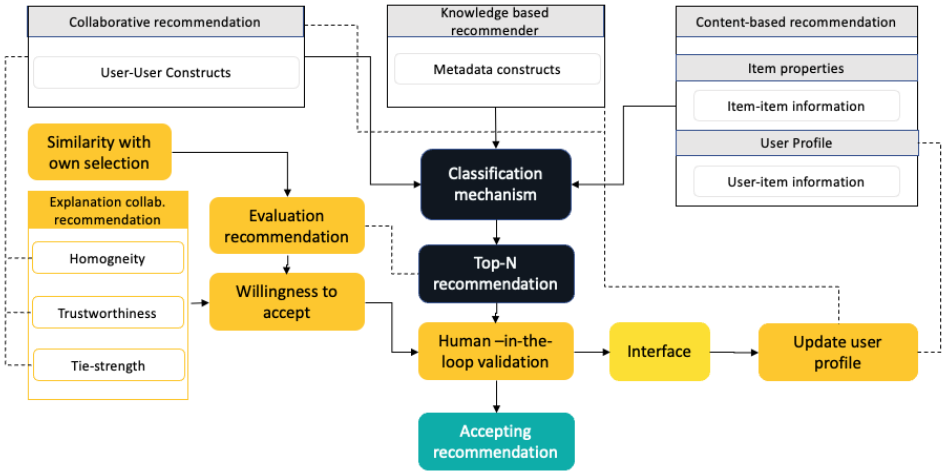


Figure 2.4: Design theory for a hybrid recommender-system for the venture capital domain.

[6]https://www.youtube.com/watch?v=u-Ez7trpNrM

3

RESEARCH DESIGN

Unknown Group wants to explore the possibilities of implementing a recommender-system in the scouting process for data driven decision making, in order to increase reliability and efficiency. Therefore the research question sounds;

- ***RQ 1:*** *Can the reliability and efficiency of the startup scouting process be improved, by implementing a hybrid recommender-system to validate startups in the strategic niche domain of 'new energies'?*

To be able to answer the question defined above, the following subquestions should be assessed:

- ***RQ 1.1****: What are the functional requirements for a data driven recommender-system that should assist startup scouting at Unknown Group?*
- ***RQ 1.2****: What are the non-functional requirements for this system?*
- ***RQ 1.3****: What knowledge, collaborative and content data can be used as an input for the hybrid recommender-system?*
- ***RQ 1.4****: What classification method should be used and how should it be created?*
- ***RQ 1.5****: How to deal with the domain challenges that are present in the venture capital domain?*
- ***RQ 1.6****: How can the effectiveness and reliability of the recommender-system be measured?*
- ***RQ 1.7****: Does the use of the recommender-system show improvement in quality and efficiency in the scouting process?*

Because this thesis involves the development of an information system artifact, it is important to build upon a suitable framework:

3.1 Design science research methodology

Because there was need for an appropriate framework for Information System (IS) Design Science(DS); Peffers et al.[43] have dedicated their research to the development of a design science research methodology(DSRM). The framework incorporates principles, practices and procedures that are required to carry out design science research.

The framework divides the design science process into six steps; problem identification and motivation, definition of the objectives for a solution, design and development, demonstration, evaluation, and communication (Figure 3.1). The entry point is variable, after which will be expanded to both sides. In this use-case it is the third point (because the artifact is partly known but needs further elaboration), making it a design and development-oriented solution.
In the introduction, various problems in the scouting process have been identified. Furthermore, the

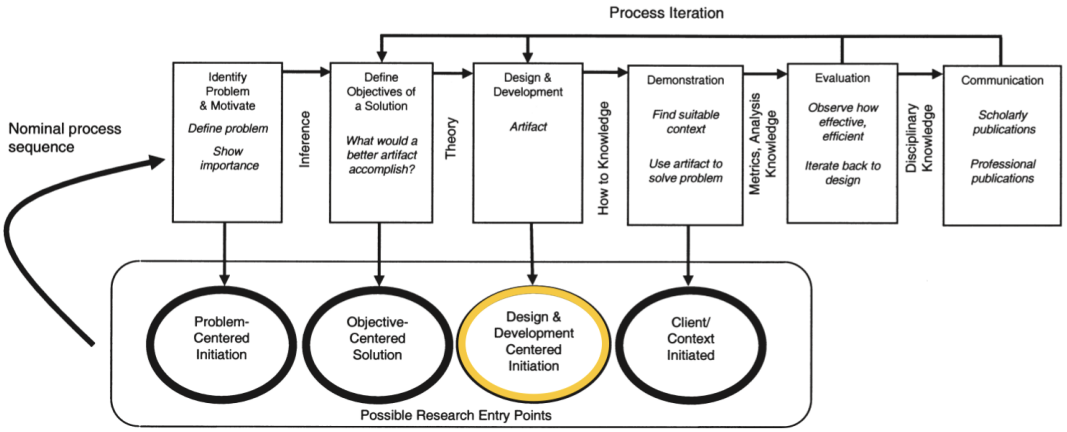


Figure 3.1: DSRM Framework by Peffers et al.

scope was set: 'Creating a data driven decision maker for validating startups and bringing down the amount of companies from about 200 to about 20 in the domain of new energies'. This means that problem 2-6 as identified in table 1.1 fall within the scope of this thesis. Additionally, a possible solution is proposed, that could supposedly solve most of these problems. In the theory (section 2), a descriptive product-oriented design theory was established on how to built the solution. Such a design theory translates into meta requirements[60], which will happen in chapter 4, this is the next step in the DSRM.

In this step, the design theory is translated into non-functional and functional requirements that have been evaluated and appended in collaboration with the direct stakeholders. Subsequently, metrics for validating the performance of the solution based on these requirements will be established in order to test the solution in a later stadium of this thesis.

In the next step, the requirements are converted into design suggestions that are collected in a final design(chapter 5). From this design, a prototype of the artifact can be developed(chapter 6). In order to do so, this design is translated to an application architecture.

The demonstration of the artifact is in fivefold. (i) First, the performance of the recommender system will be demonstrated by means of performance measures for recommender-systems, based on dummy data. (ii) The feedback structure will be tested and optimised by deriving the optimal threshold to facilitate personal recommendations. (iii) The design choices will be evaluated in a structured interview with the scouting team. (iv) The recommender-system will be tested by inserting real data from Unknown Group's scouting campaigns. (v) The solution will be evaluated based on the requirements. Subsequently, a research plan for evaluating the bias response of the recommender-system is given in the appendix F, as this could not currently be executed.

This evaluation conceptually includes all empirical evidence or logical proof. At the end of this step, the researchers face the decision to either continue to the last step DS process or to iterate back to step three in order to improve the effectiveness of the artifact. This phase is executed in in chapter 8 and 9. The further recommendations in section 10 provide suggestions for further iterations.

4

OBJECTIVES OF THE SOLUTION

In this chapter, the design theory that was established in the background literature is translated into requirements. Together with the direct stakeholders, these requirements were evaluated and appended. Furthermore, this section contains validation metrics that can later be used to measure the success of the solution.

4.1 Requirements Engineering

The design theory from the literature was translated to an initial list of functional requirements. This list, along with the identified problems, goals and motivation was proposed to (i) The COO of Unknown group, that represented the strategic organ of Unknown group, (ii) The Head of scouting and (iii) The CTO. In these interviews, the design theory requirements were verified and mapped to the problems and goals and missing requirements (mostly non-functional) were added. A motivational view of these stakeholders, goals and the related requirements are show in Figure 4.1. These requirements, belong to three core principles that emerge from the goal to implement data driven decision making[1].

- Data should be used to choose startups.
- The system is an addition to the current business activities.
- The system should be reliable.

4.2 Non-Functional Requirements

The requirements can be found in table 4.1. This table distinguishes between *functional requirements* and *non-functional requirements or constraints*. The functional requirements describe the features and what the artifact does, whereas the non-functional requirements define how the system is supposed to work[7]. At the top of table 4.1, the non-functional requirements are defined (visualized as constraints in the schematic representation in Figure 4.1).
Non-functional requirements:

- *N1 - Restricted Accessibility* and *N2 - Data Security* are met, once the recommender-system is implemented in the already existing scouting portal, as last mentioned already contains measures that prohibit unauthorised access and to warrant data security.
- For *N3 - GDPR*, the infrastructure and security is in place but because the recommender-system uses personal preferences to optimize the recommendations, it is important to adhere GDPR regulations. Though, the data that is used is not related to any personal information[1], making this requirement no problem for now.

[1] This was verified with the direct stakeholders.

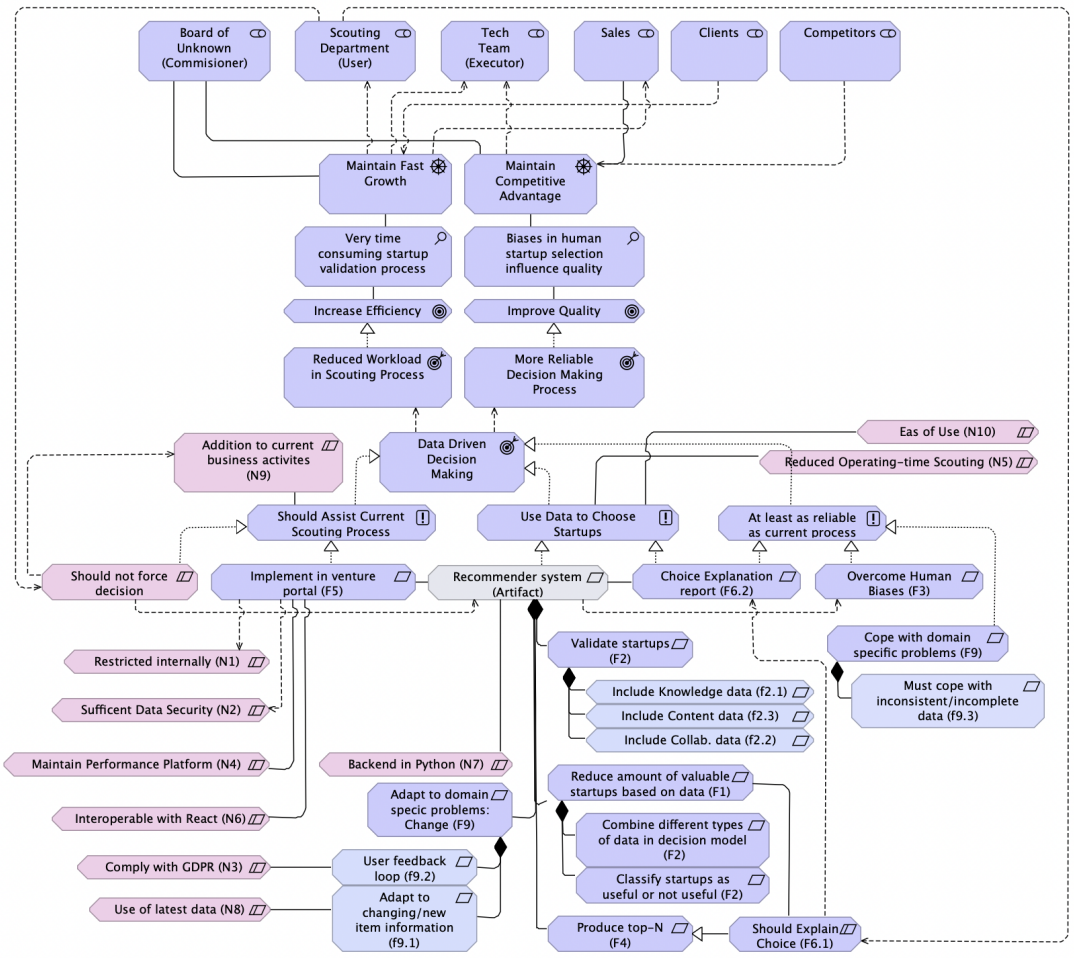


Figure 4.1: Functional and Non-functional requirements from the design theory, appended and mapped to the goals and drivers from the direct stakeholders(section 1.2).

- *N4 - Performance* Because the website renders asynchronised, loss in performance due to the new component will not cause trouble in rendering the rest of the platform. When the recommender system is implemented in the portal, this async structure is automatically in place.
- *N5 - Operating time*, involves the efficiency of the scouting process. To increase efficiency, the systems' operating time should be faster than the current scouting process. According to the head of scouting, this part of the scouting process can take months.
- For *N6 - React*, the interface of the recommender-system must be inter-operable with React[2]. This is met, when the new functionality can successfully be integrated in the scouting portal.
- Currently, all data modification and query based features of the intelligence portal are written in Python[3]. Therefore, the software engineer at Unknown Group has asked me to write the back-end in this language (*N7 - Back-end*). This, to support easy query based data integration with Google BigQuery[4] (where the data is stored), and because integration with the platform can be achieved by simply attaching it to the API of Google FireBase[5].
- *N8 - Enhance vs Replace* involves the application of the new scouting method in the current busi-

[2]https://reactjs.org
[3]https://www.python.org
[4]https://cloud.google.com/bigquery
[5]A realtime database https://firebase.google.com

	Non-Functional Requirements	ID
Non-Functional	The accessibility of the system should be restricted to people inside the organization.	N1
	Data security should be warranted.	N2
	While storing user data for personalized recommendations, the system must comply with GDPR requirements.	N3
	The system should not significantly reduce the performance of the scouting portal of Unknown Group, when integrated.	N4
	The operating time of the system should at least match the time it takes to scout startups manually.	N5
	The interface of the system should be intraoperative with React as the scouting portal is built in this framework.	N6
	The backend of the system should be built in python to support integration with BigQuery (where the data is stored)	N7
	The system should assist the current business activities instead of replacing them.	N8
	The system should be easy to use by the scouting team of Unknown.	N9
	Functional Requirements	**ID**
Functional (high-level)	The scouting team of Unknown Group must be able to scout startups in high efficiency.	F1
	The output of the recommender-system must evaluate and classify startups based on data.	F2
	The recommendations must be reliable and human biases in startup scouting should be reduced.	F3
	The output of the recommender-system must include a top-N with suggestions of startups that should be further evaluated by the scouting team.	F4
	The users should be able to use the recommender-system through an interface in the scouting portal	F5
	Adhering the recommendation must be groundable. It should contain an explanation for the choice.	F6
	The recommender-system must target startups in the strategic niche domain; New Energy	F7
	The recommender-system must recommend startups based on the most recent data.	F8
	The system must adapt to domain specific problems as identified in the background theory.	F9
Functional (low-level)	Must analyze startups based on knowledge data.	f2.1
	Must analyze startups based on content data.	f2.2
	Must analyze startups based on collaborative data.	f2.3
	Must combine knowledge, content and collaborative data in a decision model.	f2.4
	Must classify startups as useful or not useful.	f2.5
	The value of each metric must be visible.	f6.1
	Comparing the metrics should be possible through the platform.	f6.2
	Must adapt to new items.	f9.1
	Must adapt to new information about the user.	f9.2
	Must cope with inconsistent and incomplete data.	f9.3

Table 4.1: General Requirements Table.

ness activities. Because the system is new and the risk in the venture capital domain is very high, it is important to use the system as an advisory tool that should be used parallel to current scouting process.

- The system must be easy to use for scouts (*N9 - Usability*), who are often highly educated but not necessarily technically. A full usability study does not fit within the scope of this thesis, but the design should at least be usable, which should be evaluated with the scouting team.

4.3 Functional Requirements

In the section below, the functional requirements will be specified. This leads to design suggestions that can be merged into a final design in chapter 5 of this thesis.

- **F1 - High Efficiency**: Manual scouting times can be decreased by using data driven decision making. Instead of selecting the startups (which can take months), the scouts now only need to

validate the recommended startups.

- **F2 - Classify Based on Data**: The design theory (Figure 2.4) requires a combination of knowledge, content and collaborative data in a classification algorithm. The knowledge of the scouting team of Unknown Group in combination with literature about success determinants for startups, can be used to establish such a data model in section 5.2.1. Following from the onboarding course of the scouting team, startups should be evaluated by their team, the technology the traction and achievements, the competitive advantage and financial records. These constructs align with the different data types as identified in the theory (section 2.1.1):
  - **f2.1 - Knowledge Data**: *team*, *traction*, *technology* and *competitive advantage* involve metadata constructs and thus this can be seen as knowledge data. Modeling this data will say something about the characteristics of the item.
  - **f2.2 - Content Data** involves user-item and item-item relations that are not dependent on generic knowledge and can be extracted from user behavior. Theory distinguishes between implicit and explicit user behavior. Implicit user behaviour could be retrieved from matching startups based on similarity with previous successes. Explicit user information can be collected by evaluating the relevance of the recommendations, by letting the user classify each of the startups in the top-N as 'useful' or 'not-useful'.
  - **f2.3 - Collaborative Data** determinants include financial information as funding records involves information about 'other users' in the field. Another metric for other user behavior are trends[6]. These are examples of collaborative data that do not require information about other users within the platform. This is particularly useful for use-cases like the one in this thesis, where the amount of internal users is very limited, prohibiting valuable predictions based on internal data.
  - **2.4 - Combine Data Types**: Another requirement that follows from the design theory: Combining the different data-types into one system, which can be achieved by a machine-learning classification system that is trained with succeeded campaigns. This is the prior step to the following requirement:
  - **f2.5 - Two Class Classification**: To determine which companies are 'useful' and 'not-useful' a threshold must be investigated that determines this class. With supervised machine-learning, the optimal threshold is extracted from previous cases.
- **F3 - Reliability**: As we learned from the literature, combining *knowledge, content and collaborative* data should supposedly reduce the human originated biases and this can be tested with the test plan in appendix F.
- **F4 - Top-N**: The output of the recommender-system must be a top-N recommendation list, which acts as an advice, parallel to the traditional scouting process. This can be achieved by creating a separate page with recommendations in the scouting portal.
- **F5 - Implementation Scouting Portal**: The output of the recommender-system (the explanation report and the top-N recommendations) should be accessible in the intelligence portal of Unknown Group. This portal is built in React JavaScript language. In order to comply with this requirement, a functional component[7] should be created in JavaScript React, so it can be directly implemented in the intelligence portal.

[6] https://www.gartner.com

[7] https://reactjs.org/docs/components-and-props.html

- **F6 - Explainability**: From the interviews with the stakeholders was found that adhering the advice of the recommender-system must go with grounded argumentation. Therefore, the head of scouting suggested to create a report of all the metrics that were used to validate the particular startups*(f6.1)*. This is also in coherence with the sociology constructs as were identified with arazy et al.[5], that were included in the design theory. In response, I propose to implement the possibility to download or view such a recommendation report in the intelligence portal*(f6.2)*.
- **F7 - Strategic Niche Domain**: According to the design theory this should limit problems in data availability because data about startups in such niche domains should typically be more complete. This requirement particularly depends on the process prior to this recommender-system. Resulting from the interviews with the stakeholders, the purpose of the recommender-system is that it should later be universal over all strategic niche domains of Unknown Group. Therefore, the customization of the algorithm for this niche domain should be minimal. However for small customization to support case specific priority, a feedback loop could be created. Furthermore, training a machine-learning model using data from the same domain will make the classification model recognize specific characteristics of the domain.
- **F8 - Recent Data**: Adapting to rapid changing information also follows from the design theory. However, the data in the current platform is only updated every month. Therefore, there is no need for a shorter refresh frequency in the recommender-system. For user information however, recalculations could be triggered when the user accepts or declines certain recommendations.
- **F9 - Domain Specific Problems**: Following from the design theory, building a hybrid recommender-system with a human-in-the-loop mechanism should solve the domain problems:
  - **F9.1 - New Items**: Because the startup domain is changing rapidly, the recommender-system must continuously adapt to new information, requiring frequent recalculations. The cold-start problem that was identified in the literature section will not be a problem because the recommendation is not based on other users in the system (f2.3).
  - **f9.2 - New User Information**: To make the system resilient to new information about the user, the earlier explained human-in-the-loop (section 2.1.1) functionality should be added. In an interface, the user can interact with the recommendations, hereby specifying his preferences by giving explicit user information. In the future, implicit user behavior could be monitored as well, as is explained in 10.4.1.
  - **f9.3 - Inconsistent and Incomplete Data**: There are solutions for handling missing data in data sets. However, I don't think the perfect solution lies within the recommender-system but by enriching the data by increased startup involvement or successful scraping from websites as 'Linkedin'[8], 'Twitter'[9] or the features as described in 10.2.2. This is a future project that falls outside of the scope of this thesis.

4.4 Validation

In chapter 5, these requirements will be converted into a design for the solution. To be able to evaluate the success of this solution, validation methods should be created. Some of these functional requirements can be checked off during an evaluation with the scouting team, such as the implementation in the scouting portal, the target data from the strategic niche domain 'New Energy', the top-N and the inclusion of

[8] www.linkedin.com
[9] www.twitter.com

an explanation report. Others can be evaluated with the tech team. For example the response to new data and the classification technique. However, the ones that are elaborated upon below, are more difficult to measure.

Even though the recommender-system should eventually help in selecting value creating startups, this cannot be validated within this time-frame. However, we can measure the recommenders capability to overcome the main problems of this research; inefficient navigation through the vast amount of data and the unreliability of the investors choices in startup screening, due to biases. First mentioned can be measured, by comparing the time to produce a top-N selection to the time it takes an investor to make a selection of investment opportunities.

$$PercentualdecreaseScoutingTime = \frac{newTime - OldTime}{OldTime} \tag{4.1}$$

The ability to overcome the investors biases, is more difficult to measure. The recommenders' capability to avoid each bias should be tested individually according to the test plan in section appendix F.

4.4.1 Validation of Recommender system

As was explained in the background research, the Matthews Correlation Coefficient (MCC) will be used for measuring the performance of the Recommender-System. This metric relies on the geometric mean and the formula is given in formula 4.2. This is an unbiased metric for evaluating the performance of a classification system.

$$MatthewsCorrelation = \frac{(tp*tn)-(fp*fn)}{\sqrt{(tp+fn)*(fp+tn)*(tp+fp)*(fn+tn)}} \tag{4.2}$$

$$= \sqrt{informedness*markedness} \tag{4.3}$$

The range of the Matthews Correlation Coefficient (MCC) is $[-1,1]$. An MC close to 1 indicates strong agreement with the actual classification. Whereas an MC score of -1 indicates strong disagreement.

In the next chapter, these requirements and design suggestions will be converted to a final design. At the end of next chapter, this design will be reflected upon with using the design theory that was drafted in the theory section.

5

DESIGN

5.1 Global Level Design

This chapter describes a design 5.1 that originates from the requirements that followed from the design theory in Figure 2.4 together with requirements that came from the goals and principles form the direct stakeholders. This section will also include identification and assessment of the constructs that can be used for predicting startup success. These constructs are collected in a datamodel that will act as the decision engine of the recommender-system. The coherence of the design and the design theory will be discussed in section 5.3.

As can be seen in Figure 5.1; The Data from different data sources enters the recommender-system. The system calculates scores and classifies them as a list of 'N' useful startups. This top-N list is the output along with a report with detailed explanation on how these startups ended up in the list. This list is presented in the intelligence portal. In this interface, the user can interact with the recommendation. The user behavior is explicitly retrieved using evaluation of the choices. This user data is then used by the system to improve the recommendations.

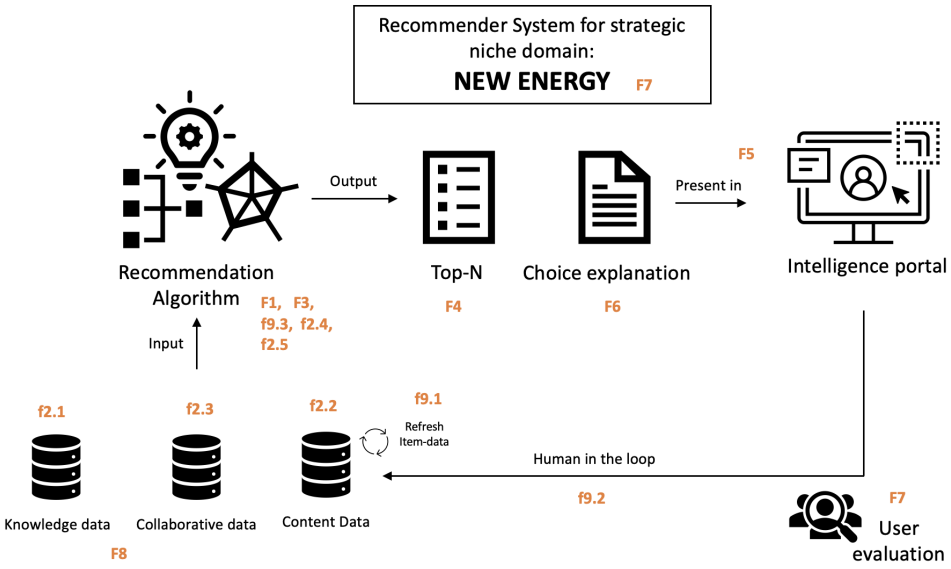


Figure 5.1: Global level schematic design.

5.2 Detailed Design

5.2.1 Data model

The head of scouting supplied the onboarding course of the scouting team, in which startup success determinants are explained. The onboarding course divides these determinants into five categories; *team, technology/product, traction/achievements, financials and investments and competitive advantage*. These categories were mapped to the datatypes from the design theory (knowledge, content and collaborative data). Five papers[51][49][32][48][14] have been used to confirm these five categories and complementary constructs. Constructs that are identified by more than one of these papers, but are currently not covered in the onboarding course, were added to the table (table 5.1). The colors in the right column relate to the presence of data from the constructs and their relevance. These constructs can be captured in a datamodel as is shown in Figure 5.2. There is a lack of information about the constructs in grey, they are therefore incorporated in the conceptual model but will not be used in the development phase in this research. The same holds for the constructs in blue. For these constructs, there is data available in the database, but they have a different reason of not being incorporated in the development phase:

- We do possess information about the roles of the team members but it is hard to map these on a measurable scale. Moreover, different branches require different team members, therefore this metric is left out of the model.
- We do possess information about the identity of the investors. However, due to lack of information about the specific type of investor, it is hard to distinguish which of these investors are venture capitalists.
- We can find similar companies to represent the competitors. However, accurate competitor identification requires qualitative analysis. Developing a representative data driven alternative is a research topic on its own and could therefore better exploited in further research. As we do not have any usable data about the competitive advantage, this category is excluded from the resulting model.

The resulting model represents the recommendation engine in the system, of which the output will be a top-N (based on the classification) and a report (with explanation of the decision for classification). The weights and relations of this model are yet to be determined by means of a machine learning back-propagation, in which the optimal weights are investigated by supervised classification. This

5.2.2 Optimizing Weights and Constructs

The data model can be used as a neural network. With time-series data, the success of startups could be monitored. The progression path of previously succeeded or successful startups can be used for training the algorithm. Unfortunately, Unknown Group has not kept this data. Though, they do possess historic data about previous campaigns. Using supervised classification[1], the model can be trained with this data as well. However, not all human biases can be eliminated because the model is trained based on previous human decision making. Nevertheless, this is a good manner to optimize the weights of the constructs[58],[55] and the algorithm can be continuously improved by the human-in-the-loop algorithm. I advised Unknown Group to start collecting time-series-data to improve the system in the future[3]. This will be explained in more detail in section 10.3.1 in the further recommendations.

[1] A ML method in which the relations of constructs are optimized by knowing the outcome of the system[16].

Category	Construct	Sources	Available in database
Team	Education	UG,Ki,Sc,CM	Y
	Prev. started comp founders	UG,S,Ki,Sc,CM	Y
	Experience in the field (in years)	K,Sc,CM	N
	Num of founders	Sh,CM	Y
	Num Employees	UG,S,Sh,Sc,CM	Y
	Different roles	UG,Sh,CM	Y
	Eagerness/Passion	UG,Ki,CM	N
	Adaptability	UG,Ki	N
	Risk sensitivity entrepreneur	Ki,CM	N
	Preparation	Sc,CM	N
Technology / Product	Focus Areas	UG,S,CM	Y
	IP	UG,Ki	N
	Technology Ready	UG,Ki,CM	N
	Problem solving	UG,Ki,CM	N
	Trends	UG,Ki	Y
Traction / Achievements	Awards	UG	Y
	Revenue	UG,S	Y
	Relevant Clients	UG,Sc	N
	Num of News Articles	UG,Sh	Y
Financials & Investments	Funding Rounds	UG,S,Sh,Ki	Y
	Funding Amount	UG,S,Sh,Ki	Y
	Revenue	UG,S,Sc	Y
	Backed by VCs	UG,S,Ki,Sc	Y
Competitive advantage	Uniqueness	UG,Ki,CM	N
	Scalability	UG	N
	Market Potential	UG,CM	N
	Number of Competitors	Sh,CM	Y
Other	Age	UG,S,Sh,Sc	Y
	Mentorship/Alliances/B-Partners	Ki,CM	N

UG =	Onboarding course Unknown Group	S =	Singh[51]
sh =	Sharchilev et al.[49]	Ki =	Kim et al.[32]
Sc =	Schutjens et al.[48]	CM =	Camelo Martinez[14]

Table 5.1: Constructs from literature mapped to the categories and constructs from the onboarding course from the scouting team of Unknown Group and their presence in the database.

5.2.2.1 Handling NULL Values

When looking at the data from Unknown Group, it stands out that there is still a lot of data missing. Missing values can be categorized into three groups; *Missing Completely At Random (MCAR), Missing At Random (MAR), Missing Not At Random (MNAR)*[42]. Depending on the category, there is a best strategy of how to handle such NULL values. First, the category of the missing data should be identified[2].

- In MCAR, the chance of missing values for all observations is completely the same. There exists no relation between the missing values and any other direct or indirect value. Common causes are human errors, malfunctioning of equipment or loss of sample. This occurs very rarely but it has the advantage that statistical analysis remains unbiased.
- In MAR, the missing values are explained by the relation with one known variable of which the information is complete. There is a pattern between the missing values in a subset of the data. For example female persons that are more likely to not share their age in a survey. This type of missing values may result in a biased statistical analysis. In order to prevent such biased results, computing the NULL values is required.
- In MNAR, the missing values are explained by relation to unobserved values. In this case, there is a pattern in the missing data but the observed values cannot explain it. A common occurrence is people being reluctant in giving the information. Doing statistical analysis may result in bias.

[2]https://www.analyticsvidhya.com/blog/2021/10/handling-missing-value/

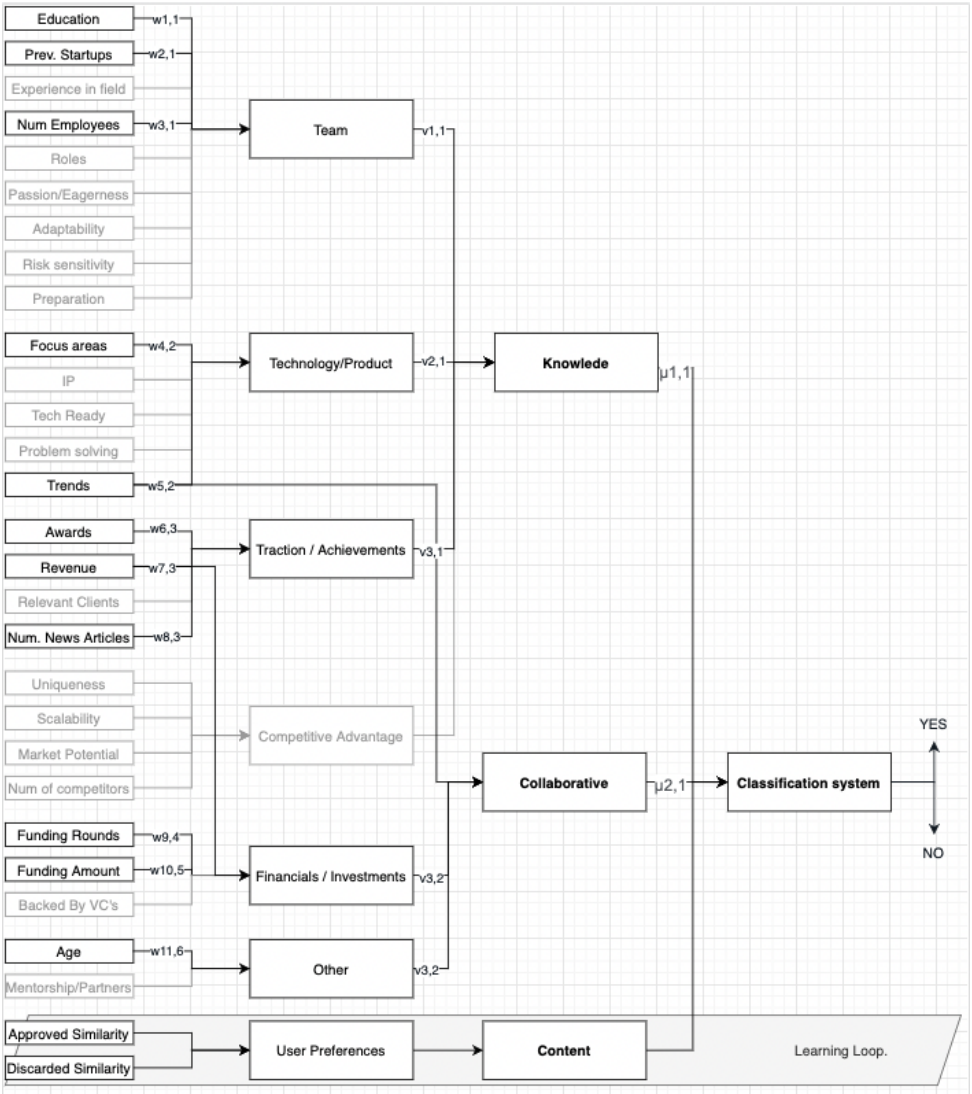


Figure 5.2: (Conceptual) Data model for the hybrid recommender-system. The learning loop is explained in section 5.2.6.

In this use case, two events may have lead to the missing data.

- The data is missing because the startup does not yet exist in all three databases.
- The startup does exist in the databases but not all information is complete.

Both these events can be caused by known and unknown variables. For example, their age may influence the fact that they have yet to complete their information or even apply for one or more of the databases. On the other hand, this could also be caused by the character of the founders. The NULL values can thus be classified as MAR or MNAR. This means that the missing data cannot simply be ignored.

Once the type of the missing data is identified, you can either delete the rows with missing values or impute the missing data[3]. The second option is the more reliable method, especially given the fact that a significant part of rows contain missing values. For imputing the data, the following strategies could be used; (i) Replacing with arbitrary value. (ii) Replacing with mean. (iii) Replacing with mode (iv) Replacing with median. (v) Replacing with previous value (Forward fill). (vi) Replacing with next value

(Backward fill). (vii)Interpolation (polynomial, liner or quadratic)[3].

The methods mentioned above, are examples of an *univariate approach*, in which only one feature is taken into account when computing the NULL values. NULL values can also be replaced using a 'Multivariate Approach' in which relations to other known values are used to predict the values of the missing entities. In the *Iterative Imputations*; using a regression model, the missing value is predicted based on the other features. In the *Nearest Neighbors Imputations*; the euclidean distance[3] is measured to find the most similar company. These values are then copied into the missing values.

Because our goal is to classify based on the features of the company instead of doing statistical analysis, a multivariate approach seems more suitable. The reason being that our machine-learning classifier will recognize patterns if all NULL values are replaced with the same new value (for example the mean, median or mode)[23].

5.2.3 Classification

As was described in section 2.4 we are dealing with a classification problem. It requires a two class non-linear classifier that can be measured with classification accuracy metrics like the Matthews-Correlation Coefficient and the mean Average Precision. Because the outcome of the test and training dataset is known, supervised machine-learning can be used to define the optimal split between the two classes. Using the neural network as described in section 5.2.2, the weights of the constructs are determined to draw the optimal line as can be seen in Figure 5.3.

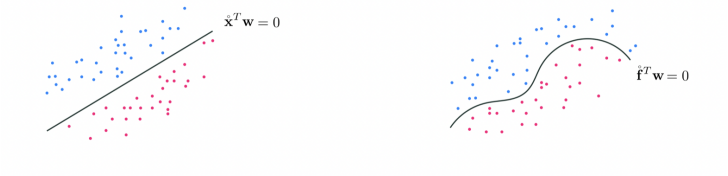


Figure 5.3: Drawing the line between Linear(left) and non-linear(rigth) classification [58].

5.2.4 Top-N

The top-N startup list will be represented on a separate page of the scouting portal. The amount of startups depend on the sample group. All startups that are classified as useful by the algorithm should be listed. The startup list will also contain an approval and discard button so direct feedback on the recommendation can be obtained. A quick sketch is represented in Figure 5.4.

5.2.5 Choice Explanation Report

When the scouts are interested in one or more of the recommendations, they are required to ground their choice for these startups. A conceptual sketch was created with an extensive dashboard on the back of each recommendation, which can be accessed with a button in the top-N window. This is shown in 5.5. The presence of sufficient information was verified with the head of scouting.

[3]In mathematics, the Euclidean distance between two points in Euclidean space is the length of a line segment between the two points. https://en.wikipedia.org/wiki/Euclidean_distance

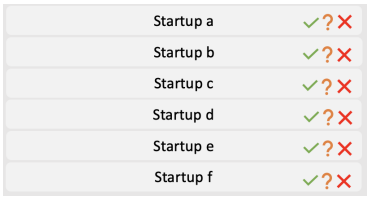


Figure 5.4: Top-N recommendation list with discard and approval buttons.

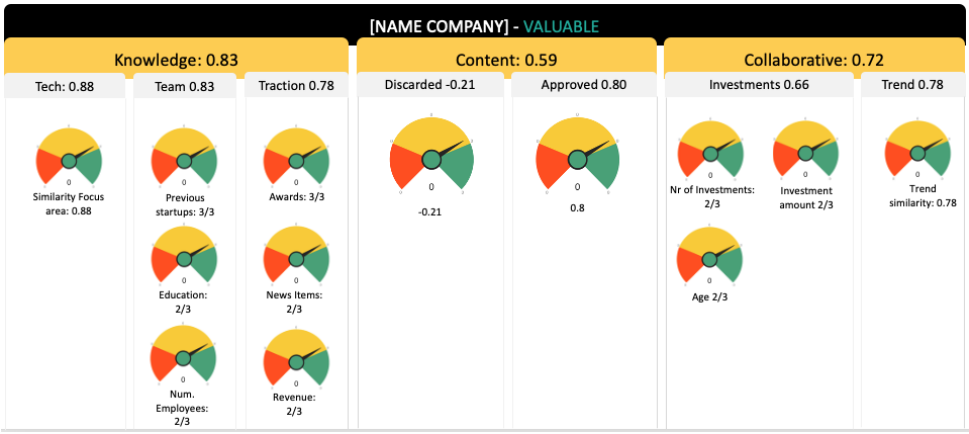


Figure 5.5: Choice explanation report from recommended startups.

5.2.6 Feedback learning loop

To avoid interference of the description being used for too many constructs (trend similarity, tech similarity, approved score and discarded score), there was decided to extend the classification system by another route. Another reason is that there exists a cold start problem for the approved and discarded score because preferences in previous cases do not imply case specific preferences in a new case. The classification system evaluates companies purely on knowledge and collaborative data, however by applying this different route, a content data learning loop can be created of which both machine and user will gain knowledge about the context[59]. The user specifies which recommendations are useful and which are not. By comparing the description of the companies to the description of approved and discarded companies, the preference of the user can be measured. Companies that score above a certain threshold with the approved companies and below a threshold for the discarded companies will be selected by the recommender-system, regardless of their recommendation classification. This way the user learns about new startups that fit their preferences. They in their turn can provide the system from feedback on the advice.

5.2.7 Intelligence Portal

The recommender system should be implemented as a new feature in the intelligence portal. This way, the scouts can access the system and it is automatically restricted to people from within the organization. Privacy measures that have been taken for the portal will also apply to the recommender system. In agreement with the front-end developer, the best way to achieve this is by creating a functional component in

React[4] in a feature branch[5], so this can be easily embedded in the platform.

5.3 Reflection Design Theory

As was mentioned earlier, the basic structure of the design is based on the design theory from Figure 2.4. In this section, the constructs for the design theory are identified with use of the onboarding course of the scouting team and five academic papers. These constructs could be categorized into the different data types, like the design theory suggests. Unfortunately, some constructs from the conceptual datamodel can not be implemented due to lack of data in the database. The classification process can be trained with the outcome of previous cases using supervised machine-learning. The value of all the metrics that the decision is based upon, can be visualized in an explanation report that helps the scouts to ground their decision to accept the advice of the recommender-system and contributes to their willingness to accept. Such an explanation report thus will replace to sociology constructs of arazy et al.[5].

In this chapter, the requirements have been converted to design features, the input data constructs from the three different data-types have been selected, the classification method has been further specified and quick sketches for the interface to support the interaction with the scouts have been drawn. In the next chapter this design will be taken to practice. Chapter 6 provides an application architecture that is used for describing the development of the recommender-system.

[4]Which is an object that can be re-used and easily embedded on different pages https://reactjs.org/docs/components-and-props.html

[5]A copy of the main code that can be used for co-working development https://www.optimizely.com/optimization-glossary/feature-branch/

6

DEVELOPMENT

In this section, the design theory will be taken to practice. To be able to test the theory in section 7, a prototype will be developed. This chapter will describe the process of building this prototype by means of the application architecture in Figure 6.1. The full code for this prototype can be accessed through the first link in the appendix A.

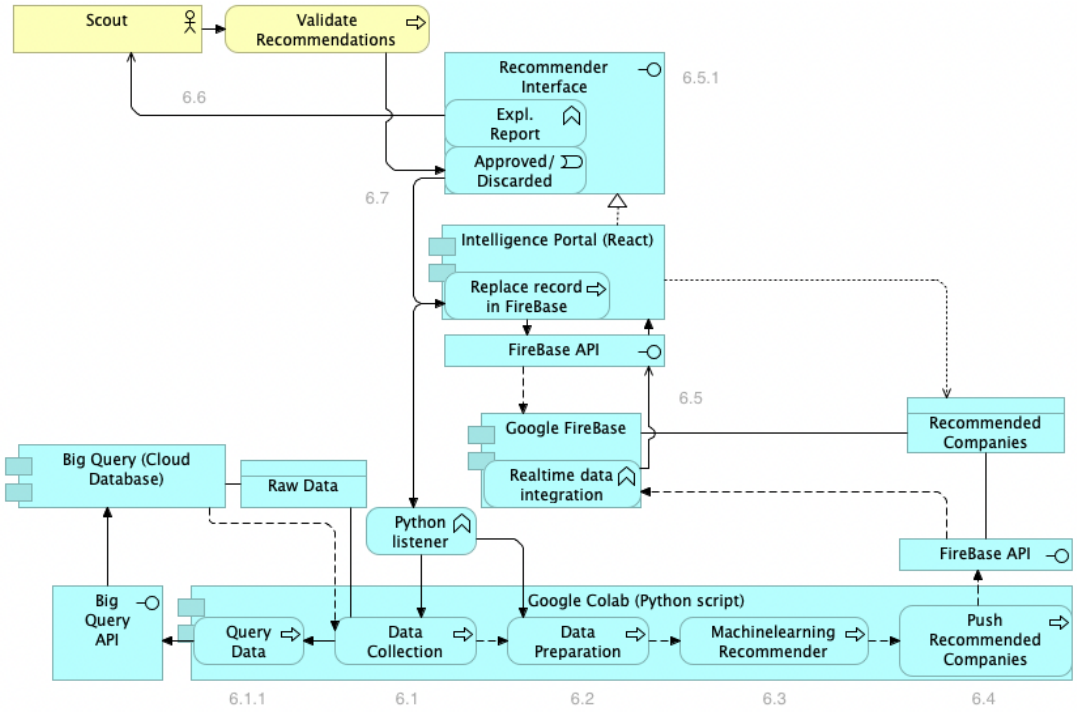


Figure 6.1: Application view architecture of the recommender software and corresponding sections (in grey).

The Archimate[1] application view architecture of the recommender software that is shown in Figure 6.1 consists of various software applications that inter-operate through API's[2]. The architecture consists of Google Big Query[3], Google Firebase [4], Google Colaboratory[5] and the portal of Unknown Group. Because of the API's and the Google Firebase, real-time data integration in the recommender interface in the intelligence portal is achieved.

The process starts with *data collection* (6.1), in which the data for the constructs of the startups are

[1] https://www.archimatetool.com
[2] Application Programming Interface, which enables communication and data transfer between two separate programs.
[3] https://cloud.google.com/bigquery, A cloud database with high performance querying functionality
[4] https://Firebase.google.com: A real-time data server, that enables pulling and pushing through different API's
[5] https://colab.research.google.com: A Python text editor that runs in the cloud for fast computing and easy access

collected through querying (6.1.1) in the Google big BigQuery database through the BigQuery API. The next step is the *data preparation*, in which this data is cleaned and prepared. This process (section 6.2), involves calculations and cleaning malicious data values, as well as preparing the structure for the data frames so they can be processed by the *ML classifier*, which is the next step(section 6.3). The resulting top-N startups are *pushed to Google Firebase* using the Firebase API, which is described in section 6.4. The intelligence portal can *pull this data from Firebase* in real time through an API as well, as is described in section 6.5. This data can then be visualized in the *interface* (section 6.5.1) in order to enable *user validation* (section 6.7). Furthermore, the data is visualized in an *explanation report* that should ground the decision making of the scout(section 6.6).

6.1 Data Collection

This system uses data from previous scouting campaigns that was selected together with the Head of scouting and another scouting lead of Unknown Group. With a Python script, the companies of these campaigns were transferred to a distinct Big Query table. The same process is executed for the to be investigated data. This Python script can be found in appendix D.
The line between data collection and data preparation in this project is thin. The reason being that BigQuery is also used for filtering and calculations as this ensures fast computations because of the cloud computing services. Though, we will distinguish between the initial collection trough the query and the second step of calculations and preparing data frames that can be processed by the ML algorithm.

6.1.1 querying

Using the query in appendix C, the data below is collected for all companies, within seconds. Retrieving the constructs can sometimes be complex because of the way the data is present in the database, this is elaborated upon below. The query starts by mapping the company IDs for the three databases to facilitate efficient navigation. To search for the first valid data record in the three databases the 'COALESCE'[6] function of BigQuery is used.

- **Number of Degrees**: Educational degrees are registered as nested items. Besides, the degrees are connected to the people, not to the companies. As a results, all degrees of all persons belonging to a company should be counted together through a complicated query. First, it unnests the data object, then it counts the amount of items in the degrees array[7]. Finally, the degrees for all persons in a company are summed. Companies with no graduated employees do not exist in the 'degree table'. Missing records should therefore be substituted by 0.
- For the number of **previous startups**, the previous involvement of founders in the foundation of other companies is investigated. For all founders of a company, their jobs with role 'founder' are summed, and 1 is subtracted for the company that we are investigating. The total sum of all these founded companies is added to the data record of the company. The 'role' field in the database is of type 'string'[8] and not 'picklist'[9]. As a result, the role of founder is sometimes written as 'Founder', 'co-founder' or 'Co-founder'. Therefore all these strings should be included in the count.

[6] A query function that looks at multiple values and takes the first that is not empty.
[7] A datatype containing a row of items, that is obtained when unnesting the object.
[8] A datatype which is a string of characters with no restrictions.
[9] A datatype with a list of limited options.

- The **awards** could be retrieved by translating the SalesForce ID to the Tracxn ID and searching for the number of awards in the Tracxn database.
- Data about **revenue** is present in multiple databases and therefore the COALESCE function delivers the first valid data record for each company.
- The **growth in number of news articles** indicates attention of the company and its speed of growth. One problem is that once a company has just started, the amount of news articles can be more than 200%. This could lead to malicious outcome of the recommender system.
- **Funding rounds and the funding amount** can also be retrieved through the COALESCE function. However, because the total funding amount largely depends on the number of years of existence of the company, the head of scouting suggested to divide this funding amount by the age. Though, because the machine-learning algorithm should be able to find this negative correlation itself, there is no point in doing it manually. Moreover, it can be that age correlates with other constructs as well and thus its value is inserted in the classifier as a separate construct.
- The **number of employees** is also available in the multiple databases, requiring the same approach as with revenue (using 'COALESCE').
- The **SalesForce status** is also retrieved form Big Query. This field takes value 1 or 0 depending if they are classified as useful or not respectively. By knowing the outcome of previous campaigns the machine-learning model can be trained by supervised learning.

6.2 Data Preparation

Now that we have retrieved the information that can be collected directly from the database, two other metrics should be calculated using Google Colabratory.

- **Trend Similarity** is calculated by comparing the description of the company to keywords of trending technologies[10]. When detecting similarity between two entities, the data attributes of the records must be equalized: Capital letters and spell mistakes should be cleared and synonyms could be added for more effective entity matching. This can be achieved by the 'spaCy'[11] NLP[12](Natural Language Processing) Python package that creates a document of synonyms of a particular string. Furthermore; punctuation, stop words and commonly used other words are filtered out because these do not relate to the properties of the data record and will therefore result in false matches. This filtering is achieved by *inverse text frequency*[13][14]. For every record in the database, a similarity score is calculated and included in the data record. The similarity of the description in relation to each technology trend is evaluated individually and the total trending tech score is a weighted sum of these individual similarities. This means that the similarity score (A-J) will be multiplied by a weight depending on the rank in the top trending list. The formula is as follows:

$$weightedsimilarityscore = w1 * A + w2 * B.....w10 * J \tag{6.1}$$

[10] https://www.forbes.com/sites/bernardmarr/2022/02/21/the-top-10-tech-trends-in-2022- everyone-must-be-ready-for-now/?sh=7b1f7670827d

[11] https://spacy.io

[12] Interactions between computers and human language where machine-learning is used to make the computer 'understand' human language.

[13] https://towardsdatascience.com/find-text-similarities-with-your-own-machine-learning-algorithm-7ceda78f9710

[14] With this technique, the importance of words is validated based on the total occurrences in the database. The fewer the word occurs, the more important it is for similarity matching.

The score for every industry of the company will be added to a Pandas data frame[15]. The mean of these scores, will act as the Trend Score of the company, as can be seen in the first code-link in appendix A and in the schematic model in Figure 6.2.

- For the **tech score**, a very similar process occurs. This calculation also includes a SpaCy similarity investigation but now on keywords from technologies in the focus area of Unknown Group. These keywords are obtained from the onboarding course from the scouting team. Nevertheless, the order of the keywords is less important because it is no longer a 'top 10', therefore the weights of the weighted sum are all equalized.

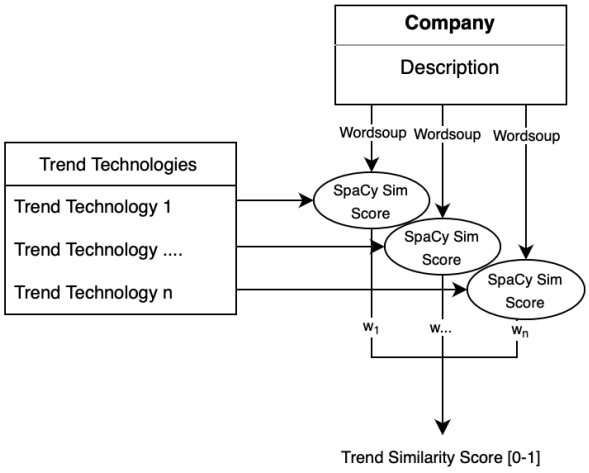


Figure 6.2: Calculation of trend Similarity Score. With w = 1.

Once the data is retrieved, enriched and cleaned, it is important to do some additional changes. At first, the data contains 'NULL' or 'None' values for all data points that are missing in the database. The machine-learning algorithm (section 6.3) cannot cope with such values. Therefore these values should be imputed. Techniques for handling NULL values are discussed in section 5.2.2.1. These methods can be applied by the code snippets in Figure 6.3 and 6.4 as was documented on Analytics Vidhya[16]. As can be seen in the code from the second link of appendix A, both these methods have been tested for data preparation.

```
IN:
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
impute_it = IterativeImputer()
impute_it.fit_transform(X)
```

Figure 6.3: Left: Multivariate Itterative Imputer Python Code Snippet.

```
IN:
from sklearn.impute import KNNImputer
impute_knn = KNNImputer(n_neighbors=2)
impute_knn.fit_transform(X)
```

Figure 6.4: Right: Multivariate K-Nearest Neighbor Imputer Python Code Snippet.

Secondly, the machine-learning algorithm can only take the constructs as an input. All other information should therefore be separated. The 'real status' should be stored in a separate 'y' value table and the 'uuids', that are used for identification of the companies, should be dropped as well. These uuids can later be merged again, as the order of the rows remains the same. This preparation phase is displayed in Figure 6.5.

[15] A 3d data structure based on the Python Pandas library. https://pandas.pydata.org
[16] https://www.analyticsvidhya.com/blog/2021/10/handling-missing-value/

```
rawDF = rawDF.fillna(0)
y = rawDF['status'].to_numpy()
X = rawDF.drop(columns =['uuid', 'status']).to_numpy()

ENI_raw = ENI_raw.fillna(0)
X_validate = ENI_raw.drop(columns =['uuid').to_numpy()
```

Figure 6.5: Data Preparation: Replacing NULL values and dropping columns.

A major concern that arose by running some prior tests while building the algorithm is: Unlike expectations, the data in these strategic niche domains is still very incomplete, I doubt that the quality is sufficient for properly training and use of the machine-learning algorithm. This will be tested extensively in chapter 7.3.4.

6.3 Machine Learning

For the machine learning program, a tutorial on 'Towards DataScience' was used[17]. In this tutorial is described how to build a two class supervised neural network that is trained using a Deep learning Neural Network. Although deep learning is known for its lack in explainability and it is more often referred to as black-box machine-learning, I choose to use this tutorial because it contained a build in explainability visualization as is displayed in Figure 6.8. This default ML model was slightly adjusted to fit 11 features. The model is built-up using two hidden layers as can be seen in the plot of the model in Figure 6.6. The output of this algorithm is the Figure 6.7. While building the algorithm, synthetic data was created to prevent that the inconsistency of the real data results in errors.

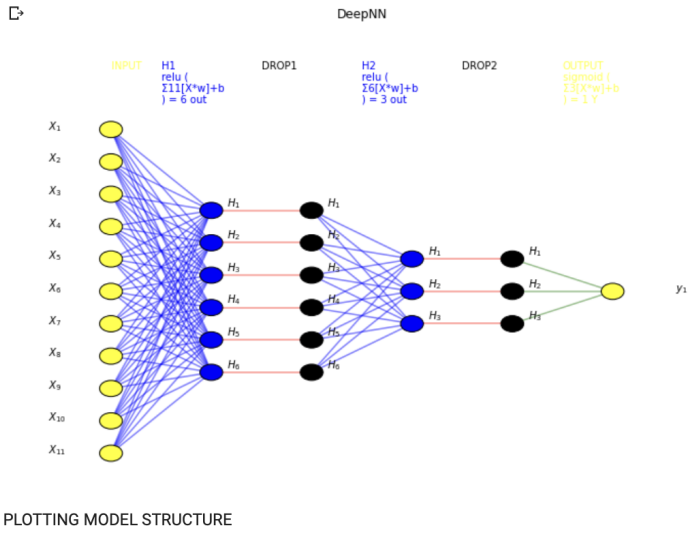


Figure 6.6: Plot of the Neural Network with 11 features and two hidden layers.

When running the real data through the trained model, a prediction can be made and the importance of the features in this outcome can be visualized using the built-in explanation visualization, of which an example is given in Figure 6.8.
To analyze the performance on a full data set, I added a function to retrieve the precision, recall and some other information about the dataset as is shown in Figure 6.9. The prediction output of the machine-learning classifier is added to the data table and the companies that are classified as '1' are selected using

[17] https://towardsdatascience.com/deep-learning-with- Python-neural-networks-complete-tutorial-6b53c0b06af0

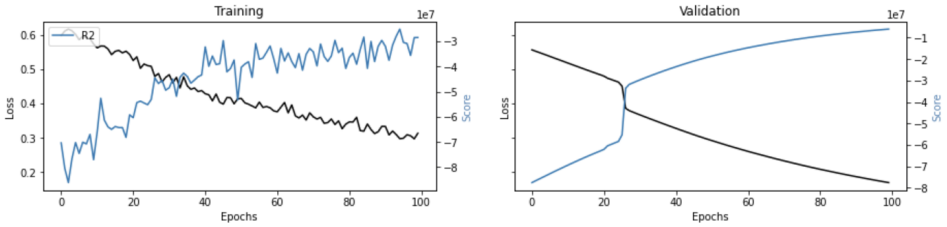
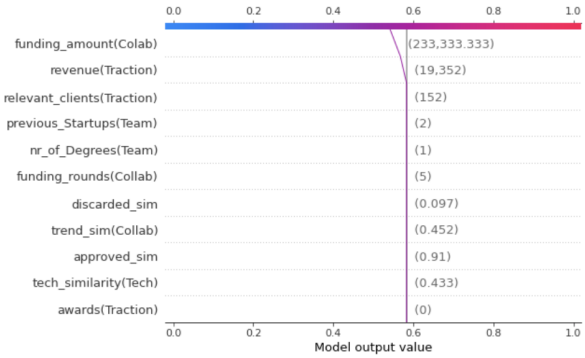


Figure 6.7: Output algorithm based on the training and test data. - *Based on synthetic data*




```
preds = np.round(model.predict(X_validate),0)
print(confusion_matrix(y_validate, preds))
print(classification_report(y_validate,preds))
```

```
[[65  3]
 [ 3  0]]
              precision    recall  f1-score   support

         0.0       0.96      0.96      0.96        68
         1.0       0.00      0.00      0.00         3

    accuracy                           0.92        71
   macro avg       0.48      0.48      0.48        71
weighted avg       0.92      0.92      0.92        71
```

Figure 6.8: Left: Output of the recommender model with explanation on importance of the metrics - *Based on synthetic data*.

Figure 6.9: Right: Performance metrics of the dataset - *Based on synthetic data*.

the code in Figure 6.10.

```
dfSelected = ENI_raw[ENI_raw.Class > 0.0]
print(dfSelected)
```

```
                                   uuid  num_of_degrees  nrofPreviousComp  \
23  8c123d92-8661-4960-8941-5424dd866e7c             4.0               4.0
36  543f7df9-ee13-40f3-a7a7-61207d9b1320             1.0               8.0
55  77291dd1-4d0b-418f-90ff-4b06680106e0             2.0               4.0

    tech_score  nrAwards   news  revenue  trend_score  num_funding_rounds  \
23    0.400000       1.0  139.0    14428     0.700000                 1.0
36    0.437506       2.0   71.0    53420     0.469179                 2.0
55    0.453913       2.0  122.0    92740     0.472839                 2.0

    fundingperage  status  discarded_sim  approved_sim  Class
23      5219342.0     0.0       0.752842      0.213847    1.0
36      4825581.0     0.0       0.107250      0.920024    1.0
55            0.0     0.0       0.240018      0.796906    1.0
```

Figure 6.10: Companies classified as '1' (useful) - *Based on synthetic data*.

6.4 Push to Firebase

Now that the predictions are known, the uuids of the selected companies should be pushed to the real-time Google Firebase database. However, the table of the data is in a Pandas DataFrame format. This is not compatible with Firebase (which only takes JSON[18] strings). For this reason, the code snippet in Figure 6.11 translates the Pandas dataframe to a JSON string.

[18] https://www.w3schools.com/js/js_json_syntax.asp

```
pdtest = pd.DataFrame()

for i in dfSelected['uuid']:
    row1 = {'ids':{i:i}}
    dfIds = pd.DataFrame(data=row1)
    pdtest = pdtest.append(dfIds)

pdtest.head()
js = pdtest.to_dict()
print(js)
```

```
{'ids': {'8c123d92-8661-4960-8941-5424
```

Figure 6.11: Translate the Pandas dataframe to a JSON string that is compatible with Firebase.

Next, Google Colaboratory needs to connect to the Google Firebase API. This is achieved using the code in Figure 6.12. The API key and IDs are hidden because of security reasons. Additionally, access-rules in the Firebase console required adjustment to allow mutations from my account ID. Once the connection is made, the data can be pushed into Firebase. Important is to adhere the proper data structure. This happens in the bottom section of Figure 6.12.

```
firebaseConfig={
  "apiKey": [redacted],
  "authDomain": "maximal-cider-305010.firebaseapp.com",
  "databaseURL": "https://maximal-cider-305010-default-rtdb.europe-west1.firebasedatabase.app/",
  "projectId": "maximal-cider-305010",
  "storageBucket": "maximal-cider-305010.appspot.com",
  "messagingSenderId": [redacted],
  "appId": [redacted],
  "measurementId": [redacted]
};



firebase=pyrebase.initialize_app(firebaseConfig)

db=firebase.database()

#Adhire Datastructure
data={"date":"21-02-22-13-45-42",
      "ids":js['ids'],
      "name":"NEW ENERGY - Recommendations"}
db.child("recommender").child("Iay3sBGSzRbylHQ0UAX3fGNEIEI2").child("companies").child("-My7MahfmZJRJRONKVbt").set(data)
```

```
{'date': '21-02-22-13-45-42',
 'ids': {'543f7df9-ee13-40f3-a7a7-61207d9b1320': '543f7df9-ee13-40f3-a7a7-61207d9b1320',
  '77291dd1-4d0b-418f-90ff-4b06680106e0': '77291dd1-4d0b-418f-90ff-4b06680106e0',
  '8c123d92-8661-4960-8941-5424dd866e7c': '8c123d92-8661-4960-8941-5424dd866e7c'},
 'name': 'NEW ENERGY - Recommendations'}
```

Figure 6.12: Connection Firebase API and Push while adhiring data structure.

6.5 Pull from Firebase

The front-end of the recommender application is built in React. How the UI is setup is explained in more detail in section 6.5.1. The data in this interface should be pulled from the real-time Firebase database. This process is similar to the behavior of the 'save-company-to-list' component, that is already implemented in the portal. Therefore, this feature was copied and slightly altered to act as a recommendation list. The Google Firebase integration happens in the code in appendix E in Figure E.1, where the companies from the recommender list are retrieved from the Firebase database. In appendix E in Figure E.2, the IDs from Firebase are parsed to the 'ResultCompany' component that creates an instance for every company in the recommender list.

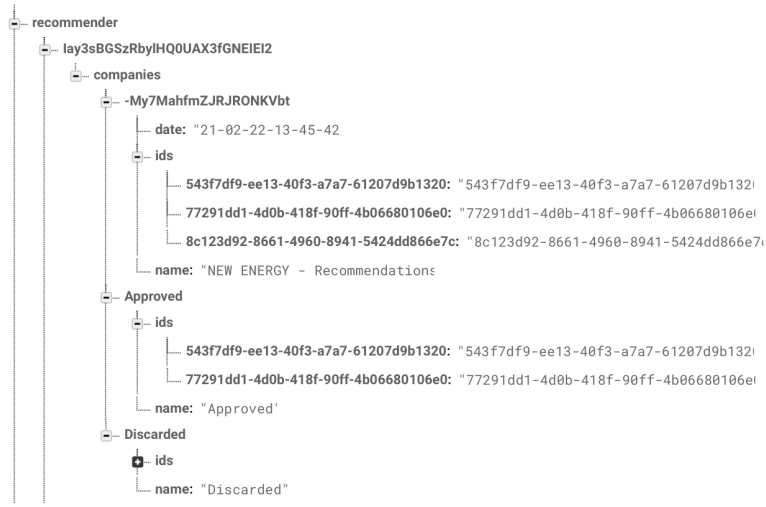


Figure 6.13: Data in real-time database in Google Firebase.

6.5.1 Interface

The interface is built in the React project of the intelligence portal. First the most recent project was copied into a branch using GitLab branch control[19]. Then, a feature-branch was created in the command-prompt window. In this feature branch, I was able to alter the code without major risks for the intelligence portal. In the *App.tsx* file, a reference was created for a new recommender page in the intelligence portal. This reference points to the *view/feature-recommender/recommender.tsx* file in the projects, which is a slightly altered copy of the *FeatureSavedList.tsx* file. This file contains a wrapper[20] for the *WidgetSaved-Profiles.tsx* component that enables saving profiles to a list in the intelligence portal. These profiles are displayed in the list by the wrapped *ResultsCompany.tsx* component as displayed in Figure E.2. In this *'ResultCompany.tsx'* component, the ID that is parsed in, is used to collect all other information about the company through Elastic Search[21], so it can be presented in the list and in the company profile. This data access through Elastic Search, was already built-in and its exact know-how is not of interest in this thesis. All files that are named in this section are copied to the feature branch, so they could be altered without affecting the current intelligence portal. The changes that are made to these files are listed below:

- At first, the normal list view does point to a database reference that is created through the portal itself. Instead, it should point to the database record that was created using Google Colaboratory (section 6.4). Therefore, I had to change the database address to the one that can be seen in Figure E.1.
- Secondly, the *'ResultCompany.tsx'* component required adjustments, because it only contained a delete functionality. Both a 'discarded' and a 'approved' button should be implemented, which was realized using the code in Figure E.3.
- The new button should trigger a function that handles the desired behavior for the company with the reference ID that is parsed to the function(as can be seen in appedix E in Figure E.3). For this, the 'addToFavorites' functionality, that already existed on another page in the portal, was copied. This type of behavior is exactly the same as what is required for the approve and discard

[19] A feature that enables co-working in the same project.

[20] An entity that encapsulates and hides the underlying complexity of another entity by means of well-defined interfaces(https://www.techopedia.com/definition/4389/wrapper-software-engineering).

[21] Elasticsearch is a distributed, free and open search and analytics engine for all types of data(www.elastic.co).

button, because the validated companies should be saved in distinct lists to be able to parse this information back to the recommender algorithm. The code for this functionality is presented in appendix E in Figure E.4. Note that the function ends with a call of the 'handleDelete()' function. This is only the case in the discard behavior as the company in this case should be erased from the recommendation.

6.6 Explanation report

The implemented explanation report slightly differs from its initial design. Instead of creating a full dashboard with a meter for every construct, the constructs are visualized together in a radar chart and different companies can be plotted on this same graph to support easy comparison. For this radar chart, the Apexcharts Library[22] for React was used. In the React code, a component was created containing the radar chart and this component was wrapped on our copy of the 'WidgetSavedList.txt' file, making it appear above the recommender-list as can be seen in Figure 6.14. The code for this radar chart is given in the appendix E Figure E.5. Please note that the radar chart is currently based on synthetic data. Due to time limitations there was no priority to finalize this part of the system. Section 10.5.1 in the future recommendations explains how to implement this radar chart to the real data.

6.7 User Feedback

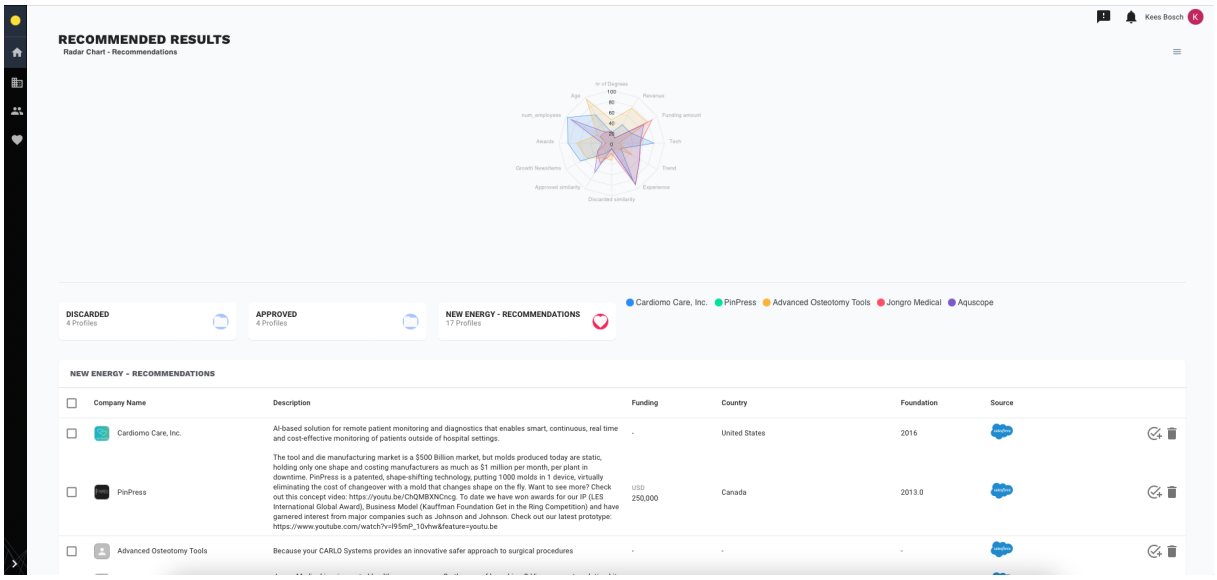


Figure 6.14: Interface of the recommender system with radarchart explanation report visualization (Based on synthetic data).

The user (the scouts of Unknown Group) retrieve the recommender information through the interface as represented in Figure 6.14. On the right of the company results, the user can either approve or discard the recommendation. These companies are then added to the 'Approved' and 'Discarded' lists that can be accessed in the portal as well. In the back-end however, these lists are stored in Google Firebase, making them accessible in real time by Google Colaboratory, for further computations. This is enabled

[22] https://apexcharts.com/docs/react-charts/

by the Goolge Firebase API in Google Colaboratory as can be seen in Figure 6.15. The description of every record in the database is compared to the descriptions of the companies that are stored in both these database records. As a result, an average discarded_similarity and approved_similarity score can be calculated. All companies that have a similarity score above a certain threshold on the 'approved companies' and below a certain threshold with the 'discarded companies' are selected independent of the outcome of the ML-classifier. These companies are added to the recommendation. The optimal value of these thresholds is to be determined with testing. The companies that remain in this filter will be added to the list of IDs that is uploaded to Firebase. Duplicates will automatically be filtered by Firebase as they run on the same ID.

```
#Retrieve Approved
ApprComp=db.child("recommender").child("Iay3sBGSzRbylHQ0UAX3fGNEIEI2").child("companies").child("Approved").child('ids').get()
print(ApprComp.val())

Approvelist = []

for ids in ApprComp.each():
  Approvelist.append(ids.val())


IDsApproved=''
print(Approvelist)
for i in Approvelist:
  IDsApproved = IDsApproved + "'"+i+"'" + ','
IDsApproved= IDsApproved[:-1]
print(IDsApproved)
pdApproved = pd.DataFrame(Approvelist, columns = ['ids'])
```

Figure 6.15: Retrieve Approved Companies through Firebase API.

7

TESTING

This chapter explains the test plan, containing all preparation necessary for the five different stages of the test. Subsequently, the equipment that is necessary for execution of the test is listed. In the last section of this chapter the results of the test are shown.

7.1 Test Plan

In order to answer the research question, it is important to distinguish between five testing goals. At first the functioning, performance and reliability of the system should be tested. To test the functioning of the recommender-system individually and not reliant on the incomplete data, synthetic data is used. Additionally, the feedback loop and the explanation report should be tested. Once these three stages have been tested successfully, conclusions can be drawn about the development of a recommender-system for startup scouting. Subsequently, a case study with real data is performed, to evaluate the effectiveness of the system. This testing will occur in stage 4. The results of the tests and the architecture will then be used to evaluate on the requirements in stage 5.

7.1.1 Stage 1 - Classification System

In stage 1, synthetic data will be created. To draw conclusions about the systems performance, the data input must be similar to the real data (but then without the missing values). To create synthetic data that can represent the real data, the real use-case was analyzed. Because investing in startups typically involved the identification of over-performing outliers, the distribution of the not valuable startups is assumed to be uniform. Therefore, a uniform data-set of 2000 fictitious companies are generated and are given 11 values with random numbers between 0 and 1. These companies are classified as 0. Then, the DataFrame is appended with another 2000 companies with random numbers between 0 and 2 (or 3)[1]. These companies are classified as 1. Given the wider range, this second batch of companies will contain outliers with respect to the first 2000 companies. Because the values can also take normal values (between 0 and 1), they will not over-perform on every metric, making it impossible to extract them with basic filtering. The data set as just described, is used for training and testing.
Subsequently, another synthetic dataset is prepared for validation. This dataset consists of 1000 companies, uniformly distributed with random values between 0 and 1, and 100 companies with random values between 0 and 2. This dataset represents the 'New Energy' dataset that must be validated using the recommender system. Because it is synthetic data, we can model the desired output and use this data to validate the system's performance by comparing the recommendation prediction with the real class of the companies. This analysis can be summarized in a confusion matrix (described in section 2.4). Because the outcome of the first few test runs was incredible, there was a chance that the synthetic data

[1] This can be altered in different runs of the test.

was too clean, making it not representative for the real data. Therefore, it was decided to distort the data by creating 20 'moles'; These are companies containing one score between 1 and 2 while the others are normal([0-1]). This, to represent the realistic event that companies perform good on one specific metric, but still do not qualify as a valuable company. These moles were added to the train/test set and the validation set.
From the confusion matrix, basic IR measures as precision, recall, accuracy and F1 score can be calculated. Because these measures are prone to biases, the recommender system can be validated by calculating the mean Average Precision and the Matthews-Correlation, of which the formulas have been given in section 2.4. Both scores can take values between '-1' and '1'. The higher the score, the better the recommender-system.

7.1.2 Stage 2 - Testing Feedback Loop

The user feedback loop is tested by running a similarity check for a particular company's description, with respect to the descriptions of all other companies. In doing so, the most similar companies and least similar companies can be found. The experiment consists of two tests:

- The four most similar companies and four most different companies will be uncovered. These will be stored in the Approved and Discarded folder respectively. The 'approve/discard filtering algorithm' (second link in appendix A) will then be ran on the whole data set. During this process, the system filters companies based on their similarity with the approved and discarded folder. Only companies above and below a threshold for the approved and discarded folder respectively, will be selected. If the system detects the pre-selected company, the experiment will be a success. This process will be repeat for 5 different companies. As a result of this test, a justifiable estimate for the optimal thresholds can be made. The average minimal thresholds for both the approved and discarded score should be a good initial value for the first prototype.
- The same process will be repeated, except now the priory selected similar and not-similar companies are not the top 4 but the most similar, the third, the fifth and the seventh most similar companies.

For preparation, one company was stored in the 'Approved' folder (the 'Discarded' folder was empty). When running the 'approved similarity' algorithm for all companies in the to be evaluated database, the similarity scores compared to the description of the selected company are calculated. All companies that do not include a description are removed as these can manipulate the results. With the Python Pandas 'nlargest'[2] method, the ten highest scores are extracted. With 'nsmallest'[3] the same is possible for the lowest scores. In the first part, the four most[4] and four least similar companies are stored in two dataframes. For the second test the 1,3,5 and 7th companies are stored in the dataframes. Next, the company that is investigated is erased from the 'Approved' folder and the identified similar and not similar companies are stored in the 'Approved' and 'Discarded' folder respectively. By calculating the average similarity from the four identified companies in the first part of the test, the optimal thresholds can be found. By running the program normally[5] with these thresholds, the functioning can be tested. If the algorithm succeeds, the prior selected company should be selected from the raw data (along with other companies that also perform above and below the approved and discarded similarity thresholds).

[2] https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.nlargest.html
[3] https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.nsmallest.html
[4] Excluding the company itself, that logically appears on top.
[5] Running it as it will run in practice (not in the setup from the first part of the test).

7.1.3 Stage 3 - Evaluating The Explanation Report

In stage three, the explanation metric will be evaluated with with three different scouting leads from Unknown group. This will be done using a survey. The goal of this interview is to evaluate the design of the explanation metric, to see if it meets their desires. The questions from this survey can be found in the appendix G. Based on the findings of this survey, further recommendations for improvement of the explanation metric can be provided. The table (G) also contains questions which were used to evaluate the (potential) added of the recommender-system in their daily scouting routine and features that could be added in the future. These questions are not relevant for this explanation report evaluation and will be discussed in section 8.3.

7.1.4 Stage 4 - Testing with Real Data

The fourth stage involves testing the system with the real data. To evaluate the performance in the discussion section, the IR metrics as well as the mean Average Precision and the Matthews-Correlation Coefficient were collected. In advance, the available data should first be analyzed.

7.1.4.1 Data Analysis

The train/test data that was appointed by a scouting lead of Unknown Group involved three prior cases for training and testing. Because a scouting campaign typically involves over 200 startups, of which only a few (4-11 companies) are selected as 'winner', there is a lot more data about startups that are not selected. Training a machine-learning model however, requires balanced data between the two classes. Therefore, startups that have been marked as winners in all prior campaigns, were added to this training/test data.
Subsequently, I noticed that the data is not as complete as was expected in such a strategic niche domain, hereby leaving a lot of data points to NULL. I had foreseen problems in using this data, because a machine-learning algorithm does not only recognize patterns in high or low values but also in exact values. NULLable values could therefore be misunderstood in their relation to the output, making proper classification impossible. Even though the value and functioning of the recommender system can be proven independently of the real data in stage 1, my goal is to make the system usable for Unknown Group. Accordingly, an attempt will be made to solve the inconsistent data in order to provide a sufficient recommendation for the domain of 'New Energy', by using different techniques for handling NULL values. Design decisions about these techniques for handling NULL values are elaborated upon in section 5.2.2.1. The implementation of the K-Nearest Neighbor imputer and the Iterative imputer is explained in 6.2.

The revenue column is completely empty, because very little companies want to share this information. For this reason, two tests were run; one in which this value was left out of the machine-learning algorithm. The other, where all values were set to 0.

7.1.5 Stage 5 - Evaluation Requirements

In this stage, the inclusion of the requirements is evaluated. All non-functional and functional requirements will be discussed with use of the metrics from the previous four tests or with grounded argumentation based on the application architecture from figure 6.1. In the conclusion, these results will be used for evaluating the success of the solution.

7.2 Test Setup/Equipment

This section describes the test setup and equipment that was used to conduct the testing in this thesis. An Apple macbook pro 2020 with M1 processor was used for the computations and builds. On this computer, I have installed 'visual studio code' to run the React feature branch as described in the development section.

- *For stage 1,* the Google Colaboratory code in the second link in appendix A was used, in which a modified version of the recommender software was prepared to support synthetic data testing. The confusion matrix and resulting IR metrics were stored in an Excel table. Additionally the mAP and MCC score were calculated using Excel. Different settings for the classification algorithm were tried.
- *For stage 2,* the second part of the Google Colaboratory project from the second link in appendix A was used for collecting the approved and discarded data. I also ran the React code to visualize the recommendations in the intelligence platform on a local host page. From here, I could validate companies by placing them in the approved and discarded folder. I opened the google Firebase project of the Intelligence portal to clean these folders for every new test run.
- *For stage 3,* an email was send to three different heads of scouting at Unknown Group, containing the questions form appendix G. The answers were also collected in this Excel sheet and a combined conclusion was drawn, this is elaborated upon in section 7.3.3.
- *For stage 4,* the data was prepared by replacing the NULL values with predictions through the Iterative imputing method and the K-Nearest Neighbor method. The code for this experiment can be accessed through the first link in appendix A. The confusion matrix and resulting IR metrics were stored in an Excel table for calculations of the mAP and MCC score. Different settings for the classification algorithm were tried.
- *For stage 5,* the requirements that followed from the design theory are evaluated based on the design features that were established. Using the first four tests, the performance of the recommender-system was measured. These results as well as the application architecture are used to evaluate if the requirements from table 7.3.5 are; *met (in green), in progress (yellow) or not met (red).*

7.3 Results

7.3.1 Stage 1 - Testing the Classification System

During this first test, the performance of the recommender-system was measured. The recommendation was based on synthetic data to prevent interference of other factors. To evaluate the performance in the discussion section, the IR metrics as well as the mean Average Precision and the Matthews-Correlation Coefficient were collected in table 7.1. This test was conducted five times, each with three runs with different randomly generated values and different splits in training and test data. The averages of these five test runs are collected in table 7.1. The description of each test is provided in appendix H. This test was conducted with two different ranges for the outliers (2 and 3 times the normal value). The results from these test were exceptional. The recommender gained a mAP score of 0.9875. It also had and average Matthews-Correlation Coefficient of 0.94 for the multiplier of two and even 0.99 for multiplier of 3. The dataset appeared to be too clean in comparison with reality. After distorting the data, by adding the moles as described in section7.1.1, the mAP and the Matthews-Correlation still remained well above

the 0.9 with a mAP score of 0.9867 and an average MC score of 0.93.

STAGE 1 Summary	Precision 0	Precision 1	Recall 0	Recall 1	Accuracy	Mean Average Precision	Matthews Random Field
AVG Test 1	0.99	0.96	0.99666667	0.93	0.99	0.975	0.938803398
AVG Test 2	0.97666667	0.99	0.99333333	0.95333333	0.986666667	0.983333333	0.95465504
AVG Test 3	0.99666667	0.99333333	0.99666667	0.98333333	0.99666667	0.995	0.984962227
AVG Test 4	1	0.99333333	1	0.99333333	1	0.996666667	0.992666667
AVG Test 5 *	0.99	0.97666667	0.99666667	0.9	0.99333333	0.983333333	0.930747603
Total average	0.99066667	0.98266667	0.99666667	0.952	0.99333333	0.986666667	0.960366987

Table 7.1: Summary of results recommender machine learning testing. (* = test run with the moles).

7.3.2 Stage 2 - Testing the Feedback Loop

In stage 2, the feedback loop was tested. The optimal thresholds for validation based on the approved and discarded similarity selection were investigated. The results are collected in table 2 in the appendix H. A summary of these results is depicted in table 7.2 below. We learn that the optimal thresholds for both tests (with the top 4 and with the 1,3,5 and 7th most similar and least similar companies) are about 0.75 for approved and 0.5 for discarded, These thresholds are therefore advised to adhere.

STAGE 2 Summary	Approved					Discarded				
Companies similarity	1	2	3	4	Threshold	1	2	3	4	Threshold
AVG Test 1	0.784	0.772	0.762	0.749	0.767	0.463	0.478	0.493	0.510	0.486
AVG Test 2	0.784	0.762	0.745	0.734	0.756	0.463	0.493	0.514	0.529	0.450
Total average	0.784	0.768	0.753	0.742	0.761	0.463	0.485	0.503	0.519	0.493

Table 7.2: Summary of results feedback model threshold investigation.

7.3.3 Stage 3 - Testing Explanation Metric

From the interview questions that were send to the three heads of scouting at Unknown Group, the following generalizations could be made (the exact results are shown in appendix G): According to the scouting lead of Unknown Group, the radar chart visualization is an effective way to display the scores on each metric. However, they mention that they require more information. The explanation could be improved by adding the weights of the constructs (their relative importance). Furthermore, not all metrics are self-explaining. Some description could help to clarify what the score is based upon.

7.3.4 Stage 4 - Testing with Real Data

In stage 4, the performance of the recommender-system on real data was measured. To evaluate the performance in the discussion section, the IR metrics as well as the mean Average Precision and the Matthews-Correlation Coefficient were collected in table 7.1. This test was conducted three times, each with three runs with different methods for handling null values. The description of each test is provided in appendix H. When querying the real data, concerns arose about the amount of NULL values. Testing

the machine-learning classifier with the real data revealed that the concerns were indeed legitimate. While running the program with all NULL values set to 0, all companies were classified of 0 or as 1. Assumingly, due to patterns being recognized in NULL values. For this reason, NULL values were predicted using the Iterative imputer or the K-Nearest Neighbors imputer as described in section 5.2.2.1. The outcome of this strategy was very inaccurate as can be seen in table 7.3. With the Iterative imputer, all companies were still classified as 'not valuable'. The K-Nearest Neighbors imputer resulted in a lot of false positives. In order to improve the performance of the algorithm, different settings for the split between train and test data were attempted but with no significant results. Similar to the tests with the synthetic data, the basic IR metrics were revealed using a confusion matrix and the Matthews-correlation and mAP score were calculated.
The MCC for the Iterative imputer could not be calculated because division by 0 is not possible. The average MCC for the test with the KNN imputer could be calculated, but is very close to 0, indicating no alignment with the real data. The mAP scores for this real data testing take values around the 0.5, suggesting that half the predictions are correct.

STAGE 4 Summary	Precision 0	Precision 1	Recall 0	Recall 1	Accuracy	Mean Average Precision	Matthews Random Field
AVG Test II (0.5 & 0.3)	0.94	0	1	0	0.94	0.50975417	-
AVG Test KNN I 1 (0.5)	0.92207792	0.05755396	0.35607249	0.66666667	0.36333333	0.531972512	-0.039624521
AVG Test KNN I 1 (0.3)	1	0.06382979	0.13666667	1	0.18666667	0.472222222	0.093599953
Total average	0.64069264	0.04046125	0.16424639	0.55555556	0.18333333	0.504649635	0.026987716

Table 7.3: Summary of results stage 4 testing with real data.

7.3.5 Stage 5 - Evaluating the Requirements

To validate the success of the solution, the level to which the requirements are incorporated in the artifact should be evaluated. The right column of table 7.3.5 indicate if the requirements are met. The groundation is provided below. The source of evidence is provided between brackets.

	Non-Functional Requirements	ID	Status
Non-Functional	The accessibility of the system should be restricted to people inside the organization.	N1	●
	Data security should be warranted.	N2	●
	While storing user data for personalized recommendations, the system must comply with GDPR requirements.	N3	●
	The system should not significantly reduce the performance of the scouting portal of Unknown Group, when integrated.	N4	●
	The operating time of the system should at least match the time it takes to scout startups manually.	N5	●
	The interface of the system should be intraoperative with React as the scouting portal is built in this framework.	N6	●
	The backend of the system should be built in python to support integration with BigQuery (where the data is stored)	N7	●
	The system should assist the current business activities instead of replacing them.	N8	●
	The system should be easy to use by the scouting team of Unknown.	N9	●
	Functional Requirements	**ID**	**Status**
Functional (HL)	The scouting team of Unknown Group must be able to scout startups in high efficiency.	F1	●
	The recommender-system must evaluate and classify startups based on data.	F2	●
	The output of the recommendations must be reliable and human biases in startup scouting should be reduced.	F3	●
	The output of the recommender-system must include a top-N with suggestions of startups that should be further evaluated by the scouting team.	F4	●
	The users should be able to use the recommender-system through an interface in the scouting portal	F5	●
	Adhering the recommendation must be groundable, The advice must include explanation of the choice for the recommended startups.	F6	●
	The recommender-system must target startups in the strategic niche domain; New Energy	F8	●
	The system must adapt to domain specific problems as identified in the background theory.	F9	●
Functional (LL)	Must analyze startups based on knowledge data.	f2.1	●
	Must analyze startups based on content data.	f2.2	●
	Must analyze startups based on collaborative data.	f2.3	●
	Must combine knowledge, content and collaborative data in a decision model.	f2.4	●
	Must classify startups as useful or not useful.	f2.5	●
	The value of each metric must be visible.	f6.1	●
	Comparing the metrics should be possible through the platform.	f6.2	●
	Must adapt to new items.	f9.1	●
	Must adapt to new information about the user.	f9.2	●
	Must cope with inconsistent and incomplete data.	f9.3	●

N1 & N2 [Architecture]: As previously mentioned in section 4.1, the first two non-functional requirements were automatically implemented once integration in Unknowns existing software was successfully accomplished.

N3 [Architecture]: The recommender system does not use or process any new privacy sensitive information as the preferences of the user are only based on knowledge and experience in the field of startup scouting [1]. Moreover, as evaluating startups is their job, the information that is passed in the algorithm is parsed with consent and is therefore owned by Unknown Group an thus legitimate to use.

N4 [Architecture]: Because the intelligence portal is built up using 'Async methods', loss in performance of one component does not cause delay in performance of others, making this requirement no longer applicable.

N5 [Stage 4]: The operating time of the recommender-system takes approximately 20 minutes. Hypothetically, when the manual actions are extracted by implementing the back-end as a python listener[6] script and saving the installs and imports, this operating time can be reduced extensively. However, with taking the traditional scouting process (that could endure for over a month[7]) as the zero measurement, tremendous improvement can be measured. The formula from section 4.4 is used to measure the ex-

[6] https://Firebase.google.com/docs/firestore/query-data/listen

[7] According to the scouts of Unknown Group

act win time-span; A month of working hours contains 9600 minutes[8]. So the percentile reduction is $\frac{20-9600}{9600}$ =-99,7%

N6 [Architecture]: The recommender interface is written in React and contains the styling and components that are also used in the rest of the platform, making it blend in perfectly. Its integration in the platform enables easy accessibility by the scouts of Unknown Group.

N7 [Architecture]: The backend of the recommender-system is written in python in a Google Colaboratory project to facilitate easy integration with Google BigQuery, Google Firebase and fast computation power because of the Google Cloud services.

N8 [Architecture]: The recommender system is accessible on a distinct dashboard in the intelligence portal. As a result, it does not affect any other functionality of the platform, making it additional to the current scouting process. On demand, the recommender-system can be started and an advice to the scouts will be presented. It will not replace any other processes. Manual evaluation by the scout is required before inclusion in the insight report.

N9 [Stage 3]: The ease of use of the recommender system was thought of while creating the design as the list-view component (that is also used on another functionality in the platform) was reused and slightly altered to present the recommended companies. The explanation visualization was evaluated with the scouting team and they all were very satisfied with the new design of the visualization. Though, they mentioned that it could be improved by adding more information about the weights and the meaning of the constructs. I marked this requirement as 'in progress', because an extensive user evaluation could be beneficial to optimize the recommender software and the features as mentioned by the scouts should be implemented to make it useful. An opportunity is to add the metric for visualizing the importance of the constructs for the classification that is created in the back-end (6.8) to the interface.

F1 [Stage 4]: The scouting team of Unknown Group must be able to scout startups in high efficiency. Because the selection process is now based on data instead of thorough evaluation, scaling down the 200 startups to about 20 will now take 20 minutes, where it used to take months. This 99,7% reduction implies an extensive win in time-span. However, till the system is really adopted by the scouting team and used in practice, we will not know the exact win in labor time. The 20 remaining startups still must be reevaluated by means of the 'human-in-the-loop' process which also takes time. In the future, the time-span can be further increased when the manual process of starting the algorithm is replaced by a python listener as is explained in 10.4. Most of the 20 minutes operating time is due to imports of libraries. In the free version of Google Colaboratory, it is not possible to save those installs and imports. However, if the system has proven its work, upgrading to the paid version to enable full integration in the platform. Because this requirement only involves the efficiency in scouting startups and do not refer to the reliability of the recommendation, this requirement has been met.

F2 [Stage 1]: The recommender system evaluates startups based on 11 metrics. synthetic data was created to simulate the 11 metrics that can be used to validate startups. Out-performers can be detected

[8] 4 weeks of 40 hours containing 60 minutes

by the machine-learning with high precision, even when the synthetic data is distorted with moles. The algorithm uses a non-linear two class classifier to detect valuable startups (f2.5). Once Unknown Group succeeds in making their data more consistent, I strongly believe they are able to validate the startups using this recommender-system. In essence, they are already able to validate based on data, but the data quality is insufficient. This does however mean that the requirement is met (though this cannot be proven without sufficient data). Regarding the different data types (knowledge data (f2.1), content (f2.2) and collaborative data (f2.3)), the input metrics have been carefully selected to include all these data types in the neural network (f2.4). The importance of each metric is determined using deep-learning back-propagation.

F3 [Architecture & Stage 1]: From the theory section about biases in startup scouting, we learned that combining knowledge data, collaborative data and content data in a recommendation system, along with a HITL mechanism, could potentially solve most human biases. Logically, this only holds if the recommender system itself is reliable. Appendix F describes a plan for bias response testing. The mAP (0,97) and the MC (0,93) score of the synthetic data testing are very promising as they indicate very strong agreement with the real classes of the validation data. Though, the recommender-system is currently not able to produce a proper recommendation due to the insufficient quality of the data. Because of this reason, the bias response tests could not be executed. This requirement is therefore marked as in progress.

F4 [Stage 2]: The output of the recommender-system is a list with startups that can be viewed in the intelligence portal. The list with ventures must be evaluated by the scouting team before actual investment or taking up in an insight report for the client. I have realized that once a startup is classified, it should be represented in the top-N. When implementing rank based classification as suggested in section 10.5.1, the optimal number of startups in the top-N should be investigated.

F5 [Architecture & Stage 2]: The recommender-system is implemented in the intelligence portal, the user eventually only needs to interact with this system. Due to time limitations and because the free version of Google Colaborate does not allow permanent installs and imports of libraries, the python back-end still requires manual actions. However, in further recommendations in section 10.4 is explained how these manual actions could be replaced by a python listener in the future. This requirement is therefore in progress.

F6 [Stage 3]: The recommendation dashboard (that is implemented in the intelligence portal *(f6.2)* contains a radar visualization that explains the strong points of the startup and thus the reason that it is selected in the recommendation *(f6.1)*. The aesthetics of this 'explanation report' are different from the initial design, though it displays the same information in a more compact manner. In an evaluation interview with the scouting team was asked if this revised design still met their expectations and desires. They all stated that the design looks very solid and that it is stronger than the one as shown in section 5.5. Though, it could be improved by adding more information about the constructs and their weights.

F7 [Architecture]: This requirement rather depends on the success of the previous phase (filtering on the scope). If everything went well, the pool of startups that are analyzed by the recommender-system only contains startups in the New Energy domain. This requirement is therefore included. Moreover, one reason for creating a machine-learning model to classify the startups was to support future imple-

mentation in other strategic nice domains without having re-investigate the model of the constructs. With machine-learning, the neural network is trained on retrieving the optimal weights for each metric. The training data contained cases in other strategic niche markets, but also a previous case in the domain of New Energy. This way the machine-learning model was slightly adapted to this domain. By training this model with other cases, the algorithm can be altered to also support other strategic niche domains.

F8 [Architecture]: The recommendations rely on the most recent data because it is directly retrieved from Google Big Query via SQL[9]. The output of the system is synchronized in real time with the intelligence portal because it is stored in Google Firebase. The same holds for the feedback from the user. Because of this infrastructure, the whole system continuously adapts to the latest information.

F9 [Stage 4]: From the theory, we learn that with building a recommender system in the venture capital domain, there are various domain specific problems that has to be dealt with. First, the continuous stream of new information about item (f9.1) and user (f9.2). These problems are covered by the real-time data integration and the human in the loop feedback that is retrieved through the interface respectively. The third problem in the requirements list is the inconsistent and incomplete data. While testing with the real data, we suffered from this domain problem. Even by trying to make predictions for the missing data, this could not be worked around. The inconsistency remains a major issue that should be fixed in order to enable successful data driven decision making. A potential solution is discussed in the further recommendations in section 10.2.2.

[9]Structured Query Language, the language to write queries.

8

DISCUSSION

This chapter contains the discussion of the results as well as assumptions and considerations that were made throughout this project. This section starts with discussing the data collection and constructs selection. Then the testing and results will be discussed.

8.1 Data Collection and Constructs Selection

Based on the theory and the onboarding course from the scouting team, the most relevant features for startup validation were identified. Most of these features have been incorporated in the Neural Network as designed in Figure 5.2. Unfortunately in practice, the database did not contain (sufficient) information to incorporate all these metrics in the prototype. As a result there are some adjustments that require explanation;

- Because competition is a concept that depends on a lot of different factors and it is hard to classify companies over one definition. It was decided to leave this metric out of the scope. Though minimal, this could affect the outcome of the recommender-system. This is something that can be worked around using the human in the loop mechanism, in which the user can check the competition manually.
- Unfortunately the database did not contain information about all constructs that were identified using the academic papers and the onboarding course (in table 5.1). This will have influence the decision power of the recommender-system. If Unknown Group will be able to retrieve data about more of these constructs in the future, they should be added to the model.
- During the real data testing, I found that for every record of the training and test data, the 'revenue' was lacking. According to the CTO of Unknown Group, startups are often not willing to share this information. While predicting these NULL values during the testing, there was no reference information making the predictions inaccurate. I hope the dialog, as explained in section 10.2.2 could be a solution to gain more data about revenue in the future. If this is not the case, a suitable alternative should be found.
- There was decided to extract the approved and discarded score from the data model and implement these in the system as a separate route to facilitate content-based information. The reason being, that the description is also used for the trend and technology score. Another reason is that the relation with the approved and discarded companies is very case specific and will thus not apply to the data that is used to train and test the algorithm. In the future, these similarity calculations could maybe be incorporated in more extensive natural language processing machine-learning algorithms higher up in the scouting pipeline. This is explained in section10.2.1.

Against expectations, the data from the niche domains contained a lot of null values. For this reason the

operation of the recommender-system was first proven with the use of synthetic data. Later, attempts were made for predicting NULL values by the multivariate approaches as discussed in ref 5.2.2.1.

8.2 Testing

8.2.1 synthetic Data

While creating the synthetic data, the characteristics of the real data where taken into consideration. In the first test, the precision and recall were rather high(implying that my synthetic data was too clean), for this reason, 'moles' were added to the data (explained in 7.1.1). This however, barely affected the result, as the Mean Average Precision and the Matthews-Correlation Coefficient were still close to '1', making the system very promising (Figure 7.1).

8.2.2 Real Data

When testing with real data, during the test with the Iterative imputer, the machine-learning model still classified all companies as not valuable. The outcome from the KNN imputed data did classify the companies over the two classes. However, unfortunately the recommendation did not have much agreement with the reality (with an average MC score of 0,027). According to the mAP score of 0,50, about half of the predictions were correct. This score strongly relies on the precision of the 0 class because false negatives are very rare due to the dis-balance between valuable and not-valuable companies.
I strongly believe that this big difference with the synthetic data testing is explained by the available data; Because the real data was very inconsistent (read; more than half of the constructs were nullable), NULL values had to be replaced by predictions based on imputer methods. This is a remedy for data sets that contain several missing values but when this exceeds the amount of present values, the recommendation becomes unreliable. Another possible cause for misconceptions of the algorithm, is the fact that Unknown Group did not possess enough data from valuable startups in recent cases. For this reason, winners from all the priory cases were added to the test/train data. These potential startups however, could have changed over time, making them no longer classify as valuable.

8.3 Evaluation Scouting Team

The scouts all agree that the recommendation-system is a valuable addition to their scouting procedure. This is because; (i) It enables quick analysis that can be presented to the client, (ii) No specialized expertise is needed for the recommendation and (iii) It enables quantitative means for startup scouting. The scouts agree that this feature is especially useful for quick analysis in reports, scans and pitches. It does however not eliminate the extensive analysis that is required for reliable startup selection. An interesting opportunity is to approach the data more vertically, for example comparing companies on technology only. It would also be interesting to explore multi-class classification to order startups based on specific metrics or to be able to better deal with edge cases that would otherwise be validated too strict. This is explained in more detail in the further recommendations in chapter 10.

9

CONCLUSION

Building upon the results and its discussion, this section will answer the sub questions and the research question. It also evaluates the research problems, the goals and the motivation.
The research question read:

- ***RQ 1****: Can the reliability and efficiency of the startup scouting process be improved, by implementing a hybrid recommender-system to validate startups in the strategic niche domain of 'new energies'?*

In order to answer this question, the following sub-questions have been investigated.

- ***RQ 1.1****: What are the functional requirements for a data driven recommender-system that should assist startup scouting at Unknown Group?*
- ***RQ 1.2****: What are the non-functional requirements for this system?*
- ***RQ 1.3****: What knowledge, collaborative and content data can be used as an input for the hybrid recommender-system?*
- ***RQ 1.4****: What classification method should be used and how should it be created?*
- ***RQ 1.5****: How to deal with the domain challenges that are present in the venture capital domain?*
- ***RQ 1.6****: How can the effectiveness an reliability of the recommender-system be measured?*
- ***RQ 1.7****: Does the use of the recommender-system show improvement in quality and efficiency in the scouting process?*

RQ 1.1 RQ 1.2: The functional and non-functional requirements have been identified in collaboration with the CTO, the COO and the Head of Scouting of Unknown Group. These requirements have been implemented in the prototype and have been evaluated in the discussion in chapter 8. Nearly all requirements have successfully been fulfilled in the prototype of the recommender system. From the non-functional requirements, the usability has been taken into consideration while designing the solution although it requires some real user testing to optimize. Regarding the functional requirements, only two main issues remain; the implementation in the scouting portal is still in progress. The system still requires some manual actions while there are plans to completely automate the recommender-system in the future by implementing a python listener. Secondly, due to unsatisfactory quality of the data, the systems reliability and bias response could not be tested meaning that the system is not resistant to all design problems. This is the only real concern that is preventing adaption of the solution. Fortunately, a potential remedy is in the pipeline for the scouting portal which will be explained in the further recommendations in section 10.2.2.

RQ 1.3: The data for the input of the recommender system was collected based upon the academic articles and thorough analysis of the scouting onboarding course and was evaluated with the Head of scouting. The presumptive relation between these data points was schematically represented in figure 5.2. Furthermore, the actual weights and relations between these metrics was calculated with back prop-

agation of the machine-learning algorithm with supervised classification. In the future one must strive to collect information about all identified constructs to improve the datamodel.

RQ 1.4: In section 2.4 in the theory section, we learned that validating startups is a two class non-linear classification problem. In section 5.2.3 was explained how such a model could be implemented in the design with supervised classification. In section 6.3, a system was created following a tutorial on building a neural network (non-linear) two class classification.

RQ 1.5: Combining different types of systems in a hybrid recommender-system, extended with human reasoning, could potentially solve these problems. While building this hybrid recommender-system during this thesis, most of these problems have indeed been mitigated. The inconsistency of data however remains a major eyesore. In section 5.2.2.1 different methods for handling nullable data were attempted. Unfortunately the lack of sufficient data caused invalid recommendations. In the future recommendations in section 10.2.2 is explained how this problem could be solved with further development of the intelligence platform.

RQ 1.6: The reliability of the recommender system was evaluated using IR metrics as described in the results in section 7.3.1. Although no proof is present with real data, the experiment strongly suggests that the recommender-system improves the quality as it showed promising results on these IR metrics. Moreover, the mean Average Precision is 0.987 and the unbiased Matthews-Correlation score is on average 0.93, even when the data is distorted with moles. The recommender system takes about 20 minutes to present a top-N, which is a 99.7%reduction in time-span of the traditional scouting method.

RQ 1.7: The last sub question could not be tested as the data from Unknown contained too much NULL values to be evaluated using the machine-learning recommender system. The test plan that was created to answer this sub-question is given in appendix F so it can be used for further research once the data consistency is improved.

To answer RQ 1:
Efficiency has been much improved as the time-frame of selecting valuable startups is reduced by 99.7%. Though, while various human actions are automatized, others emerged. Scouts now are required to validate the companies by discarding and approving the recommendations to provide feedback for the algorithm. The actual win in efficiency should be learned over time once the system has been implemented successfully.

Reliability has not been increased as the recommendations from the real data do not agree with the real classes (resulting from the average Matthews-Correlation Coefficient of 0.027). The results of the performance test with synthetic data suggest that the recommender-system is an effective method to recognize patterns in valuable startups based on the different types of data (MCC score of 0.93 and mAP score of 0.987). Though, until the data consistency is improved, this method will not be able not improve the reliability of the scouting process as the recommendations are unreliable due to the incomplete data.

To reflect on the problem, motivation and goals:

- **Too much data:** Although the vast amount of data is more a problem while selecting companies

that fit in the strategic niche domain, the amount of startups in this domain is still too extensive to be able to thoroughly analyze them all in a short time frame. This is made possible using the recommender algorithm.

- **Human biases in scouting process:** The biases in the selection procedure in the scouting process should be reduced by implementing the recommender system, because part of the human decision making is automatized. Though the test that was established to measure the extent to which the recommender-system eliminates these biases could not be executed because of the nullable data. Moreover, because the system is trained with data in which human decision making was significant, there exists a chance that the system is trained to classify with bias.
- **Inconsistent and incomplete data:** Because nullable data can be predicted using the machine-learning nullable data techniques, missing data can be processed as well. However, the amount of missing data was too extensive for the creation of a proper recommendation. Inconsistent and incomplete data remain a major problems in the venture capital domain. Though, the future implementation of the dialog as described in section 10.2.2 is promising for reducing missing values.
- **Fast changing scouting team:** Because the scouts now only need to re-evaluate the recommended options, fewer skills are required, making this a task that can be assigned to scouts with less experience [1]. Moreover, because the HITL enables learning by use[59], the scout will learn from the system.
- **Lack of technical affinity of the scouts:** The recommender algorithm contains constructs for evaluating technical attributes of the company. As a result, it recognizes valuable technologies, taking over part of the responsibility of the scouts.
- **Hard to rank different types of companies:** Because the machine-learning classifier is able to recognize underlying relations that the scouts are not able to, this problem will be no longer an issue.
- **Time consuming scouting process:** I found that time-span of the startup selection in the scouting process can be reduced with $99,7\%$. Furthermore a list of recommendations is directly visible in the intelligence portal and could potentially be shared with the client.

As the artifact is a solution to most problems and the main goals, the implementation of the recommender-system according to the design theory, is very promising with respect to increasing quality and efficiency of the scouting process. Successful implementation would support the continuation of fast growth and a better competitive advantage. However, in order to do so, Unknown Group should improve their data consistency.

[1] Appeared from the reflection with the scouting leads in appendix G

10

FURTHER RECOMMENDATIONS

During this research I came across various suggestions for improving the artifact that was developed in this thesis. This chapter explains topics that require further investigation and provides some handholds for realizing these next steps.

10.1 Roadmap

Even though the scope of this project is to cut down the amount of startups from 200 to 20 on specific strategic niche domain, this project will act as a pilot study to include more data driven decision making in the scouting process. Once it has proven its functionality in this domain, the next step will be to implement the recommender-system in multiple strategic niche domains. From there, the planning is to include more and more functionality to the system so it can facilitate decision support from the filtering from 20.000 to 200 and later even target the whole database. This roadmap is sketched in Figure 10.1. Once this last step is implemented, the recommendation scores that are now used to classify the companies binary, could be added to the profiles of all companies. This way, even when they are not recommended, the user can take this score into consideration when validating the company. However, it is questionable whether this can be consistently used in the real use case as a lot of company profiles lack sufficient information.

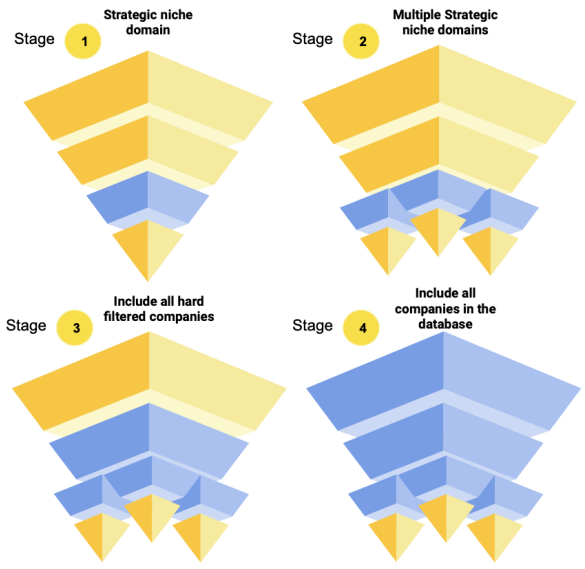


Figure 10.1: Roadmap for incorporating more steps in the scouting process.

10.2 Data

The data availability and consistency being the number one eyesore for this project has led to some further recommendations as well.

10.2.1 Constructs

The constructs that are taken as an input to the recommender system, are selected using knowledge that was obtained using the on boarding course of Unknown Group and from former research on success determinants of startups. Constructs of the three different data-types (collaborative, content and knowledge data) were selected. Unfortunately the database currently lacks information about several constructs that have been identified. In the future, Unknown Group should try to add and replenish this information by adding new data sources or by the dialog system that is explained in 10.2.2.

The recommender system that is now created could be infinitely expanded. The machine-learning algorithm will determine which constructs are useful and which are not. An interesting opportunity to explore is: to acquire more data out of the description by using Natural Language Processing. The total profile of the company could be matched with the scope of the client. This however might better fit to one of the earlier stages of the scouting process, when the pool of startups should be sharply reduced based on the scope of the client. Another interesting source for collaborative data is google trends, in which the interest in different technologies or startup types can be measured.

10.2.2 Data Quality and Consistency

One major drawback form this research is that the data that is used as the input of the algorithm was not consistent enough to provide satisfactory results. In the pipeline for the intelligence platform is to give the startups access to their own data by creating a dialog on the platform (by using forms). This way, they can make sure their data is up-to-date and complete, potentially leading to less NULL values. Because most startups want to attract capital from investors, they supposedly assure this information is elaborated and up-to-date. Once this is implemented and in use, the data of the companies in the strategic niche domains should be much more consistent. The reliability testing (calculating the IR metrics and the Matthews-Correlation) should then be repeated to see whether the system does indeed meet the high quality recommendations as it does with the synthetic data. Once the consistency of the real data is improved, the plan (in appendix F) can be executed. The output of the recommender system can be compared with the output of the manual scouting process and the differences can be evaluated using the metrics as proposed in the validation plan.

10.2.3 Data distribution

For this research there was assumed that the companies that should be validated as 'not-valuable' are uniformly distributed, meaning that on average, every score on every metric occurs equally. This choice was made based on assumptions and to simplify the problem during the development of the recommender-system. The actual distribution of the startups' constructs could be investigated to create better synthetic data. A normal distribution for example could influence the performance of the classification system.

10.3 Machine-Learning Classifier

10.3.1 Investigation for Optimal Settings Classifier

In this research, the machine-learning classifier tutorial from 'towardsdatascience.com'[1] was used. The default settings were slightly adjusted to fit the data model that was established during this research. Furthermore, slight variations in the split between test and training data were tried in order to improve the recommendation from the real data, but unsuccessfully. Once the consistency of the data is improved, an investigation to the optimal settings of the classifier could be useful in order to improve the recommendations. Additionally, the scouting team mentioned that experimenting with multi-class classifiers could also be useful for classification based on different metrics and for valuation of edge cases (section G).

The data that was handed over by the scouts of Unknown Group did not contain enough companies that were classified as useful. For this reason all companies that have ever came out of a campaign as a winner were used to train/test the algorithm using supervised classification. In the future, this should be improved. Unknown Group grows rapidly and so will data about prior cases. Other chances lie within the storage of historical data. Before the start of this research, Unknown Group only possessed the latest information about all companies in their database. However, on my advice, they started saving previous data pulls from the data sources, hereby enabling analysis based on historic data. If, in the future, we are able to investigate successful companies' previous data, we can make predictions for the success of current startups. This knowledge type data could improve the recommendations.

10.4 Feedback Loop

In this thesis, user feedback is collected through re-evaluation of the recommendations. However, this feedback loop is not completely automated and implemented into the intelligence portal. In the future however, this could be implemented using a python listener functionality attached to FireBase[2]. Alterations in the feedback data cause the script to run automatically, making this human action no longer necessary.

10.4.1 Implicit User Behavior

The feedback data is currently explicitly supplied by the user. However, there are methods for implicitly monitoring user behavior. For example, the export to SalesForce from the winners of a campaign, could be used for retraining the algorithm. Additionally, click behavior, saving to lists and user notes to company profiles could be useful sources of user behavior (content recommender).

10.4.2 Adjustable Thresholds

Usage will determine what the optimal structure of the feedback-mechanism will look like. It could be interesting to explore adjustable thresholds (appendix G).

[1] https://towardsdatascience.com/deep-learning-with-python-neural-networks-complete- tutorial-6b53c0b06af0

[2] https://firebase.google.com/docs/firestore/query-data/listen

10.5 Usage Recommendation

The potential use and valuable contribution of the recommender-system was investigated with an evaluation with the scouting team of Unknown Group.

10.5.1 Radar Chart, Analyzing Tools

In order to ground the recommendations, the system currently outputs a radar chart with the scores to all constructs of which the recommendation is based. This metric should be extended with more explanation about the constructs, their weights and preferably even the possibility to adjust the weights. In the future, many more graphs and visualizations of the data could be added that help in explaining the value of the recommended items. The explanation metric currently does not work on the real data because of time-limitations of this thesis. This however can be achieved in the same manner that the ids from the recommender are parsed to the interface via FireBase. A struct type data object[3] with all constructs and their values should be uploaded along with the ids in the Firebase folder. These can then be accessed in the front end and synchronized in real time with the radar chart.

Another important topic is the value of adding recommendation scores to all company profiles in the database. This could help scouts in validating companies during the traditional scouting method. Unfortunately, due to the inconsistency, properly scoring each company will currently not be possible. Once the dialog is implemented (as explained in section 10.2.2), this topic could be reconsidered.

10.5.2 Onboarding

Once the recommender-system is entirely integrated in the platform and runs on the real data, the scouts should start using it in their daily routine. First some extensive user-testing is required to investigate how the system can be used most effectively and which new tasks emerge around the recommender-system. Second, because the machine learns from the user and the other way around[59], some initial interaction is required to overcome the cold-start problem. Once this has been investigated, it is important to give all the scouts a course in:

- How to use the recommender-system.
- How to provide feedback.
- Which additional evaluation steps are important after the recommendation is presented in the portal.
- How to accomplish other additional tasks that emerge from the recommender-system.

10.6 Contribution to Literature

The application-view architecture that is established in this research could potentially be implemented for other domains. From the literature review that was executed prior to this research was found that there exists a gap in information of an approach to combine content, knowledge and collaborative in

[3] https://docs.informatica.com/data-integration/powercenter/10-2/developer-tool-guide/data-type-reference/transformation-data-types/complex-data-types/struct-data-type.html

one hybrid recommender-system. This thesis has developed a design theory for a hybrid recommender-system for the venture capital domain, which has been enhanced with requirements from the field and put to practice according to the application architecture in Figure 6.1. This design theory and architecture can especially be useful for other cases where there exists little information about internal users. By carefully selecting the constructs, the machine-learning recommender-system can classify items based on these the three datatypes without requiring a lot of user-data. Examples for domains in which such a system could be used, are job-application procedures or for scouting campaigns in other domains, for example the football industry. Especially because these fields also require grounded decision making.

Bibliography

[1] 2018 reform of eu data protection rules.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

[3] Gilseung Ahn, Hyungseok Yun, Sun Hur, and Si-Yeong Lim. A time-series data generation method to predict remaining useful life. *Processes*, 9(7):1115, 2021.

[4] Vito Walter Anelli, Alejandro Bellogín, Tommaso Di Noia, and Claudio Pomo. Reenvisioning the comparison between neural collaborative filtering and matrix factorization. In *Fifteenth ACM Conference on Recommender Systems*, pages 521–529, 2021.

[5] Ofer Arazy, Nanda Kumar, and Bracha Shapira. A theory-driven design framework for social recommender systems. *Journal of the Association for Information Systems*, 11(9):2, 2010.

[6] Javier Arroyo, Francesco Corea, Guillermo Jimenez-Diaz, and Juan A Recio-Garcia. Assessment of machine learning performance for decision support in venture capital investments. *Ieee Access*, 7:124233–124243, 2019.

[7] Pablo Becker, Guido Tebes, Denis Peppino, and Luis Olsina. Applying an improving strategy that embeds functional and non-functional requirements concepts. *Journal of Computer Science and Technology*, 19(2):e15–e15, 2019.

[8] Fabio Bertoni, Massimo G Colombo, Anita Quas, and Francesca Tenca. The changing patterns of venture capital investments in europe. *Journal of Industrial and Business Economics*, 46(2):229–250, 2019.

[9] Kees Tjalle Dries Bosch. Implementing a recommender-system in the venture capital sector, a literature review. 2021.

[10] Svetla Boytcheva and Andrey Tagarev. Company investment recommendation based on data mining techniques. In *International Conference on Business Information Systems*, pages 73–84. Springer, 2019.

[11] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of library and information systems*, 69(Supplement 32):175–186, 2000.

[12] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[13] Robin Burke and Maryam Ramezani. Matching recommendation technologies and domains. In *Recommender systems handbook*, pages 367–386. Springer, 2011.

[14] Diego Camelo Martinez. Startup success prediction in the dutch startup ecosystem. 2019.

[15] Gürol Canbek, Tugba Taskaya Temizel, and Seref Sagiroglu. Benchmetrics: a systematic benchmarking method for binary classification performance metrics. *Neural Computing and Applications*, 33(21):14623–14650, 2021.

[16] Emilio Carrizosa and Dolores Romero Morales. Supervised classification and mathematical optimization. *Computers & Operations Research*, 40(1):150–165, 2013.

[17] Stefano Cassella, Benjamin Golez, Huseyin Gulen, and Peter Kelly. Horizon bias and the term structure of equity returns. *Available at SSRN 3328970*, 2021.

[18] Francesco Corea. Ai and venture capital. In *An Introduction to Data*, pages 101–110. Springer, 2019.

[19] Douglas Cumming and Na Dai. Local bias in venture capital investments. *Journal of empirical finance*, 17(3):362–380, 2010.

[20] Luis M De Campos, Juan M Fernández-Luna, Juan F Huete, and Miguel A Rueda-Morales. Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks. *International journal of approximate reasoning*, 51(7):785–799, 2010.

[21] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[22] Christian Desrosiers and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. *Recommender systems handbook*, pages 107–144, 2011.

[23] Martínez-Plumed Fernando, Ferri Cèsar, Nieves David, and Hernández-Orallo José. Missing the missing values: The ugly duckling of fairness in machine learning. *International Journal of Intelligent Systems*, 36(7):3217–3258, 2021.

[24] Sumantra Ghoshal, Harry Korine, and Gabriel Szulanski. Interunit communication in multinational corporations. *Management science*, 40(1):96–110, 1994.

[25] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[26] Paul A Gompers, Will Gornall, Steven N Kaplan, and Ilya A Strebulaev. How do venture capitalists make decisions? *Journal of Financial Economics*, 135(1):169–190, 2020.

[27] Tor Grønsund and Margunn Aanestad. Augmenting the algorithm: Emerging human-in-the-loop work configurations. *The Journal of Strategic Information Systems*, 29(2):101614, 2020.

[28] Morten T Hansen. The search-transfer problem: The role of weak ties in sharing knowledge across organization subunits. *Administrative science quarterly*, 44(1):82–111, 1999.

[29] Bernd Heinrich, Marcus Hopf, Daniel Lohninger, Alexander Schiller, and Michael Szubartowicz. Data quality in recommender systems: the impact of completeness of item content data on prediction accuracy of recommender systems. *Electronic Markets*, 31(2):389–409, 2021.

[30] Jonathan L Herlocker, Joseph A Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250, 2000.

[31] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[32] Boyoung Kim, Hyojin Kim, and Youngok Jeon. Critical success factors of a design startup business. *Sustainability*, 10(9):2981, 2018.

[33] Pigi Kouki, Shobeir Fakhraei, James Foulds, Magdalini Eirinaki, and Lise Getoor. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 99–106, 2015.

[34] Juntao Liu, Caihua Wu, and Wenyu Liu. Bayesian probabilistic matrix factorization with social relations and item contents for recommendation. *Decision Support Systems*, 55(3):838–850, 2013.

[35] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. *Recommender systems handbook*, pages 73–105, 2011.

[36] Fabiana Lorenzi and Francesco Ricci. Case-based recommender systems: A unifying view. In *IJCAI Workshop on Intelligent Techniques for Web Personalization*, pages 89–113. Springer, 2003.

[37] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296, 2011.

[38] Roger C Mayer, James H Davis, and F David Schoorman. An integrative model of organizational trust. *Academy of management review*, 20(3):709–734, 1995.

[39] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.

[40] David McSherry. Explanation in recommender systems. *Artificial Intelligence Review*, 24(2):179–197, 2005.

[41] Eli Pariser. *The filter bubble: What the Internet is hiding from you*. Penguin UK, 2011.

[42] Margot Peeters, Mariëlle Zondervan-Zwijnenburg, Gerko Vink, and Rens Van de Schoot. How to handle missing data: A comparison of different approaches. *European Journal of Developmental Psychology*, 12(4):377–394, 2015.

[43] Ken Peffers, Tuure Tuunanen, Marcus A Rothenberger, and Samir Chatterjee. A design science research methodology for information systems research. *Journal of management information systems*, 24(3):45–77, 2007.

[44] Sebastian Raschka, P Linear, R Gaussian, and Locally-Linear Embedding LLE. Kernel tricks and nonlinear dimensionality reduction via rbf kernel pca. *Blog, September*, 2014.

[45] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[46] Christina Maria Schmidt. *The impact of artificial intelligence on decision-making in Venture Capital Firms*. PhD thesis, 2019.

[47] Gunnar Schröder, Maik Thiele, and Wolfgang Lehner. Setting goals and choosing metrics for recommender system evaluations. In *UCERSTI2 workshop at the 5th ACM conference on recommender systems, Chicago, USA*, volume 23, page 53, 2011.

[48] Veronique AJM Schutjens and Egbert Wever. Determinants of new firm success. *Papers in Regional Science*, 79(2):135–159, 2000.

[49] Boris Sharchilev, Michael Roizner, Andrey Rumyantsev, Denis Ozornin, Pavel Serdyukov, and Maarten de Rijke. Web-based startup success prediction. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 2283–2291, 2018.

[50] Upendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, 1995.

[51] Shivalik Singh and MH Bala Subrahmanya. Quantum of finance obtained by tech startups over the lifecycle: an analysis of its determinants. *International Review of Applied Economics*, pages 1–18, 2021.

[52] Barry Smyth. Case-based recommendation. In *The adaptive web*, pages 342–376. Springer, 2007.

[53] Richard Hermann Anselmo Stahl. Leveraging time-series signals for multi-stage startup success prediction. 2021.

[54] Thomas Stone, Weinan Zhang, and Xiaoxue Zhao. An empirical study of top-n recommendation for venture finance. In *Proceedings of the 22nd ACM international conference on information & knowledge management*, pages 1865–1868, 2013.

[55] Lingam Sunitha and M Bal Raju. Multi-class classification for large datasets with optimized svm by non-linear kernel function. In *Journal of Physics: Conference Series*, volume 2089, page 012015. IOP Publishing, 2021.

[56] Gabriel Szulanski. Exploring internal stickiness: Impediments to the transfer of best practice within the firm. *Strategic management journal*, 17(S2):27–43, 1996.

[57] Mona Taghavi, Kaveh Bakhtiyari, and Edgar Scavino. Agent-based computational investing recommender system. In *Proceedings of the 7th ACM conference on recommender systems*, pages 455–458, 2013.

[58] Jerm Watt. 10.4 nonlinear two-class classification - github pages.

[59] Fons Wijnhoven. Organizational learning for intelligence amplification adoption: Lessons from a clinical decision support system adoption project. *Information Systems Frontiers*, pages 1–14, 2021.

[60] Fons Wijnhoven and Michel Brinkhuis. I nternet information triangulation: Design theory and prototype evaluation. *Journal of the Association for Information Science and Technology*, 66(4):684–701, 2015.

[61] Guang Xiang, Zeyu Zheng, Miaomiao Wen, Jason Hong, Carolyn Rose, and Chao Liu. A supervised approach to predict company acquisition with factual and topic features using profiles and news articles on techcrunch. In *Sixth International AAAI Conference on Weblogs and Social Media*, 2012.

[62] Hao Zhong, Chuanren Liu, Junwei Zhong, and Hui Xiong. Which startup to invest in: a personalized portfolio strategy. *Annals of Operations Research*, 263, 04 2018.

[63] Dávid Zibriczky12. Recommender systems meet finance: a literature review. In *Proc. 2nd Int. Workshop Personalization Recommender Syst*, pages 1–10, 2016.

A

Appendix - Code

The code for the recommender system is quite extensive and is therefore publicly accessible through the following weblinks:
Complete code algorithm (Also for testing real data(Test 4)):
`https://colab.research.google.com/drive/1mXlqkMayALHVEe_N0jPzI8v9UGtkBJwp?usp=sharing`
Code for testing dummy data (Test 1) and for testing the feedback loop (Test2):
`https://colab.research.google.com/drive/1r7Nvyo5lEtpJ70f1oCzXm5r1a_9sVR0a?usp=sharing`

Because the query for collecting the data is referred to multiple times, this code is added to appendix C

B

Appendix - Questions for validation metric

Schröder and Thiele[47] provide a list of questions that should help to choose a proper validation metric. These questions are answered below to identify the goal of the recommender system and the corresponding accuracy metrics to later be able to measure the success of the recommender system.

1. "Is there a distinction between rated and unrated items?"
   This question determines the need for predictive accuracy metrics. When all items are rated, there is no need for such metrics. However, in the case of the startup recommender, not all items are rated.
2. "Are items rated on a numerical or binary scale?"
   The items are rated numerically but the choice if the startup is relevant or not is binary. For this reason the classification should be binary.
3. "Are users interested in rating predictions or only in the top-ranked items?"
   At first glance, the top-ranked items are most important as the 'irrelevant' items will not be displayed in the top-N list. However, it can be, that later will be decided to include the relevance score to the company profiles but this is not for the scope of this project.
4. "Is a limited list of top-ranked items shown?"
   For the startup recommender, this is the case. This means that a recommender that measures the overall accuracy of the overall prediction of the ranking is not suitable as the exact ratings of the unclassified items are not relevant to the user of the recommender system.
5. "Do the recommended items have or imply an order?"
   In principle they do, however, the exact order is not very important as all the startups in the top-N list will be evaluated manually. This means that metrics that do not consider the exact order of the data set are appropriate to use. An example of such metrics are the basic information retrieval metrics (IR) such as Recall, Precision, Markedness and Informedness.
6. "How fast does the user's interest in lower ranked items decay?"
   As mentioned in the answer to question 3, for now, the user is not interested in the lower ranked items. For this reason metrics like *Area Under the Curve (AUC)*[1] or *Kendall's tau measure*[2] are not relevant. Instead, metrics that lay their focus on the first recommendations such as mAP or GmAP are more suitable.

Following from the guidelines above, we are dealing with a classification problem and thus require classification accuracy metrics. These metrics do apply to our type of recommender-system as this type of validation tries to measure the successful decision making capacity of the recommender system. In this technique, the amount of correct and incorrect classifications for relevant and irrelevant items is measured. Schröder and Thiele[47] argue that this type of validation is useful when the goal of the rec-

[1] https://flowthytensor.medium.com/some-metrics-to-evaluate-recommendation-systems-9e0cf0c8b6cf
[2] https://statistics.laerd.com/spss-tutorials/kendalls-tau-b-using-spss-statistics.php

ommender is to convince users to make a certain decision. Which is true in this particular case.

C

Appendix - Query

C.1 Test/Train Data

```
SELECT
  DISTINCT(uuid),
  mapping.cb,
  mapping.tx,
  mapping.sf,
  COALESCE(map.name.cb, map.name.tx, map.name.sf)as name,

  txn.nrAwards,
  txn.revenue,
  txn.news,
  crunch.total_funding_usd,
  crunch.num_funding_rounds,
  crunch.founded_on,
  crunch.age,
  crunch.fundingperage,
  prevComp.nrofPreviousComp,
  nrDegrees.num_of_degrees,
  sfexport.status,
  crunch.short_description,
   COALESCE(map.short_description.cb, map.short_description.tx,
   map.short_description.sf)as short_description,
  COALESCE(map.employee_count.cb, map.employee_count.tx,
  map.employee_count.sf) as num_employees,
  COALESCE(map.total_funding_rounds.cb,
  CAST(map.total_funding_rounds.tx AS INT))as funding_rounds,
   COALESCE(map.total_funding.cb.amount,
   CAST(map.total_funding.tx.amount AS INT))as Total_funding,
   COALESCE(map.score.tx, map.score.cb)as Score

FROM
  `maximal-cider-305010.intermediates.int_companies_es__mar_22` map
LEFT JOIN (
  SELECT
    status,
    SFuuid
  FROM
    `maximal-cider-305010.Kees.TestSalesforceExport`)sfexport
ON
  sfexport.SFuuid = map.mapping.sf
LEFT JOIN (
  SELECT
    id,
    name,
    awardsInfo.awardsCount AS nrAwards,
    annualRevenue.amount AS revenue,
    newsInfo.numberOfNewsArticlesLastYearGrowthPercent AS news
  FROM
    `maximal-cider-305010.tracxn_argo.companies__mar_22`)txn
ON
  txn.id = mapping.tx
LEFT JOIN (
  SELECT
    uuid AS cbId,
    name,
    total_funding_usd,
    num_funding_rounds,
    founded_on,
    CURRENT_DATETIME() AS now,
    DATE_DIFF(CURRENT_DATE, founded_on, YEAR) AS age,
    IEEE_DIVIDE(total_funding_usd,
      CASE
        WHEN DATE_DIFF(CURRENT_DATE, founded_on, YEAR) IS NULL THEN 1
      ELSE
      DATE_DIFF(CURRENT_DATE, founded_on, YEAR)
```

```
    END
      ) AS fundingperage,
    short_description
  FROM
    `maximal-cider-305010.crunchbase_argo.organizations__jan_22`)crunch
ON
  crunch.cbId = mapping.cb
LEFT JOIN (
  SELECT
    COUNT(*) nrofPreviousComp,
    org_uuid
  FROM
    `maximal-cider-305010.crunchbase_argo.jobs__mar_22`
  LEFT JOIN (
    SELECT
      COUNT(*)-1 AS nrOfPrevComp,
      person_uuid AS p_uuid
    FROM
      `maximal-cider-305010.crunchbase_argo.jobs__mar_22`
    WHERE
      (title = 'Founder'OR title = 'founder'
        OR title = 'Co-Founder'
        OR title = 'Co-founder'
        OR title = 'co-founder')
    GROUP BY
      person_uuid)nrperPerson
  ON
    nrperPerson.p_uuid = person_uuid
  GROUP BY
    org_uuid)prevComp
ON
  prevComp.org_uuid = mapping.cb
LEFT JOIN (
  SELECT
    organizations.uuid AS uuid1,
    organizations.name AS name,
    ARRAY_LENGTH((
      SELECT
        ARRAY_CONCAT_AGG(items)
      FROM
        UNNEST(ARRAY(
          SELECT
            degrees
          FROM
            UNNEST(people.items)))) AS num_of_degrees,
    ARRAY(
    SELECT
      degrees
    FROM
      UNNEST(people.items)) AS test_2
  FROM
    `maximal-cider-305010.crunchbase_argo.organizations__mar_22`
    organizations
  LEFT JOIN (
    SELECT
      people.featured_job_organization_uuid,
      ARRAY_AGG(STRUCT(degrees)) AS items
    FROM
      `maximal-cider-305010.crunchbase_argo.people__mar_22` people
    LEFT JOIN (
      SELECT
        person_uuid,
        ARRAY_AGG(degrees) AS items
      FROM
        `maximal-cider-305010.crunchbase_argo.degrees__mar_22` degrees
      GROUP BY
        person_uuid ) degrees
    ON
      people.uuid = degrees.person_uuid
    GROUP BY
      people.featured_job_organization_uuid ) people
  ON
    organizations.uuid = people.featured_job_organization_uuid )nrDegrees
ON
  nrDegrees.uuid1= mapping.cb
WHERE
  mapping.sf IN (
  SELECT
    SFuuid
  FROM
    `maximal-cider-305010.Kees.TestSalesforceExport`)
```

C.2 Validate Data

```
SELECT
  DISTINCT(uuid),
  mapping.cb,
  mapping.tx,
  mapping.sf,
  COALESCE(map.name.cb, map.name.tx, map.name.sf)as name,
  txn.nrAwards,
  txn.revenue,
  txn.news,
  crunch.total_funding_usd,
  crunch.num_funding_rounds,
  crunch.founded_on,
  crunch.age,
  crunch.fundingperage,
  prevComp.nrofPreviousComp,
  nrDegrees.num_of_degrees,
  sfexport.status,
  crunch.short_description,
   COALESCE(map.short_description.cb, map.short_description.tx,
   map.short_description.sf)as short_description,
  COALESCE(map.employee_count.cb, map.employee_count.tx,
  map.employee_count.sf) as num_employees,
  COALESCE(map.total_funding_rounds.cb,
  CAST(map.total_funding_rounds.tx AS INT))as funding_rounds,
  COALESCE(map.total_funding.cb.amount,
  CAST(map.total_funding.tx.amount AS INT))as Total_funding,
  COALESCE(map.score.tx, map.score.cb)as Score
FROM
  `maximal-cider-305010.intermediates.int_companies_es__mar_22` map
LEFT JOIN (
  SELECT
    status,
    SFuuid
  FROM
    `maximal-cider-305010.Kees.TestSalesforceExport`)sfexport
ON
  sfexport.SFuuid = map.mapping.sf
LEFT JOIN (
  SELECT
    id,
    name,
    awardsInfo.awardsCount AS nrAwards,
    annualRevenue.amount AS revenue,
    newsInfo.numberOfNewsArticlesLastYearGrowthPercent AS news
  FROM
    `maximal-cider-305010.tracxn_argo.companies__mar_22`)txn
ON
  txn.id = mapping.tx
LEFT JOIN (
  SELECT
    uuid AS cbId,
    name,
    total_funding_usd,
    num_funding_rounds,
    founded_on,
    CURRENT_DATETIME() AS now,
    DATE_DIFF(CURRENT_DATE, founded_on, YEAR) AS age,
    IEEE_DIVIDE(total_funding_usd,
      CASE
        WHEN DATE_DIFF(CURRENT_DATE, founded_on, YEAR) IS NULL THEN 1
      ELSE
      DATE_DIFF(CURRENT_DATE, founded_on, YEAR)
    END
      ) AS fundingperage,
    short_description
  FROM
    `maximal-cider-305010.crunchbase_argo.organizations__jan_22`)crunch
ON
  crunch.cbId = mapping.cb
LEFT JOIN (
  SELECT
    COUNT(*) nrofPreviousComp,
    org_uuid
  FROM
    `maximal-cider-305010.crunchbase_argo.jobs__mar_22`
  LEFT JOIN (
    SELECT
      COUNT(*)-1 AS nrOfPrevComp,
      person_uuid AS p_uuid
    FROM
      `maximal-cider-305010.crunchbase_argo.jobs__mar_22`
```

```
    WHERE
      (title = 'Founder'OR title = 'founder'
        OR title = 'Co-Founder'
        OR title = 'Co-founder'
        OR title = 'co-founder')
    GROUP BY
      person_uuid)nrperPerson
  ON
    nrperPerson.p_uuid = person_uuid
  GROUP BY
    org_uuid)prevComp
ON
  prevComp.org_uuid = mapping.cb
LEFT JOIN (
  SELECT
    organizations.uuid AS uuid1,
    organizations.name AS name,
    ARRAY_LENGTH((
      SELECT
        ARRAY_CONCAT_AGG(items)
      FROM
        UNNEST(ARRAY(
          SELECT
            degrees
          FROM
            UNNEST(people.items)))) AS num_of_degrees,
    ARRAY(
    SELECT
      degrees
    FROM
      UNNEST(people.items)) AS test_2
  FROM
    `maximal-cider-305010.crunchbase_argo.organizations__mar_22`
    organizations
  LEFT JOIN (
    SELECT
      people.featured_job_organization_uuid,
      ARRAY_AGG(STRUCT(degrees)) AS items
    FROM
      `maximal-cider-305010.crunchbase_argo.people__mar_22` people
    LEFT JOIN (
      SELECT
        person_uuid,
        ARRAY_AGG(degrees) AS items
      FROM
        `maximal-cider-305010.crunchbase_argo.degrees__mar_22` degrees
      GROUP BY
        person_uuid ) degrees
    ON
      people.uuid = degrees.person_uuid
    GROUP BY
      people.featured_job_organization_uuid ) people
  ON
    organizations.uuid = people.featured_job_organization_uuid )
    nrDegrees
ON
  nrDegrees.uuid1= mapping.cb
WHERE
  mapping.sf IN (
  SELECT
    SFuuid
  FROM
    `maximal-cider-305010.Kees.ENI_STARTUP_raw`)
```

D

Appendix - Export SalesForce

```
import pandas_gbq
from google.cloud import bigquery
import pandas as pd
import pickle

client = bigquery.Client()

#Query for adding more '1' marked companies.
query_job = client.query(
"""
SELECT DISTINCT(Account__c), Name, Order__c,
Finalist__c, Winner__c, IF(Winner__c = true, 1, 0) as class,
FROM `maximal-cider-305010.salesforce_argo.startup_participant__apr_22`
WHERE (((Order__c = '8011n00000cXugxAAC'
OR Order__c = '8011n00000cXRGCAA4' OR Order__c = '8011n00000cWXX8AAO'
OR Order__c = '8011n00000J3f98AAB'
OR Order__c = '8010Y000000AmqnQAC')
OR Winner__c = true)AND Account__c != '0011n00002OS830AAD')
"""
)

df = pd.DataFrame()
df2 = pd.DataFrame()
for row in query_job:
        df3 = pd.DataFrame([[row[0], row[1],row[4], row[5]]],
        columns=['SFuuid','Name', 'class','status'])
        df2 = df2.append(df3)
print(df2.head())

uuidlist = ['0010Y00000niDuoQAE','0011n000029e35fAAA',
'0010Y00001GZbXNQA1','0011n00002IIBv4AAH','0011n00002T71m5AAB',
'0010Y00000niDuoQAE']
statuslist = [0,0,0,1,0,1]
df['SFuuid'] = df2['SFuuid']
df['status'] = df2['status']
df['name'] = df2['Name']


print(df.info())
pandas_gbq.to_gbq(df, 'Kees.TestSalesforceExport', project_id=None,
                  chunksize=None, reauth=False,
                  if_exists='replace', auth_local_webserver=False,
                  table_schema=None, location=None,
                  progress_bar=True, credentials=None)
```

E

Appendix - Code Snippets

```
// Get all profiles stored in firebase
useEffect(() => {

  if (user) {
    const dbRef = db.ref(`recommender/${user.uid}/${type}/`);
    dbRef.orderByChild('name').equalTo(listName).on('value', (snapshot) => {
      if (snapshot.exists()) {
        const savedObj = {
          ids: Object.values(snapshot.val())[0],
          key: Object.keys(snapshot.val())[0],
        }
        console.log("savedProfile", savedObj)
        setSavedProfiles(savedObj);
      } else {
        setSavedProfiles(false);
        console.log("no Profiles")
        // setLoading(false);
      }
    }, (error) => {
      console.log('firebase error: ', error);
    });
  }
}, [user, type, listName, value]);
```

Figure E.1: Get all profiles from the recommender list in FireBase.

```
{(hasCompanies && !loading) ? (
  profileDataCompany.map((item: any, key: any) => (
    <>
      <ResultCompany {...{ ...item, fireBaseParentKey, selected, setSelected, listName }} key={key} kid={key} />
    </>
  ))
```

Figure E.2: Parse data and create an instance of component 'Result Company' for every profile in FireBase database.

```
<IconButton edge="end" aria-label="delete" onClick={() => { handleApproved(_id)}}>
  <AddTaskIcon />
</IconButton>
```

Figure E.3: React code for approve button in the 'ResultCompany' component.

```
const handleDiscarded = (data:any) => {

  if (user) {
    //reference to database
    const dbRef = db.ref(`recommender/${user.uid}/${type}/Discarded`);
    dbRef.get().then((snapshot) => {
      //If the datarecord 'Discard' already exists
      if (snapshot.val() !== null) {
        const objfb = snapshot.val().ids;
        //In case multiple ids are parsed
        const combined = { ...objfb, [data]: data };
        const obj = {
          ids: combined,
          name: 'Discarded'
        };
        //Update database with object
        db.ref(`recommender/${user.uid}/${type}/Discarded`)
          .update(
            obj,
          );
          //if the datarecord Discard does not yet exist
      } else {
        //create the object
        const obj = {
          ids: {
            [data]: data,
          },
          name: 'Discarded',
        };
        //create record/update with object
        db.ref(`recommender/${user.uid}/${type}/Discarded`)
          .update(
            obj,
          );
      }
    });
  };//Delete record from recommender list
  handleDelete()
}
```

Figure E.4: Discard button behavior.

```
import ReactApexChart from 'react-apexcharts';

const Apexchart = ({
    data = {
        series: [{
          name: 'Berger Neurorobotics',
          data: [80, 50, 30, 40, 100, 20, 44, 91,22, 45, 78, 90],
        }, {
          name: 'Hexastorm',
          data: [20, 30, 40, 80, 20, 80, 11, 22, 68, 83, 45, 62],
        }, {
          name: 'CLARA Swiss Technologies',
          data: [44, 76, 78, 13, 43, 10, 32, 32, 55, 66 ,31, 96],
        }, {
          name: 'Pin Press',
          data: [21, 11, 88, 54, 61, 90, 20, 44, 31, 11, 25, 24],
        }],
        options: {
          chart: {
            height: 350,
            type: 'radar',
            dropShadow: {
              enabled: true,
              blur: 1,
              left: 1,
              top: 1
            }
          },
          title: {
            text: 'Radar Chart - Recommendations'
          },
          stroke: {
            width: 2
          },
          fill: {
            opacity: 0.1
          },
          markers: {
            size: 0
          },
          xaxis: {
            categories: ['nr of Degrees', 'Revenue', 'Funding amount', 'Tech', 'Trend', 'Experience', 'Discarded similarity', 'Approved similarity', 'Growth NewsItems', 'Awards', 'num_employees','Age']
          }
        }
      }

};Record<string, any>) => {

    return (
        <>

        <div id="chart">
        <ReactApexChart options={data.options} series={data.series} type="radar" height={350} />
        </div>
```

Figure E.5: Implementation code of the radar chart from the Apexcharts library. (Based on dummy data).

F

Appendix - Biases in Startup Scouting

The main problems of this research were the inefficient navigation through the vast amount of data and the unreliability of the investors choices in startup screening, due to biases. Therefore it is important to validate the recommender-system on its performance on these topics. In order to do so, this plan for evaluating the recommender-system's response to human biases was created. However, due to the inconsistency of data, no sufficient recommendations could be derived. For this reason, this section was left out of the report. Once the data consistency has been improved, this plan can still be executed to validate the biases from output of the recommender-system.
The ability to overcome the investors biases however, is more difficult to measure and the recommenders' performance with respect to each bias should be measured individually:

- Over-confidentiality and availability could be measured by the inclusiveness of the pics in the recommender process. A survey could be held, that targets to investigate whether the system recommends items that scouts would not have found themselves. The bias is caused by giving priority to items that are on top of the mind. The metric for this bias is the ratio of items in the the top-N that would not have been selected by the scouts without the recommender-system.

$$OCperformance = \frac{newItems}{totalItems} \tag{F.1}$$

- Representative bias, the halo-effect and over-confident bias, should be overcome by using a knowledge based recommender, because characteristics are scored based on theory (instead of doing own research and achieving a skewed opinion due to positive stories and news). The algorithm provides a more objective approach that does not rely on human originated communication. This can be measured by comparing base-line picks to the classification of these picks in the recommender-system. In theory, this bias should result in occurrences where startups are incorrectly classified as high value in the base line selection. Even though this metric can indicate the recommender-systems strength in overcoming the representation bias, malfunction of the recommender-system can also produce mismatches between the recommender and the baseline. In the future, when scores are added to the platform instead only a binary classifier(**??**), the relative distance can be used to work around this principle. This is because a small mismatch does still not explain a big relative distance.

$$Representative = RelativePositionDifferenceOfFalsePositivesInBaseline \tag{F.2}$$

- The confirmation bias is partly covered by the recommender-system as it recommends startups without relying on pre-existing believes. However, with the human in the loop method it could even be turned to an advantage when the list of recommendations contains a startup that is pre-believed to be successful. Therefore, I believe that validating the performance on this bias does not really apply for the recommender-system. Nevertheless, including previous picks in the system a good idea as this is content data.

- The performance of the algorithm on overcoming the bias of information overload can be measured. By comparing the average currently processed information with the amount of startups that can be validated to the use of the recommender-system in a particular time-frame.

$$ImprovedAnalyzedItemRate = \frac{NewAnalysisCapacity - OldAnalysisCapacity}{OldCapacity} \tag{F.3}$$

- Cumming et al.[19] describe a method on measuring the performance on the home or local bias. Their measure metric sounds; "The percentage difference between the weighted-average distance of the actual investment portfolio for each VC and the weighted-average distance of a hypothetical benchmark portfolio." A similar method could be used for measuring the halo-effect.
- As has been mentioned earlier, horizon Bias in my opinion does not really apply to our problem as this has something to do with risk assessment and the accompanied investment amount. This recommender-system aims to recommend the top-N best startups, risk assessment and financial considerations are the next step (contract negotiation) and are out of this scope.
- Because the recommender-system does only include previous successes of the founders and not their characteristics, the recommender-system should be able to overcome the affinity biases. However, as mentioned previously; Unknown Group names the collaboration with the founders as an important factor for startup success. This raises the question whether affinity bias is a real problem. Therefore I decide to exclude this bias from the scope of the research.

G

Appendix - Interview Scouting Team

Questions Interview	Answers: Head of Scouting (Jasper Warmenhoven	Scouting lead 1 (Evangelos Kostos)	Scouting lead 2 (Noah Paul)	General conclusion
Is the visualization of the metrics in the radarchart self-explaining and sufficient to explain the choice of the recommender-system? Can you motivate your answer?	I believe so, although a small document with how certain elements are determined would be helpful. Most are self-explanatory, but some are more open to interpretation, such as: Trend & Tech	No, I would also like to understand the relative gravity of each metric. Maybe that's not something that needs to be explained every time, but it's definitely something that we have to understand and possibly adjust. For example, in some cases the funding and revenue might be more important than tech, while other times the awards and tech might be the most important. Other than that it's nice to see it in a radar, and it's actually be the most important because our choices are very often qualitative and can't necessarily be quantified easily.	I believe the visualization is solid however certain items require additional explanation to be useful. For instance, the tech variable is quite ambiguous and could refer to many things, the same goes for nr of degrees, Approved similarity and Discarded similarity. In order for these variables to be actionable there needs to be some extra context provided and there computation explained. If this is not the case the information is superficial.	The radar chart visualization is an effective way to display the scores on each metric. However, the scouts mention that they require more information. The explanation could be improved by adding the weights of the constructs (their relative importance). Furthermore, not all metrics are self-explaining. Some description could help to clarify what the score is based upon.
Is the recommender system, as an advice, a valuable addition to your current scouting activities? Can you explain why?	Yes, I see it function in 2 ways. First, as a starting point if we need some quick scouting for examples for a presentation so that we don't have to create a full search and do a lot of filtering and evaluating ourselves. Second, as a way to ensure we have found everything, so after our "traditional" scouting it can help to see if we missed anything.	Yes. I would like to have a small recommendation, especially for quick additions and in cases where we need to give a quick recommendation to a client. Additionally, each scout is specialised in a different field so it would actually be nice to get recommendations for fields where we are not experts based on our colleagues preferences.	Yes, the advice can be very valuable as it provides points of reference that are usually not taken into account quantitively but rather evaluated by the scout through qualitative means. This is especially true for items such as: Tech, awards, trend, news.	The scouts all agree that the recommendation system is a valuable addition to their scouting procedure. This is because; (i) It enables quick analysis that can be presented to the client, (ii) No specialized expertise is needed for the recommendation and (iii) It enables quantitative means for startup scouting.
Can you think of other applications within your business activities, in which (parts of) this system could be a valuable contribution? Would you rather like to receive the recommendations based on the similarity with the approved and discarded companies in a	Could be applicable to founders also (simplified verions), IF we want to take that approach	As mentioned above, in my opinion, most likely, this would be used mostly for quick additions. These can be used in reports, or solution scans or small pitches to our clients. However, take this with a grain of salt because we really need to test the system before we make any final decisions. As I mentioned above, probably the most important metric is indeed the similarity one. If we can't make the metric gravity adjustable, then a separate folder would indeed be a good solution.	Yes, if this system was available per technology/vertical instead of startup that would be of a lot of value for various activities. For instance providing data on technologies in this format would allow for comparisons and evaluating them for different applications. This would be useful for all reports and sales intelligence when pitching clients.	The scouts agree that this feature is especially useful for quick analysis in reports, scans and pitches. It does however not eliminate the extensive analysis that is required for reliable startup selection. An interesting addition is to approach the data more vertically, for example only on technology.
Do you think it would be valuable to distinguish between more than two classes?	Perhaps we can use our Scouting Campaign distinction here, meaning three classes: Not a fit, Non-Priority & Priority	I think that might be helpful, especially if the classes are dynamic. This means that we could set up the classes based on our own metric bias. As I said, I think that giving us the freedom to adjust the metric bias and gravity would be helpful. That way we could set up the recommendations based on our clients' and projects' specific characteristics.	I have no opinion on this for now, as I would have to run through the process to see whether that would add value. Yes, I would imagine that there are a significant amount of edge cases where the system cannot evaluate a startup that may be very relevant. Therefore it may prove useful to have a third class that included these edge case startups to be evaluated by the scout.	Usage will determine what the optimal structure of the feedback-mechanism will look like. It could be interesting to explore adjustable thresholds. It would indeed be interesting to explore multi-classification to order startups based on specific metrics or to be able to better deal with edge cases that would otherwise be validated too strict.
Do you have any other recommendations to improve the system?	Not at the moment, but after using it, I surely will have some!		For now I only have one recommendation, there should be transparency if possible what constitutes the variables as this information would allow scouts to make more informed decisions.	The meaning and strength of the weights should be better explained and it would be nice to even adjust them to improve case specific classification.

Table G.1: Results from the interview with the scouting team.

H

Appendix - Results Tables

Stage 1	Description		
test #	Train/Test set	Validate	Description
1.1	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,2] (multiplied)
1.2	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,2] (multiplied)
1.3	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,2] (multiplied)
TOTAL	<-MAP		
2.1	4000 (split 50/50	240 (split 5/1)	values class 0 [0,1] values class 1 [0,2] (multiplied)
2.2	4000 (split 50/50	240 (split 5/1)	values class 0 [0,1] values class 1 [0,2] (multiplied)
2.3	4000 (split 50/50	240 (split 5/1)	values class 0 [0,1] values class 1 [0,2] (multiplied)
TOTAL			
3.1	4000 (split 50/50	240 (split 5/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
3.2	4000 (split 50/50	240 (split 5/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
3.3	4000 (split 50/50	240 (split 5/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
TOTAL			
4.1	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
4.2	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
4.3	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
TOTAL			
5.1	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
5.2	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
5.3	4000 (split 50/50	1100 (split 10/1)	values class 0 [0,1] values class 1 [0,3] (multiplied)
TOTAL			
TOTAL ALL			

Stage 2	Explanation
test #1	Test with top 4 of both Approved and Discarded
1.1	Reason, fault in the algorithm
1.1.2	
1.2	
1.3	
1.4	
1.5	
AVERAGE	Explanation
test #2	Test with every other number in the top 8 for approved an discarded
2.1	
2.2	
2.3	
2.4	
2.5	
AVERAGE	
TOTAL AVERAGE	

Stage 4	Description		
test #	Train/Test set	Validate	Description
1.1	extended with all 'winners'	split 0.5	KNN imputer (Test 01)
1.2	extended with all 'winners'	split 0.5	KNN imputer (Test 04)
1.3	extended with all 'winners'	split 0.5	KNN imputer (Test 05)
TOTAL	<MAP		
2.1	extended with all 'winners'	split 0.3	KNN imputer (Test 08)
2.2	extended with all 'winners'	split 0.3	KNN imputer (Test 09)
2.3	extended with all 'winners'	split 0.3	KNN imputer (Test 10)
TOTAL			
3.1	extended with all 'winners'	split 0.3	KNN imputer (Test 02)
3.2	extended with all 'winners'	split 0.5	KNN imputer (Test 03)
3.3	extended with all 'winners'	split 0.3	KNN imputer (Test 12)
TOTAL			
TOTAL ALL			

Table H.1: Descriptions of tests per stage.

Stage 1

test #	TP (1)	TN (0)	FP (1)	FN (0)	Precision 0	Precision 1	Macro AVG	Weighted AVG	Recall 0	Recall 1	Macro AVG	Weighted AVG	F1 0	F1 1	Accuracy	Macro AVG	Weighted AVG	Matthews Correlation AP
1.1	94	990	10	6	0,99	0,90	0,95	0,99	0,99	0,94	0,96	0,96	0,99	0,92	0,99	0,96	0,99	0,913769398 0,95
1.2	91	1000	0	9	0,99	1,00	1,00	0,99	1,00	0,91	0,96	0,99	1,00	0,95	0,99	0,97	0,99	0,949675235 1,00
1.3	94	998	2	6	0,99	0,98	0,99	0,99	1,00	0,94	0,97	0,99	1,00	0,96	0,99	0,98	0,99	0,955433282 0,99
TOTAL	93	996	4	7	0,99	0,96	0,98	0,99	0,996667	0,93	0,9633333	0,99	0,996667	0,94333	0,99	0,97	0,99	0,938803398 0,975
2.1	37	199	3	1	0,99	0,97	0,98	0,98	0,99	0,93	0,96	0,98	0,98	0,95	0,98	0,97	0,98	0,939215687 0,98
2.2	40	198	0	2	0,95	1	0,98	0,98	0,99	1,00	0,99	0,99	0,99	0,98	0,99	0,99	0,99	0,971008312 0,98
2.3	37	200	0	3	0,99	1	0,99	0,99	1,00	0,93	0,96	0,99	0,99	0,96	0,99	0,98	0,99	0,954636082 1,00
TOTAL	38	199	1	2	0,976667	0,99	0,983333	0,986666667	0,993333	0,953333	0,97	0,986667	0,99	0,963333	0,986667	0,98	0,986666667	0,954655044 0,983333333
3.1	40	200	0	0	1,00	1	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1 1,00
3.2	38	200	0	2	0,99	1	1,00	0,99	1,00	0,95	0,97	0,99	1,00	0,97	0,99	0,98	0,99	0,969842286 1,00
3.3	40	199	1	0	1,00	0,98	0,99	1,00	0,99	1,00	1,00	1,00	1,00	0,99	1,00	0,99	1,00	0,985257178 0,99
TOTAL	39,33333	199,6667	0,333333	0,666667	0,996667	0,993333	0,996667	0,996666667	0,996667	0,983333	0,99	0,996667	1	0,986667	0,996667	0,99	0,996666667	0,984962227 0,995
4.1	99	1000	0	1	1,00	1	1,00	1,00	1,00	0,99	0,99	1,00	1,00	0,99	1,00	1,00	1,00	0,994490316 1,00
4.2	99	1000	0	1	1,00	1	1,00	1,00	1,00	0,99	0,99	1,00	1,00	0,99	1,00	1,00	1,00	0,994490316 1,00
4.3	100	998	2	0	1,00	0,98	0,99	1,00	1,00	1,00	1,00	1,00	1,00	0,99	1,00	0,99	1,00	0,9891569 0,99
TOTAL	99,33333	999,3333	0,666667	0,666667	1	0,993333	0,996667	1	1	0,993333	0,9933333	1	1	0,99	1	0,996667	1	0,992666667 0,996666667
5.1	90	1014	6	10	0,99	0,94	0,96	0,99	0,99	0,90	0,95	0,99	0,99	0,92	0,99	0,96	0,99	0,910770924 0,97
5.2	84	1020	0	16	0,98	1	0,99	0,99	1,00	0,84	0,92	0,99	0,99	0,91	0,99	0,95	0,99	0,909410263 0,99
5.3	96	1019	1	4	1,00	0,99	0,99	1,00	1,00	0,96	0,98	1,00	1,00	0,97	1,00	0,99	1,00	0,972308066 1,00
TOTAL	90	1017,667	2,333333	10	0,990667	0,982667	0,987333	0,993333333	0,996667	0,9	0,95	0,993333	0,996	0,963333	0,993333	0,980667	0,993333333	0,960366987 0,986666667
TOTAL ALL																		

Stage 2

		Approved						Discarded						Optimal Threshold				Explanation
test #1	Company Name	Sim comp 1	Sim comp 2	Sim comp 3	Sim comp 4	Threshold	Similarity 1	Similarity 2	Similarity3	Similarity 4	Threshold	Succeded?	Approved	Discarded	Accuracy	Macro AVG	Weighted AVG	Test with top 4 of both Approved and Discarded
1.1	MOMALA	0,777995	0,7677	0,762917	0,75498	0,765898	0,394783	0,40327	0,405087	0,434078	0,409305	X	0,786138	0,525854	0,99	0,97	0,99	Reason, fault in the algorithm
1.1.2	MOMALA	0,811256	0,788042	0,77407	0,771184	0,786138	0,499428	0,521702	0,534536	0,547748	0,525854	√						
1.2	Looxid Labs	0,75736	0,753709	0,746825	0,743634	0,750382	0,476095	0,487626	0,522919	0,524083	0,502681	√	0,750382	0,502681				
1.3	Syqe Medical	0,771981	0,769457	0,76773	0,740255	0,762356	0,45214	0,467521	0,481604	0,489258	0,472631	√	0,762356	0,472631				
1.4	NeuroSyntek	0,756893	0,745133	0,721563	0,719403	0,735748	0,449709	0,450265	0,457014	0,497754	0,463686	√	0,735748	0,463686				
1.5	Pro Arc	0,821448	0,805368	0,798949	0,772524	0,799572	0,435944	0,462432	0,467501	0,490264	0,464035	√	0,799572	0,464035				
AVERAGE		0,783788	0,772342	0,761827	0,7494	0,766839	0,4626632	0,477909	0,492715	0,5098214	0,485777		0,766839	0,485777				
test #2	**Company Name**	**Sim comp 1**	**Sim comp 2**	**Sim comp 3**	**Sim comp 4**	**Threshold**	**Similarity 1**	**Similarity 2**	**Similarity3**	**Similarity 4**	**Threshold**	**Succeded?**	**Approved**	**Discarded**	**Explanation**			
2.1	MOMALA	0,811256	0,77407	0,769098	0,738065	0,773122	0,499428	0,534536	0,55016	0,556334	0,535165	√	0,773122	0,535165	Test with every other number in the top 8 for approved an discarded			
2.2	Looxid Labs	0,75736	0,746825	0,741365	0,726803	0,743088	0,476095	0,522919	0,527608	0,528186	0,513702	√	0,743088	0,513702				
2.3	Syqe Medical	0,771981	0,76773	0,7397	0,736923	0,754084	0,45214	0,481604	0,495101	0,52353	0,488094	√	0,754084	0,488094				
2.4	NeuroSyntek	0,756893	0,721563	0,711202	0,705956	0,723904	0,449709	0,457014	0,497754	0,527738	0,483054	√	0,723904	0,483054				
2.5	Pro Arc	0,821448	0,798949	0,764157	0,763523	0,787019	0,435944	0,467501	0,499638	0,508994	0,478019	√	0,787019	0,478019				
AVERAGE		0,783788	0,761827	0,745104	0,734254	0,756243	0,4626632	0,492715	0,514052	0,5289964	0,499607		0,756243	0,499607				
TOTAL AVERAGE		0,783788	0,767085	0,753466	0,741827	0,761541	0,4626632	0,485312	0,503384	0,5194089	0,492692		0,761541	0,492692				

Stage 4

test #	TP (1)	TN (0)	FP (1)	FN (0)	Precision 0	Precision 1	Macro AVG	Weighted AVG	Recall 0	Recall 1	Macro AVG	Weighted AVG	F1 0	F1 1	Accuracy	Macro AVG	Weighted AVG	Matthews Correlation AP
1.1	0	62	4	6	0,94	0,00	0,47	0,86	0,94	0,00	0,46	0,86	0,93	0,00	0,86	0,46	0,87	-0,073127242 0,47
1.2	4	5	63	0	1,00	0,06	0,53	0,95	0,07	1,00	0,54	0,12	0,14	0,11	0,12	0,12	0,14	0,066255684 0,53
1.3	4	4	64	0	1,00	0,06	0,53	0,95	0,06	1,00	0,53	0,11	0,11	0,11	0,11	0,11	0,11	0,058823529 0,53
TOTAL	2,666667	23,66667	43,66667	2	0,922078	0,057554	0,51	0,92	0,356072	0,666667	0,51	0,363333	0,393333	0,073333	0,363333	0,23	0,373333333	-0,039624521 0,509754167
2.1	4	8	60	0	1,00	0,06	0,53	0,95	0,12	1,00	0,56	0,17	0,21	0,12	0,17	0,16	0,21	0,085749293 0,53
2.2	4	13	55	0	1,00	0,07	0,53	0,95	0,19	1,00	0,60	0,24	0,32	0,13	0,24	0,22	0,31	0,1138469 0,53
2.3	4	7	61	0	1,00	0,06	0,53	0,95	0,10	1,00	0,55	0,15	0,19	0,12	0,15	0,15	0,18	0,079591718 0,53
TOTAL	4	9,333333	58,66667	0	1	0,06383	0,53	0,95	0,136667	1	0,57	0,186667	0,24	0,123333	0,186667	0,176667	0,233333333	0,093599953 0,531972512
3.1	0	68	0	4	0,94	0,00	0,47	0,89	1,00	0,00	0,50	0,94	0,97	0,00	0,94	0,49	0,92	#DIV/0! 0,47
3.2	0	68	0	4	0,94	0,00	0,47	0,89	1,00	0,00	0,50	0,94	0,97	0,00	0,94	0,49	0,92	#DIV/0! 0,47
3.3	0	68	0	4	0,94	0,00	0,47	0,89	1,00	0,00	0,50	0,94	0,97	0,00	0,94	0,49	0,92	#DIV/0! 0,47
TOTAL	0	68	0	4	0,944444	0	0,47	0,89	1	0	0,5	0,94	0,97	0	0,94	0,49	0,92	0,472222222
TOTAL ALL					0,955507	0,040461	0,503333	0,92	0,49758	0,555556	0,5266667	0,496667	0,53444	0,065556	0,496667	0,298889	0,508888889	0,026987716 0,504649635

Table H.2: Results testing stage 1, 2 and 4.

DEPARTMENT OF EEMCS
UNIVERSITY OF TWENTE
Enschede, Netherlands
www.utwente.nl

UNIVERSITY
OF TWENTE.