UNIVERSITY OF TURKU

ABSTRACT

	Bachelor's thesis
X	Master's thesis
	Licentiate's thesis
	Doctoral dissertation

Subject	Information Management	Date	15.8.2020
Author(s)	Aaro Askala	Number of pages	77+appendices
Title	Explanatory analytics in business dashboards		
Supervisor(s)	Dr. Emiel Caron		

Abstract

Business dashboards are visual products of business intelligence and analytics, that are used to support decision-making. This thesis studies how dashboards can be extended with explanatory analytics. Explanatory analytics are automated diagnostics that generate probable explanations to a problem, often an exceptional value. This is especially important with the advent of big data and maturing dashboard technologies.

This thesis is conducted by design science research, where an artifact is created based on previous theoretical grounding, and then evaluated through business experts. Three separate models for explanations from different fields are compared, namely explanation formalism, informative summarization, and explanation by intervention. First, the models' theoretical bases are detailed and compared. Then the extension is planned by the use of UML diagrams, and implemented through Python using object-oriented programming and Microsoft Power Bi as the dashboarding platform. This implementation is then evaluated with business experts, through semi-structured qualitative interviews.

As a result, it is found that business dashboards can be extended with explanatory analytics, and that the three models share many functions, while differing in others. The main differences found are the recursion logic, measure of impact, and visualisation of the models. Explanation formalism uses top-down recursion logic, with a measure of impact based on the absolute difference of actual and reference value and has a visualisation in the form of an explanation tree. Informative summarization, in contrast, uses bottom-up logic, with impact measure of both magnitude and ratio, and the result is in the form of a table. Explanation by intervention has no recursion, but calculates everything with a big bang method, measuring the impact by ratio, and the result is in the form of table. With a qualitative evaluation it was found that most business experts prefer the use of absolute difference in the measure and having a visualisation such as explanation tree to speed up the assimilation of information. Ratio as a measure of impact was seen as including insignificant explanations when solving business problems.

Key words	Business dashboard, Business intelligence, Analytics, Explanatory analytics

TURUN YLIOPISTO

TIIVISTELMÄ

	Kandidaatintutkielma
X	Pro gradu -tutkielma
	Lisensiaatintutkielma
	Väitöskirja

Oppiaine	Tietojärjestelmätieteet	Päivämäärä	15.8.2020
Tekijä(t)	Aaro Askala	Sivumäärä	77+liitteet
Otsikko	Explanatory analytics in business dashboards		
Ohjaaja(t)	Dr Emiel Caron		

Tiivistelmä

Business dashboardit, vapaasti suomennettuna liiketoiminnan mittaristot, ovat liiketoiminnan analysoinnin ja hyödyntämisen (BI&A) visuaalisia tuotteita, joita käytetään liiketoiminnan päätöksien tukemiseen. Tämä tutkielma tutkii, kuinka mittaristoja voidaan laajentaa eksplanatiivisilla analyyseillä. Eksplanatiivinen analytiikka on automatisoitua diagnostiikkaa, joka luo liiketoiminnallista arvoa tunnistamalla ongelmia aiheuttavia tekijöitä. Tämä on erityisen tärkeää tiedon määrän kasvun ja teknologioiden kehittymisen seurauksena.

Tämä tutkielma toteutetaan design science research -menetelmällä, jossa artefakti luodaan aiempaan teoreettiseen pohjaan perustuen, ja sitten arvioidaan liiketoiminnan asiantuntijahaastatteluin. Kolmea eri mallia syiden tunnistamiseen eri tutkimusaloilta verrataan keskenään. Nämä ovat explanation formalism-, informative summarization-, ja explanation by intervention -mallit. Ensin mallien teoreettiset pohjat käydään läpi ja niitä verrataan keskenään. Tämän jälken mittariston laajennus suunnitellaan UML-kaavioiden avulla, ja toteutetaan Pythonilla olio-ohjelmoinnin avulla Microsoft Power Bi -ohjelmaan. Tämä toteutus arvioidaan liiketoiminnan asiantuntijoilla, puolistruktu-roiduilla kvalitatiivisilla haastatteluilla.

Tuloksena huomataan, että liiketoiminnan mittaristoja voidaan laajentaa eksplanatii-visella analytiikalla, ja että tutkielmassa käytettyjen kolmen mallin välillä on paljon sa-maa, mutta kuitenkin myös eroja. Pääerot mallien välillä syntyvät rekursiologiikassa, vaikutusmitassa, ja visualisoinnissa. Explanation formalism -malli käyttää ylhäältä-alas-rekursiota, pääasiassa absoluuttisen eron mittaa, ja visualisoinnissa puumallia. Informative summarization -malli toisaalta käyttää alhaalta-ylös-rekursiota, vaikutusmittaa, joka ottaa huomioon kummatkin, absoluuttisen eron ja muutoksen suhteen, ja visualisointi on taulukko. Explanation by intervention -malli ei käytä rekursiota, vaan niin sanottua big bang -metodia, kun taas sen vaikutusmitta on pääasiassa muutoksen suhde, sekä lopputuloksen esitys on taulukkomuodossa. Kvalitatiivisella arvioinnilla huomattiin, että useimmat liiketoiminnan asiantuntijat pitivät enemmän absoluuttisesta erotuksesta vaikutusmittana, sekä visualisoinnista kuten puumallista, joka nopeuttaa tiedon assimilaatiota. Muutoksensuhde vaikutusmittana nähtiin monessa tilanteessa epäkäytännöllisenä, koska se voi ottaa huomioon epätärkeitä ja pieniä syitä.

Avainsanat	Liiketoiminnan mittaristo, Liiketoiminnan hyödyntäminen, Analytiikka, Eksplanatiivinen analytiikka

EXPLANATORY ANALYTICS IN BUSINESS DASHBOARDS

A comparison of three explanation models

Master's Thesis
in Information Management

Author:
Aaro Askala

Supervisor(s):
Dr Emiel Caron

15.8.2020
Turku

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin OriginalityCheck service.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

1 INTRODUCTION

This thesis is aimed at building a proof of concept model for the use of explanatory analytics in business dashboards. The focus is at solving the problem of explaining exceptional values in business dashboards for the use of business decision-making.

1.1 Background

As big data solutions mature and machine learning advances from a concept to a technology, business intelligence has become an important focus for companies (Agarwal & Dhar, 2014). Business intelligence has hence received widespread interest from industry and academia alike (Chen, Chiang, & Storey, 2012). However, one of the current problems with business intelligence is the actual process of value creation (Trieu, 2017).

Business dashboards are one distinct product of a company's business intelligence solution. Dashboards mainly aim to support managers in their decision-making and planning process. (Rasmussen, Bansal, & Chen, 2009). The problem is that generally dashboards only give overview of the symptoms, instead of the underlying causes. When an analyst looks at Figure 1 displaying a dashboard diagram, their attention may be drawn to the fact that the gross profit has drastically decreased on 12th month in 2013, compared to 2012.

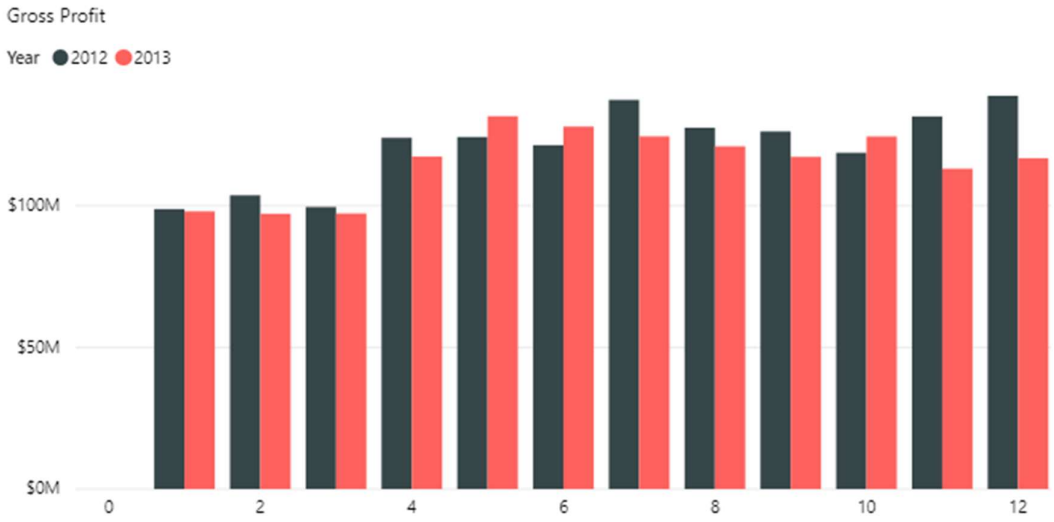

Figure 1 Example of gross profit diagram in a dashboard

However, just knowing that the gross profit has gone down does not support data-driven decision-making. First an analyst must determine if it is a normal change or an actual

exception. This can be done by referencing last year's values or an expected value based on history, budgeting, and forecasting methods. After it is established that the value is indeed an exceptional one the analyst needs to find the cause. Often an analyst will search for the cause through simple methods such as drilling-down, drilling-up, or slicing. Going through the dimensions to search specific dates, locations, or other factors in month 12 that have affected the gross profit to pinpoint the time and location of the exception. However, this becomes increasingly difficult as the data set's size grows (Caron & Daniels, 2008).

Analytics is no longer a new thing in the world (Davenport, 2013), but with larger data sets and greater demand, automation is the next logical step. The goal of this thesis is to extend traditional business dashboards with automated explanatory analytics, in order to provide causes instead of only symptoms to support managerial decision-making. This problem is two-fold:

1. first there is a need to have a method to reliably find exceptional values, *problem identification*, and
2. after that use explanatory analytics to automatically find the causes, called *explanation generation*.

Exceptional values are values that differ from the normal behaviour and expected values. (Caron & Daniels, 2008.) The first problem, coined as *problem identification* by Caron and Daniels (2013), has already been solved by previous study (Feelders & Daniels, 2001).

The main focus of this thesis is to prove that explanation generation can be automated, with the use of Python and Powerbi tools. This can be understood in the context of healthcare; it is not enough to notice the symptoms, such as fever and a running nose, but a diagnosis is made to understand the causes in order to make correct decisions on the situation. The same applies to business, knowing that the sales have gone down in the second quarter of a certain region is not enough. Traditionally, the causes have been determined manually by business analysts with simple visual operators (Caron & Daniels, 2008). Manual errors and inefficiency become problems, which this thesis aims to solve.

1.2 Research question

In this thesis, the concept of explanation generation is applied on dashboards. The idea is to seek additional efficiency and value through extending business dashboards with

automated diagnostic analytics, in order to support timely business decision-making. Therefore, the main research question is:

How can business dashboards be extended with explanatory analytics to support business decision-making?

The sub questions are:

1. *What are business dashboards?*
2. *What are models for explanatory analytics and their comparison?*
3. *How can business dashboards be extended with explanatory analytics?*
4. *How to implement sub-question 3. in Python?*
5. *How to validate and evaluate the models for explanation generation in dashboards?*

In order to answer the main research question, four sub-questions require solutions.

The first sub question is answered by a literature review on business dashboards and its surrounding concepts. Main point is to define what are business dashboards, and how are they useful to enterprise decision-making. Similarly, different models of causal reasoning are reviewed as an answer to the second sub question. Three models for explanation are discussed, compared, and applied to a business dashboard setting. The aim of the comparison is to understand important qualities of the explanatory models and how they affect their usage.

UML models such as use case and class diagrams are used to answer the third sub question. These models provide the framework for the artifact itself, by looking at the big picture and grounding it to best practices. The fourth sub question's answer is implementing the third one, by writing the code in Python language. The most important and distinct parts of the code is reviewed in the thesis, while the whole code can be found in the appendices. The final sub question: evaluation and validation of the model is conducted by interviews. This is an important part of ensuring that the model is useful and provides new insights into automated diagnostics.

1.3 Research relevance

The research relevance for this thesis is two-fold; the contribution to theory and the relevance for business managers. Theoretical contributions come from the comparison of models, proving implementation in business dashboard environment, and evaluating the models by business experts. The relevance to business managers come from the outcome of the design and its evaluation: showing a possible path to enhancing business

intelligence of a company and adding the business experts' voice to the research of explanatory analytics.

1.3.1 Theoretical contribution

This thesis adds onto an existing research stream of explanatory analytics and exceptional values in OLAP databases and business dashboards (Feelders & Daniels, 2001; Sarawagi 2001; Caron & Daniels, 2008; Caron, 2013; Roy & Sucio, 2014), as well as to the overall research of business intelligence. This thesis applies theories from OLAP environment (Feelders & Daniels, 2001; Sarawagi, 2001; Caron, 2013) and database/sql environment (Roy & Sucio, 2014) to a business dashboard environment (Eckerson, 2010), bridging the different fields of business intelligence and analytics.

Furthermore, the comparison of a variety models for explanation from different fields contribute to the general research of explanations. As a design science this study contributes by creating an artifact that enables automated explanation generation in business dashboards, therefore proving that future implementations are possible. By evaluating the artifact through qualitative semi-structured interviews, this thesis adds to previous knowledge about business experts' needs and views on automated analytics and business problem-solving.

1.3.2 Managerial relevance

The relevance to business managers can be seen in the possibility of automated and reliable explanations of exceptional values to speed up and enable the process of data-driven decision-making. According to Larson and Chang (2016), the trend of "fast analytics" has been on the rise after agile methods were introduced to business intelligence. This thesis adds to fast analytics by making the visual analytics even more efficient for business managers and analysts.

This thesis shows a possible direction for extending business dashboards for the use of managers by proving implementation possible, comparing various models for explanation, and giving voice to business experts in their evaluation. By proving the implementation possible, current commercial solutions gain assurance on a possible direction for extending dashboards. Furthermore, with the comparison and evaluation of various models both commercial dashboard software vendors and business managers can make more informed decisions on their analytical software architecture. Automating the finding, filtering, and causal relationship explanations of exceptional values allows data to be

available faster for decision-makers, enabling data-driven decision-making and fast analytics.

1.4 Research method

The main research methodology for this thesis is design research. Hevner, March, Park, and Ram's (2004) design research model is adopted for the creation of the explanatory analytics model. The design research model has three main parts; Knowledge base, Environment, and the Information System research itself. These three parts are combined to create utility, in this thesis the artifact for using explanatory analytics in business dashboards. (Hevner et al., 2004.) Figure 2 shows an adapted design science framework for this thesis.

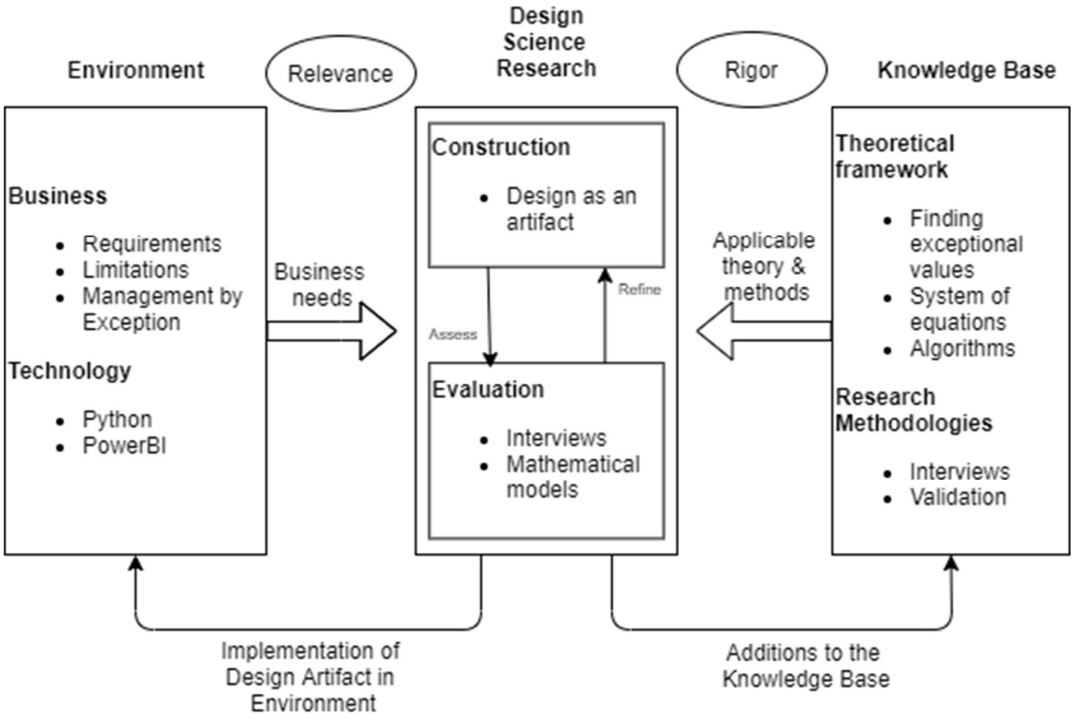

Figure 2 Design science research framework for this thesis (adapted from Hevner et al. 2004)

Knowledge base of this thesis is divided into two categories: theoretical framework and research methodologies. Knowledge Base provides rigor to the design, whereas the outcome of the design will add to the existing base. As seen in Figure 2, the theoretical framework consists of previous literature, such as finding exceptional values, system of equations and algorithms. Literature provides background for concepts such as business

dashboards and models of causal reasoning. Main algorithms used in this thesis come from Feelders and Daniels (2001) Sarawagi (2001), and Roy and Sucio (2014)'s works on explanation generation. Research methodologies include methodologies used in the evaluation of the artifact, such as qualitative analysis of semi-structured interviews.

On the other hand, environment grounds the design with relevance. Business factors such as managerial decision-making and business requirements ensure that the output artifact is relevant and useful. Two main technologies are used in the creation of the artifact: Python and PowerBi. Both technologies are chosen based on their popularity in the environment, as well as suitability to the design. After the artifact is created and ready, the final goal is to implement the design into the environment.

Following Hevner et al.'s (2004) design research framework, creation of the model is also divided into two parts: construction and evaluation. This process is used to ensure that the artifact is constructed along the design and then evaluated. The evaluation is conducted by interviews and mathematical models with the goal of refinement and validation of the artifact.

The artifact is validated by interviews of PowerBI users that are qualitatively assessed. A general interview guide approach is used to provide structure to the interviews while allowing freedom to go in-depth in order to receive thick and rich validation (Turner, 2010). This approach means semi-structured interviews, where both the respondent and the interviewer go through specific topics aimed at discussing the pros and cons of the artifact. The population of the interviews is a convenience population, the respondents are selected from available users of PowerBI to gain business insight on the topic.

1.5 Scope

Automated business diagnosis is a broad topic, which requires multiple fields of research to come together. Business dashboards could be automated in order to provide storytelling of the company's past, current, and future state. Another research option is to integrate Natural Language Processing into business dashboards to provide detail in text format. The dashboard itself could change depending on the circumstances to draw attention where needed. However, in this thesis the scope is to give exceptional values' probable causes in business dashboards.

Furthermore, system of equations does not fit every kind of value; the objective is to find causes for financial values, such as revenue, profit, and costs. Here the scope is limited, so non-financial values such as customer satisfaction and production lead-time are

not taken into consideration. The artifact itself is a proof of concept model, as the algorithms and the idea for automated business diagnosis already exists (Feelder & Daniels, 2001).

The technologies chosen for the artifact, namely Python and PowerBI limit the scope as well. Python is a versatile high-level language that runs on many platforms (Oliphtant, 2007). This makes it the ideal language for the artifact creation. PowerBI is arguably one of the most popular data visualization tools, and therefore well-suited for this model. The results of this thesis should be applicable to a wider range of technologies.

1.6 Outline

This thesis is structured to provide the required information for the validation of the solution in an ordered manner. In chapter 2 a literature review is conducted for business dashboards, including the concepts of business intelligence, decision-support systems, and management by exception. Additionally, the choice of technology for this thesis is assessed. Chapter 3, explanatory analytics, gives an overview of different models for explanation. Three models, namely explanation formalism by Feelders and Daniels (2001) and Caron (2013), informative summarization by Sarawagi (2001), and explanation by intervention by Roy and Sucio (2014) are detailed and compared. Chapter 4, extending business dashboards with explanatory analytics, the requirements and solutions for integrating these two concepts are shown. In chapter 5, artifact creation, the process and steps taken when building the artifact are explained. The evaluation conducted in the form of interviews is discussed and main points are shown in chapter 6, design validation. Finally, the results and summarised in the discussion and conclusion chapter.

2 BUSINESS DASHBOARDS

In this chapter the focus in on business dashboards and its surrounding concepts. To understand what business dashboards are, first the idea of business intelligence requires clarification. In a sense, business dashboards are a product of business intelligence, and more specifically decision support systems. Business intelligence systems such as business dashboards are aimed at supporting decision-making. Therefore, this chapter has three main concepts, business intelligence, managerial decision-making, and business dashboard technology itself. Additionally, the choice of technology for this thesis is explained after an overview of prominent business dashboard technologies.

2.1 Business intelligence

According to Chen et al. (2012) business intelligence field has evolved from using internal and structured data, to external and unstructured, to using mobile and sensor-based content. Alongside the technologies and analytics have kept pace by accommodating the need for more data and increasing amount of heterogeneous data sources. In this thesis, the data used is structured, as the source is a dashboard platform where the data has already been cleaned.

To understand what business intelligence (BI) means and how the concept has evolved, looking at the varied definitions over the years is a good starting point. Božič and Dimovski (2019) point out that the definitions of business intelligence & analytics are many and none are uniformly accepted. Golfarelli, Rizzi, and Cella (2004) define it as a process of turning data into information, and information to knowledge. Additionally, BI has been defined as "both a process and a product", where the product helps organisations in their decision-making process predicting the environment (Jourdan, Rainer, & Marshall, 2008).

According to Joshi, Chi, Datta, and Han (2010) and Bose (2009) technology is also a part of business intelligence definition. Chen et al. (2012) define business intelligence & analytics (BI&A) most comprehensively "as the techniques, technologies, systems, practices, methodologies, and applications" that aid organisations in making decisions. Božič and Dimovski (2019) note that the value creation model of BI&A depends on the definition used, since it varies based on the perspective. In this thesis, BI is defined as a process, technology, and a product that aid organisations in their decision-making

process. This definition mirrors the automatic diagnosis, technological platforms, and the results used and provided by the artifact in the thesis.

The main value of BI is to support decision-making of an organisation (Božič & Dimovski, 2019). This can be done through BI products, that give valuable knowledge to support data-driven decision-making. However, the full picture of how that value is created can be complicated (Božič & Dimovski, 2019). Therefore, the process, product, and technology, in addition to value creation are explained in sub-sections 2.1.1 and 2.1.2 respectively.

2.1.1 Process, Product, and Technology

Golfarelli et al. (2004) describe the BI process as more than just data warehousing; it is the act of turning data into useful information and knowledge to operational, tactical, and strategical levels. This process combines the ETL process and data warehouse with business rules to form user interfaces that help with strategical decision-making. March and Hevner (2007) add, that to gain business intelligence the data must be contextualized and rooted to business knowledge. Moreover, they argue that acquisition and integration of systems is not enough, the procedures and infrastructure to use the resultant information and knowledge is vital. Golfarelli's (2004) BI process does not consider external data sources, unlike newer studies and definitions that include external and unstructured data (Chen et al., 2012; Ain, Vaia, DeLone, & Waheed, 2019).

Golfarelli et al. (2004) defined four BI interfaces, namely OLAP, reports, dashboards, and alerts. The separation of process and product is seen as important in the sense of how business intelligence provides value: it is not the process itself, as some information systems do, but use and integration of those products that create value (Jourdan et al., 2008; Trieu, 2017; Božič & Dimovski, 2019). However, in addition to the process and the product, infrastructure requires optimization for business intelligence operations to succeed in an enterprise (Bose, 2009; Chen et al., 2012; Božič & Dimovski, 2019).

IşıK, Jones, and Sidorova (2013) found empirically, that technological capabilities and data quality are important for BI to support an enterprise's decision-making process. This supports the argument that not only does an enterprise require proper BI process in place, but also invest in the technological capabilities to exploit that process. Technological capabilities can make the product more presentable, visualized (Božič & Dimovski, 2019), faster, and more accurate (Larson & Chang, 2016).

For this thesis' definition of BI, the focus is on internal and structured data due to convenience limitations. The ETL process has already been done before the artefact, although some data conversion is still required. The business dashboard itself is a visualized product; but the artifact extension requires its own visualized product to conform with the BI standards. The technological aspect will be further discussed in the choice of the commercial platform and coding language in section 2.4.

2.1.2 Value Creation

IT value creation has been studied extensively, whereas BI value creation has been based on the IT value creation models and has not received as much attention (Fink, Yogen, & Even, 2017). According to Fink et al. (2017), the major problem with the IT value creation models have been that they are focused on the overall IT assets and capabilities, instead of specific technologies and systems. However, most BI value creation models are still based on the IT value creation studies (Trieu, 2017).

Trieu (2017) propose a value creation model based on Soh and Markus (1995), where the framework is divided into three parts: BI conversion process, BI use process, and BI competitive process (See Appendix A1). During conversion process, investments are turned into assets, while the use process uses the assets for impact, which translates to organisational performance. During the use and competitive process, both contextual and environmental factors as well as latency effects affect how the investment turns into value. (Trieu, 2017.) This is a very general model, that explains how investing in BI can affect organisational performance.

Fink et al. (2017) propose a more specified model, where the BI capabilities and value are divided into operational and strategical (See Appendix A2). Strategic capabilities are repeatable actions where BI is used to support strategic organisational activities, such as measuring performance. Operational capabilities are repeatable actions that use BI assets to support operational activities, such as transactional data analysis and production optimization. Furthermore, in this model BI assets are divided into BI Team and BI Infrastructure, differentiating between the people capabilities and physical assets. (Fink et al., 2017.) The division between operational and strategical capabilities can be useful, as the process and product for optimizing and securing operations can be very different from strategical analysis.

Sharma, Mithas, and Kankanhalli (2014) raise the problem of decision-to-value in BI research. They argue that making better decisions does not mean more value, since many factors affect if the decision is realised, such as the acceptance rate of a decision. Božič and Dimosvki (2019) empirically found, that visual and easily understood BI products are important to realise the value of the BI process. Visualization of the BI product increase the acceptance rate and therefore the value of BI products. Torres, Sidorova, and Jones (2018) divide BI capabilities to sensing and seizing, based on their organisational dynamics. Sensing corresponds to identifying threats, opportunities, and inefficiencies. Seizing is the act after an opportunity or threat is identified, such as building consensus among stakeholders. (Torres et al., 2018.) Therefore, good business intelligence process can both increase the quality and the acceptance of decision-making in an enterprise, creating value through fact-based decision-making.

2.2 Managerial Decision-making

According to Sauter (2014), decision is a choice among alternatives available too an individual. Decision making process in its simplicity is intelligence, design, and choice, where many paths and alternatives emerge during design phase (Simon, 1977; Eisenhard & Zbaracki, 1992). These decisions have traditionally been divided to fast and intuitive decisions and slow, strategical decisions (Starcke & Brand, 2012; Beshears & Gino, 2015). The process is the same even in fast decisions (Beshears & Gino, 2015), but much more streamlined and more affected by emotion and intuition (Starcke & brand, 2012). This is problematic in a business environment, where decisions need to be made fast, but rationally and logically.

Eisenhardt and Zbaracki (1992) already noted that decision makers often satisfice instead of optimising, and rarely comprehensively search through their options. Even decades later, Sauter (2014) argue that most managers do not necessarily make decisions based on facts but depend on who presents those ideas. Even when uncertainty is low and a clear strategic decision can be made, the decision can later add bias to future decisions (Starcke & Brand, 2012). These are some of the reason why most organisations have complex decision-making process, with the aim of minimising bias and committing to a rational and solid choice. It is important to ensure the quality of decisions as they are a major driver of lasting competitiveness (Rejikumar, Aswathy Asokan, & Sreedharan, 2020).

However, as Fiedler's (1964) contingency theory argues that there is no single best way of leading, the same can be said of making decisions. Starcke and Brand (2012) argue that often both intuitive and slow decision-making processes act in concert, where rationality is merged with intuition. This aligns with data-driven decision-making (DDD), where the idea is to base decisions on the analysis of data rather than purely on intuition (Provost & Fawcett, 2013). Provost and Fawcett (2013) continue to add, that different organisations engage in DDD to lesser or greater degree. Therefore, there are some similarities with individual human decision-making and business decision-making processes.

As this thesis focus is on business analytics and the decisions made based on it, data-driven decision-making is further focused on in the sub-section 2.2.1. This is aimed at providing background for how the artefact can be used in real-life situation, where analysts and managers co-exist in an enterprise, with the goal of making and ensuring execution of good quality decisions.

2.2.1 Data-driven decision-making

Data-driven decision-making (DDD) has been on the rise with the advent of big data and increase in ease of implementation (Brynjolfsson & McElheran, 2016; Rejikumar et al., 2020). Rejikumar et al. (2020) argue that drivers of DDD include lasting competitiveness, understanding of environment and developing strategies. DDD has become an important part of the business environment, where companies rely on the strength of data to ensure high quality decisions. Brynjolfsson and McElheran (2016) note that especially in companies with high levels of information technology and educated workers DDD adoption has seen most success and popularity.

Provost and Fawcett's (2013) definition of DDD has been the most widely used in its simplicity (see chapter 2.2). Mandinach (2012) defined DDD as "the systematic collection, analysis, examination, and interpretation of data". Going forward with these definitions, if a company verifies and trust the strength of data turned into information as a basis for decision-making, they are using DDD in some form. It is no wonder, as it has been empirically proven to improve productivity (Brynjoolfsson & McElheran, 2016).

DDD can be divided into two kinds of decisions, (1) ad-hoc decisions where new discoveries are made from the data and (2) decisions that repeat and can be more easily automated (Provost & Fawcett, 2013). These decisions still follow the line of intelligence (data to information), design (options), and choice (commit). Provost and Fawcett (2013) note that the second kind of decisions benefit majorly from even a small increase in data

quality or quantity, while the first kind not as much. These two decisions can be likened to strategical and operational levels of business intelligence in terms of ad-hocness, continuity, and impact level.

Even in DDD it is not all about the numbers and data: it is important to turn that data into actionable knowledge by creating insights (Mandinach, 2012; Sharma et al., 2017). According to Lycett (2013) and Sharma et al. (2017), insights emerge from the engagement of analysts and managers using data and analytics tools to discover new knowledge. This means that efficient use of DDD require analytical skills from both analysts and managers (Provost & Fawcett, 2013; Rejikumar et al., 2020). In fact, DDD shows a much higher adoption and success rate in organisations with high levels of information technology and educated workers (Brynjolfsson & McElheran, 2016).

2.3 Dashboard technology

The terminology of dashboards is originally from vehicle dashboards, where the driver can glance at important metrics while driving (Yigitbasioglu & Velcu, 2012). According to Sharda, Delen, Turban, Aronson, and Liang (2014), dashboard are not a new concept and can be traced all the way to the 1980s. However, the concept of a driver looking for important information with a single glance still holds true. Sharda et al. (2014) define dashboards as "visual displays of important information that is consolidated and arranged on a single screen so that information can be digested at a single glance and easily drilled in and further explored". Eckerson (2010) argued that dashboards are complete business information systems that are built on BI and data infrastructure of a company. Dashboard can also be thought of as a data-driven decision support system (Yigitbasiouglu & Velcu, 2012). The current business dashboards are not only the one-page view; but a system that includes the related tools and extensions that enable necessary functionalities to support decision-making.

According to Eckerson (2010) dashboards have three applications:

1. Monitoring application
2. Analysis application
3. Management application

Monitoring application monitors processes and activities and alerts creates alerts based when performance falls below target value. Analysis application is used to find the root causes of said problems, by exploring the layers and perspectives of measures.

Management application is used to manage processes and people to enhance decision-making, optimising performance, and aligning strategy and operations. (Eckerson, 2010.)

Dashboards have become ubiquitous in the business world and are used for many purposes, the main one still being decision support (Sharda et al., 2014; Rahman, Adamu, & Harun, 2017; Magdalena, Ruldeviyani, Sensuse, Bernando, 2019). According to Eckerson (2010), the purpose of a dashboard is to make decision-making and management overall more effective. The fundamental problem with this is to ensure fast assimilation of the required information in the dashboard (Few, 2005). Sharda et al. (2014) proposed to use context to speed the assimilation of the numbers, such as comparing real values to budgeted ones.

In Sharda et al.'s (2014) definition a dashboard is easily drilled-in and explored. According to Rahman's (2017) literature review, the only functionality that business dashboards had to causal reasoning was drill-down mechanism. However, with increases in the sizes of data sets the usage of manual drill-down becomes convoluted and problematic (Caron & Daniels, 2008). To solve this problem, automatic diagnostics in business dashboards can be used to ensure the speed of assimilation and management efficiency. This aligns with Eckerson's (2010) idea that business dashboards are meant to automate management. Implementing automatic diagnosis can be divided into two distinct problems, belonging in two different application layers: *Problem Identification* (Caron & Daniels, 2008) belongs to the Monitoring application. However, instead of looking for only bad performance, all exceptional values are found and noticed. The second problem, *explanation generation* (Caron & Daniels, 2008), is part of the Analysis application. The idea is to codify business logic not only to find exceptions, but to generate possible causes for the exceptions found. The goal is to eliminate or at the least distinctly decrease the time-spent on having to explore the dashboard to find relevant information for decision-making.

2.3.1 Dashboard architecture

There are several commercial applications for implementing business dashboards. However, positioning the business dashboard into an enterprise's business intelligence architecture follow a standard structure. According to Sharda et al. (2014), business intelligence, and therefore business dashboards, are enabled by data warehousing. Data

warehousing refers to the entire process of collecting, integrating, and organising data from various data sources to facilitate decision support, provide access to business information, and create business insights. The process begins when data is collected from various data sources (e.g. ERP, Legacy systems, external data sources). Then the collected data is put through the ETL (Extraction, Transformation, Loading) process. This can be done through separate code, software, or commercial applications. Once the data is loaded into the Enterprise Data Warehouse (EDW) it is ready to support decision-making through relevant summarised and detailed information. Data marts can be used in the data warehousing architecture, depending on the organisational structure, planning, and needs. Middleware tools provide front-end applications access to the organisation's data warehouse. In commercial applications the middleware is often built-in the software. This process is illustrated in Figure 3 "Data Warehousing Framework".

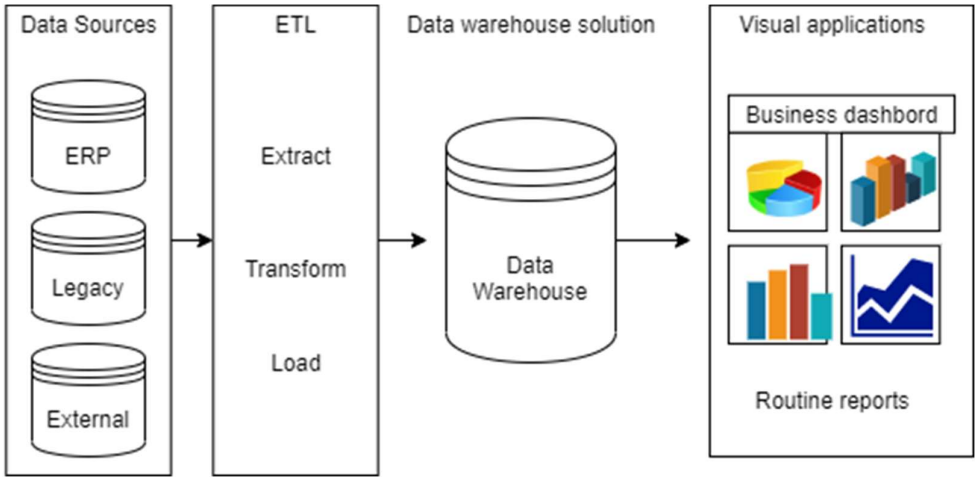

Figure 3 Positioning business dashboard to Data Warehouse Framework (adapted from Sharda et al. 2014)

In Figure 3, dashboards are positioned in the final layer, visualisation applications. However, the complete versions of commercial business dashboard technologies offer more than just the visualisation; they often also provide the middleware for connecting to various kinds of data warehouses and sometimes their own data warehouse. Rasmussen et al. (2009) argue that to thrive in the current data-overloaded environment, dashboards require a proper back-end infrastructure such as data warehousing and online analytical processing (OLAP). Most commonly the back-end infrastructure of a business dashboard is built on a star database schema (Rasmussen et al., 2009; Sharda et al., 2014).

The main parts of a business dashboards back-end infrastructure are the data and database schema. The database schema, often star schema, provides the information model to the raw data that is inputted. These enable business dashboard systems to provide visualisation, analysis, and monitoring. With the star schema explaining dimensions of a measure (= unit of analysis) we can view the data from multiple angles and drill down for increased details (Eckerson, 2010). Figure 4 illustrates the interaction of back-end and front-end infrastructure in a business dashboard.

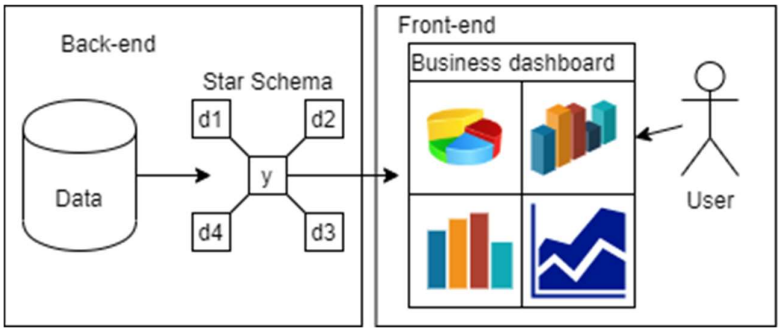

Figure 4 Business dashboard framework

When a business dashboard uses the star schema, an analyst can drill-down or drill-up by using the different dimensions. Every measure in the front-end dashboard can be explained through dimensions. This is illustrated in Figure 5.

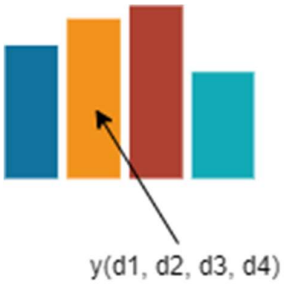

Figure 5 Measure interaction with dimensions

In the front-end level business dashboard also share similarities (Eckerson, 2010). According to Eckerson (2010) these similarities can be summarised into “three threes”: Three applications (monitoring, analysis, management), three layers (graphical metrics, multidimensional, and detailed view), and three types (operational, tactical, and strategic). These are required properties of a dashboard, otherwise the dashboard does not cover the needs of an organisation. The three applications are looked at in chapter 2.3. Most

users start at the graphical view of the business dashboard, which can then be drilled down to multidimensional and detailed views for further analysis. The three types correspond to the purpose of the dashboard; operational dashboards monitor the core operational metrics, rather than focusing on analysis like tactical dashboards. Strategical dashboards emphasize the management application as the focus is on monitoring the execution of strategical objectives.

2.3.2 Star schema

As mentioned in sub-chapter 2.3.1 the star schema is the most common and simplest style of multidimensional modelling (Sharda et al., 2014). In order to facilitate multidimensional modelling, the first step is to transform the online transactional processing database (OLTP) to a star schema (Rasmussen et al., 2009). The star schema consists of a central fact table surrounded and linked to several dimension tables. A fact table contains the decision analysis attributes, measures, needed to perform analysis and query reporting (e.g. sales, profit margins, production costs). Dimension tables are linked to the fact table with foreign keys. Dimension tables contain the aggregation and classification information about the central fact table measures. Dimension tables have a one-to-many relation with the fact table. (Sharda et al., 2014.) A simple star schema is illustrated in Figure 6.

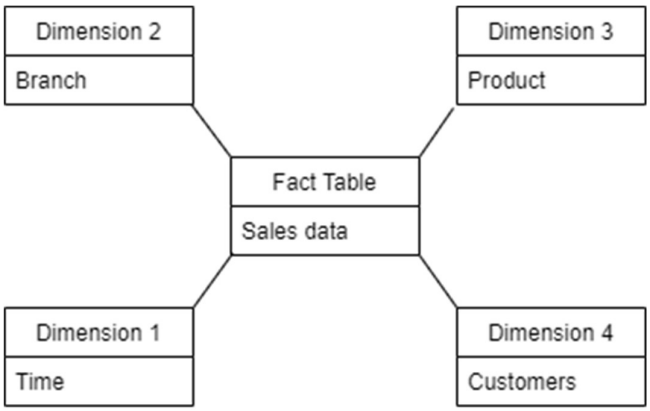

Figure 6 Example of Star Schema

To note is that the star schema is not the only way of modelling an organisation's database. Whereas star schema aims to minimise the number of joins, increase simplicity, and fast query-response time, snowflake schema removes redundancy by normalising the dimension tables (Sharda et al., 2014). This looks like a snowflake, where dimension tables

branch off. The problem with snowflake schema is that it does not have a standard structure, due to the normalisation. This thesis holds the assumption of a star schema in the artifact. Star schema's simplicity and standardisation are the main reasons for this assumption.

2.4 Choice of technology

To evaluate the choice of technology in this thesis, alternative options must be reviewed. There are two different technologies used in the thesis, a dashboard technology and a programming language. As the extension of the dashboard technology is the primary goal, the choice of dashboard technology must be made first and then choosing a language best suited to it. The main requirements in this thesis for a dashboard technology is popularity, offering affordable solutions, and supporting use of data with a popular object-oriented programming language. Supporting the use of data with a programming language in a business intelligence solution is part of self-service of the solution. Self-service means customer interaction with the technology to provide a service independent of the employee/supplier involvement (Meuter, Ostrom, Roundtree, & Bitner, 2000). Business intelligence augmentation with ML- and AI-driven solutions lessens the need for self-service but does also often decrease the customisability. These metrics are used to evaluate potential BI software solutions.

Gartner's (2020) Magic Quadrant for Analytics and Business Intelligence Platforms is used to establish a list of preliminary technologies, which are then reviewed one by one and compared. The Magic Quadrant is presented in Figure 7, where the y-axis is measured as the "ability to execute" and x-axis presents the "completeness of vision". The chart is divided into four quadrants, as the name implies: Niche players, Visionaries, Challengers, and Leaders. The first quadrant, Niche players, does not necessarily mean their software is of low quality. On the contrary, as seen from the name, the solutions these vendors present are focused on specific domains or customers, such as IBM, whose primary interest is to provide existing IBM Cognos customers modern analytical tools (Gartner, 2020). The second quadrant, Visionaries, has high completeness of vision, with comparatively lower ability to execute. As seen in Figure 7, the current Visionaries are not far from the Leader category and includes software technology giants such as SAP, Oracle, and SAS. They each have advanced tools with the ability to bring visualization, user-friendliness, and strong augmentation to their set of tools. The third quadrant, Challengers, have strong ability to execute their respective procedures, but lack completeness in

their vision that the Leaders have. Finally, the Leaders, have both the ability to execute and high completeness in their vision.

Figure 7 Gartner's Magic Quadrant for Analytics and Business Intelligence Platforms (Gartner, 2020)

The “Leaders” category holds 4 technologies: Microsoft, Tableau, Qlik, and ThoughtSpot. Qlik and ThoughtSpot are remain as Leaders, same as last year, but there is a clear divide between them and the top 2 (Gartner, 2019). In fact, it could be said that Tableau and Microsoft are in a league of their own, distinctly ahead of the mass. Nevertheless, to ascertain the technology choice of this thesis, all four Leaders are compared. The four leaders are reviewed and compared in the subchapter 2.4.1. to assess the choice of business dashboard technology in this thesis. In subchapter 2.4.2 the programming language choice of Python is reviewed.

2.4.1 Comparison of dashboard technologies

The three main comparison points of the four business intelligence solutions are popularity, price, and integration of programming language, as discussed in chapter 2.4. These three qualities are assessed mainly through Gartner's (2020) extensive report and the software's own website. To ensure robustness, additional scientific sources are used when available. Results of the comparison are summarised in Table 1, with each quality assessed as Good, Moderate, or Low fit for this thesis. To note here, is that the comparison is made with the focus being on how good a fit the technology is for the purpose of this thesis and not for general or commercial use.

ThoughtSpot released its early version in 2014, premiering as eliminating the need for traditional BI tools (ThoughtSpot, 2014). The technology is mainly used by non-technical person, to make analysing available without the use of analysts. ThoughtSpot is innovative and newest member of the Leader quadrant. Its main attractiveness is in search-based augmented analytics, with customer friendly interface. (Gartner, 2020.) This means it can be used without prior knowledge of analytics as its (NPL) can search for answer to questions. Gartner (2020) states that ThoughtSpot's "cautions" include its lacking dashboards and requirement of using external ETL tools. ThoughtSpot offers only enterprise pricing, with every product available for unlimited users (ThoughtSpot Pricing, n.d.), which does not support the usage for a singular project such as this thesis. Popularity is assessed as Low for this thesis, pricing as Low due to unavailability for individuals, and integration as Low due to lack of data interaction.

Qlik is the next Leader on the review list and likewise has strong ML- and AI-driven augmentation elements. Qlik is a long-standing competitor in the BI business, this being its 10th year in row in the Leader category of Gartner's magic quadrant (Qlik, 2020). Qlik was founded in 1993 with a product named QuikView, which is known as QlikView today (Christiansen, 2014). Qlik's new product Qlik Sense supersedes the old QlikView, adding modern processing techniques to increase the amount of data it can handle and make it increasingly extendible (Vizard, 2019). However, the extensibility of the software requires specific developer skills and can be challenging (Reddy, Sangam, and Rao, 2019). Qlik Sense pricing starts at 30$ per month per user (Qlik Pricing, n.d.), which is slightly steep for a single project usage. Qlik Sense is a solid option for the technology choice of this thesis, even with the pricing and extensibility issues. This technology is assessed as having Moderate popularity, Good pricing, and Moderate integration.

Tableau is the current main competitor of Microsoft PowerBi as seen in Figure 3. Tableau offers interactive visualisations with user-friendly interface and a soft learning curve (Federer & Joubert, 2018). Gartner's (2020) report states that the active customer-base and ease of visual exploration are the strongest points of tableau. Future, however, is unknown as Salesforce acquired Tableau in June 2019. Additionally, Gartner (2020) cautions that new innovations may disrupt Tableau's current position. Tableau has some free online solutions, but in those cases the visualisations are only available in Tableau's public website (Federer & Joubert, 2018). For individuals Tableau's pricing starts at 70$ per month per user, which is costly when not used commercially (Tableau Pricing, n.d.). According to Reddy et al. (2019), Tableau has high support for third party integration. Tableau has multiple interfaces to access data and build extensions, the data itself extractable with C/C++, Java, and Python (Tableau, 2020). Tableau would work very well for this thesis, however; the steep price is an issue. For the comparison table, the pricing is evaluated as Moderate, popularity as Moderate, and the integration as Good.

Microsoft's PowerBi was launched in 2013, and after years of being in the "follower" quadrant has entered the market overtaking most other software (Gartner, 2020). Power Bi's completeness of vision can be partly attributed to its strong integration with other Microsoft services such as Excel and Azure (Reddy et al., 2019; Gartner, 2020), but Microsoft has additionally invested strongly in Power Bi in 2019 adding hundreds of features to its cloud service capabilities (Gartner, 2020). Power Bi is the most viral business intelligence solution, due to its pricing, marketing, and spread through packaged solutions such as Office 365 E5 (Gartner, 2020). Microsoft Power Bi offers individual users a strong free version and additional 10$ per month per user "Pro" version (Power Bi Pricing, n.d.). While PowerBi does not offer quite as comprehensive interfaces as Tableau does, the Pro version offers variety of tools to extract and import data as well as using Python scrips directly. As seen in Table 1, the popularity is Good, pricing is Good, and integration Moderate.

Table 1 Comparison of potential business dashboard solutions for this thesis

Technology	Popularity	Pricing	Integration
ThoughtSpot	Low	Low	Low
Qlik Sense	Moderate	Good	Moderate
Tableau	Moderate	Moderate	Good
Microsoft Power Bi	Good	Good	Moderate

In conclusion, both Qlik Sense and Tableau could be used for this thesis. Qlik Sense has less options for extensions, whereas Tableau's issue is with pricing. Tableau is also less complete with its vision (see Figure 3), and not as popular as Microsoft PowerBi (Gartner, 2020). According to Gartner (2020), some of the PowerBi's strongest points are the cost (free version or 10$ per month per user), viral spread, and product capabilities. The choice of business dashboard technology for this thesis is the marginally best fit, PowerBi, although it is certainly not the only option.

2.4.2 Python as a data manipulation language

According to GitHub's (2020) PopularitY of Programming Language (PYPL) ranking, Python is distinctly in the first place. After Python, the rankings are Java, Javascript, C#, PHP, C/C++, and R in the 7th place. For this thesis, a popular data manipulation language is required, and in the top 10 most suited choices are either Python or R. Python being far more popular, and with Python being the main scripting language for PowerBi it is the natural choice. By itself, Python is a good choice as a steering language, but with additional tools it becomes a high-level language well suited for both scientific and engineering code (Oliphant, 2007).

Originally Python was most used for scripting and not so much for data manipulation; however, with the advent of Pandas tool in 2008 it has become a leading data science language (Pandas, n.d.). As Oliphant (2007) argued, what truly makes Python excel are the extensions, on top of useful general-purpose libraries. Using Python as a data manipulation language is easy due to its simple approach to object-oriented programming and syntax in general (Python Software Foundation, n.d.). Pandas is "an open-source Python Library providing high-performance data manipulation and analysis tool" (Pandas, n.d.). Perhaps the most crucial feature in Pandas is the DataFrame object that allows efficient data storage and indexing with many powerful and intuitive methods (Pandas, n.d.).

Python's clean syntax and simple object-oriented programming, together with its popularity and the Pandas extension create a valuable tool for data manipulation, especially in scientific proof-of-concept scenario. Additionally, the environment used in this thesis is Anaconda Spyder that is designed for and by scientists, engineers, and data analysts (Spyder, n.d.). Spyder is useful for syntax problem-solving and fast variable checking, as well as data configuration. In conclusion, the choice of programming language; Python, is very ideal and useful for the purpose of this thesis.

2.4.3 Extending dashboards

Extending dashboards with additional features, besides the basic one-glance charts makes dashboards truly a complete information system (Eckerson, 2010). According to Gartner (2020), BI solutions are no longer differentiated based on their data visualisation capabilities, but integrated support for enterprise reporting and augmented analytics. These are extension to traditional dashboarding that are currently being researched and offered. Integrated support for enterprise reporting means that dashboard platforms are increasingly used in place of traditional reporting software such as SAP (BusinessObjects) and IBM (Cognos pre-version 11) (Gartner, 2020). Automatic diagnostics would belong to the category of augmented analytics. According to Gartner (2020), ML- and AI-assisted data preparation, insight generation, and insight explanation are the future sources of competitive differentiation. Additionally, automated insights and advanced analytics and two of the key functionalities of future business dashboards (Gartner, 2020).

Most of the upcoming vendors of BI solutions are focusing on augmented analytics, with older vendors acquiring functionalities to keep ahead. Microsoft PowerBi offers a decomposition tree feature since November 2019 (PowerBi decomposition tree, 2020). This tree offers a visualised alternative to drilling-down, with the idea that visualisation will make finding root causes easier and faster. The tree is used to find causes similarly to a decision tree; based on splits, which can be decided manually. An example of a decomposition tree is shown in Figure 8. However, this kind of manual decomposition does not offer comparison nor any kind of automatic explanation for a symptom. It does however present an easy to understand visual for decomposing a measure in a dashboard.

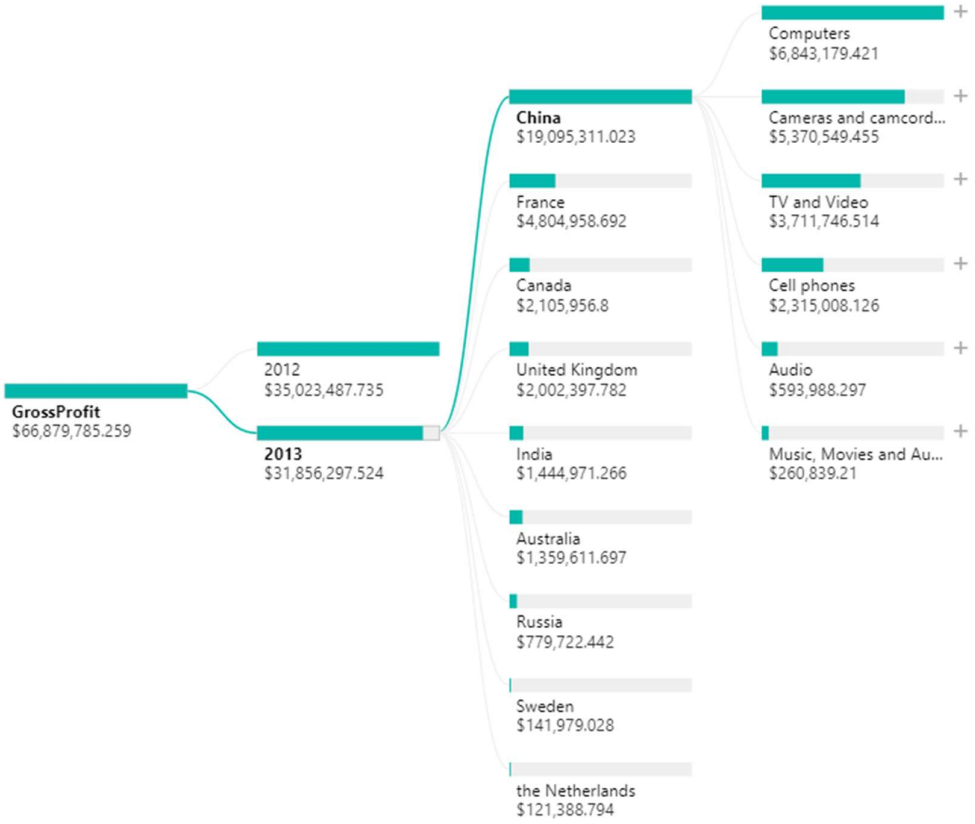

Figure 8 Decomposition tree in Power Bi

Additionally, Microsoft Power Bi has launched a concept of "quick insights" that goes through a dataset to search for insights with the aid of statistical algorithms. It can detect category outliers, trends, correlations, low variance, seasonalities, and time series outliers. (Power Bi: quick insights, 2020.) However, the quick insights function does not offer targeted diagnosis over measures, but tries to guess what a user may want from the whole dataset.

Other vendors have their own extensions, such as ThoughtSpot that uses Natural Language Processing (NPL) to help managers find answers to questions written in plain text (Gartner, 2020). New dashboarding platforms, such as Alibaba Cloud, focus on how quick they can provide the analytics with as little effort as possible (Gartner, 2020). Here again, the focus shifts from actual dashboarding to extending the system with additional features. In conclusion, dashboarding technology has matured and most vendors look to focus on more than just the visual aspects or extend the platform with new analytical features such as NLP.

3 EXPLANATORY ANALYTICS

This chapter details various models for explanation and compares them, in order to answer RQ2 and create a foundation for the implementation. First, various terms and concepts used in the comparison of models are defined. Then, the models are detailed in-depth one by one, and lastly their theoretical parts are compared and combined to Table 8.

3.1 Use of the models

Explanatory analytics, or models for explanations, are used to find *explanations* for a symptom or a problem. Feelders and Daniels (2001) argue that *models* for diagnostic purposes are suited for financial and business problems, as opposed to *heuristic* problem-solving. Mainly due to existing and known cause-effect relations and systems of equations. Most models use some form of intervention or influence to measure the explanations (Feelders & Daniels, 2001; Sarawagi, 2001; Caron, 2013; Roy & Sucio, 2014). Models for explanation are therefore a natural solution for automating diagnosis of business problems.

Comparing multiple models for explanations has the benefit of finding similarities and differences between them, and hence contributing towards better understanding and usage of these models. The comparison in section 3.5 is based on the theoretical frameworks of the models and the application of said models on the running example.

The models are developed by different authors for different situations, and therefore the terms the authors use may differ, even if the meaning is the same. In order to unify the different models for the purpose of easier understanding and comparison I define the following terms and concepts, which I use for this thesis:

Definition 1: Reference value

Henceforth, the term *reference value* is used when meaning a value that is used to compare against the actual value. This term is used instead expected value, which is similar in definition to reference value. In the context of explanatory models in this thesis, the terms are combined and only one is used. The reference value itself can be derived from multiple different *reference models*, for example using historic data, normative models, managerial knowledge, or predictive models. In this thesis, the reference values are derived using historic model, i.e. the reference value is the previous year's value.

Definition 2: Candidate explanation

Henceforth, the term *candidate explanation* is used instead of probable causes or probable explanation. The term means an explanation where it is not yet known if it is a significant explanation or not.

Definition 3: Hidden causes

There is only one model that explicitly considers hidden causes, but as it is used in the comparison of the models it is formally defined. *Hidden causes* term is used for explanations that may be missed due to cancelling-out effects. It is an explanation whose effects are neutralised by a counteracting cause(s) in the same dimension.

Definition 4: Measure of impact

The terms degree of explanation and measure of explanation are combined, and only *measure of impact* is used in this thesis. As there are models for explanation, there needs to be a way to measure the impact of a particular explanation. Therefore, a measure of impact is an explanation's (variable's) impact on the symptom, measured in some way.

Definition 5: Filtering measure

Each of the models filter the candidate explanations in some way, in order to finally present only the most probable significant causes as explanations. Measure of impact is used in some way for filtering, but the models do have differences in the way they use this measure.

The running example is based on Figure 1 where a manager of an enterprise finds that the gross profit in December 2013 is significantly lower compared to December 2012. There are two dimensions used in the simplified example; CountryName and ProductCategory. The dimension CountryName consists of {Australia, Canada, China, France, India, Russia, Sweden, the Netherlands, United Kingdom} and ProductCategory: {Audio, Cell phones, Computers, Movies and Music, TV and Video}. The same problem is solved with all three models to understand and compare how the different models work.

In the next sections we describe the following models of explanation:

- Explanation formalism – EF (Section 3.2)
- Informative summarization – IS (Section 3.3)
- Explanation by intervention – EbI (Section 3.4)

3.2 Explanation formalism

Explanation formalism is essentially based on Feelders and Daniels' (2001) explanation method, which has then been extended by Caron (2013). This section is a summarisation

of these two articles and to ensure readability most further references to these articles are omitted.

Explanation formalism was originally developed for the use of OLAP data rather than business dashboards. Both share similarities however, as the assumption is that business dashboards use a star database schema. The main difference may be that in business dashboards, there are less dimensions included in one graph or problem. These similarities make it natural to adapt a OLAP based model to business dashboards.

3.2.1 Canonical formalism

The model is based on the idea of *contributing* and *counteracting* causal influences. The causal explanations' canonical formalism is:

$$< \boldsymbol{a}, \boldsymbol{F}, \boldsymbol{r} > \text{ because } Cb, \text{ despite } Ca. \quad (3.1)$$

In the formula, <*a,F,r*> is the event that requires explanation. The actual value is a, property F shows the deviation of the variable from its reference object *r*. *Cb* and *Ca* stand for contributing and counteracting sets of causes, respectively. *Cb* is non-empty while Ca can be empty. The actual explanation consists of the causes from the contributing set, *Cb*. Counteracting causes, *Ca*, is not part of the actual explanation, but does give information on how exactly the *Cb* has affected the event.

In this thesis, the event to be explained is a business measure, where an exceptional value has been found and there exists a normative reference value. This mean that the actual object *a,* and reference object *r* are both clear from the context. Therefore, it is possible to simplify the canonical explanation to:

$$\delta y = q \text{ because } Cb, \text{ despite } Ca. \quad (3.2)$$

Here the δy = q corresponds to the event, which is the difference of the actual value y^a and the reference value y^r. In this formula q can have 3 distinct qualitative values as seen in Table 2. $\delta y = Low$

Table 2 Mapping q to qualitative value, adopted from Feelders and Daniels (2001)

Actual vs reference value	q
$y^a > y^r$	High
$y^a = y^r$	Normal
$y^a < y^r$	Low

When q = normal it is not required to find an explanation, since there is no deviation. In other words, the two cases we have is when q is either a negative or a positive value.

Explanations generated by the explanation formalism are based on general laws that express the relations between events. These laws can be divided into two when discussing business problems; the system of business model equations M (e.g. revenue = price*quantity) and the system of drill-down equations (such as in business dashboards).

Example 3.2.1 In the running example, the manager had noticed that there was a notable decrease in the gross profit of December 2013 in physical stores compared to December 2012. Here the actual value a is GrossProfit(December 2013, Store), reference value r is GrossProfit(December 2012, Store), and F stands for the difference between the two. By mapping these values to Table 2, the manager can define that in this problem $q =$ Low.

3.2.2 Measure of influence

A business model M can be expressed as a quantitative function y = *f*(x), where x = (x_1, x_2, x_3,…,x_n) denotes an n-component vector. Then a measure of influence can be defined as:

$$\mathrm{Inf}(x_I, y) = f(\mathbf{x^r_{-i}}, x_i^a) - y^x, \tag{3.3}$$

where $f(x^r{}_{-i}, x^a{}_i)$ denotes the value of *f*(x) with all other variables evaluated with their reference values, except the measure x_i. In words, inf(x_i,y) indicates what would have been the difference between the actual and the reference value of y, if only x_i had deviated from its reference value. By using this hypothetical situation, it is possible to estimate the inf-measure. However, the interpretation of the inf-measure changes depending on the form of the function *f*. For additive functions, the inf(x_i,y) can be interpreted as a quantitative specification of the change in y, explained by the change in x_i.

Calculating the inf-measure enables the establishment of the set of contributing and counteracting causes (Ca and Cb). This can be done with the equation:

$$\mathrm{inf}(x_i, y) \times \delta y > 0 (< 0). \tag{3.4}$$

In practice, this means that those variables whose influence value has the same sign as δy are contributing causes, and counteracting causes are those with values of the opposite sign.

Parsimonious set of causes is established by means of reduction method (RM). It is calculated as the smallest subset of Cb, such that its influence on y is more than a chosen fraction of the influence in the complete set (T+). The formula for this is

$$\frac{\mathbf{inf}\ (\boldsymbol{Cb_p}, \boldsymbol{y})}{\mathbf{inf}(\mathbf{Cb}, \mathbf{y})} \geq \boldsymbol{T^+} \tag{3.5}$$

where the fraction T^+ is an empirically chosen number between 0 and 1. Parsimonious set of counteracting causes Ca_p is defined similarly, with T^- as the fraction.

Example 3.2.2 Continuing the running example, after the manager has identified the problem situation, measure of influence can be calculated. In Table 3 the example data has been separated by the dimension "country", and shows the actual, reference, and influence value for each. The calculated parsimonious set of contributing causes with $T^+ = 0.85$ for the example data is Canada (48%), United Kingdom (21%), Australia (10%), and France (7%). Parsimonious counteracting causes is only China (90%).

Table 3 Measure of influence of countries on total gross profit

CountryName	Actual (2013)	Reference (2012)	Influence	Influence %
Canada	2105956	2885247	-779291	48%
United Kingdom	2002397	2348615	-346218	21%
Australia	1359612	1527150	-167538	10%
France	888137	996577	-108440	7%
Russia	779722	872523	-92801	6%
the Netherlands	121389	196580	-75191	5%
India	1444971	1499366	-54395	3%
Sweden	141978	113385	28593	10%
China	3091110	2839041	252069	90%

This interpretation of the influence measure requires that two different constraints are fulfilled: *consistency constraint* and *conjunctiveness constraint*. Consistency constraint states that the reference values and actual vales must satisfy the same functional requirements, i.e. $y^a = f(x^a)$ and $y^r = f(x^r)$. Conjunctiveness constraint holds for a quantitative equation when for all subsets $X \subset \{x_i, \dots, x_n\}/\{x_i\}$:

$$\mathbf{inf}(\boldsymbol{x_i}, \boldsymbol{y}) \geq \mathbf{0} \rightarrow \mathbf{inf}(\boldsymbol{X} \cup [\boldsymbol{x_i}\}, \boldsymbol{y}) \geq \mathbf{inf}(\boldsymbol{X}, \boldsymbol{y}), \tag{3.6}$$

$$\mathbf{inf}(\boldsymbol{x_i}, \boldsymbol{y}) \leq \mathbf{0} \rightarrow \mathbf{inf}(\boldsymbol{X} \cup [\boldsymbol{x_i}\}, \boldsymbol{y}) \leq \mathbf{inf}(\boldsymbol{X}, \boldsymbol{y}). \tag{3.7}$$

In words, this constraint means that the influence direction of a single variable should not change when it is considered in conjunction with other variables. So, if the influence of a single variable x_i is positive, the influence is still positive when more variables are added into the equation. This constraint ensures that significant causes can be added together as a total set. When considering business problems, it is of note that two large groups of functions satisfy the conjunctiveness constraint: *monotonic* and *additive functions*. Monotonicity in this context means monotonicity of all variables separately. Nearly all financial models are monotone, meaning this constraint holds in almost all financial problems.

3.2.3 Maximal explanation

Explanations derived from a single equation of a business model M generates "one-level" explanations. However, it is purposeful to continue an explanation of $\delta y = q$ by explaining the differences of actual and reference values of its contributing causes. This way, causes can be further explored, from level one to the next level in the business model. This can be done until a *maximal explanation* is achieved.

Here the different effects of contributing and counteracting causes can be seen clearly. Only the contributing causes are continued for maximal explanation, whereas counteracting causes are not further explained. By going through the different levels of contributing causes, an *explanation tree* is formed. Usually only parsimonious causes are added into the tree, to ensure simplicity and robustness of explanation.

Example 3.2.2 In the example the parsimonious set of contributing causes (Canada, United Kingdom, Australia) was determined. To continue the explanation for maximal explanation the manager can look for another dimension: Product Category. In Figure 9 an explanation tree is created from the parsimonious causes of the two dimensions: country and product category. From this tree it is possible to already intuit that the most significant causes are these certain countries and product categories, whereas Canada and Cameras seem to be the main causes. In the figure, the counteracting causes are denoted with a dashed line, whereas the percentages represent the fraction of influence measure of the symptom.

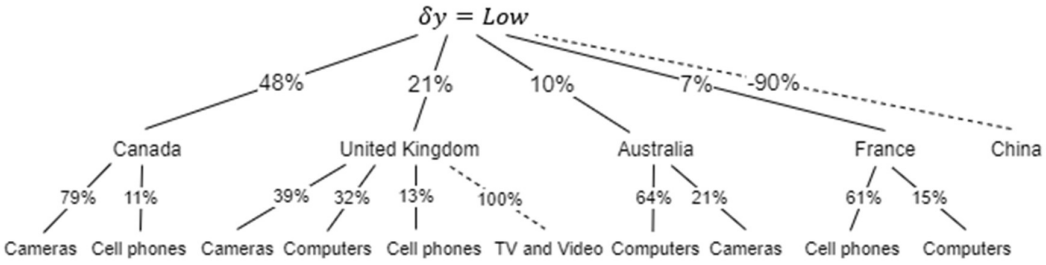

Figure 9 Explanation tree formed from maximal explanation

3.2.4 Look-ahead explanation and hidden causes

Feelders and Daniels' (2001) maximal explanation's shortcoming is the inability to deal with *neutralisation* or *cancelling-out effects*. Cancelling-out effects can occur at any level in the business model. This phenomenon occurs when the effects of lower-level variables cancel each other out, so that the higher-level variables' influence is partly or fully

neutralised. For example, the more than normal profits in January may be cancelled out by less than normal profits in February and March. In this case, on the quarter level the aggregated measure might seem normal and no exception is found. These are *hidden causes* which are often found in financial data. Caron (2013) presented a look-ahead explanation to solve the issue in maximal explanation, by making the hidden causes visible by substitution.

When explaining a symptom $\delta y = q$ where the following equations are from the business model M:

$$\boldsymbol{y = f(x) \subset M^{p;(p+1)}}, \tag{3.8}$$

$$\boldsymbol{x_i = g_i(z) \subset M^{(p+1);(p+2)}}, \tag{3.9}$$

where $x = (x_1, \dots, x_n)$ and $z = (z_1, \dots, z_n)$. In the equations 3.8 and 3.9 a subset of equations from the business model is represented by $M^{(p;p+1)}$. This means that a variable on the level *p* is expressed with variables of level *p+1*. Now suppose that we have a variable x_i which is not significant (does not belong to the found parsimonious set of causes), meaning the variable's influence on *y* is marginal. In order to make sure we have a complete explanation, all children of x_i are investigated for possible cancelling-out effects. Therefore, a new equation is derived:

$$\boldsymbol{y = h_i(x, z) \subset M^{p;(p+2)}}. \tag{3.10}$$

This is from the result of substituting all equations on the lower level into the parent equation. The equation 3.10 can be used for finding causes that are possibly not captured by the maximal explanation method. The procedure is called *one-step look-ahead*. Figure 10 illustrates how the one-step look-ahead works in practice. The figure is a diagram that represents how the derived algorithm $M^{p;(p+2)}$ works.

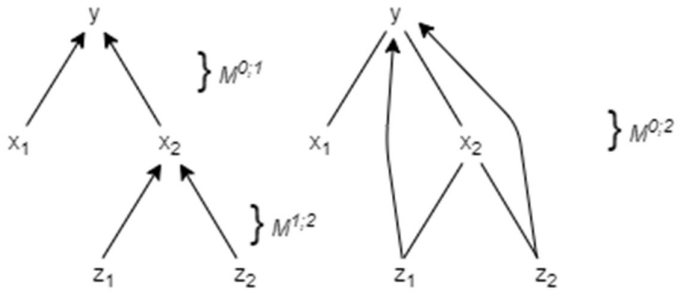

Figure 10 Explanatory graphs for one-step look-ahead (Caron, 2013)

Finally, the definitions for contributing and counteracting hidden causes, and their influences on symptom δy:

"Variable z_j is a *contributing hidden cause* when $z_j \subset Cb_p(y)$ and x_i not $\subset Cb_p(y)$, where z_j is a successor of x_i."

"Variable z_j is a *counteracting hidden cause* when $z_j \subset Ca_p(y)$ and x_i not $\subset Ca_p(y)$, where z_j is a successor of x_i." (Caron, 2013)

The influence of z_j on y is given by:

$$\mathbf{inf}(\boldsymbol{z_j}, \boldsymbol{y}) = \boldsymbol{f}\left(\boldsymbol{x^r_{-i}}, \boldsymbol{g_i}(\boldsymbol{z^r_{-j}}, \boldsymbol{z^a_j})\right) - \boldsymbol{f}(\boldsymbol{x^r_{-i}}, \boldsymbol{g_i}(\boldsymbol{z^r})), \tag{3.11}$$

While the influence of x_i on y is given by:

$$\mathbf{inf}(\boldsymbol{x_i}, \boldsymbol{y}) = \boldsymbol{f}(\boldsymbol{x^r_{-1}}, \boldsymbol{x^a_i}) - \boldsymbol{f}(\boldsymbol{x^r}) = \boldsymbol{f}\left(\boldsymbol{x^r_{-i}}, \boldsymbol{g_i}(\boldsymbol{z^a})\right) - \boldsymbol{f}(\boldsymbol{x^r_{-i}}, \boldsymbol{g_i}(\boldsymbol{z^r})). \tag{3.12}$$

In words, the other variables in vector z neutralise the effect of z_j.

Example 3.2.3 Continuing the last example's explanation tree, it is possible that the manager did not notice a significant cause from one of the countries that were deemed insignificant. With the algorithm for hidden causes the manager goes through the insignificant causes of Table 3. The insignificant dimensions (India, Russia, Sweden, and the Netherlands) are combined into Table 4 with the calculated influence of the highest and lowest influencing z_j on y, according to Equation 3.11. In this case the manager notices that India's highest contributing hidden cause (Computers, 5%) is higher than India's contributing influence in Table 3. This is a probable hidden cause but is not part of the parsimonious set since it is lower than France's total influence of 7%.

Table 4 Influence of hidden causes on total gross profit

CountryName	ProductCategory	Gross-Profit2012	Gross-Profit2013	Influence	Inf %
India	Computers	611174	525527	-85647	5%
India	TV and Video	294287	329426	35139	13%
Russia	Cameras	265624	207660	-57964	4%
Russia	Computers	290560	322388	31828	11%
Sweden	Cell phones	11332	5218	-6114	0%
Sweden	Cameras	39370	43959	4589	2%
the Netherlands	Computers	94202	66279	-27923	2%
the Netherlands	Movies and Music	3215	4594	1379	0%

To conclude, the explanation formalism consists of business model M together with dimensions to form a maximal explanation, which is then extended by the dimensional hierarchies to find hidden causes. This is used to form an explanation tree for visualisation

of the parsimonious causes. Additionally, explanation formalism relies on two assumptions; that consistency and conjunctiveness constraints hold.

3.3 Informative summarization

In this section an alternative explanatory model is presented, known as *informative summarization* developed by Sarawagi (2001). Sarawagi (2001) created a framework for explaining differences in aggregated functions for dimensional data. Therefore, it is possible to extend this for business dashboards as they use star database schema. Further references to Sarawagi are omitted from this section to ensure readability.

Informative summarization was developed for OLAP data, like explanation formalism. As discussed in section 3.2 the OLAP to business dashboard adaptation is natural and straightforward.

3.3.1 The general framework

The general framework of informative summarization works through iDiff operator, which returns summarized explanations for increases or decreases in aggregated measures. The original framework was developed for OLAP cubes, where the idea is to consider the differences between two aggregated quantities g_a and g_b. Both quantities are formed by aggregating their values over one or more dimensions. C_a and C_b denotes the two sub-cubes generated by expanding the aggregated dimensions of g_a and g_b. The sub-cubes' cell values are defined by $v_a \subset C_a$ and $v_b \subset C_b$.

Example 3.3.1 In the running example, g_a denotes the gross profit for the cell (All countries, All product categories, December 2012) = 13278484, and g_b denotes the gross profit for (All countries, All product categories, December 2013) = 11935272. C_a is the sub-cube with dimension {CountryName, ProductCategory} for Time = December 2012 and C_b denotes a sub-cube with the same set of dimensions, but Time = December 2013. v_a is a value in C_a, for example with dimensions {Australia, Cell phones} $v_a = 123847$ and with the same dimensions $v_b = 155103$.

Sarawagi's (2001) model groups cells with similar changes together to form a compact summary of the difference of C_a and C_b. For the summarisation there are several different methods that arise from the choices made for grouping criteria, error function, grouping format, and objective function.

Firstly, the *grouping criteria* determines how to measure commonality and similarity in the relationship between the values v_a and v_b that belong in C_a and C_b respectively.

For example, it is possible to group the values together if there is a similar absolute difference between v_a and v_b or if they have the same ratio. Then each collection of cells that have a similar value of the chosen relationship are grouped together and summarised. Ratio is a better grouping criterion than the absolute difference for most dimensions such as geographical locations or product categories. Table 5 shows how the absolute differences vary by a large amount, whereas the ratios have significantly less variance. The r for gross profit over the dimension {CountryName} is 0.9 as can be seen in Table 5.

Table 5 Gross Profit over CountryName with calculations

CountryName	2013	2012	Difference	Ratio	Error
Australia	1359612	1527150	-167538	0.89	12478
Canada	2105956	2885247	-779291	0.73	10147961
China	3091110	2839041	252069	1.09	10338161
France	888137	996577	-108440	0.89	6526
India	1444971	1499366	-54395	0.96	677957
Russia	779722	872523	-92801	0.89	2635
Sweden	141978	113385	28593	1.25	1328198
United Kingdom	2002397	2348615	-346218	0.85	573983
the Netherlands	121389	196580	-75191	0.62	2076294
Total	**11935272**	**13278484**	**-1343212**	**0.90**	

The *error function* determines how much is the error of the actual value v_a and the reference value derived from v_b with the summary statistic of its group. Sarawagi (2001) presented an error function for dimensional data:

$$\mathrm{Err}(\boldsymbol{v_a}, \boldsymbol{v_b}, \boldsymbol{r}) = (\boldsymbol{v_b} - \boldsymbol{r v_a}) \log \frac{\boldsymbol{v_b}}{\boldsymbol{r v_a}}, \tag{3.13}$$

which shows the difference between the actual value v_b and the reference value rv_a. In Equation 3.12 r denotes the ratio of the group of cells that the values v_a and v_b belong to. This error function gives high weightage to both larger absolute differences and to a larger ratio, making both magnitude and ratio of change important. Table 5 shows the error calculated on example data.

The *grouping format* is used to determine what cells are considered for grouping together. It is not meaningful to group together an arbitrary subset of members over a dimension, as dimensions are normally categorical. Dimensional hierarchies, however, usually provide natural boundaries for grouping. This results in easily interpretable and meaningful outcome which can be further analysed.

The *objective function* has two different choices: user specified limited number of rows that minimises the total error, or user specified limited total error e_{ma} where the

objective is to find the smallest subset of rows that meets this limit. In the running example, the limited total error e_{max} is used.

As seen in Table 5, since the error is always positive it may not be intuitive when looking for causes to a certain symptom. To solve this problem, it is possible to use *One-sided errors*. In this case, we modify the error function by having it change the value to "0" if the sign of the change is opposite of that of the symptom, i.e. the topmost parent.

Table 6 Summarisation of gross profit over country and product category

ProductCategory	CountryName	2012	2013	Ratio	Error
(All)-	(ALL)-	5071062	5181549	1.02	0
Cameras	Australia	495108	375740	0.76	117250
Cameras	Canada	894617	511863	0.57	1320100
Cameras	France	296347	182775	0.62	314840
Cameras	Russia	290560	207660	0.71	122670
Cameras	the Netherlands	94202	25052	0.27	726090
Cameras	United Kingdom	758187	405870	0.54	1428410
Cell phones	Sweden	19155	5218	0.27	143250
Computers	(ALL)-	4463284	4430822	0.99	416300
Computers	Australia	604404	461588	0.76	133070
Movies and Music	(ALL)-	162339	101319	0.62	162680
Movies and Music	Canada	67182	20530	0.31	430000
Movies and Music	India	34278	15263	0.45	109210
Movies and Music	Russia	27759	10023	0.36	136150

Example 3.3.2 In the running example, the manager wonders why the gross profit of the company has fallen in 2013 compared to 2012. Therefore, the manager can summarise the rest of the rows. However, since the manager is wondering on a decrease of gross profit, he uses one-sided error correction to only take into consideration the rows that have affected the decrease. In this example, $e_{max} = 10000$. Table 6 shows the outcome of the summarisation, where the first line shows all categories and countries, except the ones listed below it. According to this summarisation, the manager can see that Cameras, Computer, and Movies are the largest causes of the decrease in gross profit, whereas in certain categories, such as Cell phones there is only Sweden that is cause for concern. Largest singular causes according to Table 6 are {Cameras, Canada} and {Cameras, United Kingdom}.

To conclude, informative summarization assumes a star database schema to go through dimensions. The number and hierarchy of dimensions do not matter to the model, and there are multiple different options to choose from depending on the type of question and data. The model does not support non-additive measures, although Sarawagi (2001)

does suggest some bypass methods. The method is bottom-up in the sense it starts at the most detailed level and rolls-up from there. It relies heavily on the order of the dimensions to find hidden causes, since on the higher levels they can be easily missed.

3.4 Explanations by intervention (Roy & Sucio)

The third framework to be discussed is Roy and Sucio's (2014) formal approach to explanations for database queries. In this section, I adapt the framework from databases to a more general framework that can be used with Python and in business dashboards. Most further references to Roy and Sucio (2014) are omitted from this section to ensure readability.

The formal approach to explanations was developed for databases and to be used with SQL queries. The adaptation from database queries to business dashboard context is not completely straightforward. Additionally, the original framework uses count() data as the variables, whereas in this thesis the framework has been adapted to be used with sum() financial data. As the equation for measure of impact as well as the algorithm is generic enough, it could be possible that the model be extended to values such as avg() and even max() and min().

3.4.1 Formal framework

The framework is based on two main concepts; intervention and causal paths.In this framework, a user question is expressed by (Q, dir), where Q denotes the numerical query and $dir \subset \{high, low\}$ is a direction specifying whether the Q is thought to be higher or lower than the reference value. Q can be thought of as an aggregated measure instead of a numerical query when dealing with business dashboard data that follows a star schema. $Q = E(q_1, \dots, q_m)$ expresses how Q is composed of q_i that are any single aggregate operator, and E is an arithmetic expression consisting of numerical operators.

A candidate explanation ϕ is a combination of predicates on attributes, therefore

$$\phi_j = [R_i, A \text{ op } c], \tag{3.14}$$

where R_i is a relation, and A is an attribute of that relation, and $\text{op} \subset \{=, <, \leq, >, \geq\}$. In this thesis a relation can be understood as a dimension, and the operator used is only $\text{op} \subset \{=\}$.

Table 7 Aggravation and intervention for gross profit in United Kingdom

ProductCategory	2012	2013	Q	μ_{aggr}	μ_{interv}
Cameras	495108	405870	0.82	-0.82	0.92

Computers	1110167	1035215	0.93	-0.93	0.87
Cell phones	201628	171640	0.85	-0.85	0.91
Movies and Music	51894	31390	0.60	-0.60	0.91
Audio	67182	50568	0.75	-0.75	0.91
TV and Video	296347	307714	1.04	-1.04	0.88
Total	**2222326**	**2002397**	**0.90**		

Example 3.4.1 The manager is trying to find an explanation for the decrease in gross profit in United Kingdom in the year 2013, compared to the year 2012. Here $Q = \frac{q_1}{q_2} = 0.90$ and $dir = low$. The candidate explanations are the product categories that are present in the country.

3.4.2 Measure of candidate explanation

There are two methods for calculating the measure of a candidate explanation in Roy and Sucio's (2014) framework: *aggravation* and *intervention*. The first option is to *aggravate* the candidate explanation, which simply means calculating the value of Q for the all subsets ϕ and then rank them by decreasing order of their measure of impact. The sign of the measure of a candidate explanation ϕ is therefore

$$\boldsymbol{\mu_{aggr(D,Q,dir)}(\phi)} = \begin{cases} \boldsymbol{-Q(D_\phi)\ if\ dir\ =\ low} \\ \boldsymbol{Q(D_\phi)\ if\ dir\ =\ high} \end{cases}, \tag{3.15}$$

where D denotes the database. In words, the explanation measure's sign follows the sign of the symptom's direction, meaning explanations with even lower values compared to $Q(D_\phi)$ will have top ranks. Aggravation is a simple method but has certain limitations: by calculating only on the percentage of changes, it may find explanations that are small but have changed majorly.

An intervention is a set of measures to be deleted from database D. Intervention is denoted as $\Delta = (\Delta_1, \dots, \Delta_k)$ and $D' = D - \Delta$ is the residual database instance after removing the intervention. The *measure of impact by intervention* is the amount by which the measures in $Q(D')$ move to the direction expected by the user. The equation for intervention is

$$\mu_{interv(D,Q,dir)}(\phi) = \begin{cases} Q(D - \Delta^\phi) \text{ if dir} = \text{low} \\ -Q(D - \Delta^\phi) \text{ if dir} = \text{high} \end{cases}. \tag{3.16}$$

Interestingly, the sign for μ_{interv} is the opposite of that for μ_{aggr} as seen from Equation 3.14. It is explained by the goal; here we want to intervene on the problem by changing the value of Q to the opposite direction of *dir*.

Example 3.4.2 In the example, because the $dir = Low$ the aggravation measure changes signs, whereas the intervention does not. According to μ_{aggr} in Table 7 the manager notices that the top reason for France's decreasing gross profit is Movies and Music product category. However, the manager knows this can be misleading since the category is comparatively smaller. Therefore, the manager looks at the measure by intervention which gives a different answer: the leading explanation is the Cameras category.

When considering multidimensional problems, blindly looking at the μ_{interv} or μ_{aggr} can include many redundant explanations. Therefore, a filtering measure called *minimal append* is used to comb through the explanations and remove redundant ones. A good example is if the top explanation is {ProductCategory: Audio, CountryName: (ALL)}, the second explanation {ProductCategory: (ALL), CountryName: France}, and third explanation would be {ProductCategory: Audio, CountryName: France}. In this case the third explanation is dominated by both the first and second, and therefore is redundant. The first two explanations are called *minimal explanations*. To find a minimal append of explanations, they are ranked from on descending order of intervention or aggravate measure, and then appended one by one to a new list, while checking for redundancies.

3.5 Comparison of the three models

In this section the models are compared to find similarities and differences between them. The findings are gathered to Table 8 in order to show a summarised and clear distinction between the models. In the next paragraphs Table 8 is explained in words and the argumentation for the choice of comparison points are presented.

Dimension	Explanation formalism	Informative summarization	Explanation by intervention
Recursion	Top-down	Bottom-up	Big Bang
Equation for impact	$\mathrm{Inf}(x_I, y) = f(\mathbf{x}^{\mathbf{r}}_{-\mathbf{i}}, x_i^a) - y^x$	$\mathrm{Err}(v_a, v_b, r) = (v_b - rv_a)\log\frac{v_b}{rv_a}$	$\mu_{interv(D,Q,dir)}(\phi) = \begin{cases} Q(D - \Delta^{\phi}) \text{ if dir = low} \\ -Q(D - \Delta^{\phi}) \text{ if dir = high} \end{cases}$
Filter mechanism	Fraction T to filter parsimonious set, ranking	Summarise and rank by Error	Rank by Intervention, create minimal append
Measure of impact	Absolute	Both magnitude and ratio	Ratio of change
Assumptions	Consistency & Conjunctiveness, set of equations	Additive	Additive or ratio based on count()
Conditions set by user	Choose value T	Choose an order for dimensions, Choose err_{max}	None
Hidden causes	Yes	Some	Yes
Visualisation	Explanation tree	List	List
Original framework	Originally OLAP data	Originally for OLAP data	Originally for database and SQL queries

Table 8 Comparison of explanatory models

The main difference between the models, besides the differing algorithms for measuring the explanations, is the recursion method. The EF model by Feelders and Daniels (2001) and Caron (2013) use a top-down approach, where the model starts by going through the nearest dimension and moving downwards based on contributing causes. The IS model by Sarawagi (2001) on the other hand start on the most detailed level and rolls up along the dimensions by summarising the non-significant causes. The EbI model by Roy and Sucio (2014) however, use a big bang method, where all the different candidate explanations are gathered in one table and simultaneously calculates the measure of impact for each possible cause.

EF's algorithm for measuring the explanations consider the absolute difference if the candidate explanation is substituted to the reference value, whereas EbI considers the ratio of change when the candidate explanation is removed from the reference value. IS, however, offers a powerful algorithm to calculate the error value, which considers both magnitude and ratio of the difference between the actual and the changed. Intuitively these differences affect the outcome of the candidate explanations, where EF has the highest absolute impact, explanation by intervention the ratio of the impact, and informative summarization somewhere in the between.

The assumptions for the different models are similar, with the main outlier being explanation by intervention which can handle ratios such as profit margins. The conditions set by user means the setup that a user must do before the model is usable. EF and IS both require a setup for the threshold of significant causes, although this can be automated through repeated testing. IS additionally requires the user to order the dimensions on the similarities. EbI requires no setups or thresholds. To conclude, there are possible setups that a user may need to do, although it is possible to circumvent these by recursive testing. Therefore, all three models are suitable for automatic diagnostics.

All three models can find hidden causes in their own way. EF finds them by the look-ahead algorithm, whereas explanation by intervention due to the fact is calculates the measure for all possible candidate explanations. IS may miss some hidden causes due to the dependency on the order of the dimensions.

EF strong point is the possibility to visualise the explanations in the form of a tree, whereas the other models only provide a list of top explanations. The original framework of the models is for OLAP data, which is extremely suitable for business dashboard integration, except for explanation by intervention that was developed for databases and SL queries.

4 EXTENDING BUSINESS DASHBOARD WITH EXPLANATORY ANALYTICS

This chapter presents how business dashboards can be extended with explanatory analytics, through UML diagrams. First, the modelling of the implementation is discussed, then use-cases for the extension are presented. Finally, the class diagram for the package is detailed.

4.1 Proof-of-concept modelling

In the thesis the business dashboards are not fully integrated with the models, but instead it is shown that it is possible to use the models in the dashboard environment through a prototype implementation. This proves the possibility of future integration, with the foreknowledge of how the implementation works. Additionally, the modelling itself is a proof-of-concept build and is not fully optimised for commercial use.

However, the artifact is suitable for comparisons on their functionality and usability in theory and in business environment. This is ensured by grounding the models in their theoretical framework and creating a foundation for comparison by using same data and same methods for data preparation.

4.1.1 Exporting data from power bi

In order to verify that the python implementation works in a business dashboard environment, data exported from Microsoft Power Bi is used. This is to ensure authenticity and validity of the concept, as commercial solutions for business dashboards use data in a slightly different form when compared to OLAP data. For one, the dimensions are less and there can be some discrepancies mixed in with the data.

Microsoft Power Bi PRO version offers an easy-to-use export function which can be used to export the data to a comma-separated-value (csv) format. Csv files are a popular format for data and can be used in many ways. In Figure 11 the Export data function is shown as it is presented in Power Bi.

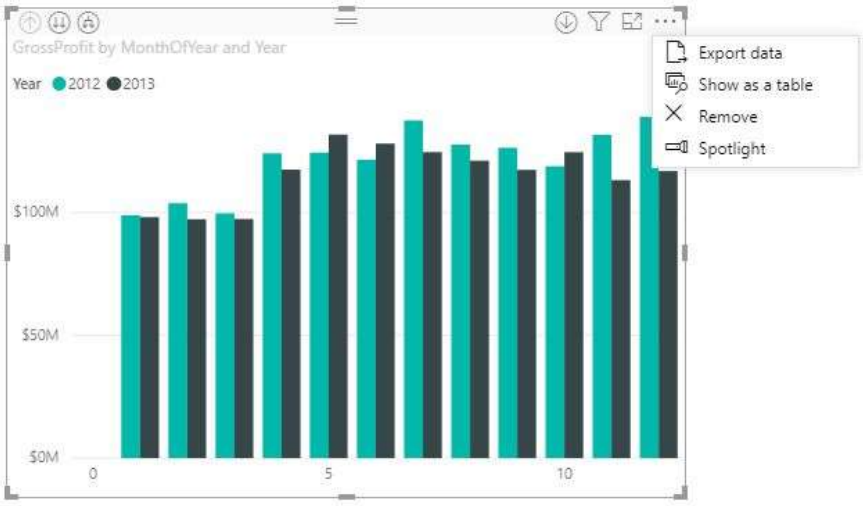

Figure 11 Exporting data from Power Bi

To note is, that the function exports the data as it is shown; the filters and dimensions in place anywhere on the graph are exported. For example, if the whole dashboard page has been filtered to not include the product category “Audio” then even if it is not included in the graph’s dimensions, the data is still filtered through the product category dimension.

Since the data exported from Power Bi is in .csv format, and the exact format and types of data can vary, it needs to be standardised so the explanatory models can use it. In section 4.2 and 4.3 the use-cases and class diagram illustrate that the extension of business dashboards require a Data module, that can extract, clean, and transform the data into usable and uniform format.

4.2 UML Use Case diagrams for implementation

Unified-Modelling Language (UML) use cases are used to show a high-level view of the implementation, in order to represent how the artifact works. First, a use case for explanation formalism is presented in sub-section 4.2.1 as an example, and then the combined use case for the implementation is shown and further explored in subsection 4.2.2.

4.2.1 Use case for explanation formalism

The use case for explanation formalism is presented in Figure 12, which consists of 3 parts: the analyst, the diagnosis system, and power bi. The analyst in this case can be an actual analyst or a manager seeking to analyse data for insights. There are four main

activities in the use case diagram: Importing the data, computing the causes, visualising the data, and exporting the data to a dashboard.

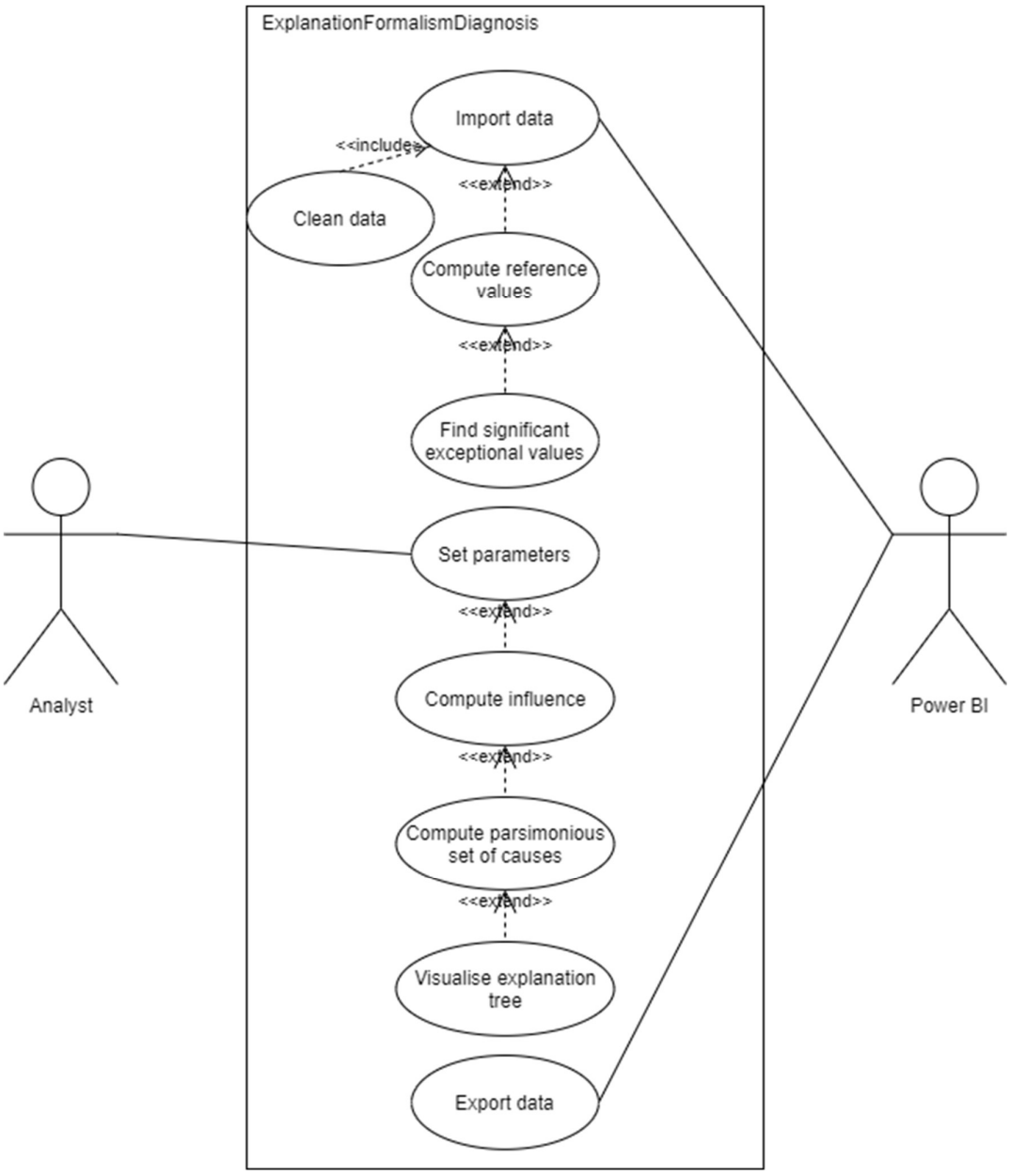

Figure 12 Use case for explanation formalism

Importing the data from Power BI is simple, as Power BI supports the activity by an easy-to-use export-to-csv file function. The diagnosis system then cleans and computes reference values of the imported data, in preparation for the actual diagnosis.

The analyst sets the parameters for the diagnosis, such as the threshold for significant exceptional values and parsimonious causes. The system runs the diagnosis with the set

parameters and returns the computed causes, which can then be used to provide visualisation in the form of an explanation tree. After this the data can be exported back to Power BI, with the goal of providing additional insights and increased automation for analysing business data through business dashboards.

4.2.2 Combined use case

As all the explanation models have similar use case in the data preparation phase, the use cases were combined to ensure ease of use and implementation. The use cases for informative summarization and explanation by intervention are found in the Appendix B. Figure 13 illustrates the combined use case for the three models and how they can be implemented with 4 moduless: DataManipulation, ExplanationFormalism, InformativeSummarization, and ExplanationByIntervention. These are combined into one system: AutomaticDiagnosis to create an artifact that allows the use and comparison of all three models.

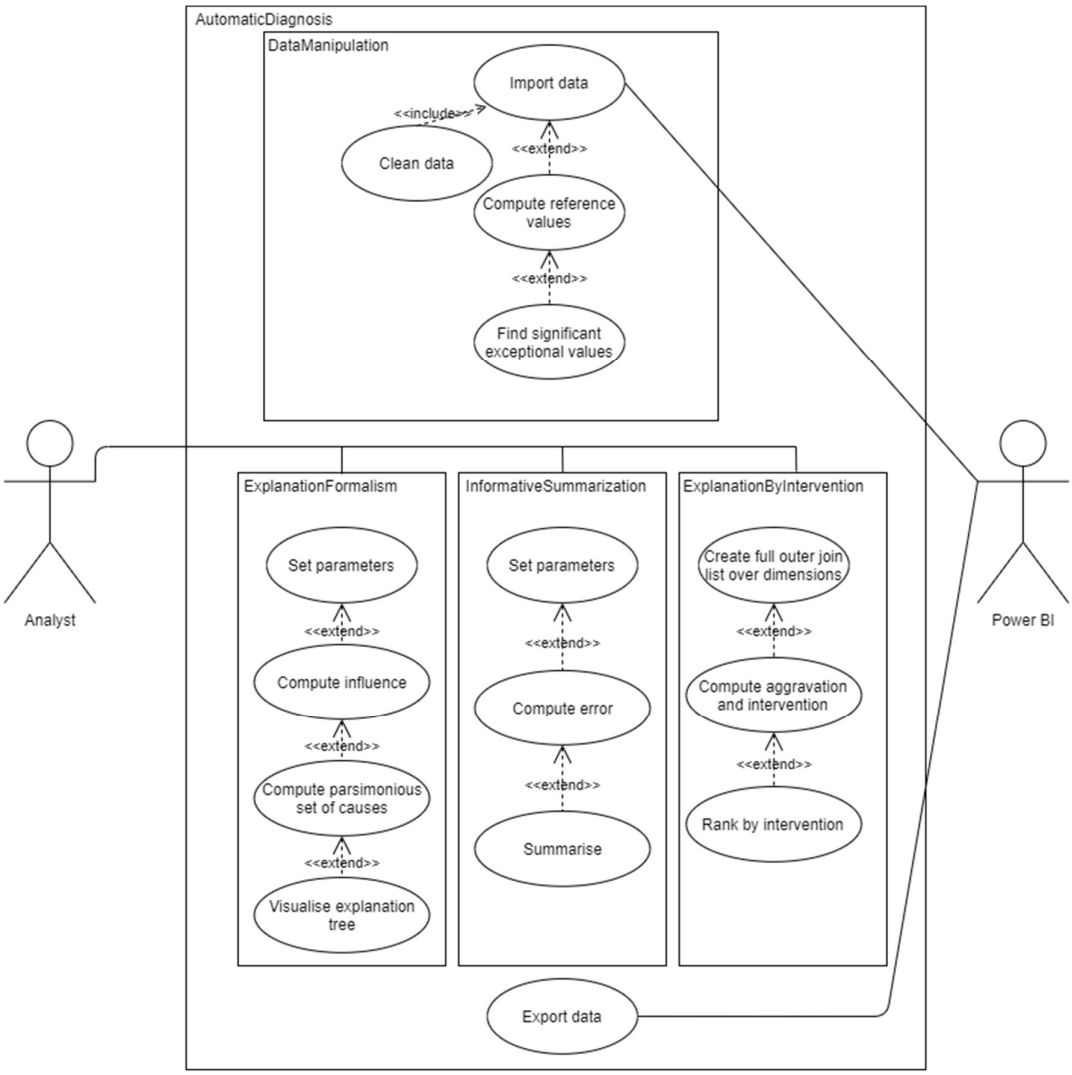

Figure 13 Combined use case for three models

The reason for combining the three models into one use case is to increase code reusability and ensure solid foundation for comparison. In the combined use case, there are similarly two actors: human actor analyst and a system actor “Power Bi”. The human actor could also be a manager or any user in need to explanations. The diagnosis starts with the system importing data from Power Bi, which includes cleaning it and then computing reference values and finding significant exceptional values.

After the data is done with initial preparation, the analyst can then choose to use any of the three models on a certain exceptional value found from the data. Both explanation formalism and informative summarization requires the user to set parameters, whereas explanation by intervention can be done automatically, just with data and a set direction of the problem. To note is that all use cases for explanation models follow similar

guidelines; start, computing measurements, and presenting product. Explanation formalism stands out by offering visualisation in the form of an explanation tree.

4.3 Class diagrams

Class diagram is an UML diagram that represents how classes are built and the way they interact with each other. In this section, a class diagram for all three models for explanation are shown in a single class view, together with the Data module, which extracts and cleans the data from a BI software into usable format. The diagram is presented in Figure 14. The class diagram depicts the relationships between the different classes, as well as the main attributes and functions that the classes hold.

The class diagram is divided into the same four modules; First is the Data, which extracts and cleans the raw data from a BI software. This is done with the Data class, whereas the Exceptions class allows the user to identify exceptional values in the data. The Dimension class holds supportive functions, such as groupDimensions() and reduceTo() which change the dimensions and values seen in the data.

ExplanationFormalism has the main class, Model, which computes the measure of impact for each candidate explanation and returns contributing and counteracting parsimonious sets of causes. Additionally, the ExplanationTree class allows for the visualisation of the parsimonious set of causes by creating an explanation tree. The ExplanationFormalism's Model class is the only explanatory model that directly uses the Dimension class.

InformativeSummarization has the class Model and its main functions are summarise(), group(), and computeStatistics(). As the framework relies on summarising similar values, the main functions follow the general idea. Similarly, the Model class in ExplanationByIntervention module holds three functions: dataPreparation(), computeStatistics(), and minimalAppend().

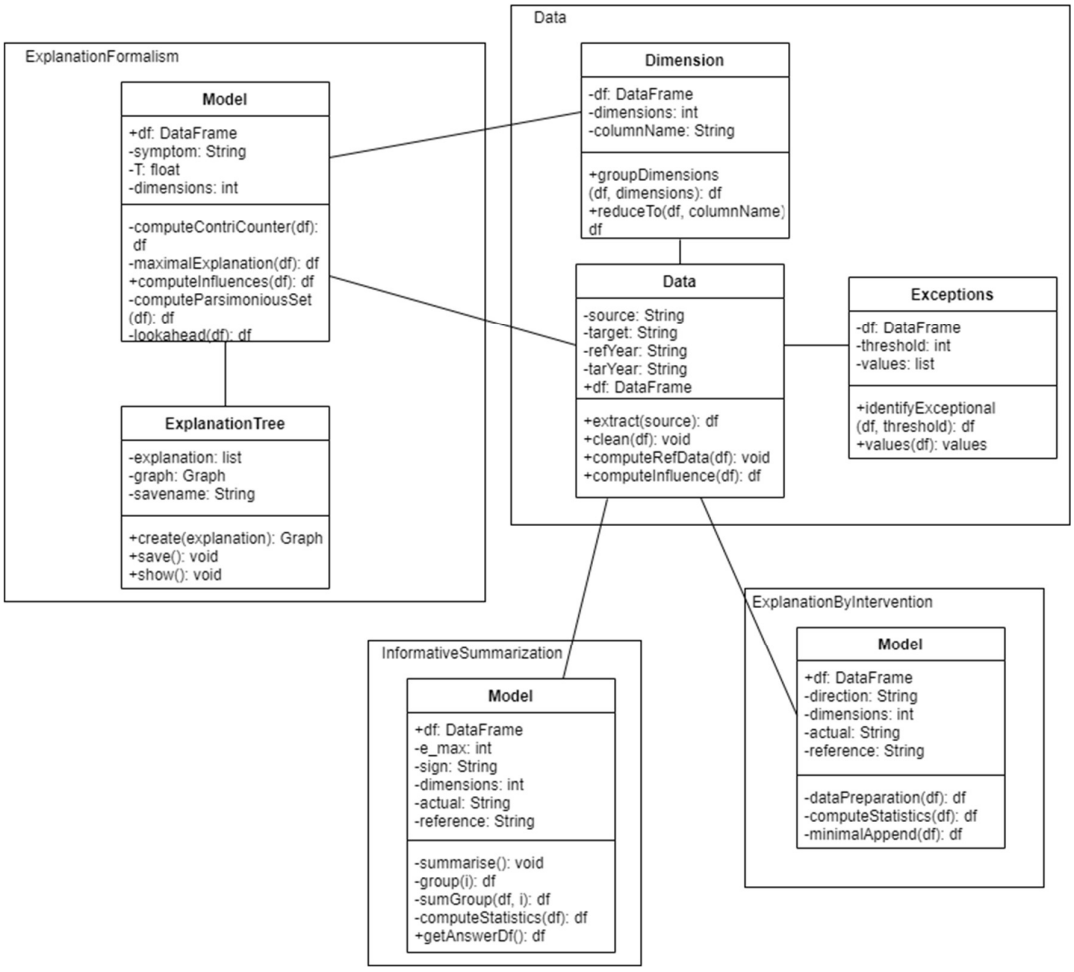

Figure 14 Class diagram of three models

In conclusion, the class diagram adequately represents how the artifact itself is built and the relationships between them. In the next chapter the creation of the artifact is shown in increased detail, as the code itself and various choices made in building the artifact are presented.

5 ARTIFACT CREATION

The design of the artifact is object-oriented and it consists of 4 modules, Data, ExplanationFormalism, InformativeSummarization, and ExplanationByIntervention. In this chapter each of the modules are explained in the same order. Interesting and useful solutions in the code are showcased, as well as each of the measures of impact for the explanatory models.

5.1 Data module

In this section some of the details and functions of the Data module are presented. The module is used for the manipulation of the data imported from Power Bi system. The purpose is to automate and standardise the data that is imported, as well as give the tools for further data manipulation to the explanatory models that they require. The Data module holds three classes: Data, Exceptions, and Dimension. The full Data module code is in Appendix C1.

The module heavily relies on pandas library and the DataFrame object and its functions for data manipulation. Pandas is a popular data manipulation and analysis tool, built upon Python to ensure flexibility and ease-of-use (Pandas, n.d.). Most popular object and the main reason behind its usage in this thesis is the DataFrame object. DataFrame allows for fast and efficient data storage and manipulation, with many powerful and intuitive built-in functions (Pandas, n.d.).

The Data class uses pandas to read .csv file and turn it into a DataFrame column with *pandas.read_csv* function. The cleaning is performed through type changes, type confirmations, and the regex code which removes extraneous letters from the values, such as the dollar or euro sign. Since the reference data models are not in the scope of this thesis, the reference data is simply computed by using the previous year's values as the reference values. The Data class requires the variables source (filename), target (column name of target, e.g. Sales), refYear (reference value year), tarYear (target year, e.g. current year).

The Exceptions class allows for identifying exceptions from the Data class, with a user determined threshold. The exceptions found can be either significantly lower or higher than the reference value. These exceptional values can then be used to form DataFrames based on them, which can be used in any of the three models. The Dimension class is a help class to deal with dimensional functions, such as grouping dimensions together or reducing dimensional values.

5.2 ExplanationFormalism module

The first model for explanation, as detailed in section 3.2 is the EF model by Feelders and Daniels (2001) and extended by Caron (2013). The main class is *Model* and the supporting visualisation class is called *ExplanationTree*. The class *Model* runs automatically when initialised with the required variables: a DataFrame, symptom, fraction T, and the number of dimensions. The first variable, a dataframe, is produced by the data module for proper fitting of the model.

The main functions of the class are maximalExplanation(), computeContriCounter(), computeParsimoniousSet(), and lookahead(). The maximalExplanation() function holds the logic behind fitting the model and the order of usage of the other functions. The maximal explanation starts by going over the first dimension and checking for contributing and counteracting parsimonious sets of causes. The code for this is:

```
self.computeContriCounter(self.df)
parsContri = self.computeParsimoniousSet(self.dfContri,1,Contri = True)
parsCounter = self.computeParsimoniousSet(self.dfCounter,1,Contri = False)
self.explanation.append([parsContri,1, 'Contri'])
self.explanation.append([parsCounter,1, 'Counter'])
```

It then goes over all the dimensions, based on parsimonious contributing sets, and appends the explanation. The explanation variable is a list of lists that is the final output of the model.

The measure of impact for explanation formalism, as shown in equation 3.3, is computed with:

```
df['Influence'] = df[actual]-df[reference]
df['InfPerc'] = df['Influence']/df['Influence'].sum()
```

where actual stands for the name of the column with the actual value and reference the name of the column with reference value. The variable df is an object of type DataFrame, and the influence percentage underneath the influence is computed for the purpose of comparing with the fraction T. After this, the measure of impact's sign is changed depending on the direction of the symptom with a simple *if* clause.

As the name suggests, computeContriCounter() computes the contributing and counteracting sets of a dataframe, whereas computeParsimoniousSet() returns the parsimonious set of a dataframe. Both of these functions depend on the direction of the symptom, e.g. whether the actual value is higher or lower than the reference value. The lookahead() function provides the possible hidden causes through comparing the higher dimensional

level values of insignificant explanations to the parsimonious set of causes. The algorithms for these functions are simple and therefore omitted. The whole class can be found in Appendix C2 for further details.

The ExplanationTree is the second class in the Module *ExplanationFormalism*. It holds the visualisation part of the model. The visualisation is created through an imported libraries *network* and *pydot* and saved with the helpful function for the library *os*. The tree is built recursively from the explanation list of lists, where contributing causes are shown with full lines, and counteracting causes with dashed lines. An example explanation tree is illustrated in Figure 15. The percentages are the measures of impact per explanation below them, the tree in question is formed by T = 0.8. The topmost part of the tree is the "symptom", and from there it moves on to the next level of explanations. Full lines represent parsimonious contributing causes, whereas dashed lines are counteracting causes. Following the model's logic explained in section 3.2, only the parsimonious contributing causes are then further explained.

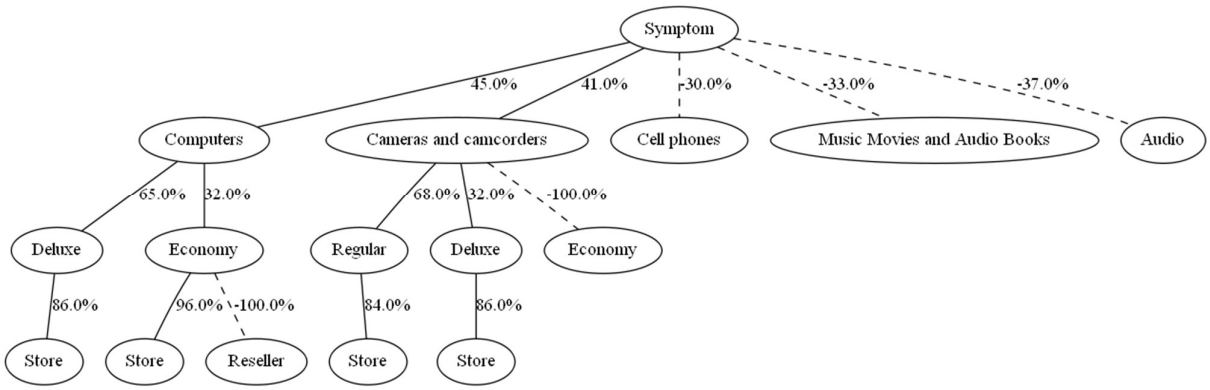

Figure 15 Explanation tree formed by the ExplanationFormalism module

5.3 InformativeSummarization module

The second model for explanation, which was detailed in section 3.3, is informative summarization by Sarawagi (2001) and is implemented in this module. It uses the Data module as a basis for receiving standardised DataFrame as an input, and then iterates over the dimensions to provide a summarised ranking of explanations. The full code of the module is in Appendix C3.

The module holds only the main class, Model, which receives the inputs of DataFrame, e_max, sign, dimensions, actual, and reference. Additionally, an optional variable is *onesided = True,* which applies the one-sided error correction from section 3.3 to the

class by default. All except e_max and direction can be derived from the Data module but are still inputs for increased flexibility. The e_max is an arbitrary number defined by the user, as discussed in section 3.3. The main function of the class is *summarise()* which holds the recursion logic of the class.

The *summarise()* method first computes the error of each row, with the method *computeStatistics()* and then iterates over the dimensions, going from the most detailed to the least detailed dimension.

```
self.df = self.computeStatistics(self.df)
self.columns = list(self.df.columns.values)
for j in range(self.i,0, -1):
  self.df = self.group(j-1)
```

where *i* is the number of dimensions in the original data, and *df* the DataFrame. The actual summarisation is computed through the function *group()* with the help of *sumGroup()*. The first method checks which rows require grouping and which do not. Then the rows to be grouped are given to *sumGroup()* and then combined back to a single DataFrame object.

The computeStatistics() method calculates the Equation 3.13 for each row of a DataFrame.

```
va = df[self.reference].astype('float')
vb = df[self.actual].astype('float')
#Ratio:
df.loc[:,'Ratio'] = (vb/va).astype('float')
ratio = (np.sum(vb)/np.sum(va)).astype('float')
#Error:
df.loc[:,'Error']=((vb-ratio*va)*np.log(vb/(ra-
tio*va))).astype('float')
```

In addition to computing the Ratio and Error rate of the rows, the *computeStatistics()* applies one-sided error correction according to the class variable *onesided*. The method uses numpy, which is imported as np for summing and logarithmic function of the error calculation.

Finally, the results of the summarization can be accessed with the method *getAnswer(top)*, where the 'top' variable is a user-defined number for how many rows the resulting DataFrame object has. The numerals in the answer are rounded to 0, and the actual and reference values have thousand separator applied to them with the lamba code

(lambda x : "{:,}".format(x)). Figure 16 shows an example of the resulting table, based on example data exported from Power Bi. The topmost explanation in the rankings is {ChannelName: Store, ClassName: Regular, ProductCategory: TV and Video, RegionCountryName: United States}. By looking at the table, one can derive that United States is a large cause, as well as Store, since both are in all the top 5 explanations.

ChannelName	ClassName	ProductCategory	RegionCountryName	GrossProfit2013	GrossProfit2012	Ratio	Error
Store	Regular	TV and Video	United States	4,327,990	6,600,313	0.66	724825
Store	Economy	Computers	United States	3,215,262	4,563,125	0.7	337277
Store	Regular	Cameras and camcorders	United States	6,142,176	7,973,866	0.77	303879
Store	Deluxe	Computers	United States	2,930,709	3,916,379	0.75	190360
Store	Regular	Computers	(ALL)-	15,614,972	15,640,049	1	35960
Online	(ALL)-	(ALL)-	(ALL)-	22,750,264	23,228,814	0.98	19326
Online	Regular	Cameras and camcorders	(ALL)-	4,827,104	5,328,802	0.91	11733
(All)-	0	(ALL)-	(ALL)-	9,962,562	10,356,363	0.96	0

Figure 16 Informative summarization result table on example data

5.4 ExplanationByIntervention module

The third model for explanation, explanation by intervention developed by Roy and Sucio (2014) for databases, which was detailed in section 3.4 is implemented in the module *ExplanationByIntervention*. The main class of the module is *Model*, which is initialized by the variables *df, dimensions, direction, actual,* and *reference*. All of these variables can be gained from the module Data.

The main method of the class is *run()*, which holds the recursion logic of the model. By using itertools the method iterates over all possible combinations of each dimensions with the following code:

```
for d in range(0,self.dimensions):
    combinations = itertools.combinations(dimens,d+1)
    for c in combinations:
        l = list(c[0:d+1])
        temp = Dimension.groupDimensions(self.df, l)
        temp = self.computeStatistics(temp)
        output = output.append(temp)
```

The code runs through all dimensions, and all possible combinations of values in those dimensions with the above code.

The measure of impact, as shown in Equations 3.15 (aggravate) and 3.16 (intervention) are computed with the method *computeStatistics()*. The below code is an abbreviated version, shown only for the case of *direction = 'Low'*. See appendix C4 for the full code.

```
df['Q'] = df[self.actual]/df[self.reference]
Qratio = df[self.actual].sum()/df[self.reference].sum()
#calculate aggravation:
df['Aggravate'] = df['Q']*(-1)
#calculate intervention:
df['Intervention'] = Qratio - (Qratio-(df[self.actual].sum()-
df[self.actual])/(df[self.reference].sum()-df[self.reference]))
```

The measures are computed with the help of 'Q' which is the ratio of each row, and *Qratio*, which is the parent ratio of the DataFrame.

The minimal append, as detailed in section 3.4 is computed by creating an empty dataframe *output*, and appending it with the highest-ranking row, as long as the row is not dominated by the earlier rows existing in the *output*. The computation for this is, that it goes through the dimensional values of a row, and if the value is not (ALL), and it exists in the *output* DataFrame already or is dominated by an (ALL) clause, a score *append* is increased by 1. If this score is equal or higher than the number of dimensions, the row is dominated and is not appended into the *output* DataFrame. The code for this is presented below:

```
for i in range(df.shape[0]):
  #If dominated by previous appends, then do not append.
  append = 0
  for d in dimens:
        if not df[d].iat[i] == '(ALL)':
              If df[d].iat[i] in output[d].values:
                    append = append +1
              if '(ALL)' in output[d].values:
                    append = append +1
        else:
              append = append +1
  if append < len(dimens):
        output = output.append(df.iloc[i],ignore_index = True)
```

Similar to the informative summarization, the *Model* for *ExplanationByIntervention* has a methot *getAnswer(top)* which returns a presentable table of results. The variable *top* stands for the number of rows the resulting table will have. As in the *InformativeSummarization* module, the results are rounded, and thousand separators are applied. By default, the table is ranked by Intervention, which is the better measure of impact, according to Roy and Sucio (2014). Figure 17 shows an example of this answer table, created on the example data. Similar to the IS model's table, the topmost explanation is the one with the highest impact. According to the table, United States with all channels, classes, and product categories is the highest explanation. From there the explanations go down in the rankings. To note is, that since this model uses ratio of change (intervention) in the rankings, smaller explanations such as {Store, Music Movies and Audio Books, Malta} are shown high in the rankings.

ChannelName	ClassName	ProductCategory	RegionCountryName	GrossProfit2013	GrossProfit2012	Aggravate	Intervention
(ALL)	(ALL)	(ALL)	United States	63,087,723	71,240,153	-0.89	1.04
Online	Regular	(ALL)	China	6,188,763	6,712,418	-0.92	0.95
(ALL)	Economy	Audio	Spain	160	257	-0.62	0.95
Store	(ALL)	Music Movies and Audio Books	Malta	268	359	-0.75	0.95
(ALL)	Deluxe	Cell phones	Poland	225	312	-0.72	0.95
(ALL)	(ALL)	TV and Video	Ireland	15,484	13,673	-1.13	0.95
(ALL)	(ALL)	Cameras and camcorders	Denmark	18,597	16,254	-1.14	0.95
(ALL)	(ALL)	Computers	Slovenia	49,654	41,590	-1.19	0.95

Figure 17 Result table from ExplanationByIntervention module

6 DESIGN VALIDATION

This chapter has the purpose of validating and evaluating the artifact shown in chapters 4 and 5. First, the methodology for the validation is presented and then data collection method is shown. Lastly, the results of the validation are presented theme by theme.

6.1 Methodology

The chosen methods for this research are design science research and qualitative research. Design research science gives the overview and direction for the artifact creation, while qualitative research is used for the last part of design science; evaluation of the artifact. The design research science is detailed in section 1.4 and Figure 2, and therefore this section focuses on the methodology of qualitative studies. Furthermore, the chosen method out of many qualitative research types — such as ethnography, grounded theory, and narrative analysis — is case studies. The purpose of case study is to produce detailed and holistic knowledge, based on analysis of multiple empirical sources (Erickson & Kovalainen, 2015).

Qualitative research works well for the purpose of the artifact evaluation in this thesis, because the original problem is a business problem. According to Myers, Kappelman, and Prybutok (1997), when the research focuses on solving managerial problems instead of technological development, qualitative research is important. Therefore, it is sensible to choose qualitative over quantitative methods.

Case study as a qualitative method is chosen, due to its fit for the purposes of artifact evaluation. It is used to answer RQ5:

How to validate and evaluate the models for explanation generation in dashboards?

The case is the comparison of the three models for explanation, through interviews with employed persons on the managerial level. There are always one or more cases in a case study, with the goal of creating new or understanding old theory better through research questions.

Erickson and Kovalainen (2015) describe interviews as a discussion with questions and answers. Additionally, interviews and be held face-to-face, over telephone, or over the internet. In this thesis, all the interviews are held over the internet with the use of Zoom calls. The interviews are then recorded for data collection, transcribed into text, and analysed.

According to Erickson and Kovalainen (2015), semi-structured interviews are often used to answer ‘what’ and ‘how’ questions. In this thesis the interviews are semi-structured, with certain topics pre-determined, but with plenty of leeway to dig deeper into the problem and evaluation of the artifact. This method is chosen to ensure that the business experts can express themselves and give opinions on the artifacts freely, as opposed to structuring questions in order to achieve a certain result.

6.2 Data collection

The interviewees were chosen based on both availability and suitability. In a sense, the population is convenience population as they were the available business experts to interview for this thesis. The interviewee group consists of three business experts, all working in a large multinational company. Each of the interviewees deal with business data in their work environment and periodically make analyses on it. The number of interviewees was chosen to provide sufficient validation for the artifact, while not increasing the number until the answers start repeating each other. Table 9 shows the position and interview time of each interviewee, henceforth the interviewees are referred to as P1, P2, and P3.

Table 9 Interviewee group demographics

Interviewee	Position	Interview time
Person1 (P1)	Key Account Manager	30 min
Person2 (P2)	Key Account Manager	45 min
Person3 (P3)	Director of European Sales	45 min

The interviews were held during a one-week time period and followed the same guidelines: First, background questions are asked and answered. Second, the problem of the thesis is shown manually inside Power Bi environment, and then solved through the three models for explanation. Third, discussion on the three models and their possible uses in a business environment is generated through a variety of questions.

6.3 Interview results

In this section, the results of the interviews are showcased with citations from the interviews. First, general comments about the benefits and uses of the models the group thought about are presented, and then models and their comparison, as well as suggestions for improvement are shown.

Generally, the models were well accepted by the interviewed business experts and each interviewee found something they like about the implementations. All saw the benefits in saving time by using automated diagnosis, in order to streamline problem-solving. Additionally, P1 highlighted the that the models are helpful in combating human errors:

P1: "My conclusions might not actually be fully accurate, if I rely on what I can find by viewing."

Additionally, hidden causes, that were discussed in sub-section 3.2.4 along with Explanation Formalism, were known problems to the group. This is not so much human error as it is the benefits of using algorithmic models and P3 summarised the problem very nicely in words:

P3: "If you have one large product line increasing in sales, and other decreasing, the graphs themselves don't show this – so it is important to then see these kinds of models and figures."

In conclusion, the group were overall positive about the models, although some more than others. P3 offered a statement that these kinds of models are needed in the business environment as starting points for solving real issues and aiding the understanding of the company:

P3: "You need these sorts of top-level soundbites to understand how the business is going, obviously breaking it (explanation) down helps."

6.3.1 Validation of the three models

The explanation formalism module received the most positive feedback, mostly due to its graphical representation in the form of an explanation tree. The explanation tree, when compared to the other two models having tables as their representation, received many positive comments, including P3: "wow". Additionally, both P2 and P3 commented on how the percentages in the tree enable faster assimilation than the arbitrary intervention and error-based rankings in the tables. The group was divided on the usefulness of the counteracting causes in the tree however, as P1 commented them being slightly unnecessary:

P1: "The (counteracting causes) are increasing profits, so they are maybe a bit extra?"

The informative summarisation module, which is detailed in section 5.3, received positive feedback on the amount of information and ability to offer help in analysing. P1 summarised this in the following:

P1: "I think the first one (Informative summarization) gives you a lot of information – useful if you simply want to analyse what is going on, to understand what is going on in sales."

Additionally, P2 highlighted that he liked the idea of the Equation 3.13 including both magnitude and ratio, commenting how the value of the model increases due to the mathematical function.

The explanation by intervention received the most critical feedback, likely due to the fact, that as the example was implemented to business data such as sales and total profits, the absolute value is more important than the ratio of change. P1 and P2 both pointed out the problem, commenting:

P1: "10 percent change in a product line of 10 million matters much more than a 70% change in product line that sells a couple thousand."

P2: "We should look at the absolute value of loss and then who contribute most to the loss and focus on those. The rest are not important in this case."

However, P3 thought that the explanation by intervention table can be useful as well, as long as it is used with the knowledge of how it calculates the intervention and ranks the explanations:

P3: "Yea, I see the model (explanation by intervention) as useful tool for highlighting variance."

When asked to compare the models and which would they prefer to use in a business environment, the explanation formalism received most positive feedback due to the explanation tree. P1 commented on the models from personal point of view:

P1: "I am a more visual person, and I would like the tree. I think it is easy to look through, and gives you and overview, which is pretty good. I get the two tables, but… they may be a bit unclear."

In conclusion, all the models may have their use in problem-solving in business environment. The respondents preferred the equations to lean on the side of absolute difference instead of ratios, as well as having graphical representation.

6.3.2 Improvement suggestions

The interviewee group was asked for suggestions, improvements, and what would they like to change in the models. Most of the suggestions were about the presentation of the results themselves, understandably as the suggestions are coming from a group of end-users.

Everyone in the group raised the issue of having no visualisation for the models of informative summarization and explanation by intervention, with P2 and P3's responses giving concrete suggestions:

> *P2: "My suggestions is to make the results more readable, for the non-IT people, the users. The (float numbers) are not in an easily readable format."*

> *P3: "Having color-coding or graphical representation helps, it gives you that sort of instant representation of what the business is currently doing."*

P2's suggestion on making the float numbers readable have already been implemented to both tables, as detailed in section 5.3 and 5.4 by the *getAnswer()* method. Rounding the numbers and adding thousand separators ensures that even large numbers become readable. Color-coding or graphical representation for the tables would help the assimilation of the information and ensure that the models fit the theme of business dashboards.

Additionally, P3 suggested that the models should be extended with predictive modelling. Stating that most business problems are better solved before they become too large. In conclusion, the largest improvement suggestions were regarding the visualisation of the two tables.

7 DISCUSSION AND CONCLUSION

7.1 Limitations

There are several limitations to this study, as the study was conducted in a proof-of-concept method, the artifact is not a full software that can be evaluated. Furthermore, the artifact was not properly optimised but instead simply a working prototype. It would be important to create a fully optimised version that could be evaluated more completely. The results of this thesis are generalisable to a degree: It offers a path and shows how business dashboards can be extended with explanatory analytics. However, since commercial solutions for dashboards can vary in their execution, the same model that was implemented for Microsoft Power Bi may not work for other platforms.

The population for the artifact evaluation was small and a convenience population. The evaluation should be conducted with greater number and variety of population in order to ensure more robust and trustworthy validation. Moreover, with a greater population there would have been increased points of views that could have given additional insight. The current population belonging to a single group of business experts may have skewed the results due to monotonicity. Then again, the qualitative method for the interviews was used to gain deeper insights on the benefits and critique of the models, something that a larger sample may not have corrected.

7.2 Conclusion

The research question of this thesis is:

How can business dashboards be extended with explanatory analytics to support business decision-making?

In order to answer the research question, several sub-questions were formed. Firstly, to extend a business dashboard, the concept of business dashboards required defining. The sub-question "*What are business dashboards?"* is answered in chapter 2 through a literature review. The conclusion can be summarized as business dashboards being visual and interactive products of business intelligence and analytics, that are used to support decision-making process. Furthermore, a comparison of commercial solutions that can be used for the purposes of this thesis' artifact creation was conducted. Microsoft Power Bi was concluded to be affordable, popular, and suitable dashboard platform to use for this thesis. It was noted that commercial solutions such as Microsoft Power Bi have already

been extending their dashboard platforms with AI- and ML-augmented analytics. However, automated explanatory analytics are missing at least in Power Bi.

The second sub-question: *"What are the models for explanatory analytics?"* is answered in chapter 3. Three models for explanations, namely explanation formalism by Feelders and Daniels (2001) and Caron (2013), informative summarization by Sarawagi (2001), and explanation by intervention by Roy and Sucio (2014) are explained in-depth and compared in section 3.5. The comparison provided several interesting results: Although all three models are similar in many ways, such as using some measure of impact to rank their explanations, the underlying way they go about it differs. The recursion method is different for all three models, since explanation formalism uses top-down, informative summarization bottom-up, and explanation by intervention big-bang to go through all possible explanation candidates. Similarly, the measures of impact differ in their logic. Explanation formalism's measure focuses on absolute difference, while intervention is mainly the ratio of change, and informative summarization considers both magnitude and ratio. These differences and similarities are summarised in Table 8, which can be taken as the concluding answer to the second sub-question.

The third sub-question: *"How can business dashboards be extended with explanatory analytics?"* combines the knowledge of from first sub-question and the second. The answer was shown through UML-modelling in chapter 3, with both use-case and class diagrams that represent the way the artifact is built. Through use-case modelling it was noticed that all three models for explanation require similar inputs from the business dashboard, and therefore a Data module was created. The module cleans the data imported from a business dashboard platform and transforms it into a standardised format that can be used by any of the three models.

The artifact creation is detailed in chapter 5 to provide a partial answer to the fourth sub-question: *"How to implement sub-question 3 in Python?"*. The answer in chapter 5 is partial, because the complete implementation code is not shown in the chapter, but instead can be found in Appendix C. Pandas library is heavily used in all models and the DataFrame object is the most crucial part of the implementation. The visualisation of the explanation tree was completed through Networkx's graph function. Each module for an explanatory model has commonalities; they have a method for calculating the measure of impact and a method for providing the final results in a readable format.

The last sub-question: *"How to validate and evaluate the models for explanation generation in dashboards?"* is answered in chapter 6. For the purpose of evaluation semi-

structured case interviews were held, with the target population of business experts. The focus was on the uses, benefits, and improvements that the different models have and require. In the outcome, the explanation by intervention received critique for the use of ratio of change as a measure, as the business experts considered absolute difference to be more important. Moreover, the lack of visualisation in the two models, informative summarization and explanation by intervention, was a definite con. This follows the literature on business dashboards; visualisation helps with assimilation of information, and therefore the explanation tree aids in the understanding of a problem the fastest. In general, however, the business experts concluded that automated explanation generation would be helpful in their work, to both save time and reduce human errors.

According to the comparison, implementation, and evaluation conducted in this thesis, I believe that the EF model is most suited for software implementation in business dashboards. The reasoning is that it most closely follows the general idea of business dashboards; providing visual aid for analysing and problem-solving. Furthermore, the measure of absolute difference is suited to most business problems, whereas the ratio of change is not as often used. The other models have their uses in business problem-solving, especially if their explanations can be extended with visualisation, but in the current state the EF model has the clearest path for software implementation.

7.3 Future research

This thesis opens many more possibilities for future research. Since the artifact was a prototype version, a clear future research path would be to implement a fully integrated software solution for explanatory analytics. Another possibility is to continue with the comparison of the three models. This thesis focused on the business use of the models; however, the models could be used in other environments than the business environment. Furthermore, a technical comparison on their performances for large and heterogenous data sets is needed.

The three models compared in this thesis were from OLAP and database implementations. Since other fields, such as medical diagnosis, also use automated diagnoses, it stands to reason that there can be more possible models for explanatory analytics that can be derived. Therefore, the comparison can be extended to include more models to increase the robustness of the study.

The chosen dashboard platform for this thesis is Microsoft Power Bi, and although it is extremely popular and widely used platform, it is not the only one. Therefore, I suggest

that other platforms should be researched and seen if the implementation method holds or require different solutions.

The main critique for informative summarization and explanation by intervention was their lack of visualization in their answer tables. A possible research direction would be to implement a suitable visualization for these two models, to enhance the understandability and readability of the results.

REFERENCES

Ain, N., Vaia, G., DeLone, W., & Waheed, M. (2019). Two decades of research on business intelligence system adoption, utilization and success–A systematic literature review. *Decision Support Systems*, Vol. 125

Agarwal, R., & Dhar, V. (2014). Big data, data science, and analytics: The opportunity and challenge for IS research. *Information Systems Research.* Vol. 25(3)

Beshears, J., & Gino, F. (2015). Leaders as Decision Architects, *Harvard Business Review*.

Bose, R. (2009). Advanced analytics: opportunities and challenges. *Industrial Management & Data Systems.* Vol. 109(2)

Božič, K., & Dimovski, V. (2019). Business intelligence and analytics for value creation: The role of absorptive capacity. *International journal of information management*, Vol. 46, 93-103.

Brynjolfsson, E., & McElheran, K. (2016). The rapid adoption of data-driven decision-making. *American Economic Review*, Vol. 106(5), 133-139.

Caron, E., & Daniels, H. (2008). Explanation of exceptional values in multi-dimensional business databases. *European Journal of Operational Research*, Vol. 188(3), 884-897.

Caron, E., & Daniels, H. (2013). Explanatory business analytics in olap. *International Journal of Business Intelligence Research*, Vol. 4(3), 67-82.

Caron, E. (2013). *Explanation of exceptional values in multi-dimensional business databases.* Doctoral dissertation, University of Rotterdam, Rotterdam, the Netherlands.

Chen, H., Chiang, R., & Storey, V. (2012). Business intelligence and analytics: From big data to big impact. *MIS quarterly*. Vol. 36(4), 1165-1188.

Christiansen, E. (2014) The Success Stories You Probably Haven't Heard Of: Qlik. *Orendsunstrartups*. Retrieved 26.6.2020 from https://oresundstartups.com/success-stories-might-missed-qlik/

Davenport, T. (2013). Analytics 3.0. *Harvard business review*, Vol. 91(12), 64-72.

Eckerson, W. W. (2010). *Performance dashboards: measuring, monitoring, and managing your business*. John Wiley & Sons.

Eisenhardt, K., & Zbaracki, M. (1992). Strategic decision making. *Strategic management journal*, Vol.13(2), 17-37.

Eriksson, P., & Kovalainen, A. (2015). *Qualitative methods in business research: A practical guide to social research*. Sage.

Federer, L., & Joubert, D. (2018). Providing library support for interactive scientific and biomedical visualizations with tableau. *Journal of eScience Librarianship*, Vol. 7(1), 2.

Feelders, A., & Daniels, H. (2001). A general model for automated business diagnosis. *European Journal of Operational Research*, Vol. 130(3), 623-637.

Few, S. (2005). Intelligent dashboard design. *Information Management*, Vol. 15(9), 12.

Fiedler, F. (1964). A contingency model of leadership effectiveness. *Advances in experimental social psychology,* Vol. 1, 149-190.

Fink, L., Yogev, N., & Even, A. (2017). Business intelligence and organizational learning: An empirical investigation of value creation processes. *Information & Management,* Vol. 54(1), 38-56.

Gartner (2020). Magic Quadrant for Analytics and Business Intelligence Platforms. *Gartner Group*

GitHub (2020). PYPL PopularitY of Programming Language. *GitHub.com* Retrieved 27.6.2020 from http://pypl.github.io/PYPL.html

Golfarelli, M., Rizzi, S., & Cella, I. (2004). Beyond data warehousing: what's next in business intelligence? *Proceedings of the 7th ACM international workshop on Data warehousing and OLAP*, 1-6.

Hevner, A., March, S., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS quarterly*, Vol. 28(1), 75-105.

IşıK, Ö., Jones, M., & Sidorova, A. (2013). Business intelligence success: The roles of BI capabilities and decision environments. *Information & management*, Vol. 50(1), 13-23.

Joshi, K., Chi, L., Datta, A., & Han, S. (2010). Changing the competitive landscape: Continuous innovation through IT-enabled knowledge capabilities. *Information Systems Research*, Vol. 21(3), 472-495.

Jourdan, Z., Rainer, R., & Marshall, T. (2008). Business intelligence: An analysis of the literature. *Information systems management*, Vol. 25(2), 121-131.

Larson, D., & Chang, V. (2016). A review and future direction of agile, business intelligence, analytics and data science. *International Journal of Information Management*, Vol. 36(5), 700-710.

Lycett, M. (2013) 'Datafication': making sense of (big) data in a complex world. *European Journal of Information Systems,* Vol. 22(4), 381-386.

Magdalena, R., Ruldeviyani, Y., Sensuse, D., & Bernando, C. (2019). Methods to Enhance the Utilization of Business Intelligence Dashboard by Integration of Evaluation and User Testing. *International Conference on Informatics and Computational Sciences (ICICoS)*, 1-6.

Mandinach, E. (2012). A perfect time for data use: Using data-driven decision making to inform practice. *Educational Psychologist*, Vol. 47(2), 71-85.

March, S., & Hevner, A. (2007). Integrated decision support systems: A data warehousing perspective. *Decision support systems*, Vol. 43(3), 1031-1043.

Meuter, M., Ostrom, A., Roundtree, R., & Bitner, M. (2000). Self-service technologies: understanding customer satisfaction with technology-based service encounters. *Journal of marketing*, Vol. 64(3), 50-64.

Myers, B.., Kappelman, L., & Prybutok, V. (1997). A comprehensive model for assessing the quality and productivity of the information systems function: toward a theory for information systems assessment. *Information Resources Management Journal (IRMJ)*, Vol. 10(1), 6-26.

Oliphant, T. (2007). Python for scientific computing. *Computing in Science & Engineering,* Vol. 9(3), 10-20.

Pandas (n.d.) About pandas. Retrieved 27.6.2020 from https://pandas.pydata.org/about/

PowerBi Pricing (n.d.). Retrieved 27.6.2020 from https://powerbi.microsoft.com/en-us/pricing/

PowerBi decomposition tree (2020). Create and view decomposition tree visuals in Power BI. Retrieved 28.6.2020 from https://docs.microsoft.com/en-us/power-bi/visuals/power-bi-visualization-decomposition-tree

Power Bi: quick insights (2020). Types of insights supported by Power BI. Retrieved 24.7.2020 from https://docs.microsoft.com/en-us/power-bi/consumer/end-user-insight-types

Presthus, W., & Canales, C. (2015). Business Intelligence Dashboard Design. A Case Study of a Large Logistics Company. *In Norsk konferanse for organisasjoners bruk at IT*, Vol. 23(1).

Provost, F., & Fawcett, T. (2013). Data science and its relationship to big data and data-driven decision making. *Big data*, Vol. 1(1), 51-59.

Python Software Foundation (n.d.). The Python Tutorial. Retrieved 27.6.2020 from https://docs.python.org/3/tutorial/index.html#tutorial-index

Qlik (2020). Choosing an analytics solution? Choose a 10-time Gartner Magic Quadrant Leader. *Qlik.com* Retrieved 27.6.2020 from https://www.qlik.com/us/gartner-magic-quadrant-business-intelligence

Qlik Pricing (n.d.). Retrieved 27.6.2020 from https://www.qlik.com/us/pricing

Rahman, A., Adamu, Y., & Harun, P. (2017). Review on dashboard application from managerial perspective. *2017 International Conference on Research and Innovation in Information Systems (ICRIIS)*, 1-5.

Rasmussen, N., Bansal, M., & Chen, C. (2009). *Business dashboards: a visual catalog for design and deployment.* John Wiley & Sons.

Reddy, C., Sangam, R., & Rao, B. (2019). A survey on business intelligence tools for marketing, financial, and transportation services. *Smart Intelligent Computing and Applications,* Vol. 105, 495-504.

Rejikumar, G., Aswathy Asokan, A., & Sreedharan, V. (2020). Impact of data-driven decision-making in Lean Six Sigma: an empirical analysis. *Total Quality Management & Business Excellence*, Vol. 31(3-4), 279-296.

Roy, S., & Suciu, D. (2014). A formal approach to finding explanations for database queries. *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 1579-1590.

Sarawagi, S. (2001). iDiff: Informative summarization of differences in multidimensional aggregates. *Data Mining and Knowledge Discovery*, Vol. 5(4), 255-276.

Sauter, V. (2014). *Decision support systems for business intelligence.* John Wiley & Sons.

Sharda, R., Delen, D., Turban, E., Aronson, J., & Liang, T. (2014). *Business intelligence and analytics: System for Decision Support*. Pearson.

Sharma R., Mithas S., & Kankanhalli A. (2014). Transforming decision-making processes: a research agenda for understanding the impact of business analytics on organisations. *European Journal of Information Systems*, Vol. 23(4), 433-441.

Simon, H. A. (1977). *The structure of ill-structured problems. Models of discovery,* 304-325. Springer, Dordrecht.

Soh, C., & Markus, M. (1995). How IT creates business value: a process theory synthesis. *ICIS 1995 Proceedings*, 4.

Spyder (n.d.) Retrieved 27.6.2020 from https://www.spyder-ide.org/

Starcke, K., & Brand, M. (2012). Decision making under stress: a selective review. *Neuroscience & Biobehavioral Reviews*, Vol. 36(4), 1228-1248.

Tableau (2020). Extensibility, *Tableau.com*. Retrieved 27.6.2020 from https://help.tableau.com/current/blueprint/en-us/bp_extensibility.htm

Tableau Pricing (n.d.). Retrieved 27.6.2019 from https://www.tableau.com/pricing/individual

ThoughtSpot (2020). Don't BI. Just Search: ThoughtSpot Brings Google-like User Experience to Business Data. Retrieved 26.6.2020 https://www.thoughtspot.com/don%E2%80%99t-bi-just-search-thoughtspot-brings-google-user-experience-business-data

ThoughtSpot Pricing (n.d.) Retrieved 27.6.2020 from https://www.thoughtspot.com/pricing

Torres, R., Sidorova, A., & Jones, M. C. (2018). Enabling firm performance through business intelligence and analytics: A dynamic capabilities perspective. *Information & Management*, Vol. 55(7), 822-839.

Trieu, V. (2017). Getting value from Business Intelligence systems: A review and research agenda. *Decision Support Systems,* Vol. 93, 111-124.

Turner III, D. (2010). Qualitative interview design: A practical guide for novice investigators. *The qualitative report,* Vol. 15(3), 754.

Vizard, M. (2019) Qlik Sense vs Tableau: 2019 Comparison. *It-businessedge*. Retrieved 26.6.2020 from https://www.itbusinessedge.com/business-intelligence/qlik-sense-vs-tableau.html

Yigitbasioglu, O., & Velcu, O. (2012). A review of dashboards in performance management: Implications for design and research. *International Journal of Accounting Information Systems*, Vol. 13(1), 41-59.

APPENDIX A

Business intelligence value creation models:

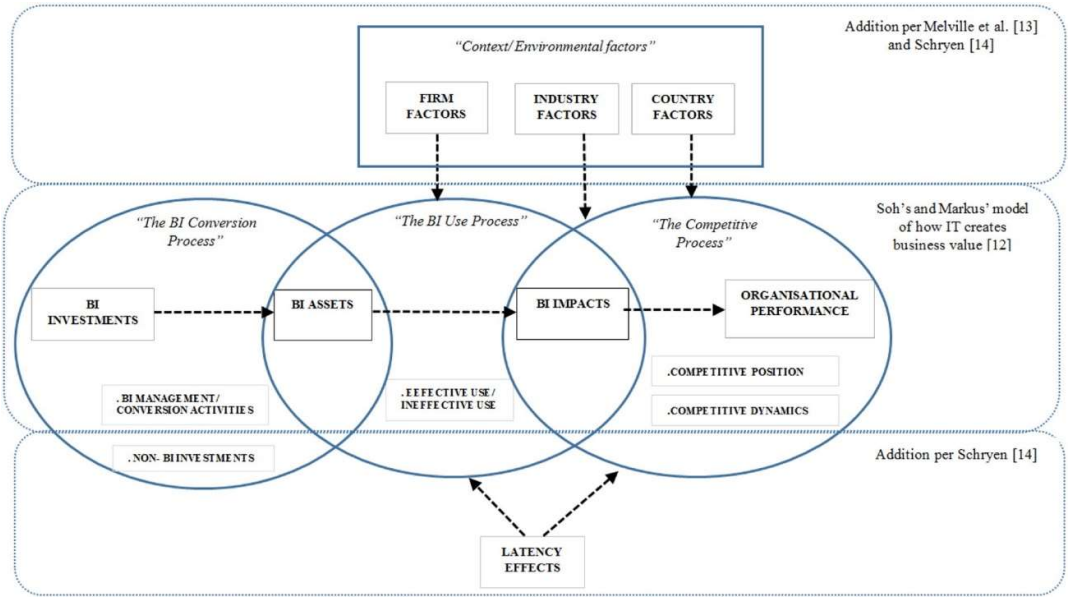

Figure A1 Trieu (2017) Framework of how BI creates business value.

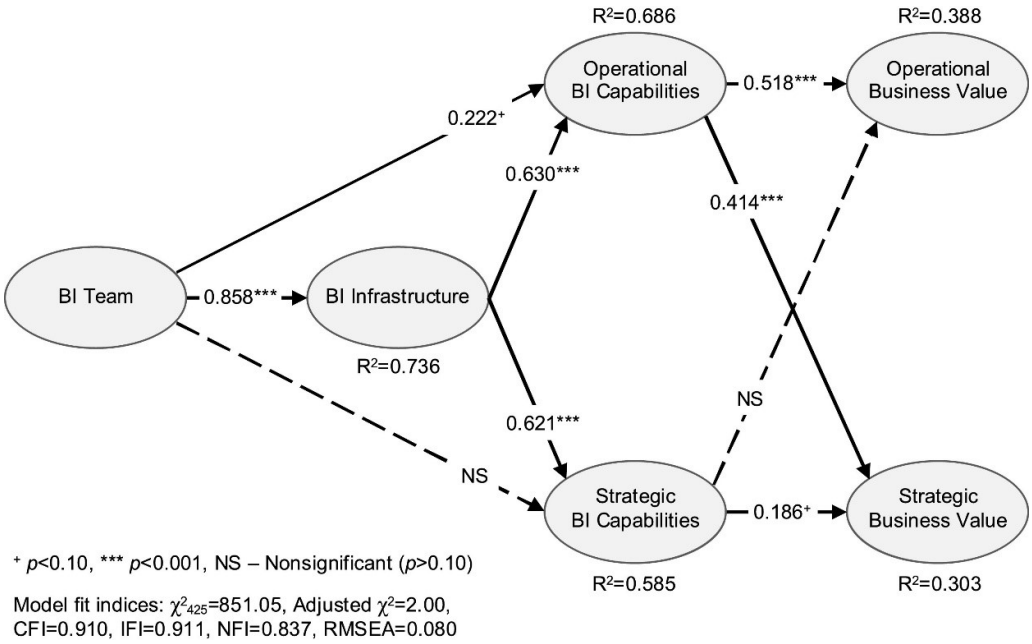

Figure A2 Fink, Yogen, & Even (2017) Standardized solution of structural model.

APPENDIX B

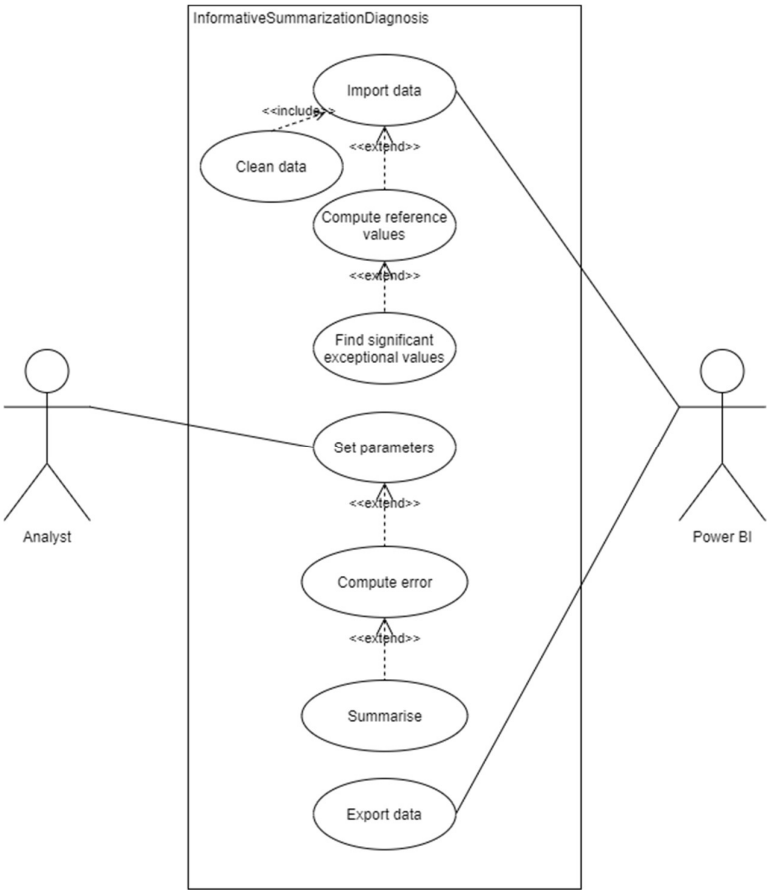

Figure B1 Use case for informative summarization.

Figure B2 Use case for explanation by intervention.

APPENDIX C

C1 Data Module:

```
import pandas as pd

class Data:
    def __init__(self, source, target, refYear, tarYear):
        #initializes class, source = file source, target = target column name, refYear = reference year, tarYear = target year
        self.target = target
        self.refYear = refYear
        self.tarYear = tarYear
        self.source = source
        self.actual = target + tarYear
        self.reference = target + refYear
        self.dfExceptional = pd.DataFrame()
        #run the automatic functions:
        self.df = self.extract(source)
        self.clean()
        self.computeRefData()
    def extract(self,source):
        #extract df from source file, file expected to be csv
        df = pd.read_csv(source)
        df = pd.DataFrame(df)
        df.columns = df.columns.str.replace(' ', '') #remove spaces'
        return df
    def clean(self): #clean the values
        for i in self.df.columns:
            self.df[i] = self.df[i].astype('str')
            self.df[i]   =   self.df[i].replace({'[^A-Za-z0-9.]+': ''}, regex=True)
```

```
            self.df[self.target]       =       self.df[self.tar-
get].astype('float')
        def computeRefData(self):
            #Combines reference and target year values on the col-
umn "target"
            self.df['Year'] = self.df['Year'].astype('str')
            df2013                                                =
self.df[self.df['Year']==self.tarYear].drop('Year', axis = 1).re-
name(columns  =  {self.target:  self.actual}).reset_index(drop  =
True)
            df2012      =      self.df[self.df['Year']==self.re-
fYear].drop('Year',  axis  =  1).rename(columns  =  {self.tar-
get:self.reference}).reset_index(drop = True)
            self.df = df2013
            self.df.insert(len(self.df.columns.values), self.ref-
erence, df2012[self.reference])
            self.df[self.reference]     =     self.df[self.refer-
ence].astype('float64')
            self.df[self.actual]        =        self.df[self.ac-
tual].astype('float64')
        def setInfluence(self, df):
            df  =  Data.computeInfluence(df,self.actual,  self.ref-
erence)
            return df
        def computeInfluence(df, actual, reference):
            #Calculate influence, df = any dataframe,
            #actual = name of column with actual values
            #reference = name of column with reference values
            df['Influence'] = df[actual].astype('float') - df[ref-
erence].astype('float')
            df['InfPerc']               =               df['Influ-
ence'].astype('float')/df[reference].astype('float')
            return df
```

```
        def getExceptionalDf(self, threshold, i, listOfDimen-
sions):
            #threshold = threshold to compute exceptional values
            #i = row of the exceptional value to get
            #listOfDimensions = list of dimensions to include
            df1 = self.df.copy()
            df1 = Dimension.groupDimensions(df1,'MonthOfYear')
            df1     =     Data.computeInfluence(df1,     self.ac-
tual,self.reference)
            exceptions = Exceptions(self, threshold).values
            if len(exceptions) > 0:
                dfException = Dimension.reduceTo(self.df, excep-
tions[i][0], exceptions[i][1])
                print('The exception is: ', exceptions[i])
                dfException = Dimension.groupDimensions(dfExcep-
tion,listOfDimensions)
                return dfException
            else:
                print('No Exceptions found with threshold of
',threshold)
                return pd.DataFrame()
        def getExceptions(self, threshold):
            df = self.df.copy()
            df = Dimension.groupDimensions(df,'MonthOfYear')
            df = Data.computeInfluence(df, self.actual,self.ref-
erence)
            self.dfExceptional = df
            exceptions = Exceptions(self, threshold).values
            return exceptions
        def getMonth(self, month, listOfDimensions):
            #Reduce the dataframe to a certain month
            df = self.df.copy()
            df = Data.computeInfluence(df, self.actual,self.ref-
erence)
```

```
            df = Dimension.reduceTo(df, 'MonthOfYear',month)
            df = df.drop('MonthOfYear', axis = 1)
            return df
    class Exceptions:
        def __init__(self,data, threshold):
            #data = Data class object
            #threshold = some percentage, usually around 0.05-0.25
            #Initialises the class
            self.data = data
            self.threshold = threshold
            self.dfExcp        =        Exceptions.identifyExcep-
tional(self.data, self.threshold)
            self.values = Exceptions.values(self.dfExcp)
        def values(dfExcp):
            #Creates a list of exceptional values
            values = []
            #List holds columnName and columnValue of the excep-
tional values
            for i in range(dfExcp.shape[0]):
                values.append([dfExcp.col-
umns[0],dfExcp.iat[i,0],dfExcp.iat[i,-1].round(2)])
            values.sort()
            return values
        def identifyExceptional(data, threshold):
            #data = data object
            #Get exceptional values
            df = data.dfExceptional
            actual = data.actual
            reference = data.reference
            df  =  df[((df[actual]-df[reference])/df[reference]  >
threshold) | ((df[actual]-df[reference])/df[reference] < -thresh-
old)]
            #Select DataFrame rows on index
            return df
```

```
class Dimension:
    #Get dimensional data from a dataframe
    def groupDimensions(df, dimensions):
        #Input: any df, list of dimensions that stay in output
        #output: dataframe grouped along the wanted dimensions
        df = df.groupby(dimensions, as_index = False).sum()
        return df
    def reduceTo(df, columnName, columnValue):
        #Reduces the dataframe to only hold certain values
        df = df[df[columnName] == columnValue]
        return df
```

C2 Explanation Formalism Module:

```
import pandas as pd
from Data import Dimension
import networkx as nx
import os

class Model:
    def __init__(self, dfExceptional, symptom, T, dimensions,
actual, reference):
        # dfExceptional = Data.dfExceptional
        #symptom = High if actual > reference, Low if actual
< reference
        #T = fraction for parsimonious set of causes, e.g. 0.85
        #dimensions = Number of dimensions
        self.df = dfExceptional
        self.df0 = self.df.copy()
        self.symptom = symptom
        self.T = T
        self.dimensions = dimensions
        self.exceptional = []
        self.actual = actual
```

```
            self.reference = reference
            self.dfContri = pd.DataFrame()
            self.dfCounter = pd.DataFrame()
            self.parsContri = pd.DataFrame()
            self.parsCounter = pd.DataFrame()
            self.columns = self.df.columns.values.tolist()
            self.explanation = []
            #Up until this is initialisation, now need to run the
code
            self.maximalExplanation()
        def maximalExplanation(self):
            #set local variables
            parsContri = pd.DataFrame()
            parsCounter = pd.DataFrame()
            #Group over dimensions
            self.df            =            Dimension.groupDimen-
sions(self.df0,self.columns[0])
            #Compute Contributing and Counteracting sets for first
level
            self.computeContriCounter(self.df)
            #Add to parsimonious sets
            if not self.dfContri.empty:
                parsContri      =      self.computeParsimonious-
Set(self.dfContri, 1,Contri = True)
            if not self.dfCounter.empty:
                parsCounter      =      self.computeParsimonious-
Set(self.dfCounter, 1,Contri = False)
            #Append  explanation  with  the  contributing  set  of
causes
            self.explanation.append([parsContri,1, 'Contri'])
            self.explanation.append([parsCounter,1, 'Counter'])

            #Go over dimensions for maximal explanation
```

```
            if self.dimensions > 1: #Only go over if more than one
dimension
                for dimension in range(1,self.dimensions): #Go
over all dimensions
                    #assign local variables
                    tempContri = pd.DataFrame()
                    tempCounter = pd.DataFrame()
                    for row in range(parsContri.shape[0]): #Get
parsinomious contributing set
                        #Choose only children of row
                        self.df = self.df0.copy()
                        for i in range(dimension):
                            self.df                                =
self.df[self.df[self.df.columns[i]] == parsContri.iat[row,i]]
                        #Group over dimensions
                        self.df      =      Dimension.groupDimen-
sions(self.df,self.columns[0:dimension+1])
                        #Compute Influences for the subset
                        self.df       =       Model.computeInflu-
ences(self.df,self.actual, self.reference, self.symptom)
                        #Compute Contributing and Counteracting
sets
                        self.computeContriCounter(self.df)
                        #Add to parsimonious sets
                        if not (self.dfContri.empty):
                            tempContri      =      tempContri.ap-
pend(self.computeParsimoniousSet(self.dfContri, dimension+1,Con-
tri = True), ignore_index = True)
                        if not (self.dfCounter.empty):
                            tempCounter  =  self.computeParsimoni-
ousSet(self.dfCounter, dimension+1,Contri = False)
                    parsContri = tempContri.copy()
                    #Append explanation with contributing set:
```

```
                    self.explanation.append([tempContri,   dimen-
sion+1,'Contri'])
                    self.explanation.append([tempCounter,dimen-
sion+1,'Counter'])
        def computeContriCounter(self, df):
            #Divide a DataFrame to contributing and counteracting
causes
            if self.symptom == 'Low':
                self.dfContri = df[df['Influence']<0].copy()
                self.dfCounter = df[df['Influence']>0].copy()
            if self.symptom == 'High':
                self.dfContri = df[df['Influence']>0].copy()
                self.dfCounter = df[df['Influence']<0].copy()
            self.dfContri  =  Model.computeInfluences(self.dfCon-
tri, self.actual, self.reference, self.symptom)
            self.dfCounter          =          Model.computeInflu-
ences(self.dfCounter, self.actual, self.reference, self.symptom)
        def computeInfluences(df, actual, reference, symptom):
            #Calculate Influence and Influence percentage
            df['Influence'] = df[actual]-df[reference]
            df['InfPerc'] = df['Influence']/df['Influence'].sum()
            if symptom == 'Low':
                if df['Influence'].sum() >0:
                    df['InfPerc'] = df['InfPerc']*(-1)
            else:
                if df['Influence'].sum() <0:
                    df['InfPerc'] = df['InfPerc']*(-1)
            return df
        def computeParsimoniousSet(self, df, dimensions, Contri =
True):
            #returns pars set with total InfPerc > T
            #Contri  =  True  if  input  df  is  contributing  causes,
otherwise False
            localT = 0
```

```
            pars = pd.DataFrame()
            if Contri == True:
                while localT <=self.T:
                    pars                =                pars.ap-
pend(df.loc[[df['InfPerc'].idxmax()]], ignore_index = True)
                    localT          =          localT          +
df.at[df['InfPerc'].idxmax(),'InfPerc']
                    df = df.drop(df['InfPerc'].idxmax(), axis = 0)
                #check for hidden causes
                TotalInf = df['Influence'].sum()
                TMin = pars.iat[pars['InfPerc'].idxmax(),-1]
                lookahead = self.lookahead(df,TotalInf, TMin, di-
mensions)
                if not lookahead.empty:
                    pars = pars.append(lookahead)
            else:
                T = self.T*(-1)
                while localT >=T:
                    pars                =                pars.ap-
pend(df.loc[[df['InfPerc'].idxmax()]], ignore_index = True)
                    localT          =          localT          +
df.at[df['InfPerc'].idxmax(),'InfPerc']
                    df = df.drop(df['InfPerc'].idxmax(), axis = 0)
            #Use lookahead to ensure parsimonious set of causes:

            return pars
        def lookahead(self,df,TotalInf, TMin, dimension):
            #lookahead method looks ahead for contributing parsi-
monious causes, with an input of insignificant causes
            #df = insignificant causes
            #TotalInf = total influence of parent dimension
            #TMin = Smallest Influence Percentage of parent dimen-
sion
            lookahead = pd.DataFrame()
```

```
            dfChildren = pd.DataFrame()
            for i in range(df.shape[0]): #Create new df with next
dimension in it
                dfChildren            =            dfChildren.ap-
pend(self.df0[self.df0[self.columns[dimension-1]]             ==
df.iloc[i,dimension-1]])
                dfChildren   =   Dimension.groupDimensions(dfChil-
dren, self.columns[0:dimension+1])
            if not dfChildren.empty:
                dfChildren  =  Model.computeInfluences(dfChildren,
self.actual, self.reference, self.symptom)
                dfChildren['InfPerc'] = df['Influence']/TotalInf
                if self.symptom == 'Low':
                    dfChildren   =   dfChildren[dfChildren['Influ-
ence']<0]
                if self.symptom == 'High':
                    dfChildren   =   dfChildren[dfChildren['Influ-
ence']>0]
                for i in range(dfChildren.shape[0]):
                    if dfChildren.iat[i,-1] > TMin:
                        lookahead.append(dfChildren.iloc[i,:])
            return lookahead
    class ExplanationTree:
        def __init__(self, symptom,explanation, savename, Counter
= True):
            #Symptom = String which describes the original symptom
            #Explanation = List of lists with dataframes for ex-
planations, from Class ExplanationFormalism
            self.symptom = symptom
            self.explanation = explanation
            self.savename = savename
            self.Counter = Counter
            self.create(self.explanation, Counter)
        def create(self, explanation, Counter = True):
```

```
            #Creates explanation tree formed from a list of expla-
nations
            G = nx.Graph()
            G.add_node(0, label = self.symptom)
            if Counter == True:
                for i in range(len(explanation)):
                    #Iterates over the list of explanations, add-
ing nodes and edges for both contributing and counteracting par-
simonious causes
                    if explanation[i][2] == 'Contri':
                        temp = explanation[i][0]
                        temp = temp.sort_values('InfPerc')
                        for j in range(temp.shape[0]):
                            G.add_node(temp.iloc[j,:explana-
tion[i][1]].sum(), label = temp.iat[j,explanation[i][1]-1])
                            G.add_edge(temp.iloc[j,:explana-
tion[i][1]-1].sum(),temp.iloc[j,:explanation[i][1]].sum(),  label
= str((temp.iat[j,-1]*100).round(0))+'%')
                    elif explanation[i][2] == 'Counter':
                        temp = explanation[i][0]
                        if temp.shape[0] >0:
                            temp = temp.sort_values('InfPerc')
                            #Only  add  up  to  3  counteracting
causes, to prune the tree
                            if temp.shape[0] <=3:
                                for j in range(temp.shape[0]):
                                    G.add_node(temp.iloc[j,:ex-
planation[i][1]].sum(), label = temp.iat[j,explanation[i][1]-1])
                                    G.add_edge(temp.iloc[j,:ex-
planation[i][1]-1].sum(),temp.iloc[j,:explanation[i][1]].sum(),
label = str((temp.iat[j,-1]*100).round(0))+'%', style = 'dashed')
                            else:
                                for j in range(3):
```

```
                                    G.add_node(temp.iloc[j,:ex-
planation[i][1]].sum(), label = temp.iat[j,explanation[i][1]-1])
                                    G.add_edge(temp.iloc[j,:ex-
planation[i][1]-1].sum(),temp.iloc[j,:explanation[i][1]].sum(),
label = str((temp.iat[j,-1]*100).round(0))+'%', style = 'dashed')
            else:
                for i in range(len(explanation)):
                        #Iterates over the list of explanations,
adding nodes and edges for both contributing and counteracting
parsimonious causes
                        if explanation[i][2] == 'Contri':
                            temp = explanation[i][0]
                            temp = temp.sort_values('InfPerc')
                            for j in range(temp.shape[0]):
                                G.add_node(temp.iloc[j,:explana-
tion[i][1]].sum(), label = temp.iat[j,explanation[i][1]-1])
                                G.add_edge(temp.iloc[j,:explana-
tion[i][1]-1].sum(),temp.iloc[j,:explanation[i][1]].sum(), label
= str((temp.iat[j,-1]*100).round(0))+'%')
            self.G = G
            self.p = nx.drawing.nx_pydot.to_pydot(G)
        def save(self):
            #Saves the explanation tree
            self.p.write_png(self.savename+'.png')
        def show(self):
            self.save()
            #Shows the explanation tree
            os.system(self.savename+'.png')
```

C3 Informative Summarization Module:

```
import pandas as pd
import numpy as np
```

```
   class Model:
       def __init__(self, df, e_max, direction, dimensions, ac-
tual, reference, onesided = True):
           #df = DataFrame to summarise
           #e_max = maximum error allowed by user
           #dir = 'Low' or 'High', depending if user believes the
actual values are lower or higher than normal
           #dimensions = number of dimensions to summarise,
starts at 0 when one dimension
           #actual = name of column with actual values
           #reference = name of column with reference values
           #Set variables:
           self.onesided = onesided
           self.actual = actual
           self.reference = reference
           self.final = df.copy()
           if direction == 'Low':
               self.dir = -1
           if direction == 'High':
               self.dir = 1
           self.df = df
           self.columns = []
           self.e_max = e_max
           self.i = dimensions
           #run the class
           self.summarise()
           self.df = self.df[self.columns]
       def summarise(self):
           #summarises and iterates over the dataframe
           self.df = self.computeStatistics(self.df)
           self.columns = list(self.df.columns.values)
           #Iteration:
           for j in range(self.i,0, -1):
              self.df = self.group(j-1)
```

```
            self.df = self.df.fillna('(ALL)-')
        def group(self, i):
            # i = level of dimensions, where i = 0 : d = 1
            #Groups together the ones <=e_max
            output = self.computeStatistics(self.df)
            togroup = output[output['Error']<=self.e_max]
            nogroup = output[output['Error']>self.e_max]
            #print(togroup.iloc[48:53,2:7])
            answer = pd.DataFrame()
            if not(togroup.empty):
                answer = self.sumGroup(togroup.iloc[:,:-2],i)
                answer = self.computeStatistics(answer)
                answer  =  answer.append(nogroup,  ignore_index  =
True)
            return answer
        def sumGroup(self,subGroup, i):
            #i = number of dimensions
            #Groups together a subgroup on a certain dimension
            if i == 0:
                group  =  pd.DataFrame(columns  =  subGroup.col-
umns.values)
                group.loc[0] = subGroup.iloc[0]
                group.iat[0,i] = '(All)-'
                group.iloc[0,-2:]      =      [subGroup.iloc[:,-
2].sum(),subGroup.iloc[:,-1].sum()]

            else:
                columns = list(subGroup.columns.values)
                c = []
                for j in range(i):
                    c.append(columns[j])
                subGroup = self.computeStatistics(subGroup)
                group = subGroup.groupby(c,as_index = False).sum()
            return group
```

```
        def computeStatistics(self,df):
            #calculates and adds columns Ratio and ErrorRatio
            va = df[self.reference].astype('float')
            vb  = df[self.actual].astype('float')
            #Ratio:
            df.loc[:,'Ratio'] = (vb/va).astype('float')
            ratio = (np.sum(vb)/np.sum(va)).astype('float')
            #Error:
            df.loc[:,'Error']   =   ((vb-ratio*va)*np.log(vb/(ra-
tio*va))).astype('float')
            #one-sided error correction, change to 0 if opposite
dir:
            if self.onesided == True:
                if self.dir < 0:
                    df.loc[(df[self.actual]-df[self.refer-
ence]>0), 'Error'] = 0
                elif self.dir > 0:
                    df.loc[(df[self.actual]-df[self.refer-
ence]<0), 'Error'] = 0
            return df
        def getAnswerDf(self):
            #Pruning, rounding, and ranking the answer to a read-
able format
            df = self.df.copy()
            df['Ratio'] = df['Ratio'].round(2)
            df[self.actual]             =             df[self.ac-
tual].round(0).astype('int')
            df[self.reference]          =          df[self.refer-
ence].round(0).astype('int')
            df['Error'] = df['Error'].round(0).astype('int')
            df   =   df.sort_values(by   =   'Error',   ascending   =
False).reset_index(drop = True)
            df[self.actual]             =             df[self.ac-
tual].round(0).astype('int').apply(lambda x : "{:,}".format(x))
```

```
            df[self.reference]          =          df[self.refer-
ence].round(0).astype('int').apply(lambda x : "{:,}".format(x))
            return df
```

C4 ExplanationByIntervention Module:

```
from Data import Dimension
import pandas as pd
import itertools

class Model:
    def __init__(self, df, dimensions, direction, actual, ref-
erence):
            #df = DataFrame from object Data
            #dimensions = nr of dimensions in dataframe
            #direction = 'Low' or 'High' depending if user thinks
actual is lower or higher than reference
            #actual = name of column with actual values
            #reference = name of column with reference values
            #set variables:
            self.df = df
            self.dimensions = dimensions
            self.direction = direction
            self.actual = actual
            self.reference = reference
            self.topI = self.df.copy()
            #automatically run the class:
            self.topI = self.run()
            self.topI = self.minimalAppend(self.topI)
        def run(self):
            #Create table with all candidate explanations
            #set local variables:
            columns = list(self.df.columns.values)
            columns.extend(['Aggravate','Intervention'])
```

```
            dimens = columns[0:self.dimensions]
            output = pd.DataFrame()
            #Iterate over dimensions and append output with can-
didate explanations
            for d in range(0,self.dimensions):
                combinations = itertools.combinations(dimens,d+1)
                for c in combinations:
                    l = list(c[0:d+1])
                    temp = Dimension.groupDimensions(self.df, l)
                    temp = self.computeStatistics(temp)
                    output = output.append(temp)
            #Change order of columns to match the original
            output = output[columns[0:len(output.columns)]]
            output = output.sort_values('Intervention', ascending
=False).reset_index(drop = True)
            output = output.fillna('(ALL)')
            return output
        def computeStatistics(self, df):
            df['Q'] = df[self.actual]/df[self.reference]
            Qratio     =     df[self.actual].sum()/df[self.refer-
ence].sum()
            if self.direction == 'Low':
                #calculate aggravation:
                df['Aggravate'] = df['Q']*(-1)
                #calculate intervention:
                df['Intervention'] = Qratio - (Qratio-(df[self.ac-
tual].sum()-df[self.actual])/(df[self.reference].sum()-
df[self.reference]))
            elif self.direction == 'High':
                #calculate aggravation:
                df['Aggravate'] = df['Q']
                #calculate intervention:
```

```
                df['Intervention']  =  (-1)*(Qratio  -  (Qratio-
(df[self.actual].sum()-df[self.actual])/(df[self.refer-
ence].sum()-df[self.reference])))
            return df
        def minimalAppend(self, df):#Remove redundant explanations
            #df = top explanations by intervention
            #set local variables:
            output = pd.DataFrame(columns = df.columns.values)
            dimens = list(df.columns[0:self.dimensions])
            for i in range(df.shape[0]):
                #If dominated by previous appends, then do not
append.
                #Dominated = the same combination of dimensions
((ALL) can substitute anything) is already found
                append = 0
                for d in dimens:
                    if not df[d].iat[i] == '(ALL)':
                        if df[d].iat[i] in output[d].values:
                            append = append +1
                        if '(ALL)' in output[d].values:
                            append = append +1
                    else:
                        append = append +1
                if append < len(dimens):
                    output = output.append(df.iloc[i], ignore_in-
dex = True)
            return output
        def getAnswer(self, top):
            #top = number of rows
            df = self.topI.copy()
            #Round all numerals to 2 decimals:
            numerals = list(df.dtypes[df.dtypes == 'float64'].in-
dex)
            for c in numerals:
```

```
                df[c] = df[c].round(2)
            #Limit the answer to top:
            if df.shape[0] > top:
                df = df.loc[0:top-1]
            df[self.actual]             =             df[self.ac-
tual].round(0).astype('int').apply(lambda x : "{:,}".format(x))
            df[self.reference]          =          df[self.refer-
ence].round(0).astype('int').apply(lambda x : "{:,}".format(x))
            return df
```

C5 InterviewExample runscript:

```
   import Data
   import ExplanationFormalism as EF
   import InformativeSummarization as IS
   import ExplanationByIntervention as EBI



   #Local variables:
   filename = 'InterviewData.csv'
   #Dimensions = number of dimensions
   dimensions = 4
   #Direction = whether the actual value is Lower or Higher than
the reference
   direction = 'Low'



   #Clean and transform the data:
   #Data object: GrossProfit = Target to analyse, 2012 = reference
year and 2013 = target year
   asd = Data.Data(filename,'GrossProfit','2012', '2013')
   dfException                =                Data.Dimension.re-
duceTo(asd.df,'MonthOfYear','12').drop('MonthOfYear', axis = 1)
   #set actual and reference columns:
```

```
    actual = asd.actual
    reference = asd.reference


    """ Informative Summarization:
    Summarises starting from the most detailed dimension, computes
error rate
    Ranks based on both Magnitude and Ratio
    """
    #Model variable:
    e_max = 10000
    #Fit the model
    Informative  =  IS.Model(dfException.copy(),e_max,  direction,
dimensions, actual,reference)
    #Get the answer:
    InformativeAnswer = Informative.getAnswerDf()


    """ Explanation By Intervention:
    Creates a table with All possible explanations and computes
    Ranks mostly based on Ratio of difference
    """
    #Fit the model
    EBIModel  =  EBI.Model(dfException.copy(),  dimensions,  direc-
tion, actual, reference)
    #Get the answer:
    EBIanswer = EBIModel.getAnswer(8)


    """ Explanation Formalism script:
    Starts from least detailed dimension, moves on to more detailed
based on influence measure
    Ranks mostly based on Absolute difference
    """
```

```
    dfExceptionEF = Data.Data.setInfluence(asd, dfException)
    #Fit the model
    T = 0.8
    Model1                                                          =
EF.Model(dfExceptionEF.copy(),direction,T,dimensions,    actual,
reference)
    expl = Model1.explanation
    #Create the visualisation
    explanationtree = EF.ExplanationTree('24m decrease', expl,'ex-
planationtree')
    explanationTreeContri    =    EF.ExplanationTree('24m    de-
crease',expl,'explanationtreeConrtri',Counter = False)

    """ the visualisation:"""
    explanationtree.show()
    explanationTreeContri.show()
```