João Marques Paredes Mouco

Licenciatura em Ciências da Engenharia Electrotécnica e de Computadores

Insurance Fraud Detection - Using Complex Networks to Detect Suspicious Entity Relationships

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador: João Paulo Branquinho Pimentão, Professor Auxiliar, Universidade Nova de Lisboa

Co-orientador: Pedro Alexandre da Costa Sousa, Professor Auxiliar, Universidade Nova de Lisboa

Júri

Presidente: Rodolfo Alexandre Duarte Oliveira, Professor Auxiliar, FCT - UNL
Arguente: João Almeida das Rosas, Professor Auxiliar, FCT - UNL
Vogal: Pedro Alexandre da Costa Sousa, Professor Auxiliar, FCT - UNL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2019

Insurance Fraud Detection - Using Complex Networks to Detect Suspicious Entity Relationships

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

À minha familia, e à minha namorada.

Acknowledgements

I would like to start by thanking my two advisors on this journey, Professor João Paulo Pimentão and Professor Pedro Sousa. It was thanks to them that I found my passion for data science and everything I know today is because of them. I would also like to thank them for providing me the facilities of Holos for me to work on. It was a great work environment full of great people. On this note, I would also like to thank Professor Sérgio Onofre for always being in a good mood and for reminding me that my Mackintosh is not good at least 27 times a day, and also João Lisboa, for being one of the most knowledgeable people I ever met. With him in the room, no day was ever a sad day.

I want to thank my parents and grandparents for all the support they could ever give me. They supported me every step of my journey since coming to the university, I could not be here without them. I truly believe I lived up to their expectations and I hope they have pride in me. Thank you so much. I would also like to thank my godmother for opening up her home to me in these past six years and putting up with me. I hope I was a good "son" to you. Thank you for everything.

I want to thank all my friend, the ones that came with the university but also the ones that I left back home in Santarém. I want like to give my special thanks to Diogo, Pedro, Francisco, Nelson and Zuka for letting me "leech" information from them (I wont go in detail here), to Jardas and Filipa for those all-nighters we pulled of during the writing of our dissertations, to Gaita and Kiko for being with me from day one in this journey, and to Kobaim and Patchon for not letting me forget where I come from and being there for me when I needed the most. I would like to give my special thanks to Arsénio for being with my in this journey across the vast world of data science. Really really thank you.

Finally, I would like to thank Joana, my girlfriend. Without her I would not the person I am today. She made me grow in every sense of the word. Thank you for you unconditional support, courage and love. You were there for me in the most difficult times and for that, I truly thank you from the bottom of my heart.

"You can never cross the ocean until you have the courage
to lose sight of the shore"
Christopher Columbus

Abstract

The insurance industry is undoubtedly one of the main drivers of today's economy and certainly will be so for the foreseeable future. This is derived from a basic need every person has, and that is to have financial and personal safety.

Despite all this, some entities try to induce fraud on their policies or to circumvent some legal mechanics, to gain unlawful benefits and advantages. This being said, insurance fraud constitutes a grave downside to insurance companies as it directly translates to a loss of economical assets as well as the opportunity to establish a precursor to further exploitation of the system in place.

In this context, this dissertation proposes a framework to help detect the most common types of insurance fraud and scam. Most of the times, insurance scams are usually detected after they took place, this means the companies are already at a loss when they detect it. This framework, which is based upon complex networks for relationship visualization, takes into consideration the relationships already in place between the different entities of the insurance hub-world, advises the responsible entities for fraud prosecution on suspicious relationships. This way, frauds and scams can be detected early on, thus minimizing the losses associated.

This dissertation is being supported by at Holos, S.A using the online insurance management tool, also known as RIFT. This tool gathers data from actual insurance companies, giving the study a higher degree of veracity and applicability in the real world.

Keywords: Insurance fraud, Machine learning, Knowledge discovery, Complex networks

Resumo

A indústria dos seguros é, sem dúvida, uma das principais impulsionadoras da economia atual e certamente o será para um futuro próximo. Isto é derivado de uma necessidade básica que todas as pessoas têm, e isso é ter segurança financeira. As apólices de seguro são uma das ferramentas mais importantes para o controlo de danos que uma pessoa ou entidade pode ter em sua posse.

Apesar de tudo isto, algumas entidades tentam induzir fraudes nas suas apólices, para obter benefícios e vantagens não contempladas pela lei. Dito isto, a fraude em seguros constitui uma grave desvantagem para as companhias de seguro, uma vez que se traduz diretamente numa grande perda de ativos económicos, bem como a oportunidade de estabelecer um precursor para uma maior exploração do sistema.

Neste contexto, esta dissertação propõe uma estrutura para ajudar a detetar os tipos mais comuns de fraude. Na maioria dos casos, fraudes de seguros são detetadas depois de ocorrerem, o que significa que as empresas já estão em perda quando detetam a fraude. Esta estrutura, que é baseada em redes complexas para visualização de relações, toma em consideração as relações já existentes entre as diferentes entidades do mundo dos seguros e aconselha as entidades responsáveis pela regularização e auditoria em seguros, sobre relações suspeitas entre entidades. Desta forma, as fraudes podem ser detetadas precocemente, minimizando as perdas associadas.

Esta dissertação está a ser realizada em cooperação com Holos, S.A usando a sua ferramenta de gestão de seguros on-line, também conhecida como RIFT. Esta ferramenta regista dados pertencentes a seguradoras, conferindo ao estudo maior grau de veracidade e aplicabilidade no mundo real.

Palavras-chave: Fraude de seguros, aprendizagem automática, Redes complexas

Contents

List of Figures xix

List of Tables xxiii

Listings xxv

Acronyms xxvii

1 Introduction 1

1.1 Context and Motivation . . . 1
1.2 The Problem . . . 3
1.3 Proposed Solution . . . 4
1.4 Contributions . . . 4
1.5 Outline of the Document . . . 5

2 State of the Art 7

2.1 Conventional methodologies for fraud detection . . . 7
2.1.1 Prevention and detection - Insurance Company level . . . 8
2.1.2 Anti-fraud alliances . . . 10
2.2 Machine Learning Methods . . . 11
2.2.1 Data Preparation . . . 11
2.2.1.1 Extract, Transform and Load . . . 11
2.2.2 Data Mining Methodologies . . . 14
2.2.2.1 Cross Industry Standard Process for Data Mining . . . 14
2.2.2.2 Sample, Explore, Modify, Model and Assess . . . 17
2.2.2.3 Knowledge Discovery in Databases . . . 18
2.2.2.4 Overview on CRISP-DM, SEMMA and KDD . . . 20
2.2.3 Machine Learning Algorithms . . . 20
2.2.3.1 Clustering . . . 20
2.2.3.2 Artificial Neural Networks . . . 22
2.2.3.3 Random Forest . . . 25

2.2.4 Overview on Machine Learning Algorithms 27

3 Complex Networks **31**
3.1 Complex Systems 32
3.2 Network Theory 36
3.3 Graph Theory 38
3.3.1 Types of Graphs 39
3.4 Network Scale Measures 40
3.5 Structural vs. Functional Networks 41
3.6 Network Metrics 42

4 Data Engineering **45**
4.1 Data Source 45
4.2 Desired Structure 46
4.3 Used Frameworks 47
4.3.1 Python 47
4.3.2 SQLDeveloper 50
4.3.3 Pentaho Data Integration 51
4.3.3.1 Transformation - Edges File 54
4.3.3.2 Transformation - Nodes File 55
4.3.3.3 Wrapping Up with Jobs 56
4.3.4 Comparison between Frameworks 56
4.3.4.1 Complexity Comparison 57
4.4 Data File Structure and Analysis 58
4.4.1 File Dissection - *Edges.csv* 59
4.4.2 File Dissection - *Nodes.csv* 59

5 Complex Network Framework and Result Analysis **61**
5.1 Frameworks 61
5.1.1 Networkx 62
5.1.2 Gephi 64
5.1.3 Cytoscape 66
5.1.4 Overview on the Frameworks Used 68
5.2 Results and Discussion 69
5.2.1 First Stratification (Node Degree = 1) 70
5.2.2 Second Stratification (Node Degree = 2) 71
5.2.3 Third Stratification (Node Degree = 3) 73
5.2.4 Fourth Stratification (Node Degree = 4) 74
5.2.5 Fifth Stratification (Node Degree = 5) 77

5.2.6 Sixth Stratification (Node Degree = 6) 79
5.2.7 Seventh Stratification (Node Degree = [7;8]) 80
5.2.8 Eighth Stratification (Node Degree = [9;10]) 82
5.2.9 Final Stratification (Node Degree > 10) 83
5.2.9.1 High Risk of Fraudulent Practices 86
5.3 Final Thoughts on the Insurance Network Analysis 90

6 Conclusions and Future Work **93**
6.1 Conclusions 93
6.2 Future Work 98

Bibliography **101**

I Structural Analysis of Data Input Sources **109**
I.1 CSV File Structure 109
I.2 Relational Database 110

II Python Code for ETL Purposes **115**

III Pentaho Data Integration Transformation and Job Files **117**

IV Cytoscape Image Gallery **121**

List of Figures

2.1 Survey performed upon Australian organizations identifying basic fraud control and prevention techniques by *PricewaterhouseCoopers Economic Crime Survey 2007* [12] 8
2.2 Descriptive figure depicting the ETL process from left to right. DWH stands for a data storage solution which is not relevant to be addressed. 13
2.3 Descriptive image depicting the ELT process from left to right. 14
2.4 The CRISP-DM reference model. 15
2.5 The Sample, Explore, Modify, Model and Assess (SEMMA) reference model. 17
2.6 Descriptive image depicting the Knowledge Discovery in Databases (KDD) process. 19
2.7 Architectural design of an artificial neuron. 23
2.8 Schematic representation of the back-propagation algorithm. 24
2.9 Representation of a binary decision tree. 26

3.1 Image representing the Königsberg 7 bridge's problem. The objective is to design a route which passes through all seven bridges (green colored) once and only once. 37
3.2 Examples of three types of different graphs 39

4.1 Examples of simple Dataframes (left) and Series (right) 48
4.3 Visual depiction, using Venn Diagrams, of the different kinds of join operations. From left to right; Inner Join 4.2a, Left Join 4.2b, Right Join 4.2c and Full Outer Join 4.2d 49
4.4 Simple transformation environment possessing an input (Table file) and an output (XML file) 52
4.5 Simple job environment possessing a create file operation followed by a transformation. This transformation is then checked for the validity of its output file. 53

4.6 Pentaho Data Integration (PDI) schematic depicting the final transformation used for Extract, Transform and Load (ETL) purposes. It can be viewed further in-depth at III.1 54
4.7 Transformation schematic that originates the *Edges.csv* file. 55
4.8 Transformation schematic that originates the *Nodes.csv* file. 56

5.1 Karate's club network showing all the connections between elements from the club. 62
5.2 Gephi's interface window showing the graph regarding the insurance network. 64
5.3 Force Atlas algorithm applied to the Quaker Network data. 65
5.4 Properties menu for the edges of the network. 66
5.5 Web interface exported from Cytoscape. 68
5.6 From left to right, a case where every relationship is deemed normal, two abnormal connections and a strange relationship 71
5.7 Relationship between entities showing signs of polygamy. 72
5.8 Three companies relating to the same employee as contact entity. . . . 73
5.9 From left to right, three companies which have the same entity labeled as *Individual* as contact entity, one company with three individuals as contact entities 74
5.10 On the left figure, a case where the nodes present a layout which is to be expect whilst on the right, the layout has an abnormal structure. . 75
5.11 Relationships depicting a deviant layout behavior. 76
5.12 Depiction of abnormal relationships between a company and its employees. 76
5.13 Depiction of two similar connections between the same entities 78
5.14 Unusual ownership ring between six different companies 78
5.15 Unusual cluster formed from interpersonal relationships 79
5.16 Ring of entities connected by marital relationships 81
5.17 Seven companies which have the same contact entity as an insurance company's employee 81
5.18 Three *Individual* entities that have marriage relationships between them 82
5.19 One main entity with many connections of parental and marital types 83
5.20 Entity labeled as *Other* which only appears once in the entire network 84
5.21 Strange cluster layout where an entity has three parenthood relationships 85
5.22 High complexity cluster of different nodes and relationships 85

5.23 First example of a cluster having high probability for fraudulent practices 87
5.24 Second example of a cluster having high probability for fraudulent practices 88
5.25 Third example of a cluster having high probability for fraudulent practices 89
5.26 The network is composed of 100 nodes, single links are added with hopping probability p=0.1 [73] 90
5.27 Bar plot representing the evolution of the number of edges and nodes according to the stratification 91

6.1 Descriptive image depicting the KDD process. 93
6.2 Flowchart depicting the practical approach of this dissertation 97

I.1 Depiction of an example of a csv file when read with a common text editor. In this case, instead of commas, semicolons are used, which are also a valid example of a csv accepted delimiter. 110
I.2 Model representation of a one-to-one relationship. 112
I.3 Model representation of a one-to-many (or many-to-one) relationship. 112
I.4 Model representation of a many-to-many relationship. 113

III.1 PDI schematic depicting the final transformation used for ETL purposes. 118
III.2 PDI schematic depicting the final job used for ETL purposes. 119

IV.1 Overview of the final insurance network produced using Cytoscape . 122
IV.2 Depiction of the stratification based on the node degree (Degree = 1) . 123
IV.3 Depiction of the stratification based on the node degree (Degree = 2) . 123
IV.4 Depiction of the stratification based on the node degree (Degree = 3) . 123
IV.5 Depiction of the stratification based on the node degree (Degree = 4) . 124
IV.6 Depiction of the stratification based on the node degree (Degree = 5) . 124
IV.7 Depiction of the stratification based on the node degree (Degree = 6) . 124
IV.8 Depiction of the stratification based on the node degree (Degree = [7,8]) 124
IV.9 Depiction of the stratification based on the node degree (Degree = [9,10]) 124
IV.10 Depiction of the stratification based on the node degree (Degree > 10) 125

List of Tables

2.1 Summary of the features between KDD, SEMMA and Cross Industry Standard Process for Data Mining (CRISP-DM) 20
2.2 Comparison between the characteristics of machine learning methods. Key: ▲ = good; ◆ = fair; ▼ = poor. 28

3.1 Examples of the relationship between different systems and their corresponding emergences 33
3.2 Different types of scales regarding the study of complex networks [52] 41
3.3 Some of the most relevant metrics regarding complex networks [52]. . 43

4.1 Description of the Insurance_claims.csv file used as a stress test to Python's capability to deal with large datasets 57
4.2 Description of the Insurance_claims.csv file used as a stress test to Python's capability to deal with large datasets post transformation . . 57

5.1 Results from the application of the metrics described to the Zachary's Karate Club Network 63
5.2 Color coding of the relationships present in the network. Note that the color schema may differ from the one used in the software as the color spaces of Red, Green and Blue (RGB) codes may also differ. 67
5.3 Color coding of the nodes present in the network. Note that the color schema may differ from the one used in the software as the color spaces of RGB codes may also differ. 67
5.4 Comparison between the different frameworks for complex network creation. Key: ▲ = good; ◆ = fair; ▼ = poor. 69
5.5 The number of entities and connections present in the first stratification. 70
5.6 The number of entities and connections present in the second stratification. 72
5.7 The number of entities and connections present in the third stratification 73

5.8 The number of entities and connections present in the fourth stratification 74
5.9 The number of entities and connections present in the fifth stratification 77
5.10 The number of entities and connections present in the sixth stratification 79
5.11 The number of entities and connections present in the seventh stratification 80
5.12 The number of entities and connections present in the eighth stratification 82
5.13 The number of entities and connections present in the final stratification 83

III.1 Table depicting all the possible relationships (and inversely the inverse relationships) present in the database 120

Listings

2.1 K-Means Clustering Algorithm in pseudo-code [1] 22
2.2 Back-propagation Algorithm in pseudo-code [2] 25
2.3 Random Forest Algorithm in pseudo-code [3] 27
4.1 Coding example of the creation of a Series in Python 48
4.2 Coding example of the creation of a Dataframe in Python 48
4.3 Coding example for creating a view answering the problem above 51
5.1 Coding example of the creation of Zachary's Karate Club network using Networkx 62
II.1 Python code used for ETL purposes. This was achieved using several different Comma Separated Values (CSV) files. 115

Acronyms

ANN	Artificial Neural Networks.
API	Application Programming Interface.
APS	Associação Portuguesa de Seguradores.
CRISP-DM	Cross Industry Standard Process for Data Mining.
CSV	Comma Separated Values.
DMS	Distributed Manufacturing System.
EDA	Exploratory Data Analysis.
ELT	Extract, Load and Transform.
ETL	Extract, Transform and Load.
KDD	Knowledge Discovery in Databases.
OECD	Organization For Economic Co-Operation And Development.
OS	Operating System.
PDI	Pentaho Data Integration.
RGB	Red, Green and Blue.
SAS	Statistical Analysis System.
SEMMA	Sample, Explore, Modify, Model and Assess.
SQL	Structured Query Language.
SVM	Support Vector Machine.

Chapter 1

Introduction

This is the introductory chapter of this dissertation. The reader will be exposed to the context of this work, as well as the motivation behind it's research and development in Section 1.1. On Section 1.2 an elaborated explanation will, once more, expose the reader to the details of the problem spotlighted, and highlight which are the main benefits achieved by solving and answering it's intrinsic questions. Furthermore, Section 1.3, will give a generic answer to the problem stated in the previous section. On Section 1.4 the contributions or key takeaways to the industry will be discussed. Lastly, Section 1.5, will present the base outline of the remaining chapters of this document.

1.1 Context and Motivation

The insurance industry is undoubtedly one of the main drivers of today's economy and certainly will be so for the foreseen future. This is derived from a basic need every person has, and that is to have safety, or at the very best, to feel safe. Insurance Policies are one of the most important tools for damage control a person, or entity, can have in their possession. They are the result of a previous agreement between two consenting parties, one being the insured and the other being the insurance company. In their essence, insurance policies are the contract themselves, in which the risks the insured is covered from and how much the person or entity will pay for those services, also known as premium is specified. Depending on what's being insured the premium can be higher or lower.

There are two main areas that stem from the school of thought regarding insurance. The first one is life related polices, and the second one is regarding all the other types of insurance, also known as non-life policies [4]. Life insurance policies, as the name implies, cover the policyholder from life threatening events, being the most common one death. Non-life policies, regard all the other kinds of insurance policies that don't fall under the category mentioned before. These possess a wide range, being the automobile, housing and health types the most common (these types of policies are usually mandatory by law, depending on the country) [5]. Life Insurance, according to the Organization For Economic Co-Operation And Development (OECD), represent roughly 59.6% of all insurance policies performed in Portugal regarding the period of 2016. The remainder, obviously relates to the non-life insurance policy types [6].

Technological innovation has always been at the core for social, economical and personal evolution. In the beginning, for mankind, technology was comprised of small yet big discoveries and inventions that made society what it is today. Nowadays, technological breakthroughs don't have the same impact as they had in the beginning. They serve as drivers for the enhancement of systems already implemented and running. With this though in mind, the evolution of data mining techniques serves as the catalyst for new ways to see the world. These new approaches enable the discovery of overlooked knowledge, most of the times already present on the data possessed by the companies.

With this in mind, and seeing that we live in an era of data abundance and an ever-growing data integration in every industry, there is a pressing demand for methods that can analyze and extract knowledge from all this data. With infrastructures like so, it is possible to use data prediction and knowledge discovery techniques not only to detect anomalies, but also to maximize several parameters and features deemed of importance by the companies. Such sensibility to these opportunities will undoubtedly lead to a better understanding of how the data can help to predict uncommon events such as insurance frauds and scams, leading to a better exploitation of the company's tangible (infrastructures, people, economical resources) and intangible assets (reputation, experience, intellectual property) [7].

In this context, Holos, S.A. has an ongoing product which, in its essence, is an insurance management solution denominated RIFT Insurance Tool. The main objective of RIFT is to allow customers, people or companies, to manage (in every sense of the word) their insurance policies. Albeit private, a non-disclosure agreement was performed in order to extract customer information and data from the databases which support this online tool, in order to manipulate and perform

knowledge discovery techniques upon them. These procedures will be further explored in the following chapters of this dissertation. With this partnership, it was possible to use real life user databases, and to test upon them the models described on 1.3, enabling then the opportunity to assess how these models work on live data, giving a more solid foundation to the concluded hypothesis.

1.2 The Problem

One of the major issues that plagues the insurance industry is fraud. Insurance fraud can take many forms, ranging from external to internal, soft and/or hard, underwriting or claim. Internal fraud is committed by the insurance company or entity itself, while the external fraud is employed by the policyholders. One type of the internal variant is the embezzlement of funds, while, on what regards external fraud, there is the submission of fraudulent claims. Soft fraud refers to the kind of fraud originated in opportunistic behavior of normally honest people, whilst hard fraud falls in the category of a premeditated action[8]. Lastly frauds can be committed both at underwriting or claim phase. Underwriting refers to the omission of information to obtain dishonest advantages. Claim refers to the deliberately create false or fictitious claims about the events that triggered the insurance.

According to W. Steve Albrecht, one of the authors of Fraud Examination [9], it is even easier to commit fraud with the help of technology as it ever was: "With the advent of computers, the Internet, and complex accounting systems, employees now need only to make a telephone call, misdirect purchase invoices, bribe a supplier, manipulate a computer program, or simply push a key on the keyboard to misplace company assets".

The way toward identifying and forestalling insurance fraud can be exceptionally challenging because of the qualities and nature of fraud. Figuring the genuine expenses of the premise for organizations and enterprises is an impressive assignment on account of the stealthy idea of fraud [10]. The recognized fraud cases speak to just a "hint of a greater challenge" of the considerable number of unlawful practices that go unnoticed [9]. What's more, fraud can be exceptionally mind-boggling and has some temporal characteristics and patterns that need to be discussed [11].

Hereupon, this dissertation addresses possible solutions to the problem mentioned above and practical, cost effective and most of all precise, ways to implement the solutions mentioned on 1.3.

1.3 Proposed Solution

To solve the problem of insurance policy fraud and profit deficit, the proposed solution is comprised of a set of knowledge discovery techniques based on complex networks for pattern and outlier analysis. These types of networks will allow the users to visualize relationships between different entities and provide insights on fraudulent behaviors. To accomplish the correct employment of these methodologies, the data underwent a process of data engineering following strict guidelines that rule over data mining problems.

The final framework will enable the user to fetch, directly from a relational database, data and pipeline it to a software that will build the complex network. After this is done, the user has the freedom to explore the network using a web app interface, giving then the opportunity for a system agnostic solution.

1.4 Contributions

The main contribution of this dissertation is clearly the methodology from which Pentaho and Cytoscape emerged as the better tools to streamline the whole data mining process when studying the possibility of fraudulent entities and relationships in the dataset. There was an extensive trial and error process behind the choosing of these tools, as it came to light that, when dealing with large quantities of data, hardware and therefore software plays a big role on the overall performance and reproducibility of the overall process.

With the study of complex networks, another contribution for this dissertation is the application of a "still not that big of a player" methodology on the data mining playing field. Complex networks have been used for biology and chemistry investigative problems for so long, but only now are they reaching the data mining world. They provide a great tool for data visualization and manipulation without having a learning curve as steep as other methods already in place.

The last, but not least, main contribution to this dissertation was the opportunity to work with real data. This data came from real insurance companies and directly translating to real people (all the data extracted went through a process of anonymization to protect the identity of customers). This added a degree of reality when dealing with the problem of insurance fraud, making the finally solution also more applicable when addressing the implementation on the industry versus a purely academic data generated approach.

1.5 Outline of the Document

The remainder of the document is presented as follows:

- **Chapter** 2 - State of the Art - Not only presents the methods used by companies, nowadays, in insurance fraud detection, but also the machine learning techniques employed in those methods. Lastly, the theoretical foundations to the proposed solution are presented.
- **Chapter** 3 - Complex Networks - In this chapter, the fundamental theory behind complex networks and what envelops them will be presented.
- **Chapter** 4 - Data Engineering - The development of the model used for pipelining and performing data manipulation operations will be explored alongside other tools that boosted the model's understanding and forthcoming.
- **Chapter** 5 - Complex Network Framework and Result Analysis - A comparison between frameworks for developing complex networks will be presented. After that, the results that stemmed from the chosen framework will also be presented and discussed.
- **Chapter** 6 - Conclusions and Future Work - Presents the conclusions of the work developed in this dissertation and where it stands regarding the foreseen future.

Chapter 2

State of the Art

The most common methods used for insurance fraud and scam detection diverge into two different ideologies: the first one being an approach based on the insurance company's previous experience dealing with situations possessing similar characteristics; the second methodology is based upon machine learning techniques, also known as artificial intelligence methods. Both have their advantages and downsides, which will be further deepened in the following sections. The conventional methods will be presented on **Section** 2.1 as well as it's subsections. The same happens for the machine learning methodologies, on **Section** 2.2

2.1 Conventional methodologies for fraud detection

Detecting and preventing insurance fraud has always been done, it's nothing innovative. The techniques used to avoid fraud take upon many mantles, but the core idea is to stop fraud (or preferably try to to prevent it) at it's early stages of conception. The two following subsections shine light upon two of the major activities that try just that - to prevent fraud, as it was said, early on. **Subsection** 2.1.1 presents what insurance companies can do to prevent losses, and **Subsection** 2.1.2 presents the idea that alliances between companies can provide a better support to a fraud detection network.

2.1.1 Prevention and detection - Insurance Company level

What can an insurance company do to protect itself from fraudulent claims at an enterprise level? The answer is not as simple as one should expect. Insurance companies perform tangible investments on their greatest assets, and those are the people who work there. The foremost defense is the person who is redacting the insurance policy. It's natural to extrapolate that, this being the person (or people) responsible for the primary contact with the costumer, they should be able to detect, from experience and training, certain red flags triggered by the costumer.

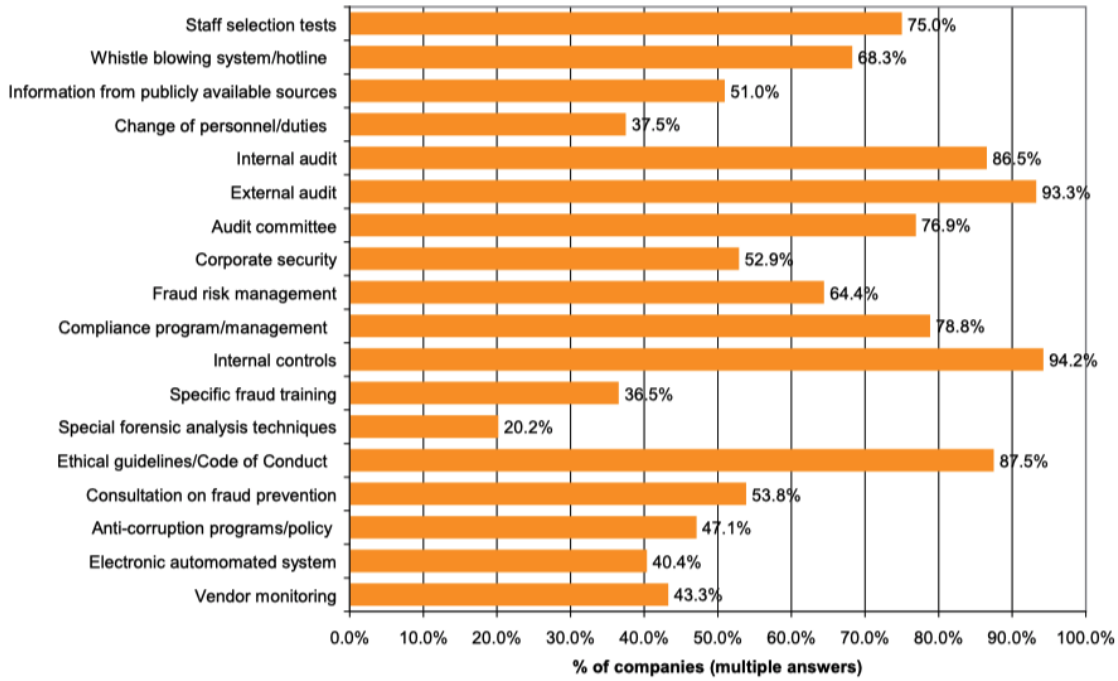


Figure 2.1: Survey performed upon Australian organizations identifying basic fraud control and prevention techniques by *PricewaterhouseCoopers Economic Crime Survey 2007* [12]

As illustrated in Figure 2.1, it is possible to infer that, according to the survey, there are four main key elements companies deem of importance to implement as measures for fraud prevention at a front-end level. Those are:

1. Internal controls;
2. External and internal audits;
3. Ethical guidelines/Code of conduct;
4. Compliance program/management.

As one can imagine, **internal controls** refer to the measures the company takes into account. These controls are usually issued by the board of directors and range from financial reports to management policies. This line of thought stems from the fact that, usually, employees feel safer, in the sense of not needing to worry about repercussions and thus, leading to a boost in performance, performing their job, if employers establish a relationship of trust and and friendship with them. Getting to know a company's workforce is the leading way, according to the study mentioned on Figure 2.1, to avoid certain situations which may increase the chances for mistakes to be made.

External and internal audits serve their purpose as methods for performance and result assessment. Despite both having the same goal, external differ from internal in the sense that a less biased approach is taken when performing the audit. Despite their forthcoming differences, they both assess several predefined parameters defined by the audit firms (in case the audit is external). These evalu--ate the financial reports the companies present at the end of the fiscal year. They look for misstatements or blatantly indices of fraud. Audits perform a crucial role as they can detect situations that can go unnoticed in the day-to-day life cycle of the company.

Ethical guidelines/Code of conduct refers to the established rules for conduct and ethical execution of tasks. Many times, unspoken and taken for granted, these guidelines strive to achieve a good performance in what regards the job implementation. Regarding insurance companies, these social and ethical guidelines, are once more, a way of minimizing the risk of falling into the fraud pitfall. Some of these approach topics like receiving gifts from customers, conflicts of interest and breaching of company's policies guidelines [12].

Compliance program/management are a way that companies have, to instill certain methodologies to their employees. Once more, according to the study, the companies interviewed deemed of relevance this mean of getting through to people. This is also one of the best ways to keep up with the current legislation and needs of the insurance industry. Lastly, keeping up with the times, once more, is an effective way of combating insurance fraud and scam. The more informed people are, the better and more appropriately they deal with unexpected situations.

Having said all this, to reiterate, keeping the people who deal directly with the costumers well informed on what is acceptable and what is not, greatly diminishes the probability of anything happening at the moment of the creation of the insurance policy. Experience also plays a big role in fraud detection and prevention as the trends for scam schemes don't diverge much from what they

used to be in the past. For example, the norm for automobile insurance fraud is to have staged accidents and then ask the insurance company to pay for the damages (activate the policy to receive the premium) [13]. After the contract is made, the likelihood the fraud is detected greatly decreases and so, **Subsection** 2.1.2 explains the main way companies found to try and decrease this trend.

2.1.2 Anti-fraud alliances

As a targeted response to the growing evidence backing up the seriousness of the insurance fraud problem, more and more companies are adhering to a strategy of establishing specialized task forces, to tackle the problem at a country-wide level. Mostly comprised of assets belonging to the insurance companies, they are also constituted by people responsible for sanctioning the fraud perpetrators. These task force's primary objective is to investigate suspicious insurance policy cases and help insurance companies deal with them accordingly [13]. Other important features of these alliances, or bureaux, is that they serve as a means to disseminate information about the offenders, they function as an awareness-raising and standardization tool for the industry norms. This way, the industry and the stakeholders have the right tools not only to protect themselves but also to change their attitude towards insurance fraud [13].

Therefore, what do, insurance companies, have to benefit from joining forces against fraud? Short answer, everything. By pooling resources, companies gain the ability to gather all the disperse and under-utilized information scattered along the system. It was well know, for the automobile insurance industry, that fraudsters gained the upper hand by exploiting this particular weakness. They would go from town to town and create policies with different alias, thus seeming that the policyholder was someone different every time [14]. Nowadays, that's much harder, for example, in Portugal, there is a system in place called *Associação Portuguesa de Seguradores (APS)* on which, insurance companies share information about the policyholders, thus leading to the creation of an ecosystem on which it is more difficult to exploit the gaps. This way, automobile insurance scammers are more easily detected and notified of their actions. The name of the tool used for this purpose is SEGURNET.

The APS is a clear example of a centralized intelligence gathering system, one which enables the linking and sharing of legal information (with some limitations) like policies, applicants, policyholders, claimants, risks, losses, doctors, repair shops, lawyers, and so on (depending on the type of industry approached). Despite this being the case, this doesn't mean the companies have access to all the

information about one another, that would compromise honest competition [13].

2.2 Machine Learning Methods

As one can imagine, given that today, we live in an era driven by buzzwords such as Big Data, Machine Learning, Data Integration and so on, the likelihood of analyzing and extracting knowledge has become increasingly more tiresome and complex. To get around this fact, the industries, and in particular the insurance industry, have devised ways to process all this information and be able to extract something of value from it. These approaches are denominated Machine Learning and Data Mining methods and are at the core of the newfound techniques for extracting knowledge from data.

But, before anything can be extracted from raw data, it must first be transformed and standardized as, once more, working with raw data can be counterproductive in the sense that the difficulty of the extraction of features and knowledge increases exponentially. Thus, on 2.2.1, ways for performing transformations and standardization will be presented.

After those transformations are made and methodologies taken into account, the possibility to perform data mining methods is enabled. Those, which are currently employed in the insurance industry, are explained in sections 2.2.3.1, 2.2.3.2 and 2.2.3.3.

2.2.1 Data Preparation

2.2.1.1 Extract, Transform and Load

ETL is the process of integrating data from multiple, typically disparate, sources and bringing them together into one central location. It is one of the main components for businesses to successfully and efficiently make use of the data stored in a data warehouse [15]. For most businesses, the potential for usefully using the data to its fullest is locked behind insurmountable quantities of raw data; one study revealed that nearly two-thirds of companies do not benefit enough, or at all, from the data in their possession [16] [17]. Given the definition of ETL, it is now of importance to approach the meaning of each step to the end goal. As implied in its name, ETL is comprised of three distinct steps, each one of equal importance on the chain of events.

The first link in the chain, called **Extraction**, aims to extract data from the source systems to be passed along, with the intent of being processed. This step

also tries to obtain the data as efficiently and with as little impact to the source system as possible. In the current days, businesses, collect and store data from an assortment of different sources, each with their own methodologies and techniques to format that data [15], and sometimes this fact can lead to the first step of the process to become the most time-consuming.

The following action of this process, **Transformation**, takes into account the data extracted in the previous step. This data goes through a process of transformation (one of the possible transformations is the conversion of the acquired data) in order to meet the requirements of the target system. This step can also involve the following transformations:

- Purging and approving data to guarantee just quality data is relocated to the new framework
- Sorting the information into segments to enhance convenience and accessibility
- Combining or blending information from various source frameworks and de-copying that information
- Applying business standards to information
- Creating rules for information approval that can be mechanized to check information for quality and precision

This process entails the previous transformations in order to ensure the quality and integrity of data. Without this step, businesses cannot be confident in the data being migrated or integrated into the target system, which can lead to an undefined amount of time and assets wasted.

The last, but not least, element on the ETL process chain, is the **Load** phase. This step ensures that the extracted and transformed data is loaded correctly into the end system. The successful completion of this step is directly connected to the volume of data, the structure of that data, and frequency at which it will be loaded at. Depending on the complexity of this step, two typical ways that data can be loaded into a data warehouse, are presented: full load or incremental load. This ensures that the systems ruling the process are not overwhelmed by the sheer amount of data in the process.

Figure 2.2 depicts the entirety of the ETL process from left to right. There is also the part corresponding to the business analytics depicted on the far right of the figure, on which the machine learning methods will be applied.

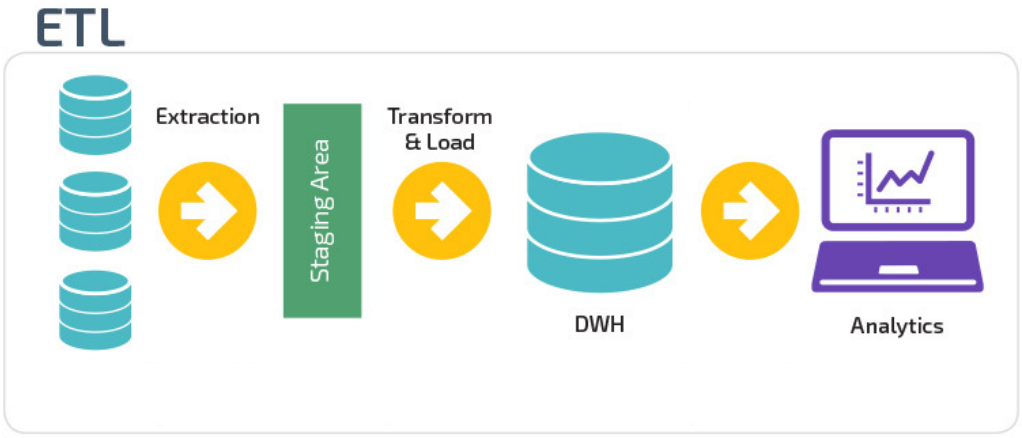


Figure 2.2: Descriptive figure depicting the ETL process from left to right. DWH stands for a data storage solution which is not relevant to be addressed.

Source: `https://panoply.io/data-warehouse-guide/3-ways-to-build-an-etl-process`

Despite being a very good and comprehensive tool, the ETL process it is not without its flaws and, therefore, some practices can be put into place to help, enhance and expedite the flow. During the ETL process, data is pulled from multiple sources, and, some of those, may be historic or legacy data and therefore may not be optimized for business and analytic practices. Through ETL, it very well may be moved to a focal information distribution center (data warehouse) in an institutionalized organization, which is considerably more reasonable for business insight. With the emphasis put on having the capacity to settle on information-driven choices, it is essential to take proactive stratagems to guarantee information quality. Purging, approving, de-copying, and profiling information are central accepted procedures that not just give quicker time-to-esteem to ETL and other information combination and relocation forms, yet engage associations in improving how they use the information in their possession [15].

ETL, once more, is a great tool to deal with the problems of today's paradigm, but as one walks through an even more data-rich environment, this methodology begins to fall behind another well known one, and that is Extract, Load and Transform (ELT). As it is illustrated in Figure 2.3, the main difference is where the transformation process occurs. Another important feature worth mentioning is the fact that, in the figure, the cloud symbol is present. This serves the purpose of identifying that the ELT process is well designed to be implemented on cloud architectures. With modern, cloud-based infrastructure technologies, frameworks can now bolster their support to extensive data storage and scalable compute power at diminished costs. In this manner, it becomes advantageous to keep data in a vast and regularly growing data pool, unending quick handling

capacity to keep up with the extracted raw data [17].

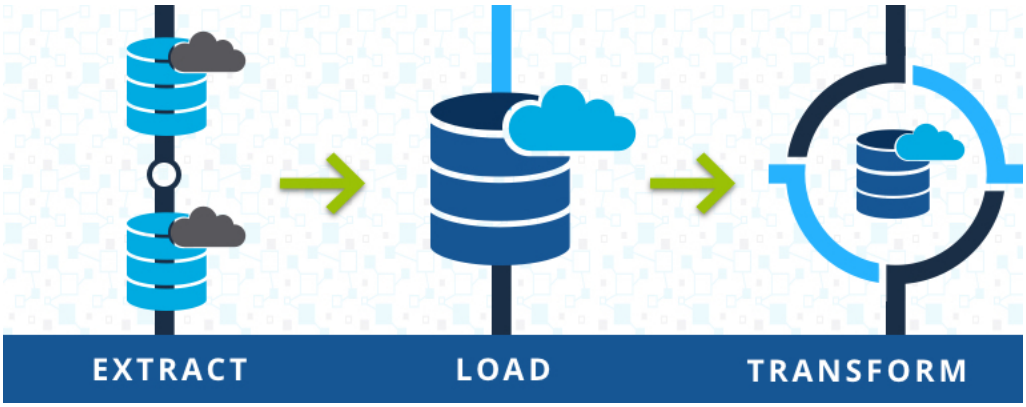


Figure 2.3: Descriptive image depicting the ELT process from left to right.

Source: https://www.talend.com/resources/elt-vs-etl/

With this in mind, if the industry, the tool is going to be implemented on, allows it, ETL should be employed. If what it is required is a scalable tool, with the future of data technologies in mind, ELT should be implemented instead. This is due to the fact that all the raw data can be stored until it is ready to be used, therefore becoming easier to divert all the resources to the extraction and loading. As with any tool, knowing when to use it is at least as important as knowing how to use it.

2.2.2 Data Mining Methodologies

2.2.2.1 Cross Industry Standard Process for Data Mining

CRISP-DM is a methodology described through a hierarchical process model, consisting of six different phases, each one having a direct impact on the next, in order for data mining experts to build analytics models. The flow of the life cycle of this methodology can also be described by how each step of the process is running and then assess how it will influence the next step as can be seen in Figure 2.4.

It is also possible to observe in Figure 2.4 how the flow is directed, and which of the steps are involved. One of the characteristics of this particular methodology is the ability of the inter-phase processes not being strict but rather dynamic, meaning it depends on the output of the previous phase. On a grander scale, this means that his methodology is not linear, allowing for more freedom when executing the tasks. As an outcome of this fact, the arrows in the figure only depict the most important and frequent dependencies between phases [18].

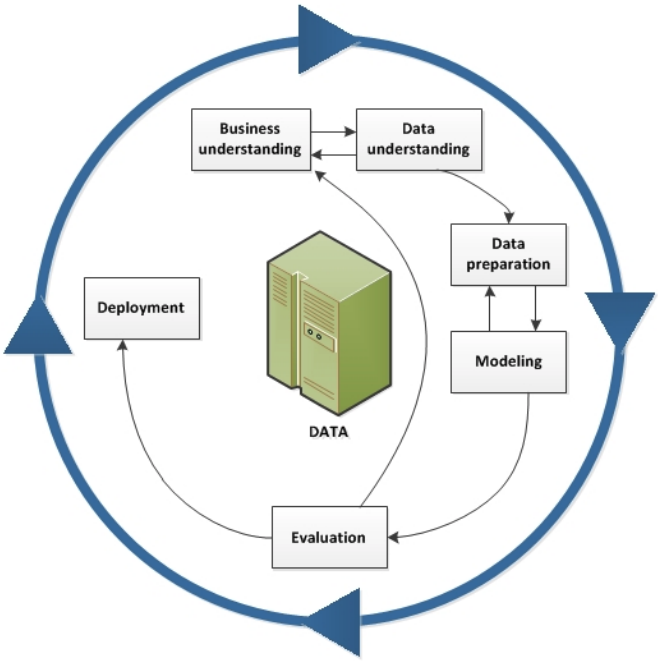


Figure 2.4: The CRISP-DM reference model.

Source: `https://tinyurl.com/yb2polyd`

This being said, the six phases, and what purpose they serve [19], are the following:

- **Business Understanding** - This crucial stage revolves around understanding the future undertaking objectives and necessities from a business perspective, and afterwards changing over this information into a data mining issue definition, and a primer arrangement planned to achieve the objectives.

- **Data Understanding** - The data understanding stage starts with an underlying information assembling phase and proceeds with activities to get to know the data, to perceive data quality issues, to discover first bits of knowledge about the data or to recognize interesting subsets to shape theories for shrouded information.

- **Data Preparation** - The data preparation stage covers all exercises to build the last dataset (information that will be fed into the modeling tool(s)) from the underlying raw data. Information readiness errands are going to be played out numerous times and in no specific order. Assignments incorporate table, record, and characteristic determination just as transformation and cleaning of information for modeling devices.

- **Modeling** - In this stage, different modeling methods are chosen and connected, and their parameters are adjusted to typical values. Typically, there are a few systems for similar data mining problem type. A few techniques have explicit necessities on the kind of information. In this way, venturing back to the data preparation stage is regularly required.

- **Evaluation** - At this stage in the endeavor, a model (or models) that appears to have high gauge has been built, from a data analysis perspective. Preceding the last arrangement of the model, it is essential to, even more through and through, evaluate the model and overview the means executed to build up the model, to make sure it suitably achieves the business objectives. A key objective is to choose whether there is some fundamental business issue that has not been adequately considered. After this stage, a decision on the use of the data mining results ought to be come to.

- **Deployment** - Despite whether the explanation behind the model is to build the knowledge of the information, the data got ought to be dealt with and showed to such an extent that the customer can use it. Dependent upon the prerequisites, the deployment stage can be as fundamental as delivering a report or as flighty as executing a repeatable data mining process. When in doubt, it will be the customer, not the data analyst, who will perform the deployment steps. In any case, paying little mind to whether the expert will not make the deployment effort, it is fundamental for the customer to understand ahead of time what exercises ought to be finished to make use of the models assembled.

Despite this phases being more or less standardized across the industry, players tend to adapt this approach to their specific needs. This being said, the core values of the methodology remain intact, only differing on the phase order taken, and how they implement the steps needed for the project completion. One other thing worth mentioning, is the fact that this methodology was created on 1999 [20], thus a previous revision on what has changed since then is good practice. For example, since today the data volume paradigm has changed dramatically, a more careful approach to the data blending from diverse sources is needed, quality control has to take a tightened approach and additional steps for optimization may be needed [21].

2.2.2.2 Sample, Explore, Modify, Model and Assess

The SEMMA method is another way to express the methodologies for building analytical models. This method was developed by the SAS Institute in order to materialize the company's ideologies regarding the data mining applications developed in-house.

Despite being dubbed as a method, the Statistical Analysis System (SAS) Institute claims it is more like a logical organization of the functional tool set of their data mining tool, the SAS Enterprise Miner, for carrying the core tasks of data mining [22]. Despite this, and for the sake of simplicity and coherence, this logical line of thought will be dubbed as a methodology.

As can be observed in Figure 2.5, and as the name implies, the SEMMA flow is comprised of five different phases, which have the following purposes:

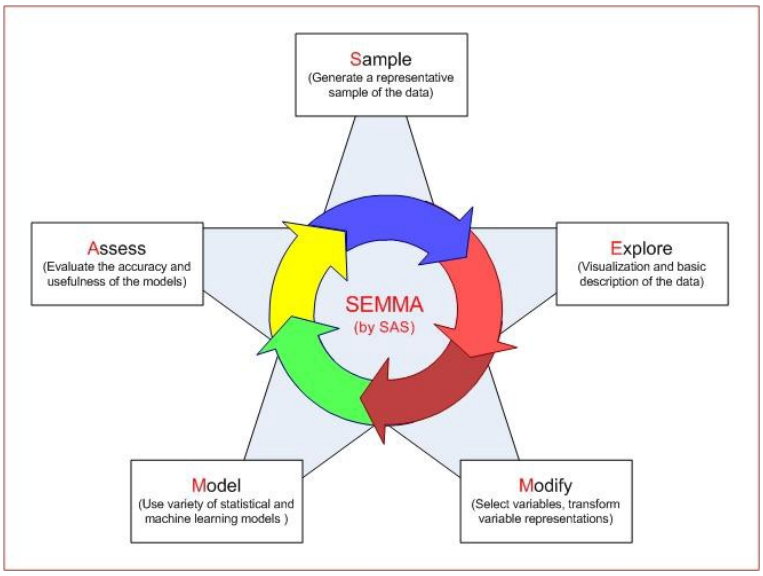


Figure 2.5: The SEMMA reference model.

Source: `https://tinyurl.com/y8h78qm9`

- **Sample** - This stage comprises inspecting the information by separating a part of a broad dataset sizable enough to contain a representative portion of the data, yet small enough to be manipulated rapidly. This stage is brought up as being discretionary.
- **Explore** - This stage comprises the investigation of the data via hunting down unforeseen patterns and peculiarities so as to increase comprehension and gain ideas. Sometimes, this phase is not feasible through visualization only. To counter that, statistical methods may be used to aid in cluster or pattern recognition.

- **Modify** - This stage comprises the adjustment of the data by making, choosing, and changing the factors to center the model selection process. Taking into account the findings from the previous phase, this step may already be pointed in the direction to the results the business wants. The modification can also occur due to the fact that the data, when it was mined, was in a dynamic stance.

- **Model** - This stage comprises modeling the information by enabling the software tools to scan naturally for a mix of data that dependably predicts an ideal result. The models used fit the category of statistical knowledge like the ones described on 2.2.3.2 and 2.2.3.3.

- **Assess** - This stage comprises surveying the data by assessing the value and unwavering quality of the discoveries from the data mining process and gauge how well it performs. In this phase, once more, it is possible to use statistical methods for assessing how good was the knowledge extracted from the data.

With this method in place, the possibility to determine how relevant the findings were for the business arises. This enables the assessment of the tool developed by SAS Institute (SAS Enterprise Miner) and how it fares for the client, giving a structured approach to the data mining problem [23].

2.2.2.3 Knowledge Discovery in Databases

The final methodology to be referenced is the KDD. This differs from regular Knowledge Discovery in the sense that, KDD is directly applied to databases (as the name implies) giving it a higher degree of abstraction. This is due to the fact that, if the data is present on a database, it has likely been subjected to some degree of ETL techniques to some extent [23]. Nonetheless, and as it is possible to observe in Figure 2.6, this method is comprised of five distinct phases, those of which have the objective to extract certain desirable features and patterns known as knowledge. The following is a brief explanation of what is the goal of each phase:

- **Selection** - This step consists on selecting all the data that fits the description of what is wanted or needed for the project (target data), with the objective of performing the data mining process.

- **Preprocess** - On this step, the selected data goes through a process of cleaning and standardization in order to obtain an even more relevant subset of data for the client to work upon.

- **Transformation** - Here is where the data transformation techniques are applied. Data dimensionality reduction and transformation methods are employed.

- **Data Mining** - One of the most important steps for the KDD method. In this phase, data is searched for patterns of interest depending on the objective.

- **Evaluation** - After the entirety of the process is concluded, the end result (knowledge) is evaluated and interpreted.

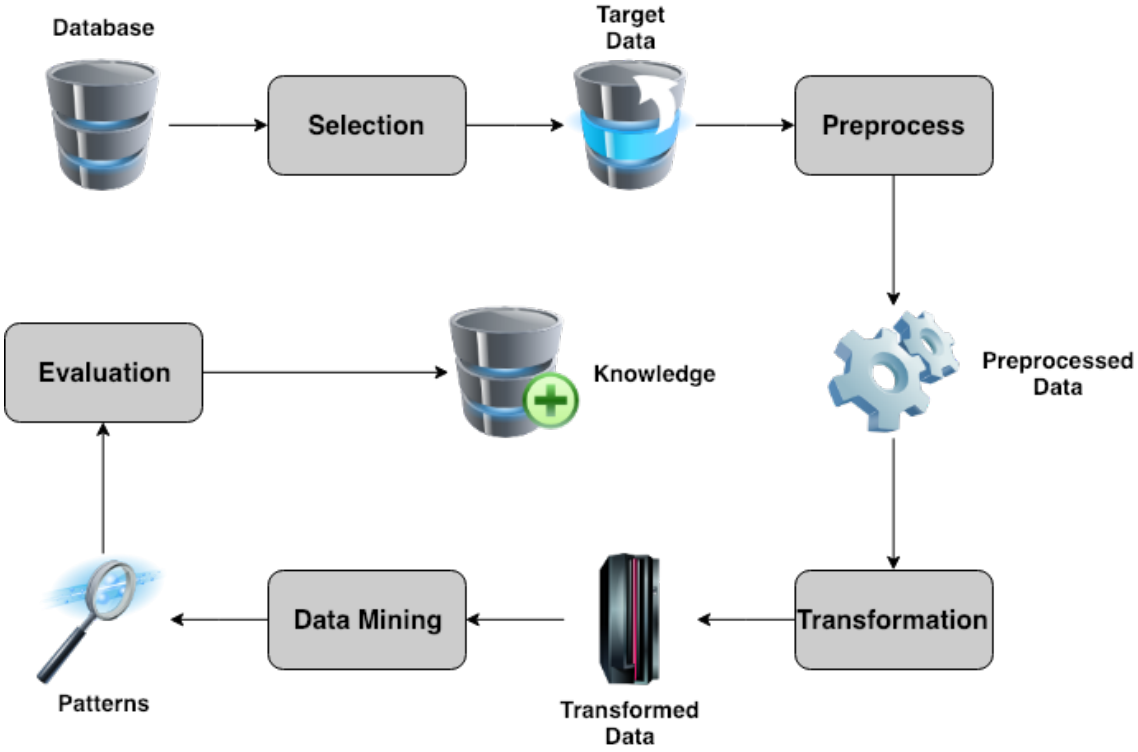


Figure 2.6: Descriptive image depicting the KDD process.

Despite the usefulness of this method, previous knowledge of the application domain is required in order to perform clear and profitable decisions for the end-user. After the knowledge is extracted at the end of this process, it can also be incorporated (and it should) into the system. Another thing worth mentioning is that there can exist loops between any two steps of this chain [24].

2.2.2.4 Overview on CRISP-DM, SEMMA and KDD

After going through the three main methodology models for a data mining process, it is now possible to compare each other for their strengths and weaknesses.

With respect to the current examination, it is conceivable to presume that SEMMA and CRISP-DM can be viewed as an execution of the KDD procedure. At first sight, it is possible to conclude that CRISP-DM is more complete than SEMMA. In any case, investigating it more top to bottom, it is possible to incorporate the improvement of comprehension of the application space, the significant prior knowledge and the objectives of the client, on the Sample phase of SEMMA, in light of the fact that the data cannot be examined except if there exists a genuine comprehension of all the exhibited viewpoints, allowing solidification by consolidating this knowledge into the framework. This prompts the way that standards have been accomplished, concerning the general procedure: SEMMA and CRISP-DM do manage to achieve their goals and implement frameworks businesses can work upon.

In table 2.1, a brief summary of the main phases of each method is presented as a way of understanding the correspondences between each model.

Table 2.1: Summary of the features between KDD, SEMMA and CRISP-DM

Adapted from [23]

KDD	SEMMA	CRISP-DM
Pre-KDD	————	Business understanding
Selection	Sample	Data Understanding
Pre processing	Explore	
Transformation	Modify	Data preparation
Data mining	Model	Modeling
Interpretation/Evaluation	Assessment	Evaluation
Post KDD	————	Deployment

2.2.3 Machine Learning Algorithms

2.2.3.1 Clustering

Clustering, or data segmentation, is the process of assigning groups to a set of objects so that they fall into classes of similar properties. The primary property of the cluster is that the objects inside the same cluster must be similar to each other while being dissimilar to objects of other clusters [25]. The great benefit from grouping up objects is that they can be treated as one, since they are in the

same cluster, thus enabling the possibility of identifying distribution patterns and correlations between attributes.

There are two main objective ways to employ the cluster method. These structural procedures take the form of data matrices (2.1) or dissimilarities matrices (2.2). Data matrices usually contain the raw data before it is processed, whereas the dissimilarities matrices contain the data after an algorithm runs through the raw data. This being said, the dissimilarities matrices contain the data of most importance for the intended objective.

Data matrix (or object-by-variable structure): This represents n objects with p variables (also called measurements or attributes). The structure is in the form of a relational table, or n-by-p matrix (**n** objects × **p** variables) [25] as seen in 2.1:

$$A = \begin{bmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i1} & \dots & x_{if} & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{bmatrix} \tag{2.1}$$

Dissimilarity matrix (or object-by-object structure): This stores a collection of proximities that are available for all pairs of n objects. It is often represented by an n-by-n table:

$$B = \begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix} \tag{2.2}$$

where $d(i,j)$ is the measured difference or dissimilarity between objects i and j. In general, $d(i,j)$ is a non negative number that is close to 0 when objects i and j are highly similar or "near" each other, and becomes larger the more they differ [25]. Since $d(i,j) = d(j,i)$, and $d(i,i) = 0$, the matrix as the one in 2.2 is created which contains all the relevant values.

Despite there being many types of features, for the sake of simplicity and getting the point through, only quantitative types will be considered as for all the other types the methods employed would be significantly different. This being said, one of the most used algorithms for data segmentation is the K-means Clustering, which measures the dissimilarity using the formula of the Euclidean distance as follows:

$$d(x_i, x_{i'}) = \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = \|x_i - x_{i'}\|^2 \tag{2.3}$$

On 2.3, i, i' and j represent the coordinates on a two dimensional plane.

This brings the conclusion that, the closer together on the plane two objects are, the likelihood of them belonging to the same cluster is higher and vice-versa. On Listing 2.1, which presents the K-means algorithm in pseudo-code, it is possible to observe just that.

Listing 2.1: K-Means Clustering Algorithm in pseudo-code [1]

```
Choose the number of clusters(K) and obtain the data points
Place the centroids c1, c2 .... ck randomly
Repeat the steps below until convergence or until the end of a
fixed number of iterations
    for each data point xi:
        - find the nearest centroid(c1, c2 .... ck)
        - assign the point to that cluster
    for each cluster j = 1..k
        - new centroid = mean of all points assigned to that cluster
End
```

The centroids $c_1, c_2 ... c_k$ define the cluster or area where the data points are assigned to. This means, in terms of a graphical representation, that the result will be a set of k well-defined clusters. The goal is achieved when no data point switches clusters in the next iteration of the algorithm.

Lastly, one important feature of this algorithm, is the ability to detected outliers or "uncommon"values. Regarding this, outliers are represented by the values that do not fit into any of the built clusters. The study of these outliers may bring fruitful insights about the problem being studied.

2.2.3.2 Artificial Neural Networks

Artificial Neural Networks (ANN) are probabilistic and statistical constructed models based and modeled with biological neurons and its conglomerates (biological neural networks) in mind. Like their organic counterparts, artificial neural networks are also capable of modeling, adjusting and processing nonlinear relationships between inputs and outputs. Notwithstanding, one of their most significant similarities and advantages is the capability of handling all the information in parallel [26].

ANN follow a well-defined mathematical model to define the adaptive weights which are present along the paths interconnecting artificial neurons. Nevertheless,

what are weights? Weights are a special kind of characteristic that is given to each one of the inputs of the network. It defines, at its essence, how vital the input is. These weights are then multiplied together with each one of the inputs so that they can be promptly fed into the network. Despite weights being of the utmost importance, there is no agreed rule on how to choose them. The objective of the neural networks is to consider the weights and modify them at will, so it can learn and adapt by itself. This is called a tuning algorithm and, as it learns from the observed data, it improves the model at hand.

Despite all of the facts mentioned above, there is one more missing factor that is of the utmost importance for the neural network to function correctly, and that component is a cost function. A cost function is a mathematical function such as a step function, piecewise-linear function, sigmoid function or a sign function [26] that pieces together all of the information from the deep hidden layers and analyzes it to "learn" the optimal solution to the problem being solved. It determines the best values for all tunable model parameters, with its primary target being the weights present at the start of the neural network. The cost function can also determine the learning rate of the network, as these functions are easily plotted, and it is relatively easy to deduce the growing function of these plots.

All these optimization techniques are applied with the result in mind. Their primary objective is to make the solution from the neural network as close as possible to the optimal solution, and when this is achieved it is safe to say that the neural network was able to solve the problem with high performance successfully.

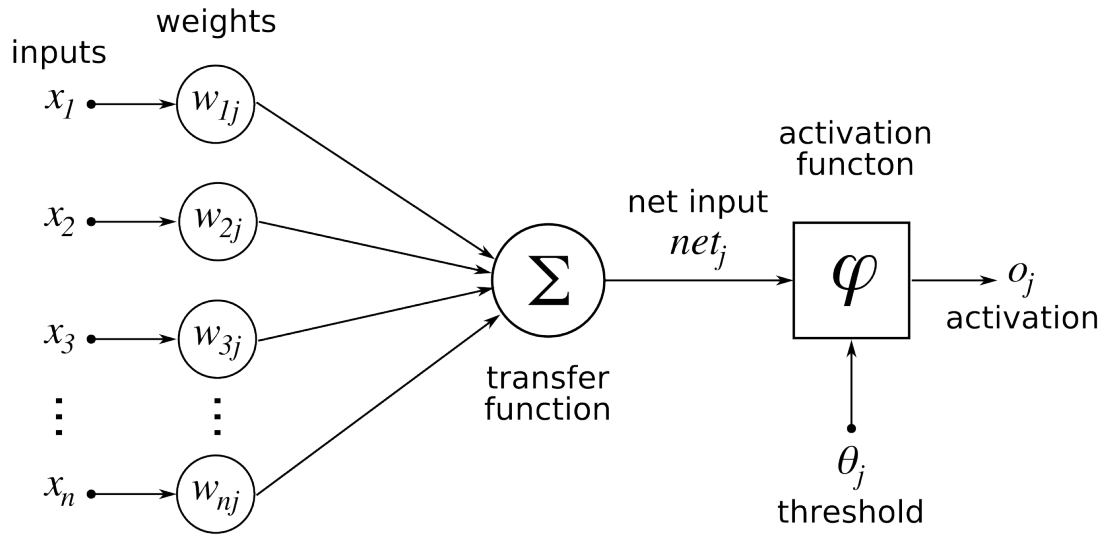


Figure 2.7: Architectural design of an artificial neuron.

Source: https://tinyurl.com/yyusr4jk

From an architectural point of view, a neural network is composed by as many layers as the user wishes to (these layers are an aggregation of as many neurons as needed), plus there are also units made specifically to compute inputs into an activation function. To this activation function it is added a certain threshold to discard certain errors and to understand if the information was correctly passed along. All the information mentioned previously is briefly outlined in Figure 2.7.

As mentioned before, one of the most important features of the network is its ability to learn, adapt and overcome obstacles. These obstacles come in the form of the final output being classified as "bad" or not good enough. This is accomplished through the use of an occurrence (algorithm) called back-propagation. Back-propagation is the method of feeding the output of the network to its input. The goal of this process is to adjust all the weights once again until we get the desired result in the end. This process is also much faster than traditional approaches that involve traditional methods for learning [27]. In the following Figure 2.8 it is possible to observe a flow chart summarizing how back-propagation works.

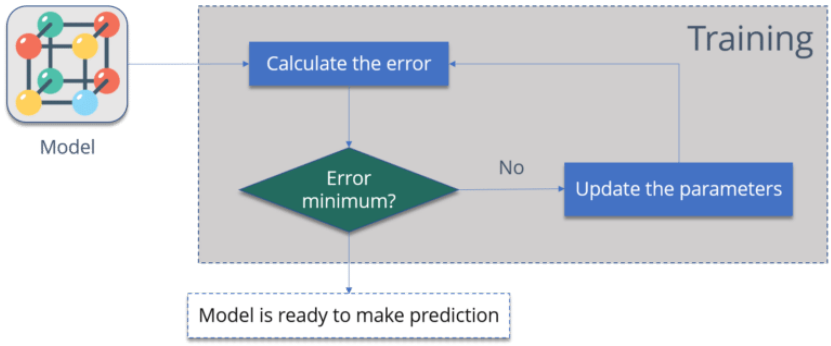


Figure 2.8: Schematic representation of the back-propagation algorithm.

Source: `https://www.edureka.co/blog/backpropagation/`

This schematic (Figure 2.8) depicts how back-propagation works at a macroscopic level. The data comes from the model and the error is calculated at the output stage. If the error does not correspond to what is expected, the parameters are altered, and the data is fed once more to the model. This process occurs as many times as necessary until the end result is satisfactory. When this happens the end result is given to the user, thus enabling the understanding of the data

Below, on another approach, in pseudo-code, the back-propagation algorithm is outlined.

Listing 2.2: Back-propagation Algorithm in pseudo-code [2]

```
initialize network weights (often small random values)

do
    forEach training example named ex
        -prediction = neural-net-output(network, ex)  // forward pass
        -actual = teacher-output(ex)
        -compute error (prediction - actual) at the output units
        -compute  δ w_h for all weights from hidden layer to
            output layer  // backward pass
        -compute δ w_i for all weights from input layer to
            hidden layer   // backward pass continued
        -update network weights // input layer not modified by
            error estimate
    until all examples classified correctly or another stopping
    criterion satisfied
return the network
```

One last aspect to note about artificial neural networks is that while extremely powerful tools for data analysis, they can also become very complex and thus are considered "black boxes" from a macroscopic point of view. This means that their inner-working processes and links are very complex and difficult to understand and infer working methods from. This being said, the employment of ANN should be carefully planned and studied prior to their implementation to avoid making a simple data analysis process more complex than it should be.

2.2.3.3 Random Forest

Before introducing the concept of random forest, it is of importance to first talk about classification trees. These types of trees constitute an algorithm that can be used to work out problems both of the regression and classification kind. As the names implies, this algorithm creates a visual representation similar to a tree, giving a top-down approach in which the topmost node (root node) creates binary splits until certain criteria is met [28] as it is possible to observe in Figure 2.9. A decision tree is also a non-parametric model in the sense that it is not assumed by any parametric form (curves and surfaces described by parameters). Branches and leaves are added during learning depending on the complexity of the problem inherent in the data [29].

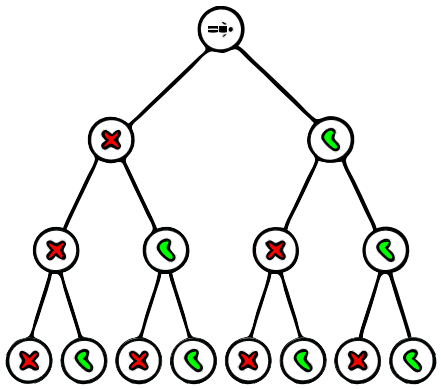


Figure 2.9: Representation of a binary decision tree.

Source: `https://tinyurl.com/yyfgqeeb`

The split is often made according to two different criteria: Gini's impurity or the measurement of Entropy. Both try to achieve the same goal, to maximize the information gain ratio induced by each split. Below, equations 2.4 and 2.5 present the overall formulae that incorporate each criteria.

$$Gini : Gini(E) = 1 - \sum_{j=1}^{c} {p_j}^2 \tag{2.4}$$

$$Entropy : H(E) = -\sum_{j=1}^{c} p_j \log p_j \tag{2.5}$$

On both equations, p_j denotes the probability of each branch existing after the split is made. The total sum of the probabilities (branches) on the same layer of splits must be always equal to one.

The Random Forest algorithm is an ensemble method, meaning that it is made up of smaller models. In this case, and as the name implies, a predetermined number of decision trees is chosen and grouped up to give form to the main algorithm. This comes as an answer to the variance problem. As it is known, each prediction has a certain variance associated to it. This being said, if several predictions are taken into account (one for each tree), the variance tends to spread around the right answer. Statistically speaking, the more predictions are taken into account, the greater the probability they land on the right one.

One of the features that makes this algorithm one of the most chosen ones for data mining processes is one particularity of it's inner workings. As stated previously, each tree performs it's own classification or prediction. Complementing this is the fact that each tree considers a random subset of the features that are present on the dataset, both for testing and training. This comes, once more, as a

statement for the algorithm's likability as it increases the diversity in the forest, leading to a more robust overall prediction.

As one can imagine, there must be a concise way to obtain the final result which is sought after. This comes after all the trees in the algorithm have made their respective predictions or classifications. After that, the mean (in case of a classification problem the mode is used instead) of all the values predicted is calculated, giving then the final result as the answer of the algorithm.

All of the steps mentioned above can be observed in a pseudo-code manner below on Listing 2.3

Listing 2.3: Random Forest Algorithm in pseudo-code [3]

```
1  Randomly select K features from total M features.
2
3      Where K << M
4
5  Among the K features, calculate the node D using the best split point.
6
7  Split the node into daughter nodes using the best split.
8
9  Repeat the above steps until L number of nodes has been reached.
10
11 Build forest by repeating the above steps for N number times to
12 create N number of trees.
13
14 Takes the test features and use the rules of each randomly created
15 decision tree to predict the oucome and stores the predicted outcome
16
17 Calculate the votes for each predicted target.
18
19 Consider the high voted predicted target as the final prediction
20 from the random forest algorithm.
```

2.2.4 Overview on Machine Learning Algorithms

As there are many different methods for making predictions, there comes a time on which, one must be chosen. This choice takes into account many variables, from which, the most relevant are the nature of the problem, and the type of the data associated with the problem.

Industrial and commercial data mining applications will, in general, be particularly testing as far as the prerequisites put on learning methods. Data sets are

frequently exceptionally substantial as far as number of observations and number of factors estimated on every one of them. In this manner, computational contemplations assume a vital job. Additionally, the data is typically chaotic: the sources of info will in general be blends of quantitative, binary, and categorical variables, the latter frequently with numerous dimensions. There are commonly many missing values, complete observations being uncommon [30].

In addition, data mining applications usually have the need for interpreting models as it is insufficient to only produce predictions. Another feature of high importance is the need for some kind of qualitative understanding of the relations between different kinds of data. Thus, black box methods such as neural networks, which can be quite useful in purely predictive settings such as pattern recognition, are far less useful for data mining [30].

Table 2.2: Comparison between the characteristics of machine learning methods. Key: ▲ = good; ◆ = fair; ▼ = poor.

Adapted from [30]

Characteristic	**ANN**	**Random Forest**	**Clustering**
Natural handling of data of "mixed"type	▼	▲	▼
Handling of missing values	▼	▲	▲
Robustness to outliers in input space	▼	▲	▲
Insensitive to monotone transformations of inputs	▼	▲	▼
Computational scalability (large N)	▼	▲	▼
Ability to deal with irrelevant inputs	▼	▲	▼
Ability to extract linear combinations of features	▲	▼	◆
Interpretability	▼	◆	▼
Predictive power	▲	▼	▲

Table 2.2 presents ten characteristics the authors from the book *The Elements of Statistical Learning - Data Mining, Inference, and Prediction* [30] deemed of relevance as a metric to a better judgment when choosing an algorithm for data mining purposes.

These characteristics are considered to be the metrics for judging and evaluating the algorithms and how they face against certain situations. For example,

in the presence of data with features from different types, Random Forests perform the best of the three algorithms. Another important field of analysis is the time algorithms take to compute the data. Regarding this matter Random Forests come out on the top again. Regarding predictive power, from the three algorithms, ANN are the clear winner. This being said and despite being one of the most easily implemented and simple algorithms, Random Forests should not be taken lightly. As it is possible to observe in table 2.2, Random Forest has a clear advantage in almost all the fields of comparison depicted in the table, giving it an all-around "goodness" when it comes to dealing with many types of problems. The comparison's table is a clear example on why Random Forests are almost always implemented on Machine Leaning problems.

Chapter 3

Complex Networks

Complex systems and therefore networks, are a fundamental yet overlooked part of how the world is described and perceived. From a not so simple amino-acid reaction to something more tangible like a network of friends a specific person has, complex networks serve as a means to discover certain insights otherwise hard or even impossible to perceive to the "naked eye". As it will be described in this particular chapter, complex networks derive from a theory with several years and are now being implemented increasingly to solve data mining problems such as the discovery of patterns and/or outliers.

According to Gordon Moore, co-founder of Intel, "The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. Certainly ... this rate can be expected to continue, if not to increase."[31] Despite being enunciated for physical components (integrated systems), nowadays this law can be extrapolated to different areas, in particular, the case of data growth.

It is common knowledge that the data present in the current paradigm's ecosystem vastly outweighs the amount of means to process it. Take for example a commercial flight spanning across the Atlantic Ocean which takes around seven to eight hours. In this short time period, a four engine jumbo plane can generate up to 640 terabytes of data [32]. Multiply this factor by all the flights that happen around the globe and it is possible to get the gist of the problem. Taking it up a notch, this only takes into account the aerospace industry, which is only one among many. According to a study performed in 2016 by IBM, "Every day, we create 2.5 quintillion bytes of data"[33].

As it was presented on the previous chapter, machine learning methods are being implemented as a way to "fight back" all the data that is constantly being gathered and need to be processed in a way to obtain knowledge. Complex Networks come as a new approach to deal with the current situation seeing that they were rarely applied to mining data in the raw. This will enable a way to extract knowledge from data and try to conceive structured opinions about it. It is important to mention that complex networks, and subsequently their analysis, were already employed in the study of biological topologies. This will be further explained and analyzed further down the chapter.

Complex Networks derive from a mathematical branch titled Graph Theory. This theory gives the exact mathematical treatment for complex networks so that these types of networks can be represented as a graph. Giving the data this kind of treatment allows a fundamental characterization of a system once the boundaries, constituent parts and relationships, are known [34].

This being said, it is now even more important to discuss and present some notions of graph theory, what it is, and how it is constructed, in order to better understand its role in the grand scheme of things considered. Despite this being true, there is also a great dependency from system theory, which will also be focused.

3.1 Complex Systems

In order to apply some degree of complexity to a system, one must first define and understand what a system is comprised of. A system is a type of model used in order to describe and understand how a certain process works [35]. Redundantly speaking, a model modulates something in order to transcribe that something into mathematical and conceptual terms. For example, when someone connects to the internet, they are being forced to take part in a larger complex system. Note the use of the word "Complex" adjacent to system. How does a system suddenly go through the process of being expressed as simple to complex?

Aristotle once said that "The whole is more than the sum of its parts"[36], but how does this phrase directly correlate with the study of complexity? Mathematically speaking, the sum of all parts in a closed system is always equal to the whole of the system itself. But does this hold true in the light of the focused study?

Let a more metaphysical approach take form. If one thinks in terms of complex systems, maybe there might be more than meets the eye. Systems, complex or not, are made up of many interacting parts or particles (ex. atoms, cells, people...)

but what makes the difference between the labeling of complex or not is the interaction of the parts that constitute the system. If these parts interact in a way, or do not even interact at all, that it would not matter if they were secluded or not, the resulting system cannot be deemed complex. In order to qualify as complex, the interactions must bring up new types of observations that would not happen at all if these particles were not interacting or the interactions were not being considered as data resulting from the observation. This directly translates to the metaphysical approach of Aristotle's phrase, in which the whole is clearly more or different [37] than the singular expressions of the particles. This is what is called Emergence [38]. Remarkably, and contrary to what one might believe, this concept first surfaced in regards of the philosophical studies [39]. Following this school of thought, the surprising difference, albeit not only in name, starts to come forth. Complex systems possess not only the particles themselves, but also the information and new behaviors that stems from the interactions between them.

Table 3.1: Examples of the relationship between different systems and their corresponding emergences

System	Emergent Behaviors
Condensed Matter	Solidity, conduction, sound...
Fluid	Flow, turbulence...
Ecosystem	Evolution, interaction chains...
Market	Price settings, bubbles, crashes...
Road Traffic	Prevailing speed, traffic jams...

Table 3.1 presents some "out of the box" examples of how emergent behaviors can be extrapolated from a system of many interacting parts. Take for example the Road Traffic scenario. If one looks at the individual parts of the system, cars for example, there are many characteristics that can be refined from analyzing the units that make up the traffic as a whole. In this case, one can consider only cars constitute the system. If only the cars are looked at, only information relative to them is obtained. Now, if the system is observed as a whole, certain bits of information start to emerge. For example, how a car accelerating directly influences the car right behind or in front of it. This is clearly a piece of information that is conceived from observing not only the particles that make up the system but their interactions as well.

This being said, emergence is not enough to give a system the classification of complex. Following the line of thought described above, there are several other characteristics deemed of major importance in order to refer to a system as

complex. They are as follows:

- **Non-linearity** - Non-linearity is one of the key aspects that define complexity. Taking a step back and comparing linearity with non-linearity, the first concept refers to an equation, groups of equations or models, that have a deterministic approach. This being said, any linear system can be well described by a set of linear equations with a predictable outcome. For example, two of the properties that guide linearity are the Superposition and Homogeneous principle. Translating these concepts to mathematical terms, it means that any sum of two equations will inevitably lead to the attainment of another one as seen in:

$$x_1 + x_2 \rightarrow y_1 + y_2 \tag{3.1}$$

  Also, if any linear equation ifs multiplied by any given factor K, the outcome, knowing that the system is completely deterministic, will also by affected with magnitude K as seen in:

$$x \rightarrow y \tag{3.2}$$

$$kx \rightarrow ky \tag{3.3}$$

  Seeing that linear systems behave in a "well-mannered" way, this is not the case for non-linear systems. These types of systems are not deterministic at all when comparing with the linear ones. Take, for example, the case of dynamics. In a linear system, if a certain force with magnitude 10 is applied to an object, it is expected that the object gives of feedback (Action and Reaction Force Pairs) 10 times as much [40]. In a non-linear system this does not happen. Taking the previous example into account, there can be the case where the reaction to the action can be several degrees of magnitude higher or lower. It is this lack of direct relationship between what is given and what is expected that is so important for complex systems that, by nature, enable the perception of new insights.

- **Feedback** - With this characteristic, it is possible to observe how complex theory starts to link up with networks. Feedback is the response a certain particle has according to the stimuli it's neighbors provide. This means, the way a particle influences it's neighbors at an earlier time will dictate, or at least point in a certain direction, how, later on, it will interact with it's neighbors [41]. Take for example a flock of birds. Each particle of the system (bird) will move according to it's surrounding neighbors. If a certain particle adjusts their route, it will start a chain reaction with it's neighboring

particles, making the entire flock change direction. These small adjustments of single (or more) particles produce a certain effect on the entirety of the system. This is also a prime example of how the human body regulates itself.

- **Spontaneous Order** - When addressing order one can consider the two ends of the spectrum: Total Disorder or Total Order. Both ends directly mean no complexity as it is easy to understand that if a system is already in a predetermined state, no useful information can be gained apart from the obvious characteristics and interactions (simple system). Spontaneous order is a sort of middle ground between the two previous absolutes. One that enables the system to have some sort of degree of freedom, but also displayed in a way that makes sense in terms or ordering. In this context, order is a really hard concept to present and can also mean a great deal of many things. One of the definitions that best suit the analysis of networks is the one that states that spontaneous order is the natural occurrence of order (or emergence) in the midst of chaos [42], thus meeting the middle ground, once again, between pure randomness and total chaos.

- **Robustness and lack of central control** - Robustness, and therefore lack of central control, directly correlates with the system's ability to cope with disturbances. Taking a complex system into account, this can be achieved by the way the system is organized. The lack of central control means that the system is distributed instead of having a centralized approach. One clear example of this methodology is one of the new emerging industry paradigm's. The Distributed Manufacturing System (DMS) uses this approach to go around certain problems present in a more centralized industrial methodology [43]. For example, if an entire supply chain of cars is dependent of a certain part which is only produced in a single factory in a certain place in the globe, if any major catastrophe is destructive enough that makes that factory go out of commission, the entire production line of all the cars in the globe will be affected. Now, if that single part is produced not only in that factory but also in other factories distributed geographically, if one of them is disrupted, the entire production chain will be shook but not stopped. Of course from this example a great many deal of things can be extrapolated to other fields, being system theory one of them. Once again, this characteristic is a fundamental part of complex systems.

- **Hierarchical Organization** - Complex systems often present some degree

of organization, be it from the system as a whole or from possible subsystems that manifest themselves. It can be stated that many, maybe even all, of the previous properties manifest themselves because of the ordering of the system [41]. Emergence, for example, starts to occur because there is some type or hierarchical organization between the interactions of the particles. So, in the end, it is safe to assume that all the previous features in addition with some type of organization, stratify the final structure of the system into layers that exchange information between themselves. One of the prime examples of hierarchical organization manifesting itself is the Earth's life ecosystem as a whole. Species are stratified by categories and other types of taxonomic classifications.

Seeing that all the previous characteristics are of major importance for a system to be classified as complex, can it still maintain this classification without a certain characteristic in particular? Short answer, no. System theory and therefore, complexity theory applied to systems is a very ambiguous and subject. For complexity to manifest itself, it is currently agreed that all the characteristics mentioned above must manifest [44]. This is due to the fact that there exist systems which are simple in nature but possess one or two of these properties.

The only reason Emergence took the spotlight from all the other characteristics is that a more "Aristotlic" approach was taken and seeing that emergence perfectly demonstrates this line of thought, it was chosen as the first one to be presented, also in more detail than the others. Regarding this emergence of new information in networks, it will be shown, further down the chapter, that this property is the goal of creating networks.

3.2 Network Theory

Networks are structural models that take up the form of a web, or more generally speaking, a web-like structure [45]. They serve the purpose of describing and enunciating the relationships between several parts of the same archetype that fit into the same sphere of knowledge. Prior to the application of network theory, certain knowledge fields seemed like insurmountable in the sense that, the spacial and temporal complexity would be extremely high. These types of structures would point to the same answers as the usually applied algorithms (i.e random forests, clustering...), but the novelty was that it would also become possible to discover new types of relationships that otherwise would go by unnoticed.

In order to better understand how complex networks work, an introduction to the essential components that make up said networks is needed. These components come derived from the Graph Theory, which in itself is originated in a branch of discrete mathematics. The first mathematician to enunciate graphs was Leonhard Euler with the publication of a solution to the famous Königsberg bridge problem [46].

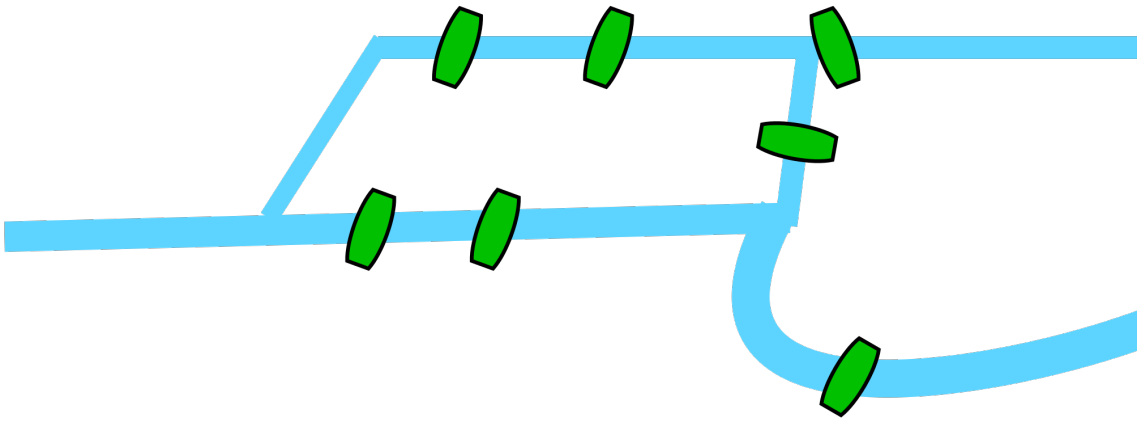


Figure 3.1: Image representing the Königsberg 7 bridge's problem. The objective is to design a route which passes through all seven bridges (green colored) once and only once.

Source: `https://www.mathsisfun.com/activity/seven-bridges-konigsberg.html`

As depicted in Figure 3.1, the problem revolves around the real world location of Königsberg in Prussia. The objective is to find a path, which is fundamentally different from a "way" in the sense that a path must only go through each destination once, that connects all the bridges. Leonhard Euler was the first to propose a solution to this problem by transforming each bridge and respective connecting path into a more simplistic approach - a Graph.

What is curious about this problem is that, the initial premises made the search for a solution impossible. Euler proved that, given any odd number, it is impossible to cross each bridge only once and visit every one only once. Therefore, the number of bridges must be an even number [47][48].

The objective of presenting this example is to stress even more the importance of graph theory. With possible solution or not, the problem was deemed solved with the aid of the newborn graph theory.

Whilst Graph Theory deals with the more mathematical side of the problem, Network Theory intends to model the problem and find useful insights from it. While graphs are abstract representations of the reality, meaning they do not always represent the truth about the world, networks approach the problem from

a more realistic point of view. They directly translate the mechanisms in place and then use graphs to analyze and deduct certain premises about the problem at hand. It is also important to mention that network theory enables the opportunity to study complex and dynamic structures such as complex systems as defined in section 3.1.

Such examples of network analysis are as follows [49]:

1. **Shortest Path Problem** - What is the shortest distance (weight wise) between any set of two nodes in a graph?

2. **Network Flow** - How does the network flow considering it's topology?

3. **Matching Problem** - Are there nodes, or pairs of nodes that make the connection to independent sets of nodes?

4. **Critical Path Analysis** - If the data present in the network is independent, or made up of independent features, which are the paths that contain a dependent nature?

Nevertheless, it is important to take a step back and discuss certain key concepts about Graph Theory in order to proceed to a more "high level" oriented approach such as network analysis. This being said, the following subsections deal with graph theory concepts.

3.3 Graph Theory

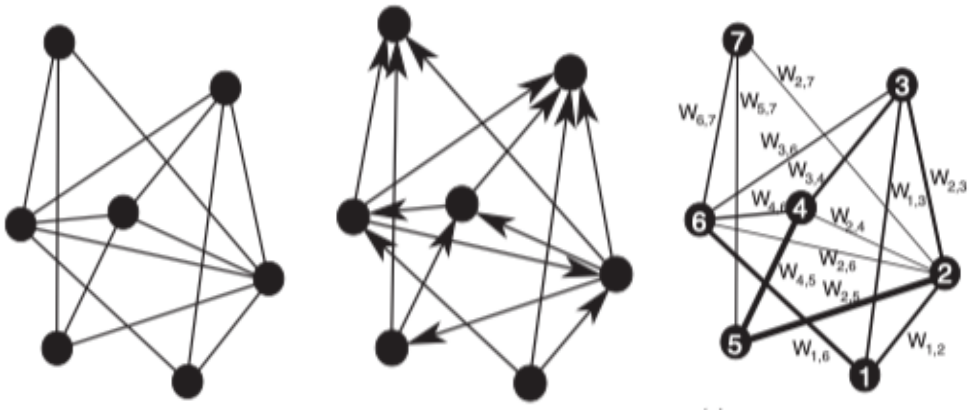
Graphs are conceptually made mathematical models which have the purpose of "graphing" certain types of information as nodes and edges. Nodes represent the entities or objects of the world whilst edges represent the relationships that said objects or particles display between one another. Despite, the collection of nodes and edges, being called a graph, and in the context of this dissertation, the title "Network" or as a matter of fact, "Social Network" is more correctly employed since in the study of these types os graphs or networks, the term "social" is usually applied due to the properties of the edges being called relationships.

A graph $G = (N, L)$ consists of two sets N and L, such that $N \neq \emptyset$ and L is a set of pairs of elements of N. The elements of $N = \{n_1, n_2, \ldots, n_N\}$ are the nodes (or vertices, or points) of the graph G, while the elements of $L = \{l_1, l_2, \ldots, l_K\}$ are its links (or edges, or lines). The number of elements in N and L are denoted by N and K, respectively so that $G(N, K) = (N, L)$ [34].

Now that the basic mathematical nomenclature for nodes and edges has been established, it is now possible to start discussing the three main types of graphs.

3.3.1 Types of Graphs

There are three main types of graphs. Undirected, directed and weighed. The first two are very similar in nature whilst the third one is fundamentally different on how it works and presents itself. Simply put, directed graphs are just like their close relatives, the undirected ones, but each link has a certain order by which the relationship guides itself. This usually means that the metrics for this type of graph are different as will be discussed further down. They also have topological differences as it is possible to observe arrows that display how the relationship is established.



(a) Undirected Graph (b) Directed Graph (c) Weighed Graph

Figure 3.2: Examples of three types of different graphs

Source: [34]

A node is usually referred to by its order i in the set N. In a undirected graph, each of the links is defined by a couple of nodes i and j, and is denoted as (i, j) or $l_i j$. The link is said to be incident in nodes i and j, or to join the two nodes; the two nodes i and j are referred to as the end-nodes of link (i, j). Two nodes joined by a link are referred to as adjacent or neighboring. In a directed graph, the order of the two nodes is important: $l_i j$ stands for a link from i to j, and $l_i j = l_j i$. The usual way to picture a graph is by drawing a dot for each node and joining two dots by a line if the two corresponding nodes are connected by a link. How these dots and lines are drawn is irrelevant, and the only thing that matters is which pairs of nodes form a link and which ones do not. Examples of a undirected graph

and of a directed graph, both with $N = 7$ and $K = 14$, are shown in Figures 3.2a and 3.2b, respectively [34].

The graph presented on Figure 3.2c is fundamentally different from the previous two examples. This type of graph possesses an additional information for each edge, and that is a certain weight. These weights usually serve their purpose has input information to feed an algorithm thought by Edsger Dijkstra, which is named after him - Dijkstra's Algorithm. This algorithm chooses the shortest path to the final destination taking into account the sum of the weights that make up the path. It iterates through multiple paths in order to choose the shortest one [50][51]. Whilst the first type of graph has a certain disorder associated to it (all the connections between vertices are dictated by the data), the other two follow a fundamental order to connect the vertices. Another way to observe this differences is to look at an API developed by Google named Google Maps. As it stands, and without any input from the users, it is possible to consider the overall map as an undirected network. Cities are connected by traffic ways and other types of roads. The next step in this line of thought is to think that to go from point A to point B you can have one, or multiple different paths. This is an example of a directed network. Multiple nodes can be connected but there will always be present a source node and a target (destiny) one. Finally, to perceive weighed networks, one can think about the end result of a Google Map's search. If someone chooses to go from A to B, despite having multiple ways or paths to complete the journey, most of the times, a person will chose the fastest one. This can be achieved because every path connecting the nodes has a certain weight associated, meaning it is possible to calculate the total amount of a certain metric, which in this case are distances/times.

3.4 Network Scale Measures

Strictly speaking, there are three main scales on network measures which come forth due to their size and respective influence. The following table 3.2 presents the scales and a brief definition.

This type of classification is not important in by itself but starts to show influence when certain types of metrics are enunciated. As will be explained in the next section, each scale of the network is expected to be evaluated in a different way from its peers as it does not make any sense to look at a network from a macro-scale point of view without first looking at its core properties present at the micro-scale. This being said, each type of metric tries to incorporate the

same network but at different class types. What this means, and looking at the examples from Table 3.2, is that a community of individuals can be analyzed from the connections a certain individual has up to how distant is a certain individual from another one in the network.

Table 3.2: Different types of scales regarding the study of complex networks [52]

Micro-Scale	This scale centers around the properties of a single connection (edge) or single hub (node). At the point when the entire network is investigated these measurements are generally found by averaging each hub contained inside the system, so as to make a global picture. Example: Number of associations an individual has in an interpersonal organization.
Meso-Scale	This is the transitional level between contemplating components of a framework independently (micro-scale) and thinking about the framework in general (macro-scale). It is possible to characterize the meso scale as any ordinary structure that influences the availability of clusters of hubs in the system. Example: Communities of individuals present inside an interpersonal organization.
Macro-Scale	The macro-scale portrays the structure all in all. It represents the general structure of the network, tending to the development and movement of data all through. Example: The average path length to contact an individual in an interpersonal community.

3.5 Structural vs. Functional Networks

Structural networks, also called physical, are described through explicit interactions. To give a direct analogy, it is possible to directly compare a physical network to that of the commercial aviation sector. The nodes represent the airports while the links between them represent the routes. This being said, however, some real-world systems lack explicit information about links. Supposing each element's dynamics is a function of its neighbors, the first step is to quantify these functions (hence the name functional networks) [52]. The result from this quantification is the network's topological features that can finally be extracted.

One such example of this is the case of the human brain. Obviously, the brain has an underlying physical structure that supports it (muscles, fibers ...) On the other hand, there is something that is metaphysical and that cannot be seen. One such example is the process of picking up a glass of water. There is clearly an actuation and therefore activation of certain areas of the brain that tell the human body how to pick up the glass, but the thought process behind everything is considered a functional network as it is not physically connected to the body.

This being said, dealing with functional networks involves working with three different parameters of the analysis process [52]:

1. **Connectivity metric**- The measure that quantifies the presence of a relationship between the dynamics of different elements.

2. **Threshold** - A value that subjects the process to an evaluation, bringing up the statistical importance of the results.

3. **Topological Metrics** - A metric that describes the structure of the resulting network.

In case there are no set rules, the three previously mentioned aspects (parameters) bring in a certain degree of subjectiveness to the problem. The threshold metric is of particular relevance as it tends to act as a regulator for the entirety of the dynamics process. It usually induces a certain value as a parameter for comparing certain values against. One clear example of the relevance of the threshold is the case of the ANN mentioned in subsection 2.2.3.2. On that particular case, once more, it induces a value into the system so that the resulting value of the neuron can be compared against. The connectivity metric can also be described in terms of the ANN as it directly stands for the weights each interconnecting path of the neuron has. On that instance, the weights act as regulators that induce a certain degree of importance to each link. The topological metrics tend to act as a way to assess the end result of the complex network. It evaluates if the topology created by the process is in accordance with what was expected or not.

This being said, there are additional steps that require some attention. One of such relevance is the filtering of certain links that are deemed unfit for the network. While sometimes these links might represent outliers of significant importance, most of the times they are the result of noisy measurements and therefore are considered invalid. While this is true, one must exert caution when deleting links from the network as it might cause unpredictable chain reactions [52].

3.6 Network Metrics

Having now the concepts of networks measures, the concept of network metrics can now be introduced. The types of metrics range from a wide variety of measurements that can be performed to the network itself. These depend on the type of network, scale and tool used to perform said operation.

Regarding scale, it is not possible to apply the same metrics to the entirety of the network. From a Macro perspective, node degree count would return values that would not correspond to the truth. If one wishes to assess the node

degree, firstly it is necessary to subdivide the network into different clusters or stratifications, the same goes for link density.

Seeing that both modularity and the motifs deal with clusters and the emergence of communities [53], it would be unwise to try and assess the network at the Micro scale. These clustering of nodes take form due to the high concentration of certain characteristics that can only be possible to discern at a more higher level of inference.

Finally, at the Macro scale, it is possible to retrieve some measurements that can also be of interest. At this scale, the measurements usually involve distances, being the most important ones the Geodesic distance and the computation of the network's diameter. These two types of measurements usually involve transversing large portions of the network (or even its entirety) hence the Macro scale. All of the previous information is summarized in table 3.3, which was partially extracted from [52]

Table 3.3: Some of the most relevant metrics regarding complex networks [52].

Metric	Scale	Description
Degree	Micro	Number of connections a node has, i.e. number of neighbors.
Link Density	Micro	Number of links in the network, divided by the number of links that could be present. Maximum link density means that all the nodes are connected to each other and vice-versa. [54]
Modularity	Meso	Assesses the presence of groups of nodes with a high degree of link concentration [55].
Motifs	Meso	Small structures (usually composed of three or four nodes) that subsequently appear in the network. Can also be identified as repetitive clusters [56].
Eccentricity	Macro	Maximum distance between a node and every other node in the network [57].
Diameter	Macro	The diameter of the network can be calculated by finding the longest shortest path between two nodes [58].
Geodesic distance	Macro	The geodesic distance between nodes is the shortest possible path that connects them [59].

Chapter 4

Data Engineering

The main focus of this chapter is to approach the data used as the focus of the study and its subsequent transformations. As it will be seen throughout this chapter, before loading the data onto the model, in order to obtain meaningful results, the data must be processed in a way that fits the objectives proposed. This being said, several approaches will be presented as different methods wield different results. There is also the mindfulness approach regarding the time and spacial complexity of the data, which will also be explored in this chapter.

4.1 Data Source

Before presenting the used data source, and therefore the one that was chosen, it is first important to refer what were the sources and types available and how they correlate with the focus of this dissertation, Complex Networks.

Since this is mainly a data mining problem, the common approach for data scraping would be the integration of data from a CSV file. The other possible approach, and the one that was used in the end, is scraping data from a relational database, which in this case is represent by an Oracle Database.

CSV files are one of the most used input structures for data mining projects, as was said before, due to their flexibility and broad range of manipulation techniques employed in two of the most used programming languages (Python and R)[60], also for data mining, as will be explained further down the chapter.

On the other hand, there exists the possibility to use "raw" data, in the sense

that is the most common formatting method employed in the industry as a whole, in the form of relational databases. The database, as mentioned before on 1.1, was presented by Holos. This database stems from the RIFT Insurance Tool, as it is the primary storage solution for this tool.

Having the possibility to chose from two fundamental different data sources, why was the relational model chosen over the CSV file? The short answer is because it was easier from a technological complexity perspective. As will be explained and explored in subsection 4.2, complex network software tools expect a certain data structure as their input, and the relational model contained in the Oracle database was already prepared structurally for this integration.

Before moving on to the exploration of the tools involved in the manipulation of the different data type sources, a more in-depth explanation of both these data structures will be presented on Annex I. This being said, it is also of relevance to go into the topic of the desired data structure that the software tools (for the creation of complex networks) expect.

4.2 Desired Structure

Having now two different types of data source inputs, before presenting the frameworks used, it is of relevance to first explain and evaluate the data structure that the final model, a Complex Network, expects from the previous step, which regards the ETL process.

As will be explained in the next chapter, the complex network software (that were tested as part of this dissertation) need two types of inputs in order to build the network itself. These are a Node input and an Edge input. Node files contain the information regarding each node in the network. They are optional, as their only purpose is to address information to each node. In the case of this dissertation, the node file, will contain information regarding each individual or collective group of people such as, nationality, type, description and name (more information about what each feature means on section 4.4).

Edge files are strictly mandatory by all of the software tested as they represent the connection between nodes in the network. Without this type of input, the construction of the network would not be possible. With this input alone, the software would be capable of constructing the network as there is a certain peculiarity about this file that enables this capability. Edge files have a strictly demand for two fields present on the data itself, a Source and a Target.

$$Source \rightarrow Target \tag{4.1}$$

As the names imply, the Source field represents the entity from which the relation stems from, and the Target represents which entity is affected by this relationship, thus forming a pair as explained on 3.3.

There is also the possibility to add additional fields like Weight, but manually doing so implies distorting the information of connectedness between entities. If this information is not already present on the data itself, it should not be added manually.

To sum up what was explained before, complex networks need a particular structure of the data fed and therefore, every transformation process that was done exclusively had this goal in mind.

This being said, it is now relevant to present the different frameworks used to achieve this goal, and which ones performed better than the others.

4.3 Used Frameworks

Having now two possible data sources, one being a CSV data file structure and the other being a relational database, which was chosen?

There were several factors that came into play on the decision, but in the end, one of the major factors that played a big part on the decision was the hardware bottleneck. As it stands, and working with large volumes of data, hardware plays a big role on dictating which methods should be chosen over the others.

This being said, there were several approaches that were taken in order to better understand which one would be selected. Before that, it is also relevant to describe some of the frameworks and technologies involved in the process.

Knowing that the focus of this dissertation is to find insurance fraud using data mining techniques, the choice of the language Python came naturally. Since this language is well served in packages for data handling and manipulation and also machine learning. This choice was one that was clearly more focused while working with CSV files, whilst, for handling relational databases, Oracle's solution, SQLDeveloper becomes the most relevant framework to work upon.

4.3.1 Python

As was said before, Python came as the obvious choice for handling CSV files. Having many packages for manipulating these kinds of files, the one of choice was Pandas.

Pandas receives a file as an input and promptly converts the file into a file structure known as a Dataframe. This structure possesses properties similar

to a table, having rows and columns. This type of structure enables the user to perform certain actions upon the Dataframe which are of interest in order to obtain the structure needed by the complex network software as described previously on section 4.2.

Pandas, which is an open source library created on top of the NumPy library [61], allows for fast analysis and data cleaning. Since it creates a data structure similar to that of an Excel file, it can be considered Python's approach to Microsoft's Office suite solution for dealing with tables. It allows, once more, the creating of structures, such as Dataframes and Series, for manipulating data.

A Series is an one-dimensional array whereas a Dataframe is a multi-dimensional one. The creation of Series and Dataframes can be done as follows:

Listing 4.1: Coding example of the creation of a Series in Python

```
1  import pandas as pd
2
3  example_series = pd.Series([1, 2, 3, 4], ['Lisbon', 'Porto', '
                                  Santarem', 'Algarve'])
```

Listing 4.2: Coding example of the creation of a Dataframe in Python

```
1  import pandas as pd
2
3  example_dataframe = pd.DataFrame([1, 2, 3, 4], ['Lisbon', 'Porto', '
                                  Santarem', 'Algarve'])
```

While the code depicted on Listing 4.1 outputs something like a dictionary, values and their respective keys, the code on Listing 4.2 outputs a tabular list of arrays. The following Figure 4.1, which is the output of Listings 4.2 and 4.1, presents such information.

	0
Lisbon	1
Porto	2
Santarem	3
Algarve	4

(a) Output of the code on 4.2

```
Lisbon      1
Porto       2
Santarem    3
Algarve     4
```

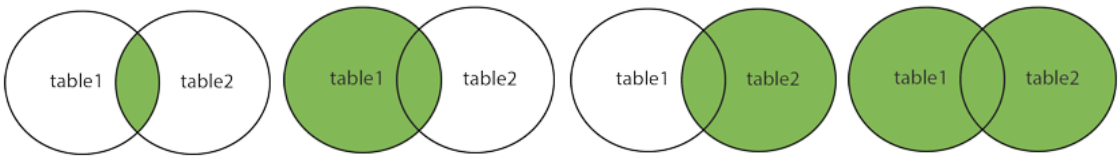
(b) Output of the code on 4.1

Figure 4.1: Examples of simple Dataframes (left) and Series (right)

Seeing both formats and having in mind that what is needed is a tabular format, Dataframes come as the approach which is the most suited for the problem.

This being said, Pandas also allow for the manipulation of Dataframes using an approach similar to that of the Structured Query Language (SQL) language. In order to join several files (that were read using Pandas and converted to Dataframes) or even "raw" Dataframes, there were 4 functions that had to be considered in order to make this process happen. These were:

1. Inner Join - Returns records that have matching values in both tables 4.2a, same as $A \cap B$
2. Left Join - Returns records from the left table plus the values that match both tables 4.2b, same as A
3. Right Join - Returns records from the right table plus the values that match both tables 4.2c, same as B
4. Full Outer Join - Returns all the records when there is a match on either table 4.2d, same as $A \cup B$



(a) (b) (c) (d)

Figure 4.3: Visual depiction, using Venn Diagrams, of the different kinds of join operations. From left to right; Inner Join 4.2a, Left Join 4.2b, Right Join 4.2c and Full Outer Join 4.2d

Source: `https://www.w3schools.com/sql/sql_join.asp`

These four types of structural operations played a big role on how the data was handled and therefore processed. Seeing that there were, at least three to four different kinds of inputs (tables on the database), the need to merge them arose in order to create the final Nodes and Edges inputs. All the work necessary is well documented on Annex II.1 in which is possible to observe that, most of the operations performed were those of the join kind.

After performing these operations, the end result should be two files: one named *Edges.csv* which possesses the structure mentioned on section 4.2, and the other named *Nodes.csv*, which contains additional information regarding each node.

4.3.2 SQLDeveloper

The other remaining viable approach (mainly to deal with relational databases) was to use a tool that allowed the extraction of data from a database, and a framework to "work", posteriorly, the data that resulted from the extraction. Seeing that there exist two different phases, Extraction and Transformation, the two different "tools" used were SQLDeveloper to deal with the extraction and Pentaho to deal with the transformation.

As the data is hosted on a remote server on a database, there is the need for an Application Programming Interface (API) to access said database, and fundamentally, a programming language in order to instruct the API to perform certain tasks (mainly retrieve information). Since the database is stored on a platform developed by Oracle, the tool of choice was the SQLDeveloper, which is also property of Oracle.

As the name implies, the objective of this tool is to manage and maintain databases using SQL as the basis for programming. Every table, and therefore every relationship, had to be created resorting to this query language.

Since this is a running database for a company (Holos, SA), all the information and structure that supports the insurance management tool (RIFT), are already in place, meaning that, no major querying had to be done. This being said, due to the flexibility of the API, it allowed for the creation of Views.

Views are, in their essence, tables that can be created resorting to pre-existing relationship models. This means that Views are a way to simplify, gather and coerce information that is, most of the times due to the logistics of the final tool, scattered across the relational model.

Views are also model agnostic. This means that they do not depend on a subadjacent model to work. They make use of tables that have already been created (tables depend on models). One can think about views like queries that have to be stored to be used recurrently.

Let there be the following example:

Formulation: Imagine that you want to gather all the information regarding a customer and their transactions. There exist two tables created for this purpose: the Customer table which has a primary key indicating the ID of the customer (Customer_ID) and a field which contains its name. The other table, promptly named Transactions contains an ID for each transaction and also a field which is a foreign key indicating the Customer_ID.

Solution: One way to approach this problem is to create a view (which will not permanently affect the database) in order to query and obtain the intended information, as seen in 4.3.

Listing 4.3: Coding example for creating a view answering the problem above

```
1 CREATE OR REPLACE FORCE VIEW "HOST_DB"."CHECK_TRANSACTIONS"
2 ("CUSTOMER\_ID", "TRANSACTION_ID", "CUSTOMER_NAME") AS
3 SELECT CUSTOMERS.CUSTOMER_ID FROM CUSTOMERS
4 SELECT CUSTOMERS.CUSTOMER_NAME FROM CUSTOMERS
5 SELECT TRANSACTIONS.TRANSACTION_ID FROM TRANSACTIONS WHERE
6 CUSTOMERS.CUSTOMER_ID == TRANSACTIONS.CUSTOMER_ID
```

In certain situations, views, can become useful when dealing with complex queries without compromising the structure of the database itself. Despite this fact and, being the objective the creation of the most intuitive framework possible, SQL and therefore SQLDeveloper still retain a certain degree of knowledge that the user has to have in order to perform the transformations.

It is important to refer that this tool, SQLDeveloper, was mainly used for Exploratory Data Analysis (EDA) purposes. This was due to the constraint of, once more, the relational database being hosted on an Oracle server. Views came into play early on as a way to summarize scattered information.

4.3.3 Pentaho Data Integration

PDI, also known as Kettle, came as an answer to reduce the knowledge needed to operate and perform decisions and operations on the database. This tool or framework was developed by Hitachi Ventara in order to streamline processes and reduce the learning curve for organizations to deal with (most of the times) complex interconnected relational models.

As the name implies, PDI is an integration tool. Instead of writing custom point-to-point integration code, this tool enables the ability to deal with environmental changes in data sources and updates which otherwise would require many hours of intensive labor to be dealt with. Using a legacy ETL mindset would not help with the volume of data that there is present in the world nowadays therefore, tools like this present new opportunities and advances on how the process of extracting, transforming and loading data functions. Holding this shift in paradigm closely to the need for more user-friendly approaches was one of the main motivations that upheld the rising of integration tools like kettle. Going from the premise of relational databases, this tool allows the extraction of

information from the database seamlessly which further backs up the goal of a expert-free knowledge approach.

Kettle presents two different ways to convey environments so the user can start doing some type of manipulation operations. The first one, entitled Transformations, serves its purpose as the main development environment. In this hub, a wide range of design options and approaches can be taken. Ranging from different types of input methods, going through different modifications and customization and finally presenting the option to output the result to different types of files.

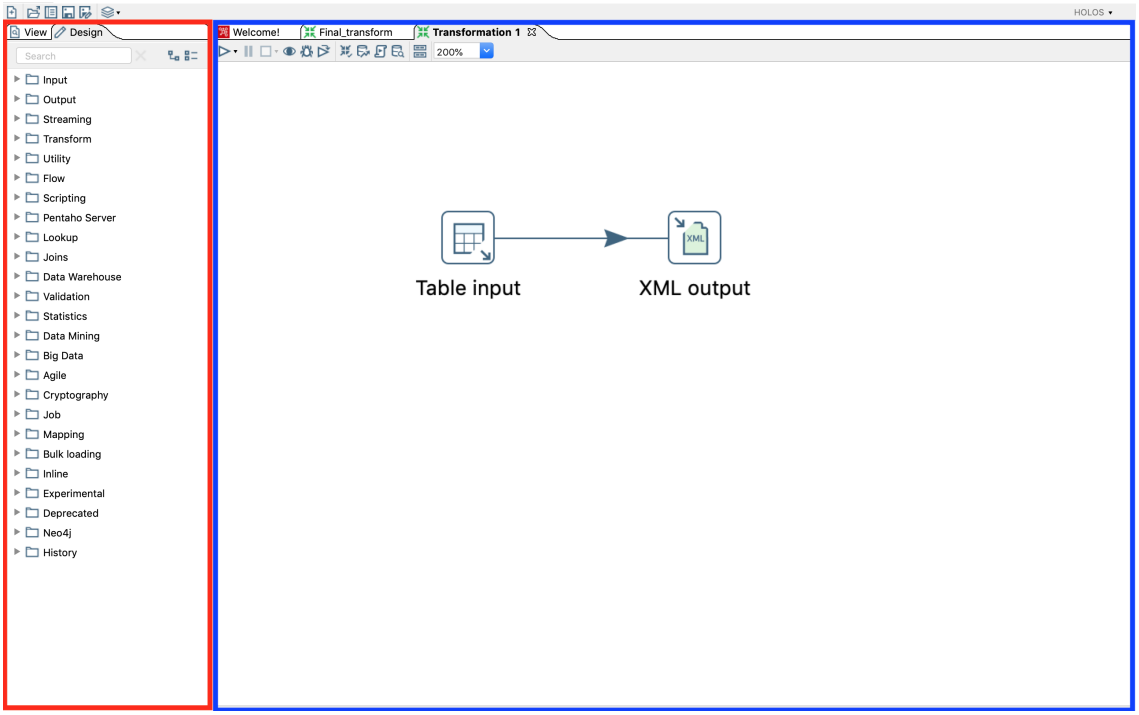


Figure 4.4: Simple transformation environment possessing an input (Table file) and an output (XML file)

Figure 4.4 presents a simple example of the main window of a transformation operation. In the area delimited by a red border there are the tools, subdivided into categories, which can be drag and dropped into the area delimited by a blue border, which is the main work area. After the elements are dragged into the blue area, they have to work with one another thus leading to the necessity of them being connected. How and why they are connected is left to the user's discretion. It is also important to refer that each element that is used in the workflow can have its parameters edited to better suit the user's needs.

The second type of environment is called a Job. Jobs are a more "high-level" design approach than transformations. They are usually used for making sure

processes are well integrated into the operating system (for example) and as a way to fail-proof the entire streamlined application. It is also of relevance to mention that jobs can invoke transformations as will be seen.

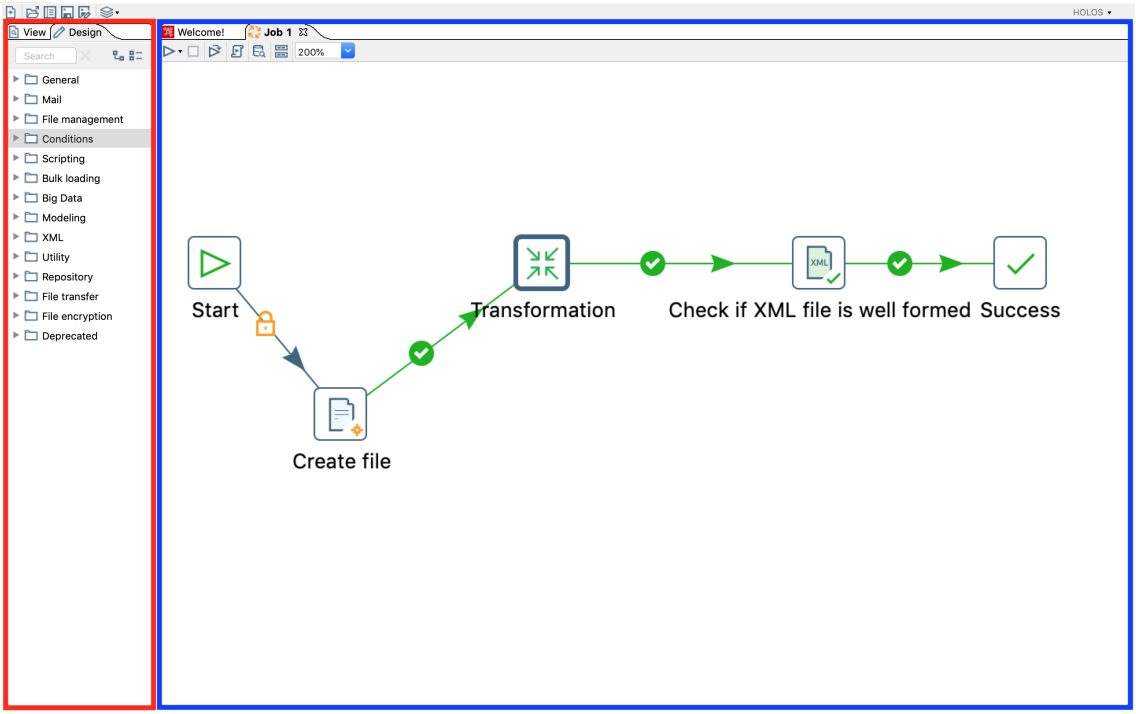


Figure 4.5: Simple job environment possessing a create file operation followed by a transformation. This transformation is then checked for the validity of its output file.

Once again, the red area marks the area from which elements can be chosen whilst the blue one is the work environment. Taking into account the transformation presented on Figure 4.4, it is now possible to transpose said transformation to a more "high-level" approach. This means that now, that transformation is incorporated in a final process ready to be deployed. As it is possible to observe in the figure above, firstly there is the creation of a file. This file, the way the flow is presented, is most likely the input file containing the information regarding the data that goes into the transformation. After that, there is a file check stage on which the output file of the transformation is checked (according some given metrics) in order to be deployed.

With this information in mind, the final transformation and job schematics can be found on Figures III.1 and III.2, respectively.

As was discussed before, the main objective of the ETL process, in this dissertation, is to obtain viable data structures that can be used for constructing

complex networks. This being said, the transformation schematic can be subdivided into two different objectives: the construction of an edges and nodes files. This process will be further deepened in the two following sections, 4.3.3.1 and 4.3.3.2 respectively.

4.3.3.1 Transformation - Edges File

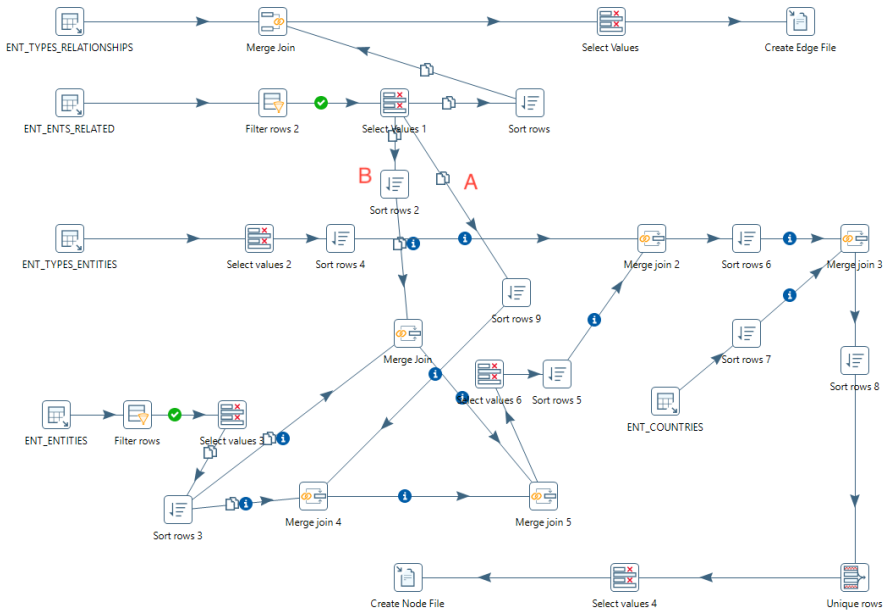


Figure 4.6: PDI schematic depicting the final transformation used for ETL purposes. It can be viewed further in-depth at III.1

Analyzing the topmost part of Figure 4.6, it is possible to observe the transformation process that creates a file containing the data necessary (and mandatory) to construct a complex network. Taking into consideration what was mentioned on 4.2, this file needs to contain at least two mandatory columns: a Source and a Target columns.

There are two main input elements that fetch data from the database (those on the leftmost side of the schematic). From a previous EDA that was performed, certain entities, whose purpose was to relate entities to the main insurance companies, had no statistical and model relevance and therefore needed to be removed, hence the need for a row filtering module. After that, the columns that did not show any relevance to the problem were removed using a select values element. Before merging the two different data streams, there is the need to use a sorting module in order to sort the rows. If this is not done previously to joining them, there might be "miss-joins". After that, another select values element was applied

in order to further clean the data structure from undesirable columns. Figure 4.7 depicts the process explained above.

The result is an *Edges.csv* file which will have its data explained in section 4.4.

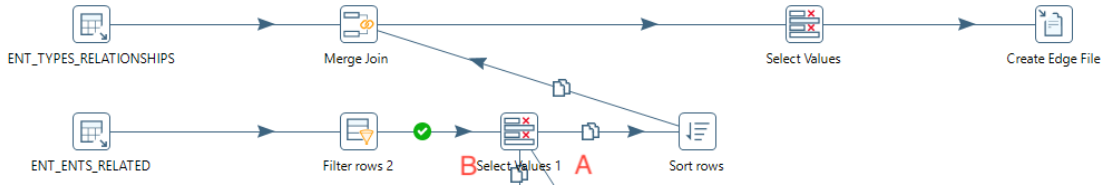


Figure 4.7: Transformation schematic that originates the *Edges.csv* file.

4.3.3.2 Transformation - Nodes File

Similarly to what was intended in 4.3.3.2, a similar process was needed for obtaining the *Nodes.csv* file. For this transformation, four input data streams were needed. One (which was also present in the previous transformation) serves as the bridge between the edges and nodes files (without this file it would not be possible to associate each node and relationship with their respective characteristics). The other three input streams are database tables which contained information deemed relevant for the study of the problem. The first table labeled *ENT_TYPES_ENTITIES* has the information regarding the type of connection (relationship) between entities, *ENT_ENTITIES* has additional information regarding each entity of the database and *ENT_COUNTRIES* has the conversion of the country codes to plain text. The data gathered from these input streams goes through a cleaning process which filters rows according to the information needed (eliminates irrelevant data) and eliminates unnecessary columns that do not add any relevance to the information gained. After this and before merging the tables, each data stream is then sorted (as explained in 4.3.3.1).

All this premises led to the need to merge all this files into one. All the operations and elements used in this transformation had the sole purpose of merging, sorting and cleaning the data structure, thus producing the *Nodes.csv* file. This was a necessary step seeing that the database is highly normalized and the information is not "concentrated" in one single table.

Despite visually intricate, as seen if Figure 4.8, if one were to reproduce the same output using only SQL code, it would take longer to develop and would be more error prone than this high-level approach

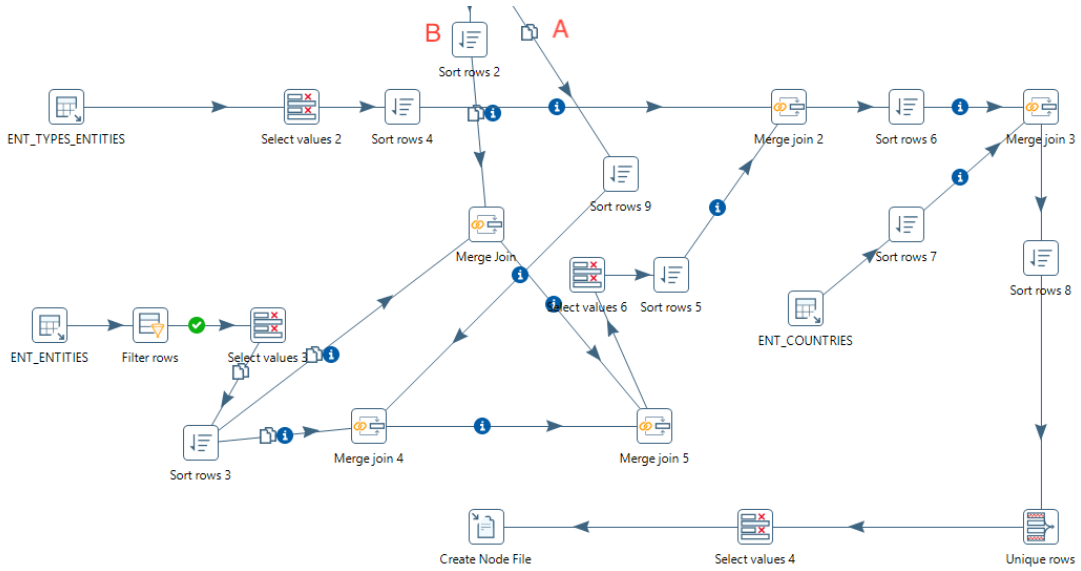


Figure 4.8: Transformation schematic that originates the *Nodes.csv* file.

4.3.3.3 Wrapping Up with Jobs

Now that the transformation is schematically complete, it is now possible to wrap it up using a job for a more streamlined and complete approach. The Figure III.2 present in the annex section presents such process in detail.

The process starts by checking if the data in the database is present or not. If not, the whole process fails whilst if the data is really present the transformation has green flag to start. After the transformation is finished it now stands at a "two-way road": it either finished successfully or unsuccessfully. In the first case, a mail is sent to the user reporting that all went well along with a file with logs from the transformation itself. The resulting files are also stored in a local partition, where they will be ready to be used in the next step (creating the complex network). In the event that the transformation did not finish with success, a folder is created in a local partition where a file will be created with logs detailing the happenings. At the same time, an email will also be sent to the user reporting the status of the process.

4.3.4 Comparison between Frameworks

Taking in all the processes and information that was exposed in the three previous subsections, 4.3.1, 4.3.2 and 4.3.3, the data source that ended up being selected was a relational database in conjunction with Pentaho. Not only it provided real-world data insights into the insurance world, but also possessed certain model typologies that greatly benefited the ingestion of data by the complex networks.

These typologies directly translate to how the data is organized in the database. As it is, the data structure is already in concordance with what is intended, as explained in section 4.2.

Despite this, both input types were digested and analyzed as a way to chose one over the other. As was said before, complexity, and therefore hardware, plays a big role on why some methods end up being chosen, so, this being said, a complexity comparison between both input methods, and therefore their respective framework tools is required.

Regarding the tool itself, Pentaho was chosen due to being the most user friendly approach between the three (Python, SQLDeveloper and Pentaho). Not only permitted the user to visually perform certain operations (like the ones proposed on Figure 4.3) but also the execution time was on par with directly querying the database (which is the fastest).

Also, and despite the use of a relational database for input purposes, the two files obtained from the transformation are those of the CSV category. This is due to the fact that all the network creation tools analyzed expect files from this extension to work with.

4.3.4.1 Complexity Comparison

Regarding complexity in itself, it is important to remind that the complex network software expects a certain structure as explained on 4.2.

Having this in mind, all the work that revolved around ETL methodologies had to have this kind of structure as the main purpose. As the data for the CSV input method, and therefore the use of Python, was scraped from the internet and had no structure whatsoever, the restructuring operations that took place converted a file that had the following properties:

Table 4.1: Description of the Insurance_claims.csv file used as a stress test to Python's capability to deal with large datasets

Features	Classes	Entries
33	1	11565

Into a massive structure that looks like as follows:

Table 4.2: Description of the Insurance_claims.csv file used as a stress test to Python's capability to deal with large datasets post transformation

Features	Classes	Entries
2	0	x

The way x is given takes into account the structure needed, meaning that from thirty-three features and one class, the file will possess only two features (Target and Source), no class (classes are mixed along with features when transforming the file) and x entries.

x is given by:

$$\frac{n!}{k! * (n-k)!} * y \tag{4.2}$$

Where:

- y is the number of entries in the dataset
- n is the number of features in the dataset
- k is the number of elements for each combination which, in this case, will always be $k = 2$

This results in about $x = 6.487.965$ entries.

With values of this scale, namely referring to n and k, the order for time and spatial complexity revolves around $O(k * 2^n)$ and $O(n * k)$ respectively. As one can imagine, these facts pose a real challenge regarding hardware capable of running these transformations in useful time.

The data coming from the database does not require the transformations as the ones applied to the file scrapped from the internet as the table of relationships between entities is already present. With a table of about 118.000 entries, the challenge of extracting and transforming data is relatively seamless.

One thing which is also worthy of mentioning is that the language itself, SQL, is intrinsically faster than Python to perform operations over tables nested in databases as the purpose for its creation was managing tables from the beginning.

4.4 Data File Structure and Analysis

Having now the files resulting from the transformation step which used PDI, it is now important to dissect each file, both the *Nodes.csv* and *Edges.csv*, and make sense of the variables present. and what they stand for. These two files will be the ones that get passed through to the software that produces the complex network that derives from the data itself.

4.4.1 File Dissection - *Edges.csv*

This file, as was expected, possesses a structure as the one described on section 4.2. It contains variables that act as a Source and Target along side other variables that give the two previous ones additional information. The variables are as follows:

- **ID_RELATION** - This variable identifies the relationship between the table's entities. It is represented by an integer with no defined boundaries (it does not start in a specific number nor does it end in one).
- **F_PRINCIPAL_ENTITY** - An integer number that identifies the Source node from where the relationship stems. It also does not possess numeric boundaries.
- **F_ENT_RELATED** - Similarly to the above variable, this one is also an integer but it identifies the Target node instead.
- **DESC_DIRECT** - String variable that identifies the type of relationship between nodes in the network.
- **DESC_INVERSE** - Much like the above variable but identifies the inverse relationship of the nodes in the network.

One of the most relevant variables for the objective of this dissertation is the relationship between nodes in the network. This relationship is given by the two variables named *DESC_DIRECT* and *DESC_INVERSE*. Table III.1 presents all the relationships that can be present in the data. All these relationships are the ones usually employed when an insurance policy is performed as they refer to what kind of connectedness type the present with each of the entities in the database.

4.4.2 File Dissection - *Nodes.csv*

This file presents some additional information to aggregate to the nodes in the network. Most of the information present in this file was deemed relevant to the problem as it brought critical information to the study and investigation of insurance frauds.

- **DESCRIPTION** - This variable handles the description of the entity. This description indicates what kind of entity it is. It is represented by a string.
- **ID_ENTITY** - This identifier is used to indicate the entity itself. This field is also used as a link (key) between the *Nodes.csv* and *Edges.csv* files. It is

represented by an integer with no defined boundaries (it does not start in a specific number not does it end in one).

- **NAME** - The name of the entity as it is identified in the database. Represented by a string.
- **ADDRESS** - The physical address of the entity which is also represented by a string.
- **DESCRIPTION_1** - Name of the country where the entity resides. Represented by a string.
- **COUNTRY** - Country code as per the standard ISO 3166-1 Alpha-2 codes [62]. It is a string made up of two letters that identify the country.

With the information present in this file it becomes possible to study, after the network is created, why certain entities are connected and evaluate empirically if those connections between entities are considered "normal" or strange.

Chapter 5

Complex Network Framework and Result Analysis

In this chapter the frameworks used for constructing Complex Networks will be explored. Three relevant software tools were used in order to accomplish the goal. Albeit this, only one of them was able to provide useful insights without suffering from hardware bottlenecks as the others did. Section 5.1.1 will address a Python's package that is extremely useful when creating networks whilst sections 5.1.2 and 5.1.3 will show a more automated approach to the problem. Finally, the results that stemmed from this study will be presented and discussed on 5.2.

5.1 Frameworks

Despite there being many tools, frameworks and software packages that can handle complex network, from plotting to evaluating certain metrics, very few can handle massive networks [63]. Firstly a definition of the word "massive" is needed. From all the tools tested during this dissertation, it is possible to conclude that massive is a word that can be applied to the data extracted and transformed from the database. Lastly, but not less important, hardware wise, it is also imperative to consider which tools can perform their jobs within acceptable times. This being said, this chapter will focus on the application of said tools.

5.1.1 Networkx

The first tool that was considered was Networkx. This is a package for Python that allows for the creation and extraction of information of complex networks using the after-mentioned language. From a Python's API it is possible to create graph objects, algorithms to analyze networks and even some rudimentary plotting capabilities.

As a way to exemplify the creation of a network using this package, the following code in Listing 5.1, presents a built-in data structure example, known as Zachary's Karate Club. This dataset originates from a study done by Wayne W.Zachary which researched an university's karate club as a way to discover the participants who interacted with one another outside the club [64].

Listing 5.1: Coding example of the creation of Zachary's Karate Club network using Networkx

```
1 import matplotlib.pyplot as plt
2 import networkx as nx
3
4 G = nx.karate_club_graph()
5 nx.draw_circular(G, with_labels=True)
6 plt.show()
```

From the above code piece, Networkx tries to plot using a circular layout the interaction model between participants. The result, as seen if Figure 5.1, is a graph of all the interactions between the participants of the study.

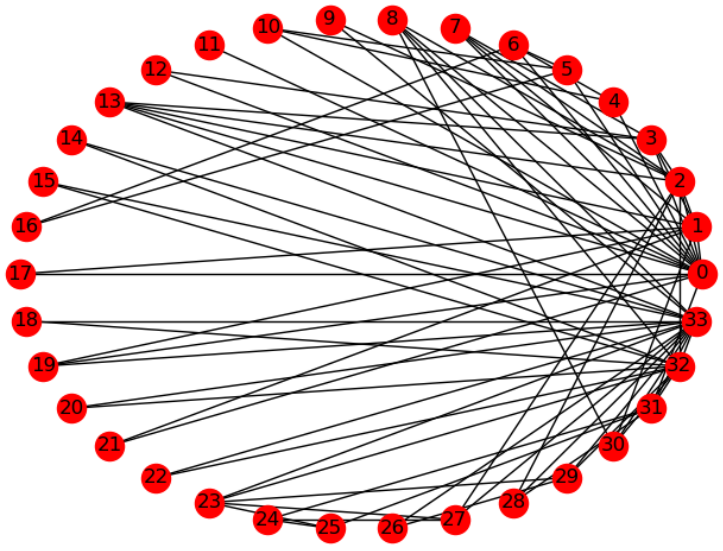


Figure 5.1: Karate's club network showing all the connections between elements from the club.

This is a great example of the power of graphs and therefore complex networks. It is intuitively easy to observe that the participants with labels on the rightmost part of the graph, clearly have more connections than those in the leftmost side.

Although this is an extremely visual tool, there are also some metrics that can be taken away from the network as a way to better understand the underlying structure of the graph. They are as follows:

- Radius - Returns the radius of the graph, counted in hops between nodes.
- Diameter - Returns the diameter of the graph, which is the distance between the two farthest nodes, counted in hops.
- Center - Returns a node or a list of nodes which have their radius equal to the eccentricity, which in turn is the maximum distance from a node to all other nodes.
- Periphery - Similarly to the above description, but this set of nodes has the eccentricity equal to the diameter.
- Density - This property infers about the connectivity of the graph. It returns a value between zero and 1 ($[0,1]$). The closer to zero it is, the less connected the network is and vice-versa. If it is equal to 0, no edges are present, if equals to one every node is connected to the other nodes.
- Transitivity - This returns a number between zero and one ($[0,1]$), which represents how likely the graph is to make complete triangles. Complete triangles refer to a concept that, if two people know the same person, it is likely that they know each other.

From the application of the previous metrics, the results were as follows:

Table 5.1: Results from the application of the metrics described to the Zachary's Karate Club Network

	Radius	Diameter	Center	Periphery	Density	Transitivity
Value	3	5	[0, 1, 2, 3, 8, 13, 19, 31]	[14, 15, 16, 18, 20, 22, 23, 26, 29]	0.1390	0.2556

From this results, it is interesting that they confirm what is possible to observe visually. Regarding center, the vector of values present in table 5.1 indicates that the nodes labeled $[0,1,2,3,8,13,19,31]$ are the ones at the core of the network, meaning that they are the ones which possess more connections. In contrast,

nodes labeled [14, 15, 16, 18, 20, 22, 23, 26, 29] are the ones which possess the least amount of connections.

Although the focus of this dissertation is detecting insurance fraud using graphs and therefore complex networks, it was not possible to create such a network from the insurance data. The sheer volume is on a scale at which the Networkx package is not able to handle the creation and thus the extraction of metrics. This lead to the exposition of Zachary's Karate Club as an example of the use of this type of Python package. This being said, this approach was abandoned in detriment of other tools that could handle large volumes of data and that would allow for a more scalable approach.

5.1.2 Gephi

Gephi is an open-source software for network analysis and visualization developed by the University of Technology of Compiègne [65]. Contrasting with Networkx, Gephi does not need any coding experience to plot graphs, albeit, usually the input file should receive some type of manipulation derived from coding APIs.

This being said, despite Gephi being a very complete tool for dealing with graphs, it still suffered from performance issues when plotting the insurance network as will be explained.

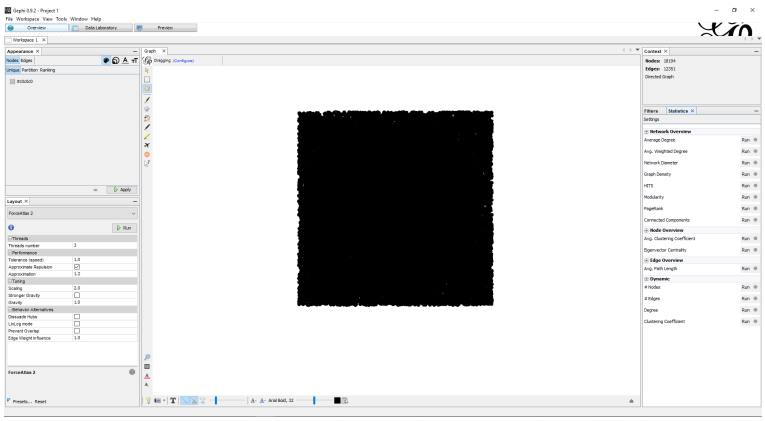


Figure 5.2: Gephi's interface window showing the graph regarding the insurance network.

As it is possible to observe in Figure 5.2, Gephi, by default, chooses a layout named "Contraction", which means that all the nodes are converging to the same center. With this kind of layout it becomes unrealistic to try and discern some kind of patterns or outliers.

With this in mind, Gephi offers a variety of layouts that the user can choose from. One of the most interesting ones is called "Force Atlas". This layout algorithm tries to emulate the force of gravity and the attraction-repulsion pairs. In practice, this means that nodes with higher concentration of neighbors should be at a more central position than nodes without as many connections.

The biggest challenge comes when applying this algorithm. As it needs to calculate many interactions between nodes depending on their number of connections, it turned out that Gephi could not handle a network this size. After the algorithm ran for about forty plus hours, it showed no sign of stabilization.

Instead, and as a way to exemplify what Force Atlas looks like, the Quaker's Network dataset was used. This dataset originated from a study in England about eighteenth-century commerce relationships [66].

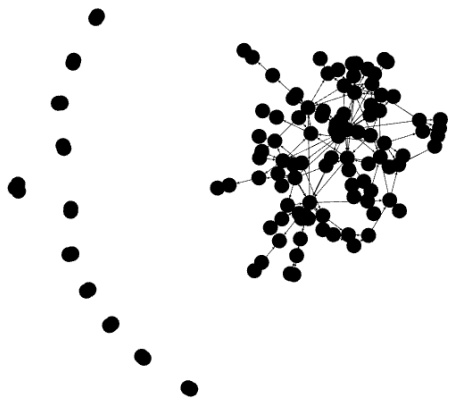


Figure 5.3: Force Atlas algorithm applied to the Quaker Network data.

As it is possible to observe on Figure 5.3, the nodes with the highest concentration of edges between them formed a cluster whilst the nodes with two or three connections started to orbitate the cluster.

Despite not being able to present a layout that can be "read", there is still the possibility to retrieve information from the network using Gephi. These types of information do not depend on the layout. This being said, the metrics extracted from the network will be presented on the next section (5.1.3), as the metrics of the network evaluated on one software framework, must be equal independently of the software.

5.1.3 Cytoscape

The answer for analyzing a big and complex structure like the one from the insurance database, came from an unexpected source. Cytoscape was first developed as an open source bio-informatics software platform with the intent of analyzing and visualizing molecular and gene interaction networks [67]. This being said, Cytoscape has a scalability that allows for the handling of nodes and edges with a magnitude of a hundred thousand [68].

Cytoscape allows for the input digestion of CSV files, therefore it is possible to use the *Edges.csv* and *Nodes.csv* files without any setbacks. Firstly, de edges file must be loaded. This is a mandatory file, and without it, it would not be possible to construct the network. Cytoscape assumes that, if there is a connection between *A* and *B*, these nodes must exist, therefore it creates them. After the edges file is loaded, the nodes file is also loaded as a way to add additional information to the already created nodes. All the information about these two files is presented in 4.4.1 and 4.4.2.

After the creation of the network, it is possible to make certain arrangements to increase the readability of the network. To achieve this, Cytoscape offers many options in the layout section of the network.

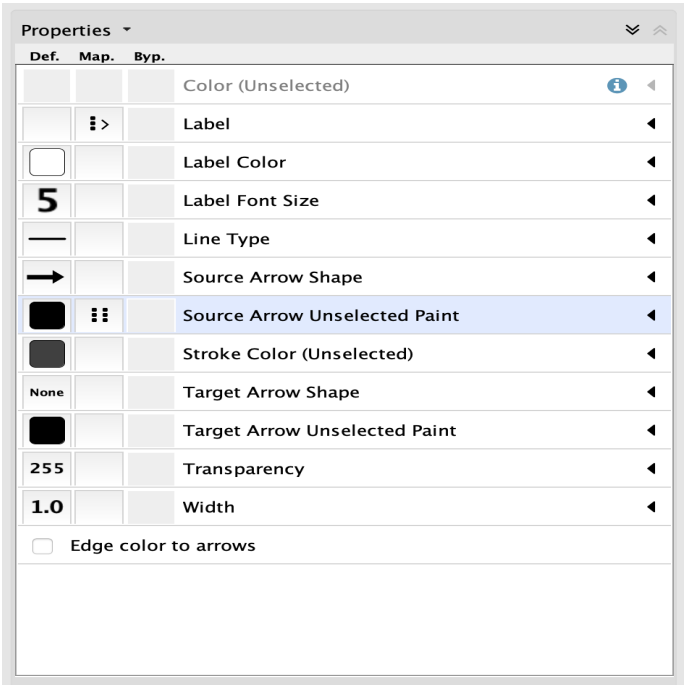


Figure 5.4: Properties menu for the edges of the network.

Figure 5.4 presents the menu with the fields that are subject to change. One of the most relevant is the "Source Arrow Unselected Paint". This field allows to

change the color of the edges between nodes according to some specific criterion the user wishes. In this case, it was opted to attribute different colors depending on the type of relationship between entities. The most prominent option is to make a discrete mapping operation in order to have a good separation of RGB values. The final mapping procedure is as follows:

Table 5.2: Color coding of the relationships present in the network. Note that the color schema may differ from the one used in the software as the color spaces of RGB codes may also differ.

Relationship Type	Color Code
has as customer	
has as employee	
has as mediator	
has as broker	
has as dependent	
has as contact entity	
is wife/husband of	
is the company to which it belongs	
is mother/father of	

Having now the edges mapping accordingly what was presented in table 5.2, the nodes would also be more visually informative if they possessed some kind of color mapping. This being said, the exact same procedure was applied to the nodes and the color schema for the nodes is as follows in table 5.3:

Table 5.3: Color coding of the nodes present in the network. Note that the color schema may differ from the one used in the software as the color spaces of RGB codes may also differ.

Node Type	Color Code
Company	
Employee	
Individual	
Others	

The resulting network is depicted in Figure IV.1. As the network is massive in scale, it is very difficult to perceive visually the underlying information about the entities and the relationships. This being the case, the next section (5.2) will delve on specif properties of the network as a means to extract results.

One particular feature of the Cytoscape's ecosystem, is the ability to export the network as a web interface. This allows for the opportunity to integrate the

results of the network cross-platform using web-apps, thus leaving behind the software dependencies that were in place.

Along the ability to change visual styles and layout properties from this interface, as seen in Figure 5.5

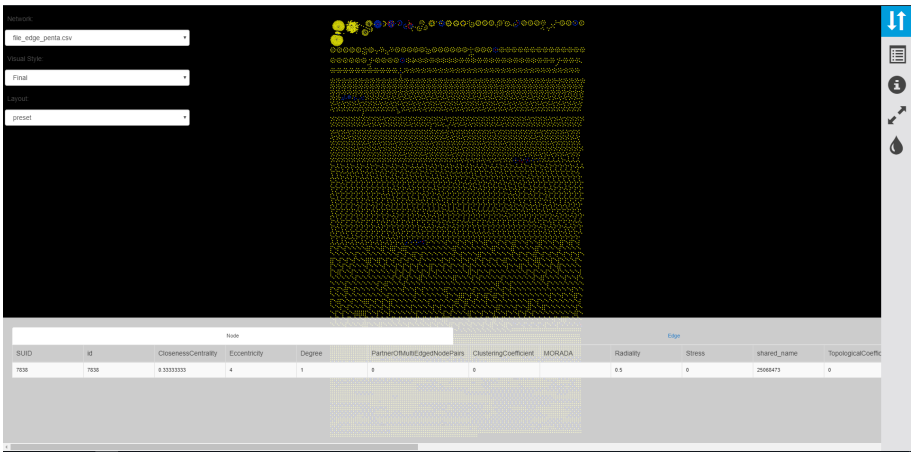


Figure 5.5: Web interface exported from Cytoscape.

Along the ability to change visual styles and layout properties from this interface, as seen in Figure 5.5, there is also the option to drag nodes and edges to better suit the visualization the user desires. This being said, one of the most import features is definitely the table menu that can show the properties of nodes and edges alongside some metrics defined by the software.

5.1.4 Overview on the Frameworks Used

Having now explored three solutions to create complex networks, it is important to retain why Cytoscape was chosen over the other two. Whilst Networkx is a great tool to create networks from scratch and possesses total freedom when extracting information about them, it suffered greatly from a network as big as the insurance one. It became impossible to render the network and even to extract more complex metrics would prove challenging. Without observing the network itself, the focus of the study would be rendered useless, seeing that complex networks have the feature of being great at presenting the information visually.

Gephi came as a pleasant surprised as it was able to plot the graph of the network whilst Networkx could not do this. It would also be possible to extract information from the network as well. Despite this, Gephi pre-defined the layout of the network as a Contraction, thus leading to the apparent square in Figure 5.2. Despite looking like a square, in fact it is not. It is a giant pile of nodes that are so packed together that it is not possible to discern any useful information.

To counter this, Gephi offers a variety of layout options to better understand the network. Despite this fact, these layouts were not possible to be applied since all of them need to perform algorithmic calculations and, once more, the size of the network was the major decisive factor.

Cytoscape was undoubtedly the best tool used for extracting results of the insurance's complex network. Performance-wise, there was an abysmal difference between Cytoscape and the other tools. Since it was created for handling large networks of molecular structures, it also handled the insurance's network pretty well. Visualization-wise, it was also the best among the three, providing many layout options and actually being able to apply them.

Bellow, on table 5.4, there is a brief and summarized comparison between the three tools on topics that are of importance when choosing a software for handling complex networks.

Table 5.4: Comparison between the different frameworks for complex network creation. Key: ▲ = good; ◆ = fair; ▼ = poor.

Characteristic	**Networkx**	**Gephi**	**Cytoscape**
Ease of use	◆	▲	▲
Ability to scale	▼▼	▼▼	▲
Visualization options and performance	▼	◆	▲
Metric extraction	▲	◆	◆
Integration with other platforms	▲	▼	▲

5.2 Results and Discussion

Having a tool that is capable of producing the network as well as an insightful visualization, it is now possible to extract information and begin to address the problem of insurance fraud.

Firstly, and before exploring the network, it is important to address the search for fraud in unsupervised data. Being unsupervised means that there are no labels, classes or categories in the data [69], making it harder to detect fraud than the likes of a supervised dataset (has labels, classes or categories). This means that there will never be a definitive answer (100% chance) for what is fraud or what it is not. The best answer one can give, is the possibility that some entity is committing fraud and it should be investigated further using mechanisms of the insurance industry (audits).

Looking at the network as a whole, it is possible to ascertain that its layout is

organized by an hierarchical order. This means that the nodes with fewer connections are at the bottom of the network whilst the ones with the most connections take their place above. Since the layout presents itself like this, it is possible to make a stratification of the network, resulting in having multiple layers separated by asymptotes whose values are the number of nodes, hence the stratifications by node degree (number of connections).

With this in mind, all the visualizations will be made possible because Cytoscape provides a filtering feature that the user can apply. These filters range from filtering the network by node attributes to edge attributes.

The final statement that should be made about this network is the fact that, almost all the entities in this network, roughly 99.97%, are from Angola. This is relevant because when looking at socioeconomic data from a certain country, it is important to address the data along that country's cultural paradigm. This will become apparent when addressing matters like high rates of fertility and polygamy.

To contextualize, the fertility rate in Angola is 5.61 in 2019, which means each woman has as an average of 5.61 children [70]. Regarding polygamy, despite not being legal, it is widely socially acceptable [71]. Both these facts, support many of the claims that will be made during the next sub-sections when evaluating relationships.

5.2.1 First Stratification (Node Degree = 1)

Figure IV.2 depicts a portion of the network that has been filtered according to the premise $NodeDegree = 1$, meaning that only one-to-one connections are present. From this sub-network, it now becomes easier to evaluate it for outliers that can, eventually, be present. Firstly, it would be useful to make a statistical census of the type of nodes and edges present in this stratification.

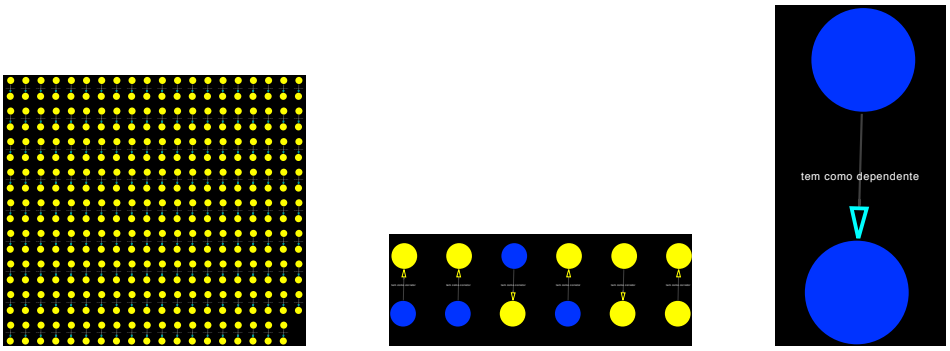
Table 5.5: The number of entities and connections present in the first stratification.

	Nodes			**Edges**						
	Individual	Company	Employee	Wife/Husband	Contact Entity	Mother/Father	Client	Employee	Broker	Dependent
	5617	1313	2	179	2518	692	1	6	6	1
Total	6932			3403						

From table 5.5, all the elements from the sub-network are accounted for.

Taking this information into consideration and, performing both a metrical and visual analysis, it is possible to conclude that this stratification does not

present many chances for fraud induced relationships. Taking the nodes connected by *Wife/Husband* connections as an example, all the nodes that are bound by this type of connection are labeled as *Individual*, which in itself is what is expected, as seen in Figure 5.6a.



(a) Marital Connections (b) Contact entities (c) Dependent of

Figure 5.6: From left to right, a case where every relationship is deemed normal, two abnormal connections and a strange relationship

Figures 5.6b and 5.6c, both present some strange relationships being formed. Whilst the a contact entity is a normal relationship between a company and an individual (the person whose contact is to be reached in order to address the company), it is strange for a person to be another person's contact entity. Regarding the last figure, the relationship present between the two nodes labeled as *Company*, is that of dependency. Between two companies, and taking into consideration other relationships between companies present in the network, is abnormal. This is a clear case of a relationship that should be further investigated by the proper audit entities.

Every other relationship from this stratification does not present itself out of place. As this stratification only shows $1to1$ relationships, it is relatively easy to detect cases that seem strange (outliers).

5.2.2 Second Stratification (Node Degree = 2)

For the stratification of the network using this type of node degree (and for every other one after bigger than 2), there is another filtering option that must be applied. Eccentricity, as explained in 3.6, regards the maximum distance between a node and every other node in the network. Instead of applying this filter to the entire network, it should be applied to the cluster of nodes being studied. With this, it is possible to eliminate clusters that have node degree equal to two and are

a sub-cluster from another cluster. This stratification can be observed in Figure IV.3.

Table 5.6: The number of entities and connections present in the second stratification.

	Nodes			**Edges**						
	Individual	Company	Employee	Wife/Husband	Contact Entity	Mother/Father	Client	Employee	Broker	Company which belongs
	2809	52	1	387	69	1421	1	17	12	1
Total	2862			1908						

As it is possible to observe, from table 5.6, the number of nodes dropped drastically. This is due to the topology of the insurance's network. As it is more common to have single connections (i.e relationships of parenthood or marriage) than other types of relationships. This being said, the relationships of parenthood still constitute a major portion of this stratification's edges.

Stating that the probability of not having deviant behavior in this portion of edges and nodes, it is now time to evaluate the behaviors of the smaller quantities of relationships and entities. This being said, the relationship of company ownership (*Company which belongs*), which is singular in this stratification, does not pose much concert seeing that is connecting two companies. This can be due to the fact that one company owns the other (partially or in totality). Lastly, the *Client* relationship is between a company and an individual, which also does not seem out of place, thus not needing further investigation.

Interestingly, in this stratification, it is possible to start observing the socioeconomic differences between countries. Figure 5.7 shows three entities related by connections of *Wife/Husband*. This being said, it is not possible to know if the people are of the female or male sex as the data was previously anonymized. Other than that, this stratification does not show any structural or statistical signs of outliers.

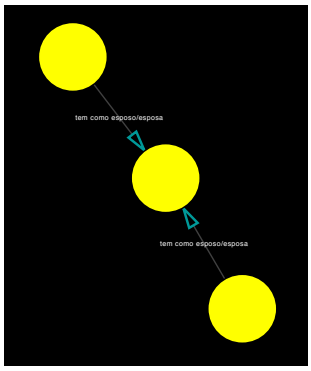


Figure 5.7: Relationship between entities showing signs of polygamy.

5.2.3 Third Stratification (Node Degree = 3)

Figure IV.4 depicts the resulting sub-network from the stratification using the premise $NodeDegree = 3$. As a result, table 5.7 presents the statistical census from the portion of the network being evaluated.

Table 5.7: The number of entities and connections present in the third stratification

	Nodes			Edges				
	Individual	Company	Employee	Wife/Husband	Contact Entity	Mother/Father	Dependent	Company which belongs
	2898	9	1	467	11	1701	1	1
Total	2908			2181				

This portion of the overall network is one of the most uneventful ones. This is due to the fact that the majority of connections and entities are related by family bounds. Most of the connections present have the label *Mother/Father* and *Wife/Husband* which also translate to a great majority of the nodes being labeled as *Individual*.

Looking at the other entities that there are in the sub-network, there are not cases that raise suspicion. Despite this, there are some interesting relationships to point out.

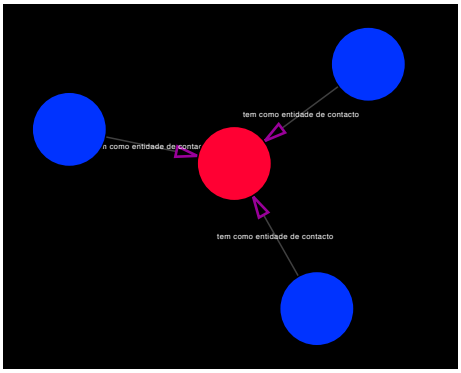
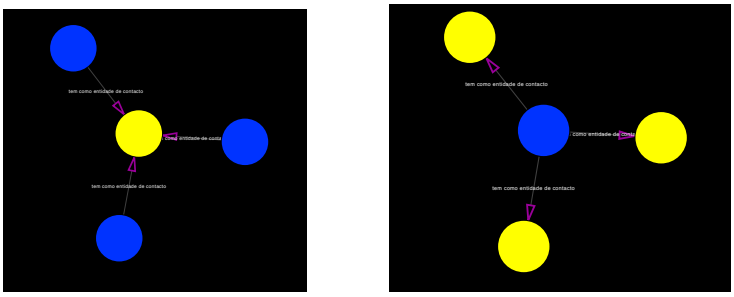


Figure 5.8: Three companies relating to the same employee as contact entity.

Figure 5.8 shows three companies that relate to the same entity with the label *Employee*. This entity marked with red, is an employee from an insurance company which is the contact entity for the three surrounding companies. This most likely means that the person who performed the insurance policies for those companies was the same. This situation is rather common and it does not raise any red flags that need further investigation.

There are two other situations worth mentioning. As seen in figures 5.9a and 5.9b, which in fact are the opposite of one another, it is possible to deduce, at

first sight, that these might be abnormal connections. Contact entities should be either employees from the company or employees from the insurance company. As there is not the label of company employee, it is possible to assume that the label *Individual* takes its place.



(a) Three companies with the same individual as contact entity

(b) One company with three individuals as contact entities

Figure 5.9: From left to right, three companies which have the same entity labeled as *Individual* as contact entity, one company with three individuals as contact entities

The first figure depicts a case where three companies have the same *Individual* entity as a contact entity, whereas the second one depicts a company having three individuals as contact entities. One possible explanation for this case is the fact that larger companies need several more people to perform the same roles.

Despite this, and as was said in the beginning of this section, this sub-network is "well-behaved" in the sense that there are no situations that arise suspicion.

5.2.4 Fourth Stratification (Node Degree = 4)

Figure IV.5 shows the layout of the network that resulted from stratifying the entire network according to the premise $NodeDegree = 4$. As a result, table 5.8 presents the statistical census from the portion of the network being evaluated.

Table 5.8: The number of entities and connections present in the fourth stratification

	Nodes			Edges			
	Individual	Company	Employee	Has Employee	Contact Entity	Wife/Husband	Mother/Father
	2561	11	3	7	14	435	1606
Total	2575			2062			

Since many of the nodes present in this portion of the network are labeled as *Individual*, and their relationships are those of *Wife/Husband* and *Mother/Father*,

these will be skipped in a first instance seeing that they represent behaviors that are deemed normal. This being said, within this stratification (and the ones to come), nodes start to arrange themselves according to a law of physics. This is Hook's Law.

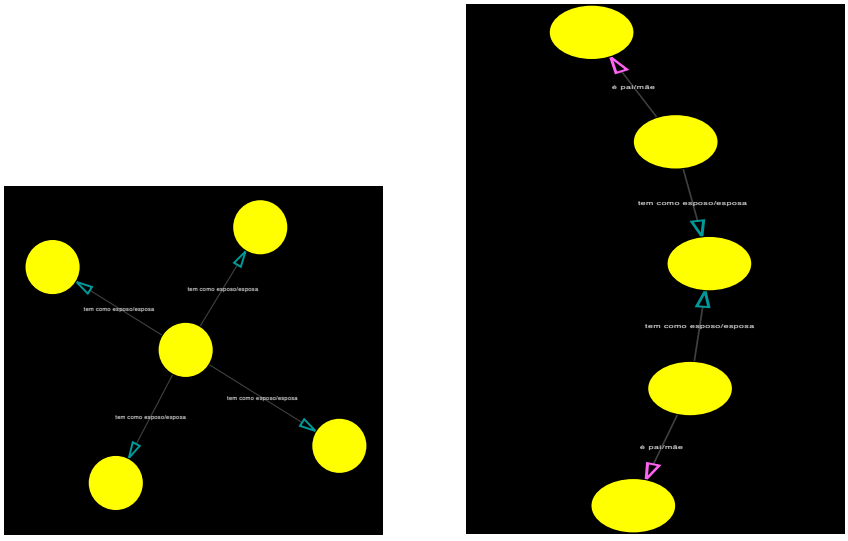
Hook's law presents a mathematical approach to explain how springs behave when applied a force to them [72], as follows:

$$F_s = kx \tag{5.1}$$

Where:

- **k** - is a constant value associated with the spring itself
- **x** - is the deformation of the spring (distance traveled)

To put it into networking concepts, the nodes are arranged according to this law. In practical terms, it is possible to compare Hook's law to the force of gravity. Nodes are expected to be disposed in a certain way around a central node (or a group of central nodes), usually orbiting the center, and if by any chance the structure of the cluster does not have an elliptical shape, it could be considered a deviant cluster. For example, Figure 5.10a shows a seemingly normal cluster whilst Figure 5.10b depicts a deviant cluster.



(a) Normal Behavior (b) Deviant Behavior

Figure 5.10: On the left figure, a case where the nodes present a layout which is to be expect whilst on the right, the layout has an abnormal structure.

In this case, the deviant cluster shows a central person which has two marital relationships, which in their turn have children. It is not certain if this is a normal

behavior or not, but in case of uncertainty, the best course of action is to always investigate the case.

Following this train of thought, there is another case of a strange layout in this stratification. This is also a case which only contain *Individual* entities which have family connections between them.

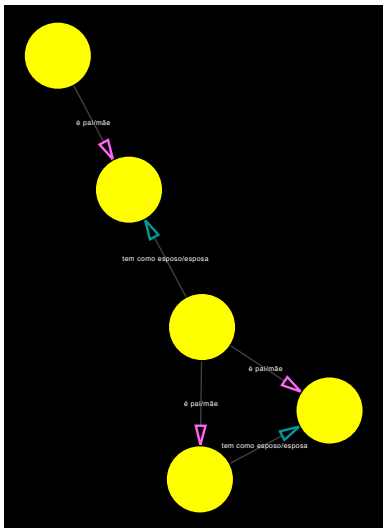


Figure 5.11: Relationships depicting a deviant layout behavior.

In Figure 5.11, it is possible to observe a relationship that is rather strange. Two individuals have a marital connection between them (which is normal) but, the two children of one of these entities are married to each other. This is by no means a normal connection and should be further investigated.

Having now skimmed over the relationships between entities labeled as *Individuals*, it is now relevant to observe the other types of entities which are in less number in the network. Despite all of them having expected connections, there is one case which rises some doubts. This is not due to the structure of the cluster, but from the connections.

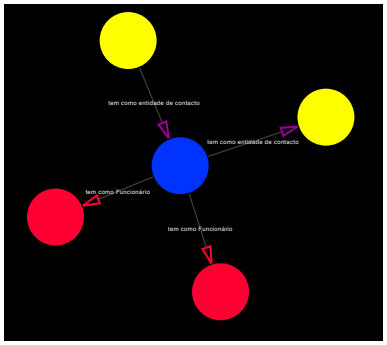


Figure 5.12: Depiction of abnormal relationships between a company and its employees.

Figure 5.12 depicts such case. The company (marked as blue) has two connections of *Employee* to two entities labeled *Employee*. Now, the nodes marked with the color red are employees from a certain insurance company, which by definition is not the company marked in blue, as the color blue is reserved for companies other than those that of the insurance industry. Succinctly, the blue company can not have as employees the ones that are from an insurance company therefore, making this cluster a major red flag that should be investigated immediately.

Other than the relationships and entities mentioned above, there is no other aspect about this stratification that is worth mentioning.

5.2.5 Fifth Stratification (Node Degree = 5)

Figure IV.6 shows the layout of the network that resulted from stratifying the entire network according to the premise $NodeDegree = 5$. Since this sub-network is, once more, greatly dominated by *Individual* nodes and, *Mother/Father* and *Wife/Husband* relationships, it is possible to assume that all of the connections between these entities using these relationships are quite normal. As per usual, table 5.9 presents a statistical census of the entities and relationships present in the sub-network.

Table 5.9: The number of entities and connections present in the fifth stratification

	Nodes		Edges			
	Individual	Company	Company which belongs	Contact Entity	Wife/Husband	Mother/Father
	1013	19	23	10	158	683
Total	1032		874			

Of course it is also important to make a visual analysis for clusters presenting strange cluster forms. Taking this into account, there is only one case, from the *Individual* node collection, which is of interest. All the other clusters of yellow nodes in the sub-network follow a well-defined structure with no visible anomalies, and also the relationships between nodes in the same cluster are what is expected.

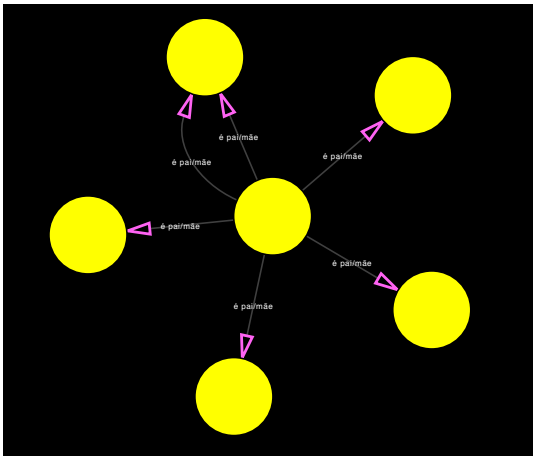


Figure 5.13: Depiction of two similar connections between the same entities

Figure 5.13 depicts just that. This cluster represents the relationships of parenthood between one entity and other five. This being said, there are two similar connections between two of the entities. Despite this not being the strangest of the cases, one possible explanation for this happening is that the *Mother/Father* might have renewed the insurance policy for that child. Other than that, no other relevant connections between *Individual* entities rises any suspicions.

After this, it is important to look at the entities labeled as *Company*. From this, there are two worrying cases that should be promptly investigated. Both cases are similar in nature so, only one will be presented and discussed.

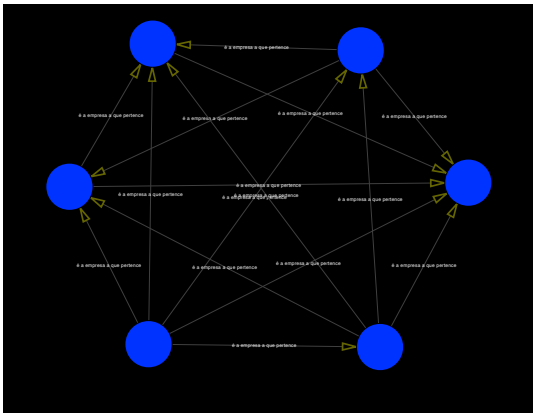


Figure 5.14: Unusual ownership ring between six different companies

Figure 5.14 depicts the unusual ownership ring that takes form between six different companies. Despite the fact that it is not that unusual for companies to own other smaller companies, what is not common is for those smaller companies to own one-another as present in the above mentioned figure. All the relationships present in the figure are labeled *Company which belongs* which indicates that a

company owns the other. If the structure was more like the one from clusters formed by parenthood relationships (one central node with the others spawning around it), it would not cause much suspicion, but with a structure like the one depicted in the figure, it is worth investigating further.

5.2.6 Sixth Stratification (Node Degree = 6)

Figure IV.7 depicts the resulting sub-network from the stratification using the premise $NodeDegree = 6$. As a result, table 5.10 presents the statistical census from the portion of the network being evaluated.

Once more, this stratification is predominantly dominated by *Individual* nodes and relationships of parenthood. This being said, it would be advised to observe the structure of the clusters, since the node degree has reached a point at which is possible to detect more easily changes in layout.

Table 5.10: The number of entities and connections present in the sixth stratification

	Nodes		Edges			
	Individual	Company	Employee	Contact Entity	Wife/Husband	Mother/Father
	286	1	3	3	34	209
Total	287		249			

This being said, there are three clusters that present changes in topology, two whose relationships do not qualify them for suspicious analysis whilst the other one might incur in doubt. Figure 5.15 depicts said situation.

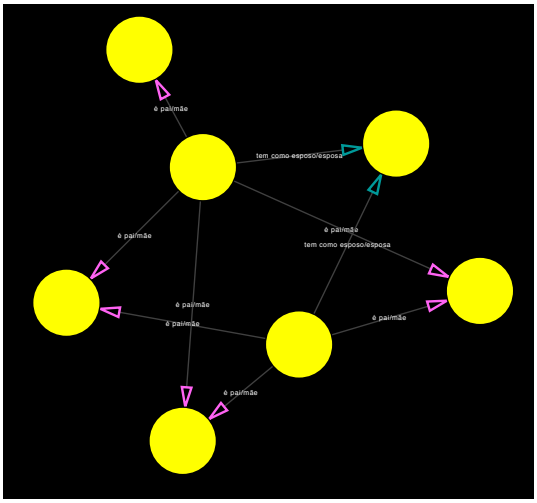


Figure 5.15: Unusual cluster formed from interpersonal relationships

In the figure it is possible to observe that the two main actors (yellow nodes with a more central display), have three children between them and are not married. What is curious about this is that, despite having children and not being married is quite common, they are then married to the same person. This should trigger an event for further investigation, as it is not legal to be married to multiple people in Angola. There is also the possibility the three people have made insurance policies covering one another for financial advantages and benefit reaping, which is also illegal. Other than that, this stratification does not present any more cases that are worth looking at.

5.2.7 Seventh Stratification (Node Degree = [7;8])

For this stratification, the decision to group nodes which have $NodeDegree = 7$ and $NodeDegree = 8$ has been made for an easier cluster analysis. As stratifications go on, it is normal to see less and less nodes and edges and therefore, this was also a factor for grouping nodes with different node degrees. As per usual, table 5.11 presents the statistical census of the entities and relationships present in this stratification.

Table 5.11: The number of entities and connections present in the seventh stratification

	Nodes			Edges		
	Individual	Employee	Company	Contact Entity	Wife/Husband	Mother/Father
	345	2	14	15	49	254
Total	361			318		

As it is to be expected, and taking what happened to other stratifications as well, with a large gap between entities labeled as *Individual* and the others, it is possible to expect that a majority of connections (edges) are also those of a family context. This, in its turn, makes the analysis of other types of entities easier (fewer entities = easier analysis).

This being said, Figure 5.16 depicts an abnormal case regarding *Individual* entities. Much like the case previously look at in 5.2.6, there are multiple entities which have a marital connection between them. Two of the entities have two separate spouses and in turn have a spouse in common. This is quite abnormal and could possible mean that these entities are registering other people as their spouses in order to gain unexpected benefits.

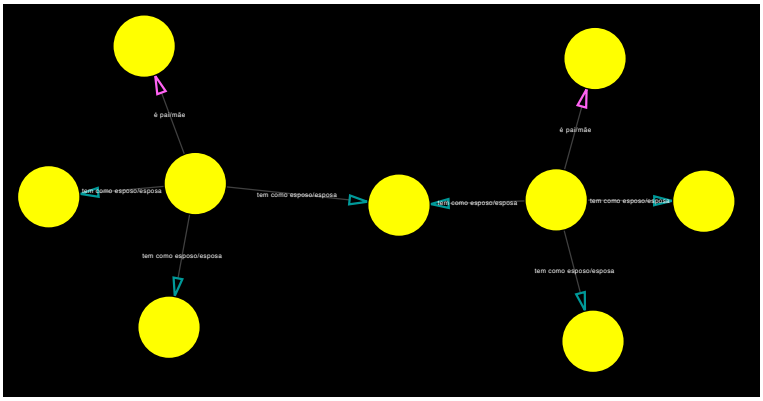


Figure 5.16: Ring of entities connected by marital relationships

Other than that, there is also a curious situation regarding entities labeled as *Company*. Figure 5.17 depicts seven entities which all have the same contact entity, which is an employee of an insurance company. Despite this not being a fraudulent behavior, it is sometimes considered a dubious practice.

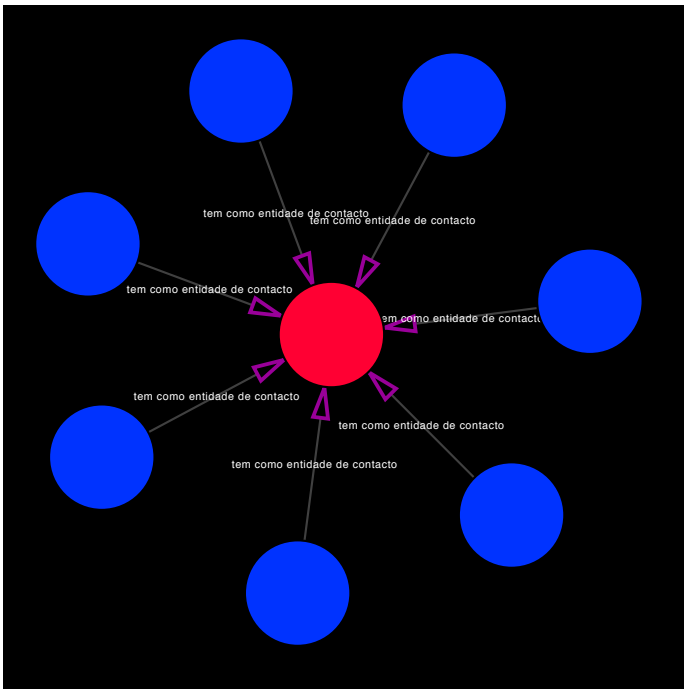


Figure 5.17: Seven companies which have the same contact entity as an insurance company's employee

This is done in order for insurance companies to protect themselves against attempts, by other insurance companies, to entice the customers to change company. The person who usually performs the policy (the *Employee*), marks the contact information for the customers as its own contact. If another insurance company tries to reach the customer, it will instead reach the employee and be promptly

rejected. Other than this, there are no peculiar clusters worth mentioning in this sub-network. There is also the possibility that, if the employee leaves the insurance company, all the entities which have this entity has a contact, will become "unreachable" as the employee will leave with the mentioned contacts.

5.2.8 Eighth Stratification (Node Degree = [9;10])

For this stratification, the decision to group nodes which have $NodeDegree = 9$ and $NodeDegree = 10$ has been made for an easier cluster analysis. As per usual, table 5.12 presents the statistical census of the entities and relationships present in this stratification.

Table 5.12: The number of entities and connections present in the eighth stratification

	Nodes		Edges		
	Individual	Company	Contact Entity	Wife/Husband	Mother/Father
	179	9	10	14	146
Total	188		170		

As seen in the above table, once more, there is a big discrepancy between types of entities. This will mean that, much likely, there will not be unusual connections in this stratification. This being said, it is always possible to argue that the case of Figure 5.18 is not usual.

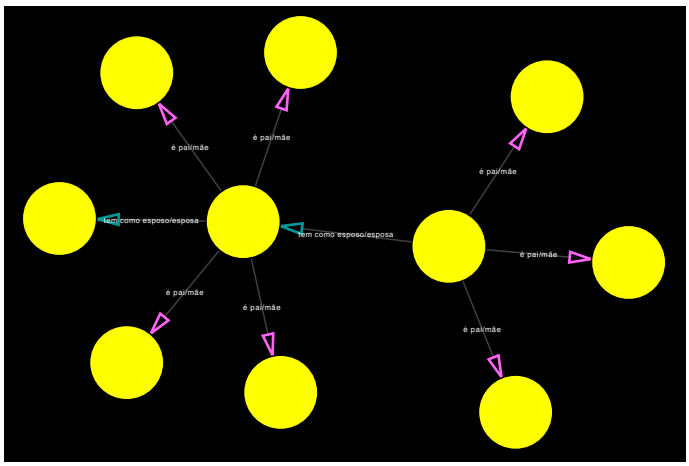


Figure 5.18: Three *Individual* entities that have marriage relationships between them

As the figure depicts, there are three of the ten entities which possess marital connections between them. Although one might speculate that the person from the middle ended their relationship with the person on the right (or vice-versa)

and then proceeded to make a new relationship with the person on the left, it should have terminated the policy before making the new one including the new relationship. Other than that, there are no strange cases on this stratification of the network.

5.2.9 Final Stratification (Node Degree > 10)

For the final stratification of the network, and as the number of connections between nodes (node degree) rises with undefined quantitative leaps, it was chosen to group the following entities and relationships using the premise $NodeDegree > 10$. As per usual, table 5.13 presents the statistical census of the entities and relationships present in this stratification.

Table 5.13: The number of entities and connections present in the final stratification

	Nodes				Edges					
	Individual	Company	Employee	Others	Contact Entity	Wife/Husband	Mother/Father	Employee	Broker	Mediator
	733	93	16	1	106	63	796	53	4	2
Total	843				1024					

As this is the stratification with the most interesting clusters of entities, each one, or a representative cluster in case of similarity to other clusters, will be discussed more in depth for a better understanding of the cases in this sub-network.

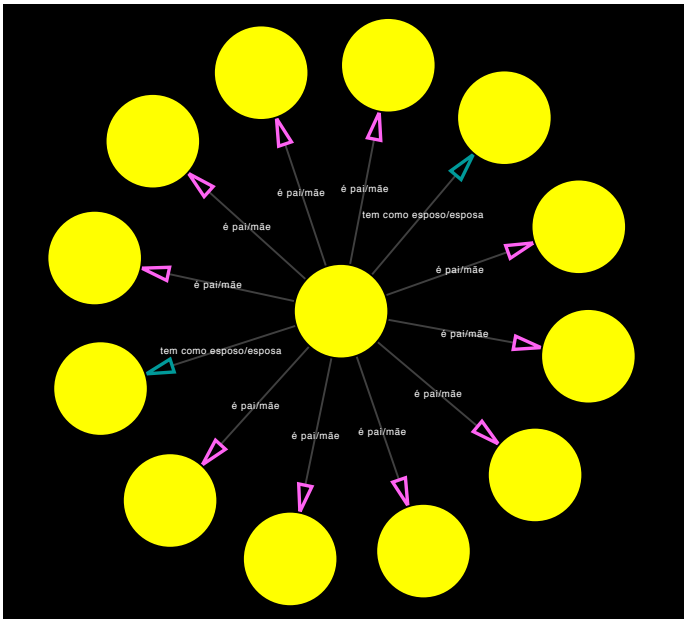


Figure 5.19: One main entity with many connections of parental and marital types

This being said, the first cluster, represented on Figure 5.19, depicts an entity which has ten relationships of parenthood and two of marriage. Now, taking into account that the fertility rate in Angola is around 5.61 children, as mentioned in 5.2, one could speculate that ten children is quite the leap. Also, it is important to consider that this case might be a normal one. It is impossible to know from data alone, so for these kinds of clusters, the best course of action is to actually perform an investigation in order to clarify if the cluster turns out to be normal or not.

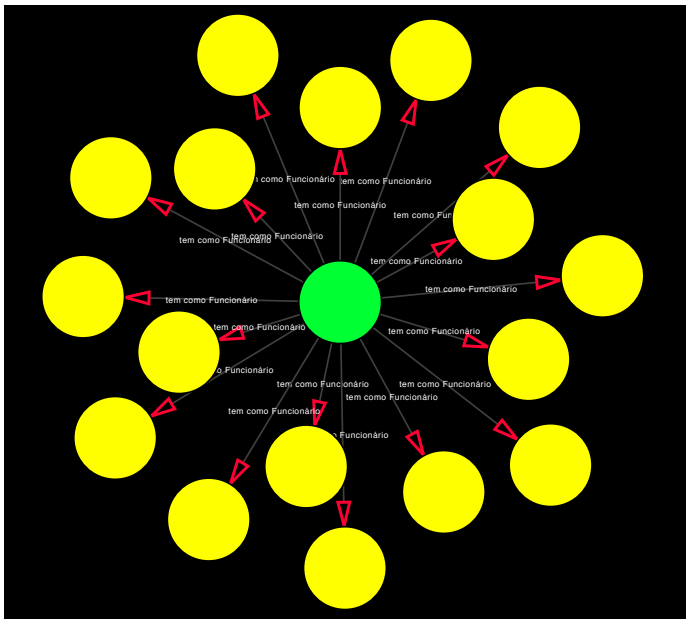


Figure 5.20: Entity labeled as *Other* which only appears once in the entire network

Figure 5.20 depicts an entity which is labeled as *Other*. This entity only appears once on the entire network, so it is interesting to try and perceive at it refers to. As it is not self-explanatory as the other labels it can not be extrapolated what it refers to. As it has many connections labeled *Employee* to individual nodes, it can be hypothesized that this entity is a company. Other than that, from the data it is not possible to go further and understand its meaning. As it rises confusing when analyzing it, this entity should be investigated on an optic to perceive its meaning.

The next Figure, 5.21, shows one of the cluster mechanics discussed previously. The yellow entity which is in the middle of the figure, is the child of three different entities. Despite not being biologically correct, one possible explanation to these relationships is the fact that the entity could have been adopted by one of the other three (from this visualization it is impossible to discover the biological parents). Although this might be the most accurate explanation, this case is (and cases like

this) are not that uncommon through the database (and therefore the network), making an investigation of these occurrences a necessity.

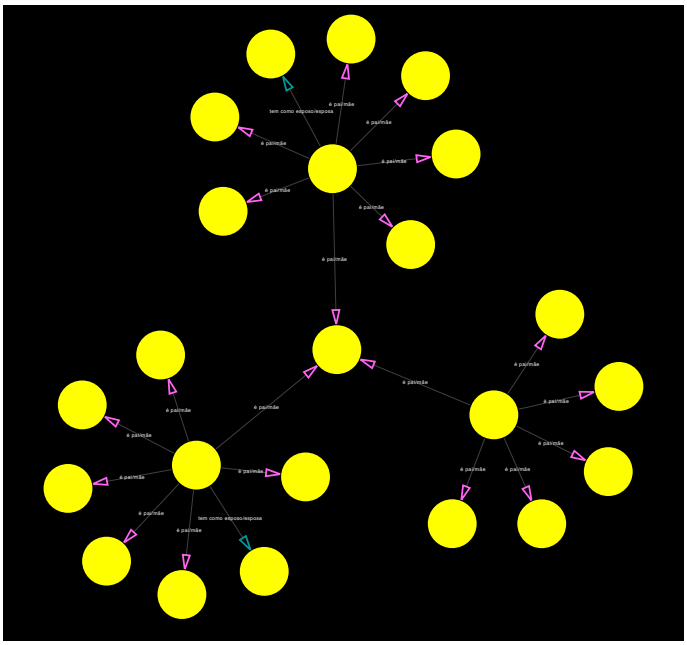


Figure 5.21: Strange cluster layout where an entity has three parenthood relationships

This being said, and before moving on to the biggest clusters of the network (*Individual* clusters), there are some smaller ones that involve multiple types of relationships and entities that show both abnormal layout and strange connections. This will be the focus of the next figure.

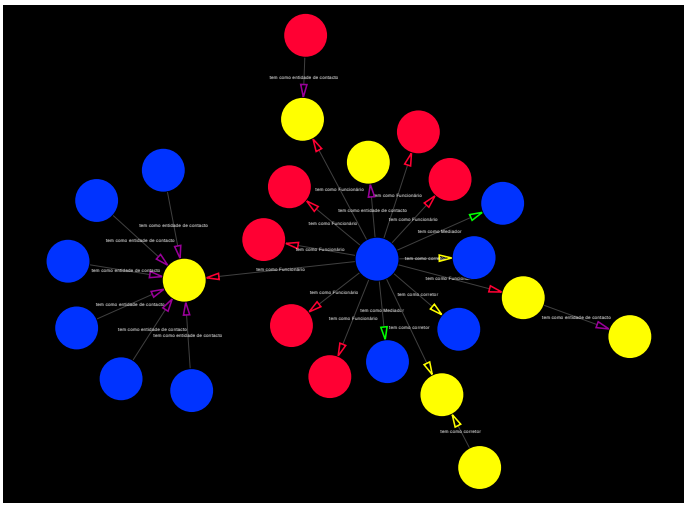


Figure 5.22: High complexity cluster of different nodes and relationships

Figure 5.22 depicts said clusters. On a first instance, there are relationships like the ones described on 5.2.4, which depict the connection (*Employee*) between

a company and and insurance company's employee, which is problematic. After that the first meaningful appearances of relationships labeled as *Broker* and *Mediator* appear. Usually, mediators and brokers are represented as companies which perform certain services (dependent on the type of entity) to the companies. While mediators operate as a subsidiary of the main insurance company (and can be *Individual* entities), brokers operate independently (and must always be labeled as *Company*). It is not unusual for bigger companies to have multiple instances of the same services, as they usually have structural dynamics that require so. The only "out of place" relationships that can be found in this cluster, apart from the ones as described in 5.2.4, are the ones that connect the main company (blue node in the most central part of the cluster) with the yellow node (lowest portion of the cluster). The blue node has a relationship labeled as *Broker* with the yellow node, which is acceptable but, in turn, the yellow node has also another yellow node (*Individual*) which is a broker for it. That might not be as normal as it seems. It is not possible to speculate on this matter any further but it is also advisable to investigate this cluster as a whole since there are certain relationships between actors in the cluster that do not fit or match.

5.2.9.1 High Risk of Fraudulent Practices

The next three examples of clusters require a section of their own. They are considered to have a high probability of being the result of fraudulent activities and it is of extreme importance that they are investigated further for a better understanding of their nature. The node degree of these clusters is much higher than all the other ones present in the overall network.

The first one, present in Figure 5.23, denotes a group of entities, two to be exact, that show a great deal of parental and marital connections. This is definitely one case of clustering that shows great promise in something being wrong. Once again, and taking into account the average number of children born in Angola, which is 5.61, this cluster presents a number of parental connections way above the expected. Of course it is possible to speculate that, as the average is 5.61, a person van have more or less children than that but, having a node degree of twenty three goes beyond the "expected".

Not only talking about parental relationships, these nodes also present a high concentration of marital ones. On average, each node (of the two main ones), has five marital connections which, as discussed before, can indicate that the two people are trying to gain unlawful advantages by adding as much people as they can to their insurance policies.

The final strange factor that this cluster present is the fact that the cluster is held together by two connecting entities. One which is the child of the two main actors and the other which is the spouse of the two same actors.

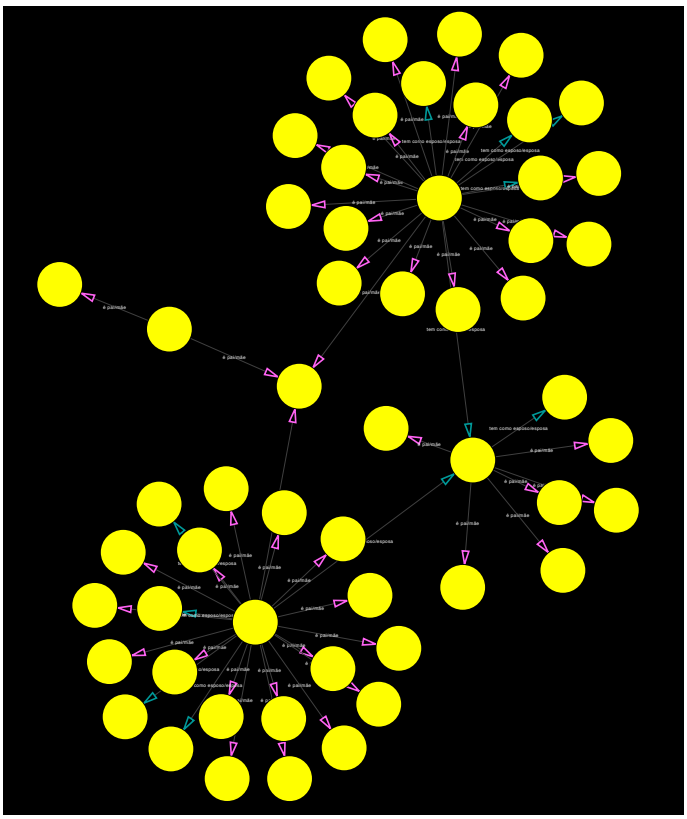


Figure 5.23: First example of a cluster having high probability for fraudulent practices

The overall classification of this cluster should be a case which points to a swift investigation. Having as many relationships as the two main actors have is definitely not normal and even-more when considering the fact that they are connected by two other entities with the label *Individual* and *Wife/Husband* respectively.

Regarding Figure 5.24, the concentration of nodes is even higher. There are two main actors (nodes) in this cluster which both have a node degree of one hundred and eight each. This case is even more worrying than the one before. One of the things that greatly differ from this cluster to the one that came before is the fact that there are only relationships of the type *Mother/Father*. This directly means that every other node in this cluster has $NodeDegree = 2$.

There is not much more to say about this cluster than what was already said. It presents a high concentration of nodes bound together by relationships labeled

as *Individual*. The sheer amount of connections of this type is sufficient enough for this cluster to be considered an anomalous presence in the network.

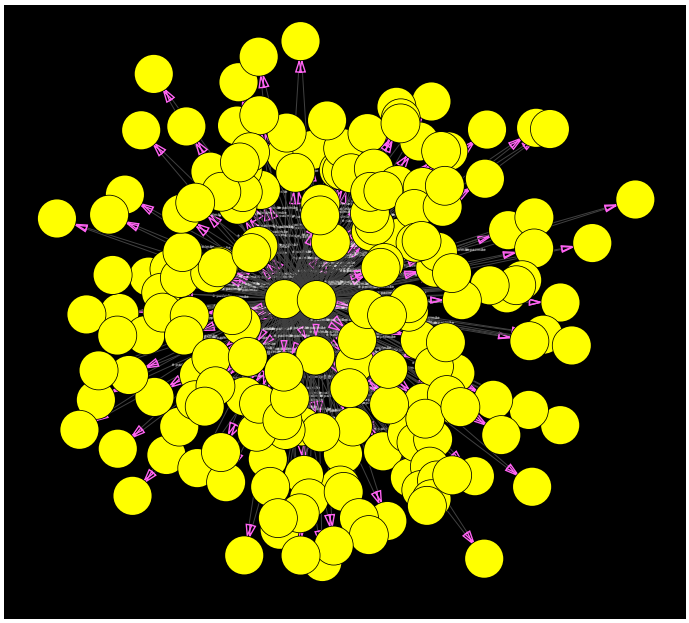


Figure 5.24: Second example of a cluster having high probability for fraudulent practices

The final cluster, which is depicted in Figure 5.25, represents the largest gathering of nodes and edges in the entire network. All of the entities (three hundred and seven) are labeled as *Individual* except one which is labeled as *Company*. Regarding the edges, there are two hundred eighty one which are *Mother/Father*, forty two labeled as *Wife/Husband* and one labeled as *Contact Entity*.

Seeing that the structure of this cluster is so dense and complex, it makes no sense to specify any of the relationships in particular (except two). It can be considered the biggest red flag regarding fraud, as the quantity of both nodes and edges is so high. This being said, the single node labeled as *Company* displays a relationships with a node labeled as an *Individual*. This relationships is a marital one, which in essence does not make much sense. Despite the data coming from a country where the belief system is fundamentally different, there is, with conviction, no place where a relationship of marriage between an individual and a company is normal. The other relationship worth looking at is the one that only appears once. Despite being a completely normal relationship of a contact entity being related to an individual, the fact that it is present in the overall (potentially) fraudulent cluster, makes it worth investigating further.

Other than that, the nodes that are most pressing for an evaluation are the two main actors that constitute the center of the bigger clusters and the ones that make up the "bridge" between the clusters. This makes sense seeing that they are the originating nodes of this complex structure and the explanation behind this unusual structure might be connected to who these entities are.

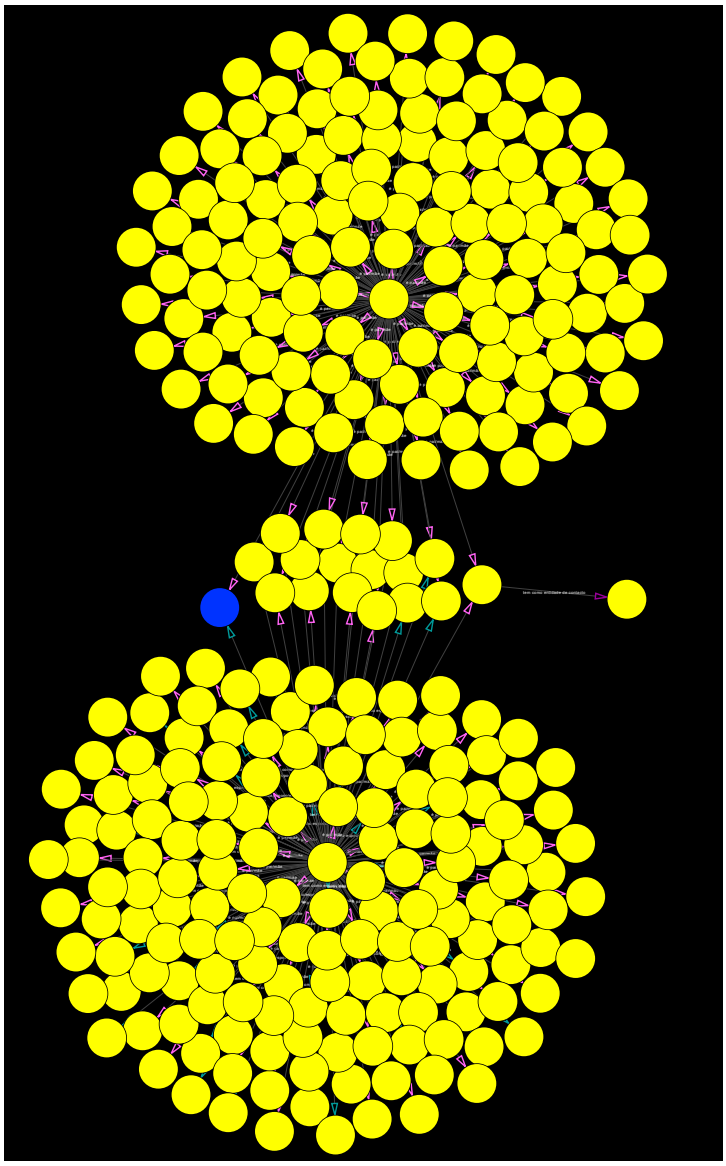


Figure 5.25: Third example of a cluster having high probability for fraudulent practices

This stratification is undoubtedly the one which possess the highest amount of cases which are deemed worth looking at, as they invalidate the expected structural foundation from an insurance network, as be explained in the next section.

5.3 Final Thoughts on the Insurance Network Analysis

Having now gone through all the substantial results from the complex network, it is relevant to succinctly go through them in succession and how they fit in the overall insurance industry's paradigm.

Firstly, the network can feel a bit sparse. This is due to the fact that, what is expect from insurance policies, is for all entities not to be connect to each other. Intrinsically, when performing insurance policies, there is a relative maximum of expected connections that are acceptable. For example, if a person goes to have an insurance policy created, it is not expected for that person to be connected to the Chief Executive Officer (CEO) of the insurance company. The relationship to be expected is to the person who performed the policy. This brings down the connectedness of the network down from what is seen, for example, in figure 5.26, much of the nodes have a high node degree because the aim of that study was to investigate quantum-systems on which particles are expected to interact with each-other.

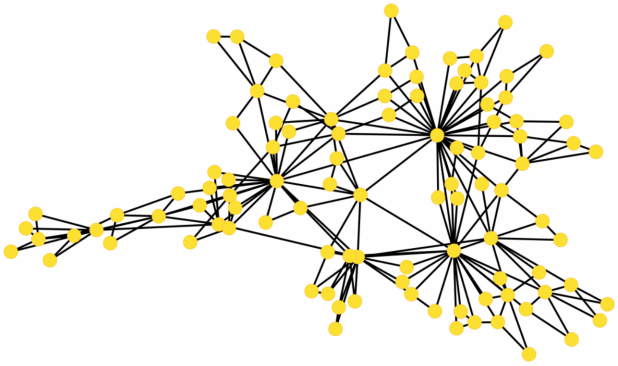


Figure 5.26: The network is composed of 100 nodes, single links are added with hopping probability p=0.1 [73]

Source: `http://www.lkb.upmc.fr/quantumoptics/quantum-complex-networks/`

This does not mean that the network targeted loses any of the properties that grant it the statues of complex. It means that the topology obtained is in accordance with what is expected from the insurance industry.

This being said, it is also important to note that, as stratifications go on, it is expected to observe a steady decline on the number of nodes and edges. This is due to what was said above. As stratifications go on, the node degree also increases, which means that it is expected to see fewer nodes with more connections, as is seen in Figure 5.27.

Despite this, it is also possible to observe that, on the ninth and final stratification, the number of nodes and edges rose unexpectedly. As was explored and explained on 5.2.9.1, this was the stratification with the highest amount of possible fraudulent cases.

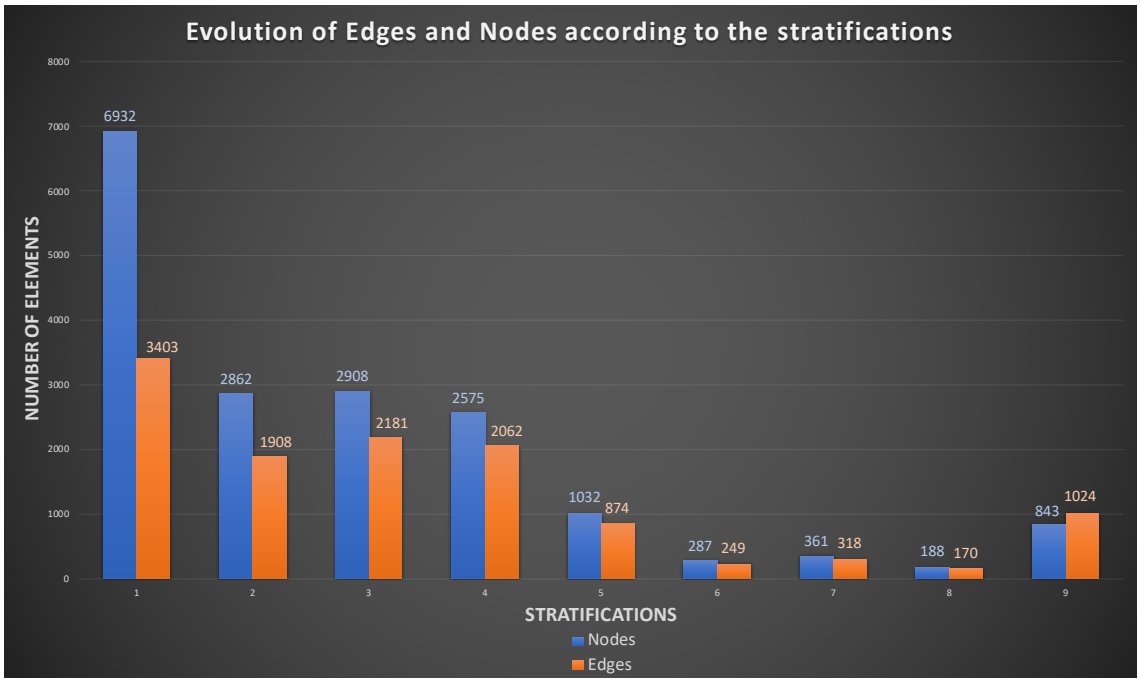


Figure 5.27: Bar plot representing the evolution of the number of edges and nodes according to the stratification

Although fraud cases are not very common and hard to detect, by stratifying the nodes and edges according to node degree, a more clear approach to the problem can be taken. This then allows for a better visualization and analysis of the sub-networks on each stratification and therefore, makes it easier to detect outliers and communities within the data.

Chapter 6

Conclusions and Future Work

This is the last chapter of this dissertation. Section 6.1 presents the conclusions of the investigation and work throughout this dissertation. Section 6.2 will present some interesting applications and directions that complex networks can take and how they will, possibly, influence the knowledge discovery industry.

6.1 Conclusions

From the beginning of this dissertation and, taking into account the area of expertise where it is inserted in, the proposed implementation always took into account the methodologies that govern and dictate how one should approach a data mining problem. From the ones presented on 2.2.2.1, 2.2.2.2 and 2.2.2.3, the clear trend of this dissertation was to follow the KDD approach.

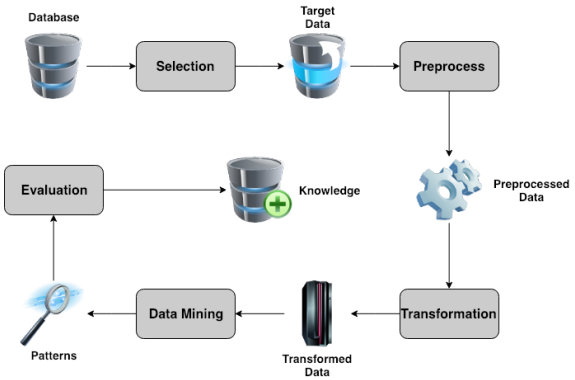


Figure 6.1: Descriptive image depicting the KDD process.

Taking this into account, and as it is possible to observe in Figure 6.1, firstly a selection of the desired data was performed. This ensured that the data to be extracted from the database would fit into the current problem's paradigm and therefore be of any statistical and holistic relevance. This phase also comprised the selection of data through views in order to not damage the database in any way possible to insure the topology did not change due to unforeseen actions. This was done, as was said before, by first doing an extensive research on the insurance industry, what is considered to be fraud and the methods, both by hand and automated, which allowed companies to look for relevant outliers on their data.

After all the data that represented the problem was in hand, the preprocess and transformation phases were applied using the help of Pentaho. Using this software it was possible to streamline these two phases and produce then a more approachable experience to the user in order to leverage most of the ETL work necessary. With Pentaho, it was possible to design a workflow in order to extract the data previously selected from the database, perform the operations needed for the next step and output the necessary files (containing the transformed data) in order for them to be of use.

Following the previous steps in the KDD methodology, the data mining phase was, probably, one of the most crucial steps. On this phase, the data which had been previously transformed, was to be loaded onto a software that could handle, not only the construction of the complex network but also handle the large volume of data that resulted from the previous step. For that, Cytoscape was chosen seeing that, the purpose it was developed for, is to handle large quantities of molecular and biological data. This being said, Cytoscape enabled the segmentation of data by stratifications in order to get a better understanding of the possible patterns that emerged. From this, several filters were applied in order to achieve what was intended. This, in its turn, allowed for the extraction of several pieces of information, thus leading to the next step of the methodology.

The evaluation phase of this dissertation differs in essence from what is expected from cases which use classical machine learning approaches. On cases like so, this phase usually means that the algorithm will be evaluated in order to assess the veracity of the results but, when handling complex networks, a more "out of the box" approach was needed. Instead of evaluating the network as it is expect from the conventional methods, seeing that it is based on graph theory, a metrical and visual evaluations could be done. From here on out, the stratifications were explored and assessed in order to detect strange communities of individuals that arose among the data as to try to obtain positive results on the search of possible

fraudulent situations.

After all the requirements for the previous steps were met, the last step was reached. This phase focuses on the extraction and presentation of the knowledge that was achieved from the data that went throughout the whole process. Taking the area where the problem fits, this knowledge has the purpose to be transmitted to the authorities responsible for investigating further the insurance policies and their intervenients. From here on out, the reach of this dissertation's work has reached its limit. Working with unsupervised data is fundamentally different to working with supervised information. There is not a 100% certainty when reaching for conclusions and every one should be treated as an advice rather than a fact. This applies to data science in general (even when working with supervised datasets) but when working with information regarding sensitive matters like the insurance industry, it is always advisable to investigate as much as possible in order to reach some type of grounded conclusions.

There is no real consensus for the exact percentage of fraudulent claims that are detected, but fraud amounts for a very small percentage of all the insurance data. This being said, there are two main possibilities for this fact, either: the number of fraudulent claims is indeed a small percentage of all claims; the number of fraudulent claims detected is lower than the actual number of unlawful claims. For either one of these cases, the work developed throughout this dissertation is of relevance as the objective is indeed to detect and pursue outlier relationships which might indicate the appearance of fraudulent claims.

Leveraging the work for the end user was also one of the main objectives of the work developed during this dissertation. To achieve this, a solution based upon low code approaches was taken. Despite this, one of the other major factors that played a big role in deciding which tools and frameworks were used during the practical work was the amount of data. The large volume of data that was extracted from the database hindered some of the processes that were initially though of, making the use of tools that could handle those kinds of volumetric values a necessity. From this, a process was designed around those tools to facilitate the extraction, transformation and loading of the data onto the tool that, in its turn, would provide the insights about the data.

With this in mind, the process, which is depicted in Figure 6.2 by the means of a flowchart, represents at a macro scale how the final implementation of the practical work was done. This follows the methodology described before, from the Selection and Preprocess phases, which are comprised of the EDA and extraction operations on the database with the use of SQLDeveloper, to the transformation phase where the application of Pentaho was employed. If the transformation was

successful in extracting the data from the database, the outputs containing the information were to be passed trough the next phase, if not, the user should go through the transformation as much times it needed to come up with a valid one. Following this comes the Data Mining phase which would receive the output files from the previous step. These files are then loaded into Cytoscape in order to create the complex network with the analysis of said network as the main objective. This analysis involved approaching the data by stratifications (which were made using a node degree filter), in order to achieve a better understanding of the overall distribution of the clusters of nodes. From here on out comes the Evaluation phase which meant going through the clusters which stood out the most from the surrounding ones. These clusters would then be analyzed in depth as a means to try and make the most sense out of their formation and structural integrity. Finally, the results that were achieved from all this process would then be passed along to the responsible entities which deal with insurance fraud.

Overall, the implementation of complex networks for insurance fraud detection proved that, in contrast to analyzing data directly from a database, visual tools have clear advantages as they allow for a smoother learning curve on what regards framework comprehension and the possibility to directly apply certain conclusions to the industry being studied. This once more strengthens the point that automation being evermore a common presence among every industry and clearly, with the amount of data circulating nowadays, is the way to go forward as a means to extract knowledge.

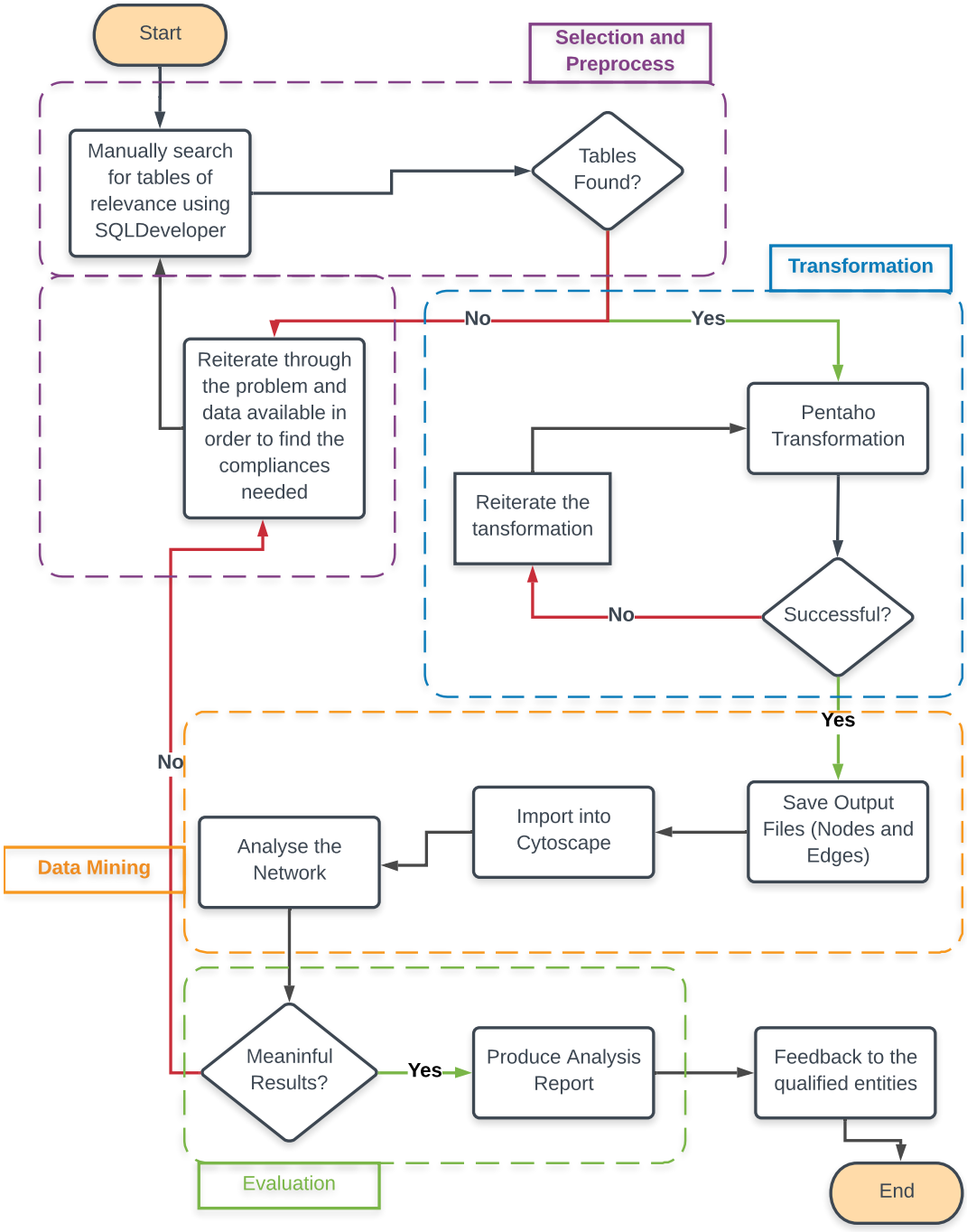


Figure 6.2: Flowchart depicting the practical approach of this dissertation

6.2 Future Work

Going from what was concluded in this dissertation and how important is to detect fraud in general (not only in insurances), going forward, automated approaches come as a fresh new take on the detection of strange patterns that might appear. From the work that was done, and talking strictly about automation, there are a plethora of manners for automating the steps of the process that present less amount of automatic behavior. As was discussed previously on 6.1, the first step of the methodology is always the understanding and search of relevant data. Going forward, if this step was to be automated, it would greatly reduce the amount of time one would spend searching for meaningful data. This being said, this type of conversion can not be of the "Black Box" type. Despite there being many machine learning algorithms and practices that can present data relevance, the input from an human being should always be necessary.

Data science and data mining are subjective topics and the techniques that exist on the current days are not yet capable of handling such subjectiveness as a human could. Despite this, if one were to make the perfect marriage "between" data mining algorithms (i.e neural networks, random forests and clustering) with complex networks and adding the subjectiveness of the human being, it would probably translate to the perfect fraud detection system. Complex networks are a great tool for visualizing how relationships, that would probably go unnoticed, stem from the data itself. With this in mind, when dealing with complex networks in conjunction with more conventional approaches (i.e neural networks, random forests and clustering), it is important to make sure these different methods do not overlap (or even cancel) each other. Using, for example, random forests with complex networks for the discovery of interesting features, would be extremely counter productive as random forests tend to be used alone for this goal. Despite this, running a clustering algorithm on the network (after it is produced) would probably be the next most useful approach. Implementing a solution like so would serve its purpose as the main detector of clusters (normal and deviant), thus mitigating most of the human work behind the analysis of the network. So, on the topic of merging different types of approaches, one should be careful and plan as much as possible their implementation and how they will affect each other before jumping to conclusions.

Going backwards to the KDD model, having now the automation of the last two phases (Evaluation and Data Mining) and the first two (Selection and Preprocess) it is still possible to argue that one could automate the process as a whole.

With the procedures and methodologies that were implemented during the practical work of this dissertation, most of the "connections" between software tools had to be done manually. Despite some of them (i.e Cytoscape) having integration with other types of tools, these are still not well documented and remain a challenge when trying to perform the linkage between them. With this in mind, having a solution that allowed for the seamless transition between phases of the KDD model would be the best approach to the problem. Not only would be easier for the user to exert, it would also mean less room for errors that stem from the application of different software tools on the same project.

With the automation of certain parts, a still pressing matter continues to delve. With the amount of information available today, systems are being bottlenecked by the physical components that make them up. One major plot point in this dissertation was the handling of massive quantities of data and how that dictated the overall course of the practical work. From the extraction to the network creation software, the volume of data was always a constant factor. This being said, nowadays there are solutions for handling these types of volumes. From cloud computing to quantum computing, the new emergent technologies to handle large quantities of data are beginning to show their presence in the world of data and knowledge discovery. Benefiting from these types of technologies would greatly increase the depth of the networks produced as it would be possible to add even more information in order to observe hidden clusters of information.

From an industry point of view, graph theory and therefore complex networks are evermore being used as a tool for fraud detection. With the incorporation of tools that benefit from these subjects, fraud detection would be possible at earlier stages of conception and thus making it possible to avoid the economical and social losses that stem from it. Undoubtedly, and stressing once more the quantity of data that is produced on a daily basis, the quantity of data companies sit on is enormous. With this in mind, it is curious to see these "ancient" mathematical theorems still being applied on current days and most important, propelling the industry's understanding of their data forward (providing the back-end solutions in place are able to deal with the volume of data).

Of course there is still room for growth, both in the way these technologies are implemented and the acceptance of them by the society. Big data, data analytics, data science, knowledge discovery are hot topics that emerged in the data paradigm that it is lived today and as society moves forward, so will these methodologies and their applicability and integration in the daily life of every person and industry.

Bibliography

[1] R. Gandhi. *K-Means Clustering — Introduction to Machine Learning Algorithms*. URL: https://towardsdatascience.com/k-means-clustering-introduction-to-machine-learning-algorithms-c96bf0d5d57a (visited on 02/11/2019).

[2] Saurabh. *What Is Backpropagation? | Training A Neural Network | Edureka*. URL: https://www.edureka.co/blog/backpropagation/ (visited on 02/11/2019).

[3] S. Polamuri. *How the random forest algorithm works in machine learning*. URL: http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/ (visited on 02/11/2019).

[4] OECD. "Insurance Markets in Figures 2018." In: June (2018), p. 5. URL: http://www.oecd.org/daf/fin/insurance/Insurance-Markets-in-Figures-2018.pdf.

[5] F. Decker. *Definition of Non-life Insurance - Budgeting Money*. URL: https://budgeting.thenest.com/definition-nonlife-insurance-27404.html (visited on 01/28/2019).

[6] *Insurance indicators: Life insurance share*. URL: https://stats.oecd.org/Index.aspx?DatasetCode=INSIND{\&}{_}ga=2.56475436.62366802.1548373781-538776792.1548373780 (visited on 01/25/2019).

[7] A. Heirman and B. Clarysse. "Which tangible and intangible assets matter for innovation speed in start-ups?" In: *Journal of Product Innovation Management* 24.4 (2007), pp. 303–315.

[8] S. Viaene and G. Dedene. "Insurance Fraud: Issues and Challenges." In: *Geneva Papers on Risk and Insurance: Issues and Practice* 29.2 (2004), pp. 313–333. ISSN: 10185895. DOI: 10.1111/j.1468-0440.2004.00290.x.

[9] W. S. Albrecht, C. O. Albrecht, C. C. Albrecht, and M. F. Zimbelman. *Fraud examination*. Cengage Learning, 2011.

[10] A. A. Shmais and R. Hani. "Data Mining for Fraud Detection." In: *International Journal of Computer Science and Information Technologies* 4.1 (2013), pp. 1–4.

[11] R. Nisbet, J. Elder, and G. Miner. *Handbook of statistical analysis and data mining applications*. Academic Press, 2009.

[12] PricewaterhouseCoopers. "Fraud: A guide to its prevention, detection and investigation." In: *PwC Forensic Service* (2008), pp. 14–16. URL: www.pwc.com/au/forensicservices.

[13] Siciliano Robert. *5 Insidious Forms of Auto Insurance Fraud | HuffPost*. 2012. URL: https://www.huffingtonpost.com/robert-siciliano/5-insidious-forms-of-auto{_}b{_}1284291.html?ec{_}carp=3030140582852774723{\&}guccounter=1 (visited on 01/31/2019).

[14] Mcintosh Jil. *How It Works: Car insurance fraud | Driving*. 2018. URL: https://driving.ca/chevrolet/auto-news/news/how-it-works-car-insurance-fraud (visited on 01/31/2019).

[15] Zhao Shirley. *What is ETL? (Extract, Transform, Load) | Experian*. 2017. URL: https://www.edq.com/blog/what-is-etl-extract-transform-load/ (visited on 02/03/2019).

[16] White Sarah. *Study reveals that most companies are failing at big data | CIO*. URL: https://www.cio.com/article/3003538/big-data/study-reveals-that-most-companies-are-failing-at-big-data.html (visited on 02/03/2019).

[17] Team Talend. *What is ETL (Extract, Transform, Load)? - Talend*. URL: https://www.talend.com/resources/what-is-etl/ (visited on 02/03/2019).

[18] P. Chapman, J. Clinton, T. Khabaza, T. Reinartz, and R. Wirth. "The CRISP-DM Process Model." In: C (1999). URL: http://crisp-dm.org.

[19] C. S. S. Chapman, Pete Julian Clinton (SPSS), Randy Kerber (NCR), Thomas Khabaza (SPSS), Thomas Reinartz (DaimlerChrysler) and R. W. (Daimler-Chrysler). "CRISP-DM 1.0." In: *ASHA presentation* (2000). ISSN: 0957-4174. DOI: 10.1109/ICETET.2008.239. arXiv: arXiv:1011.1669v3.

[20] B. S. Meeting and P. Chapman. "The CRISP-DM User Guide." In: *The CRISP-DM User Guide*. Brussels, 1999, p. 14. URL: https://s2.smu.edu/{~}mhd/8331f03/crisp.pdf.

[21] V. William. *CRISP-DM – a Standard Methodology to Ensure a Good Outcome - Data Science Central*. 2016. URL: https://www.datasciencecentral.com/profiles/blogs/crisp-dm-a-standard-methodology-to-ensure-a-good-outcome (visited on 02/05/2019).

[22] *Enterprise Miner TM SEMMA*. Tech. rep. 2006. URL: http://www.sas.com/technologies/analytics/datamining/miner/semma.html.

[23] A. Azevedo and M. F. Santos. "KDD, SEMMA AND CRISP-DM: A PARALLEL OVERVIEW." 2008. URL: https://pdfs.semanticscholar.org/7dfe/3bc6035da527deaa72007a27cef94047a7f9.pdf.

[24] *Enterprise Miner TM SEMMA*. Tech. rep. 2006. URL: http://www.sas.com/technologies/analytics/datamining/miner/semma.html.

[25] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[26] A. K. Jain, J. Mao, and K. M. Mohiuddin. "Artificial neural networks: A tutorial." In: *Computer* 29.3 (1996), pp. 31–44.

[27] A. T. Goh. "Back-propagation neural networks for modeling complex systems." In: *Artificial Intelligence in Engineering* 9.3 (1995), pp. 143–151.

[28] R. Eulogio. *Introduction to Random Forests*. URL: https://www.datascience.com/resources/notebooks/random-forest-intro (visited on 02/13/2019).

[29] E. Alpaydin. *Introduction to Machine Learning Second Edition*. Second Edi. The MIT Press, 2010, pp. 350–380. ISBN: 9780262012430. DOI: 10.1007/978-1-62703-748-8_7. arXiv: 0904.3664.

[30] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, NY, USA: 2001.

[31] R. R. Schaller. "Moore's law: past, present and future." In: *IEEE Spectrum* 34.6 (June 1997), pp. 52–59. ISSN: 0018-9235. DOI: 10.1109/6.591665.

[32] S. Rogers. "Big data is scaling BI and analytics." In: *Information Management* 21.5 (2011), p. 14.

[33] M. Winans, D. Faupel, A. Armstrong, J. Henderson, E. Valentine, L. McDonald, D. Walters, J. Waite, M. Trapani, and E. Magill. "10 Key Marketing Trends for 2017 Customer Expectations and Ideas for Exceeding Customer Expectations." In: *IBM Offering Information* (2016), p. 18. URL: https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wrl12345usen/watson-customer-engagement-watson-marketing-wr-other-papers-and-reports-wrl12345usen-20170719.pdf.

[34] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang. "Complex networks: Structure and dynamics." In: *Physics reports* 424.4-5 (2006), pp. 175–308.

[35] A. Backlund. "The definition of system." In: *Kybernetes* 29.4 (2000), pp. 444–451.

[36] Aristotle. *Aristotle Metaphysics*. Oxford University Press, 1994.

[37] P. W. Anderson. "More is different." In: *Science* 177.4047 (1972), pp. 393–396.

[38] J. Deguet, Y. Demazeau, and L. Magnin. "Elements about the emergence issue: A survey of emergence definitions." In: *Complexus* 3.1-3 (2006), pp. 24–31.

[39] G. H. Lewes. *Problems of life and mind*. Trübner & Company, 1877.

[40] E. Verlinde. "On the Origin of Gravity and the Laws of Newton." In: *Journal of High Energy Physics* 2011.4 (2011), p. 29.

[41] J. Ladyman, J. Lambert, and K. Wiesner. "What is a complex system?" In: *European Journal for Philosophy of Science* 3 (June 2013). DOI: 10.1007/s13194-012-0056-8.

[42] N. Barry. "The tradition of spontaneous order." In: (1982).

[43] D. T. Matt, E. Rauch, and P. Dallasega. "Trends towards Distributed Manufacturing Systems and modern forms for their design." In: *Procedia CIRP* 33 (2015), pp. 185–190.

[44] J. P. Sturmberg and C. Martin. *Handbook of systems and complexity in health*. Springer Science & Business Media, 2013.

[45] R. Albert and A.-L. Barabási. "Statistical mechanics of complex networks." In: *Reviews of modern physics* 74.1 (2002), p. 47.

[46] G. Alexanderson. "About the cover: Euler and Königsberg's Bridges: A historical view." In: *Bulletin of the american mathematical society* 43.4 (2006), pp. 567–573.

[47] B. Hopkins and R. Wilson. "The truth about Königsberg." In: *Leonhard Euler: Life, Work and Legacy* 5 (2007), pp. 409–420.

[48] H. Sachs, M. Stiebitz, and R. J. Wilson. "An historical note: Euler's Königsberg letters." In: *Journal of Graph Theory* 12.1 (1988), pp. 133–139.

[49] J. Colchester. *Complexity Theory*. 2016.

[50] T Cormen, C Leiserson, R Rivest, and C Stein. *Introduction to Algorithms*. Vol. 25. 4. 2009, p. 590. ISBN: 1859840566. DOI: 10.2307/2077150. arXiv: 0712.0689.

[51] M. Noto and H. Sato. "A method for the shortest path search by extended Dijkstra algorithm." In: *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*. Vol. 3. IEEE. 2000, pp. 2316–2320.

[52] M. Zanin, D. Papo, P. A. Sousa, E. Menasalvas, A. Nicchi, E. Kubik, and S. Boccaletti. "Combining complex networks and data mining: why and how." In: *Physics Reports* 635 (2016), pp. 1–44.

[53] A. Masoudi-Nejad, F. Schreiber, and Z. R. M. Kashani. "Building blocks of biological networks: a review on major network motif discovery algorithms." In: *IET systems biology* 6.5 (2012), pp. 164–174.

[54] P. E. Black. *Dictionary of algorithms and data structures*. Tech. rep. 1998.

[55] S Boccaletti, M Ivanchenko, V Latora, A Pluchino, and A Rapisarda. "Detecting complex network modularity by dynamical clustering." In: *Physical Review E* 75.4 (2007), p. 045102.

[56] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. "Network motifs: simple building blocks of complex networks." In: *Science* 298.5594 (2002), pp. 824–827.

[57] P. Dankelmann, W. Goddard, and C. S. Swart. "The average eccentricity of a graph and its subgraphs." In: *Utilitas Mathematica* 65 (2004), pp. 41–52.

[58] D. B. West et al. *Introduction to graph theory*. Vol. 2. Prentice hall Upper Saddle River, NJ, 1996.

[59] J. Bouttier, P. Di Francesco, and E. Guitter. "Geodesic distance in planar graphs." In: *Nuclear Physics B* 663.3 (2003), pp. 535–567.

[60] Amarendra Babu L. *Top 8 programming languages every data scientist should master in 2019*. 2019. URL: https://bigdata-madesimple.com/top-8-programming-languages-every-data-scientist-should-master-in-2019/ (visited on 09/13/2019).

[61] Ehi Aigiomawu. *Quick dive into Pandas for Data Science - Towards Data Science*. 2018. URL: https://towardsdatascience.com/quick-dive-into-pandas-for-data-science-cc1c1a80d9c4 (visited on 08/23/2019).

[62] International Organization for Standardization. *ISO - ISO 3166 Country Codes*. 2019. URL: https://www.iso.org/iso-3166-country-codes.html (visited on 08/30/2019).

[63] K. Xu, C. Tang, R. Tang, G. Ali, and J. Zhu. "A comparative study of six software packages for complex network research." In: *2010 Second International Conference on Communication Software and Networks*. IEEE. 2010, pp. 350–354.

[64] W. W. Zachary. "An information flow model for conflict and fission in small groups." In: *Journal of anthropological research* 33.4 (1977), pp. 452–473.

[65] M. Bastian, S. Heymann, and M. Jacomy. "Gephi: an open source software for exploring and manipulating networks." In: *Third international AAAI conference on weblogs and social media*. 2009.

[66] J. Haggerty and S. Haggerty. "Visual analytics of an eighteenth-century business network." In: *Enterprise & Society* 11.1 (2010), pp. 1–25.

[67] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. "Cytoscape: a software environment for integrated models of biomolecular interaction networks." In: *Genome research* 13.11 (2003), pp. 2498–2504.

[68] G. A. Pavlopoulos, D. Paez-Espino, N. C. Kyrpides, and I. Iliopoulos. "Empirical comparison of visualization tools for larger-scale network analysis." In: *Advances in bioinformatics* 2017 (2017).

[69] Elliott Saslow. *Unsupervised Machine Learning - Towards Data Science*. 2018. URL: https://towardsdatascience.com/unsupervised-machine-learning-9329c97d6d9f (visited on 09/02/2019).

[70] P. D. Department of Economic and Social Affairs. *Angola Population (2019) - Worldometers*. 2019. URL: https://www.worldometers.info/world-population/angola-population/ (visited on 09/03/2019).

[71] M. Grassi and J. Vivet. "Fathering and Conjugality in Transnational Patchwork Families: the Angola/Portugal case." In: *TL Network e-Working Papers* (2014), pp. 1–25.

[72] J Rychlewski. "On Hooke's law." In: *Journal of Applied Mathematics and Mechanics* 48.3 (1984), pp. 303–314.

[73] J. Biamonte, M. Faccin, and M. De Domenico. "Complex Networks from Classical to Quantum." In: (2017). DOI: 10.1038/s42005-019-0152-6. arXiv: 1702.08459. URL: http://arxiv.org/abs/1702.08459http://dx.doi.org/10.1038/s42005-019-0152-6.

[74] Margaret Rouse. *What is a Relational Database? - Definition from WhatIs.com.* 2019. URL: https://searchdatamanagement.techtarget.com/definition/relational-database (visited on 08/21/2019).

ANNEX I

Structural Analysis of Data Input Sources

I.1 CSV File Structure

As the name implies, the data is stored in a text file (albeit with a different extension), with the particularity that the text is separated by commas instead of regular topographical elements like spaces or tabs. This being said, these properties grant CSV files a tabular appearance when read, much like that of a regular Microsoft Excel file. This means that the data is allocated to a table which possesses rows and columns, much like that of a matrix, when it is read.

One thing to note is that, not always, the delimiter for a CSV file is a comma. In the case of figure I.1, the delimiter user is a semi-colon, which is also valid. This difference stems either from the local configuration of the Operating System (OS) or from the software that it originated from. There is also the possibility of a combination of both factors.

Note that, at a first glance, the text file seems hard to read as is. This is due to the fact that these types of files are not meant to be read by "human eyes" as a normal text file would. Instead, an appropriate software should be used. Allied to this fact, there is also the need to use a software which is capable of transcribing the file to the format which is needed, as the end goal is to get a tabular format.

Figure I.1: Depiction of an example of a csv file when read with a common text editor. In this case, instead of commas, semicolons are used, which are also a valid example of a csv accepted delimiter.

After being read, the file is expected to be readily ready for being manipulated or subject to any change the user desires. As will be explained further down, these changes can come from a multiplicity of places, being the most common certain programming languages, those of which Python and R which have become the preferred ones for data mining problems [60].

I.2 Relational Database

One of the most used approaches on what regards data storage and manipulation, is without a doubt relational databases. They serve as the basis for background storage of information and, in the case of this dissertation, play a big role on the overall workflow of data.

On what regards a formal description, relational databases are a set of formally described tables from which one can access information and therefore manipulate it in a way that there is not the need for formal reorganization of the whole structural model [74].

What this means is that, much like was mentioned previously on I.1, what stems from models like the ones from relational databases are tables of information that can be accessed and analyzed. Tables are also identified by rows and

columns (with corresponding headers) but with an important difference, the presence of keys. The main focus and purpose of keys is the representation of certain relationships. These relationships stem from the connections between tables. As what is intended with relational databases is the creation of a model, relationships become a fundamental force that drives the model as a whole. Still on the topic of keys, there are, apart from the definition of key itself, two more definitions that fit a key into two types, these being:

1. **Key** - A key is usually one or more columns from a certain table which are used to identify each row from the table.

2. **Primary Key** - This key is a mandatory requirement for each table. This restriction applies because there is a need to clearly identify each element on a table, thus a primary key serves this purpose. Most of the times, this type of key is also a mandatory requirement that as to be fulfilled as items on the table are have to be identified with something i.e customers on a sales database.

3. **Foreign Key** - When tables have dependencies with one another, usually the primary key from one table is transferred to the other. Present now in the second table, the first table's primary key, is now called a foreign key. This allows to have consistency in the data.

As was said before, keys partake a big role in a relational database as they enable the "linkage" of tables per se. One can only link tables and therefore share information between them when they are linked in some specific way that enables the transfer of said keys.

One other important thing to note about relational databases are the types of connections (links) between tables. These types dictate the logic structure underneath the model as well as different coding practices at the back-end model. There are 3 main types of relationships, those being:

- One-to-One - Despite being the simplest type of relationship, it is not the most common among the three. This type of relationship serves a very specific set of purposes like partitioning information that could be easily stored in a table among several others. It can also be used for security purposes.

Figure I.2: Model representation of a one-to-one relationship.

Source: https://database.guide/the-3-types-of-relationships-in-database-design

As it is possible to observe in figure I.2, it would be entirely possible to store all the information regarding the Employee in the table with the same name. The user opted to create an additional table entitled "Pay". This new table does not possess a primary key itself but instead a foreign one identifying each employee by their payment identification (PayId). The model was, much likely, constructed this way to relive stress in computational terms from the system. Additionally, this might be a good option in what regards security so that only certain individuals might access the "Pay" table.

- One-to Many (or Many-to-One) - This type of relationship is one of the most common and used in relational database models. As it stands, this type of relationship means that one element can have many matching elements from another table but the inverse is not true, meaning this type of relationship is strictly directional.

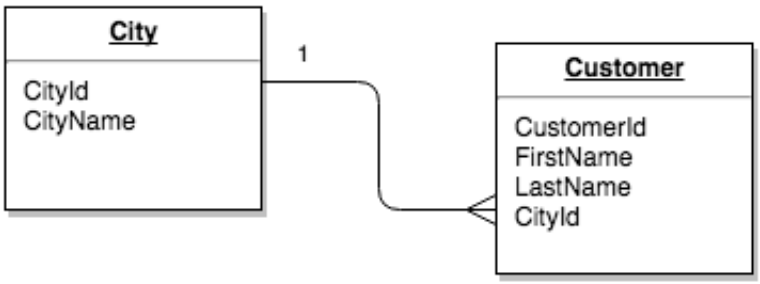


Figure I.3: Model representation of a one-to-many (or many-to-one) relationship.

Source: https://database.guide/the-3-types-of-relationships-in-database-design

Taking figure I.3 into account, the table named "City" has a connection of this type to a table labeled "Customer", meaning that a city can have multiple customers associated to it, but a customer can only have a city to its name.

- Many-to-Many - This relationship is also one of the most common ones to appear on the database world. Despite the above relationship being the most

common, most of the times, there can not be a one-to-many relationship without having first a many-to-many. This is due to the fact that databases are do not deal well, both temporarily and spatially, with a many-to-may relationship. Instead, one can view these kinds of relationships as two one-to-many ones.

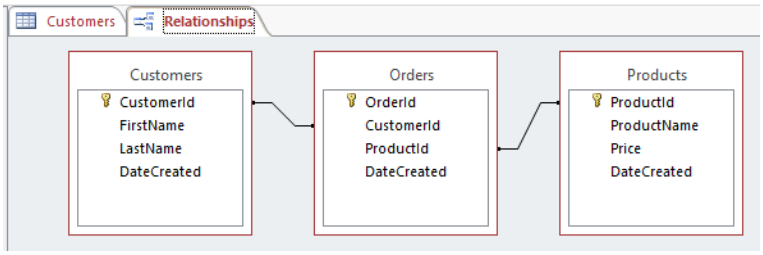


Figure I.4: Model representation of a many-to-many relationship.

Source: `https://database.guide/the-3-types-of-relationships-in-database-design`

As is seen in figure I.4, there is a bi-partition of the many-to-many relationship. Instead of linking the "customers" directly to the "Products", an intermediate table "Orders" is created. On this junction table, there are present foreign keys both belonging to the customers and products. This ends up being a good representation of what happens in the real world in contrast with the academic one. Tables are often big and complex and their underlying models are even more complex. This was a workaround that was found in order to reduce the complexity of queries to these types of databases.

ANNEX II

Python Code for ETL Purposes

Listing II.1: Python code used for ETL purposes. This was achieved using several different CSV files.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Jun 18 21:22:43 2019
@author: joaomouco
"""

import pandas as pd
import numpy as np

df = pd.read_csv('/Users/joaomouco/Desktop/HOLOS_BD/
                                    ENT_ENTS_RELACIONADAS.csv')


def select_columns(data_frame, column_names):
  new_frame = data_frame.loc[:, column_names]
  return new_frame

selected_columns = ['F_ENT_PRINCIPAL', 'F_ENT_RELACIONADA']
data = select_columns(df, selected_columns)
data.rename(columns={'F_ENT_PRINCIPAL':'Source', 'F_ENT_RELACIONADA':'
                                    Target'}, inplace=True)

mini_data = data.head(200)
mini_df = df.head(200)
```

```
24
25 edge = []
26 for i, j in mini_data.iterrows():
27   print(i, j['Source'], j['Target'])
28   edge.append([j['Source'],j['Target']])
29 for i in edge:
30   print (i)
31
32 #--------------------------------- create nodes
33 #-> fetch rows
34 #-> remove similar objects
35
36 node_df = pd.merge(mini_data, mini_df.rename(columns={'F_ENT_PRINCIPAL
                                    ':'Source'}), on='Source', how='
                                    outer')
37 joined_tables = mini_data.set_index('Source').join(mini_df.set_index('
                                    F_ENT_PRINCIPAL'))
38 joined_tables.drop(['Target'], axis=1, inplace=True)
39 joined_tables['Source'] = joined_tables.index
40
41 joined_tables.to_csv(r'/Users/joaomouco/Desktop/nodes.csv', header=
                                    True, index=False, sep=',', mode='a
                                    ')
42
43 # create adjacency matrix
44
45 edge_list = pd.crosstab(mini_data.Source, mini_data.Target)
46 idx = edge_list.columns.union(edge_list.index)
47 edge_list = edge_list.reindex(index = idx, columns=idx, fill_value=0)
48
49 writeToThis = open('matriz_adjacencia.csv', 'w')
50
51 #export file with adjacency matrix
52 np.savetxt(r'/Users/joaomouco/Desktop/matriz_adjacencia.txt',
                                    edge_list.values)
53
54 #export file as .CSV
55 mini_data.to_csv(r'/Users/joaomouco/Desktop/edges.csv', header=True,
                                    index=False, sep=',', mode='a')
```

ANNEX III

Pentaho Data Integration Transformation and Job Files

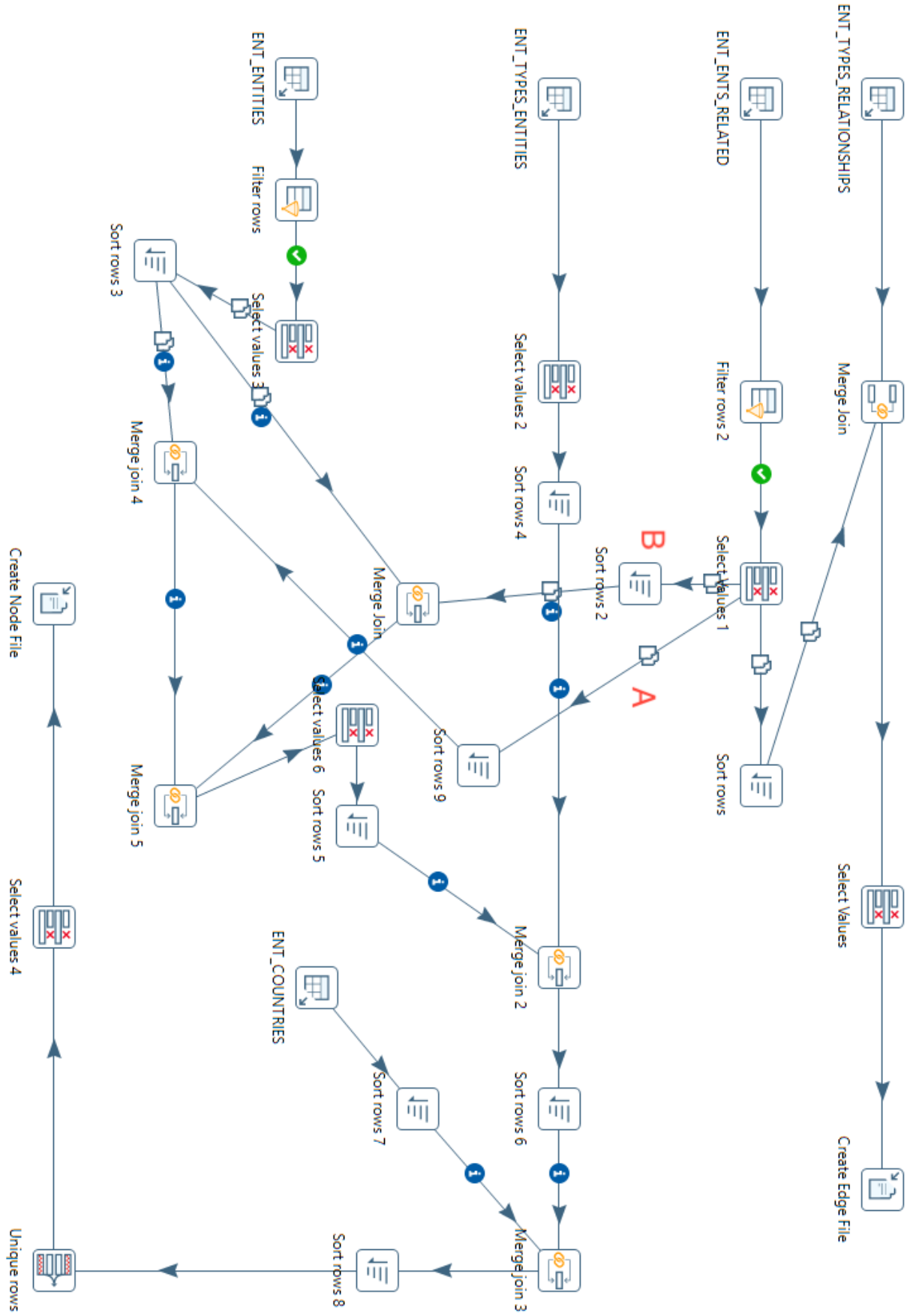


Figure III.1: PDI schematic depicting the final transformation used for ETL purposes.

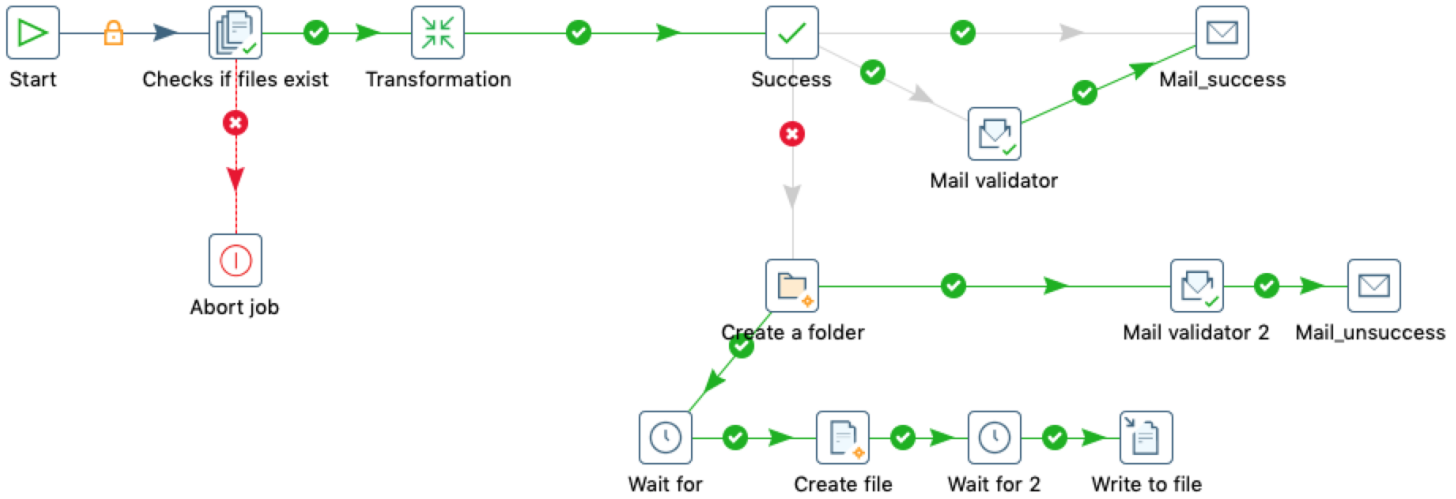


Figure III.2: PDI schematic depicting the final job used for ETL purposes.

Table III.1: Table depicting all the possible relationships (and inversely the inverse relationships) present in the database

DESC_DIRETA	DESC_INVERSA
is a fundraiser for	is a fundraiser for
have as an extranet user	is the entity to which the extranet user belongs
is headquartered the company	is the company to which the user of the environment belongs
has as owner the company	is owned by
has as an exploiting entity the company	is the exploiting entity of the environment user
has as owner / guardian	is the owner / responsible user of the territory and nature
has as an author of the coordinate	is the author of the geographic position coordinate
is a distribution list which has as a member	is entity associated with entity list
includes list of entities	is included in the list of entities
customer of	has as a customer
is mediator of	has as a mediator
is a reinsurer of	has as a reinsurer
is an insurer	has as an insurer
is a service provider of	has as a service provider
is an employee of	has as an employee
is contact entity	has as a contact entity
is a broker of	has as a broker
has as headquarters the company	is the company to which it belongs
is son of	is mother/father of
is wife/husband of	is wife/husband of
is dependent of	has as dependent

ANNEX IV

Cytoscape Image Gallery

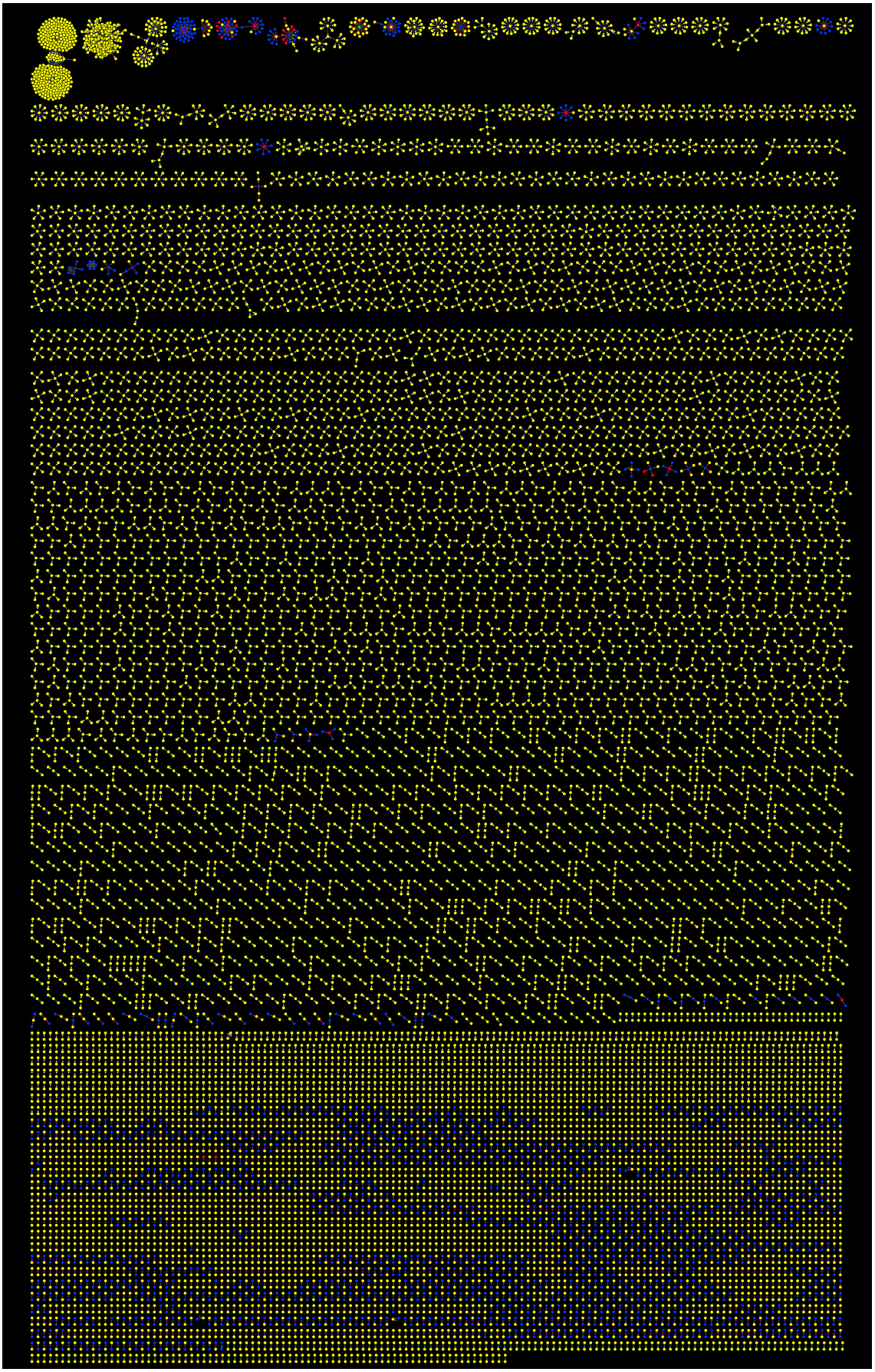


Figure IV.1: Overview of the final insurance network produced using Cytoscape

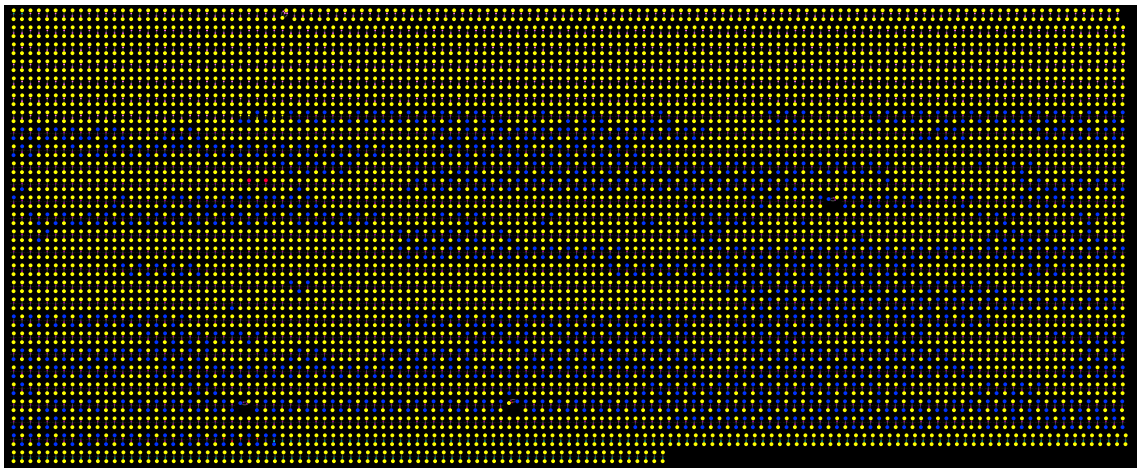


Figure IV.2: Depiction of the stratification based on the node degree (Degree = 1)

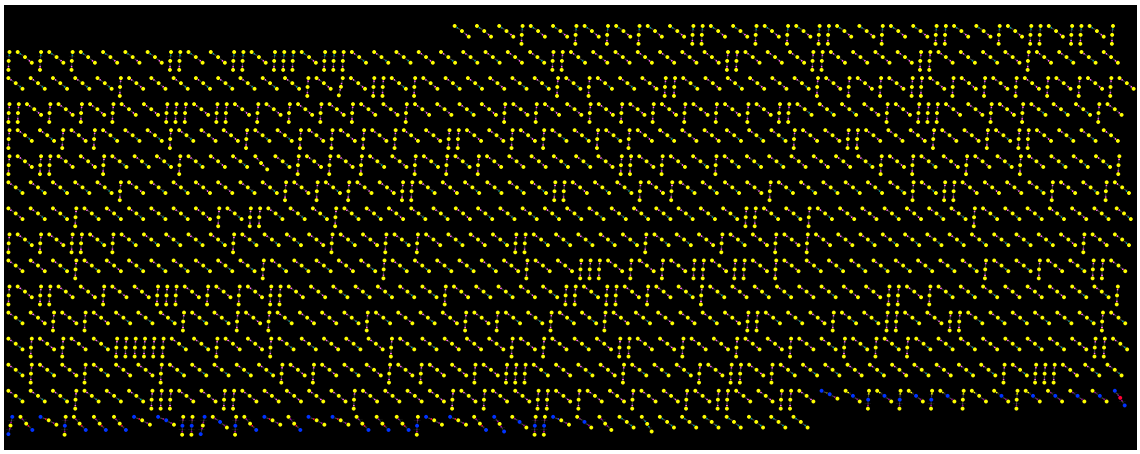


Figure IV.3: Depiction of the stratification based on the node degree (Degree = 2)

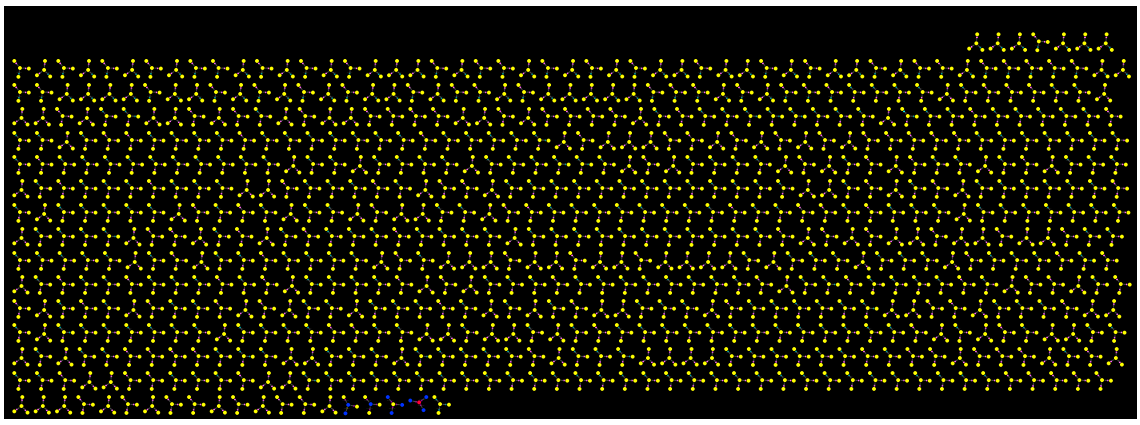


Figure IV.4: Depiction of the stratification based on the node degree (Degree = 3)

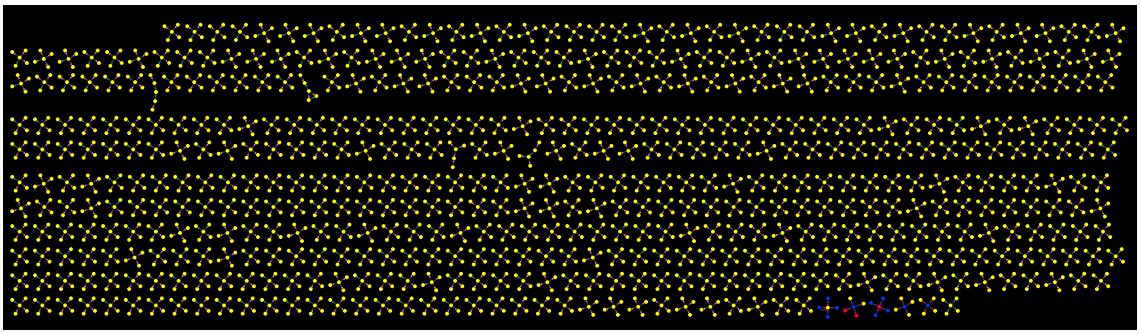


Figure IV.5: Depiction of the stratification based on the node degree (Degree = 4)

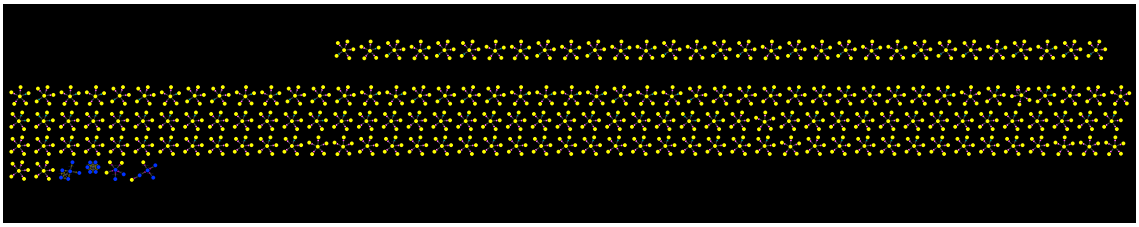


Figure IV.6: Depiction of the stratification based on the node degree (Degree = 5)

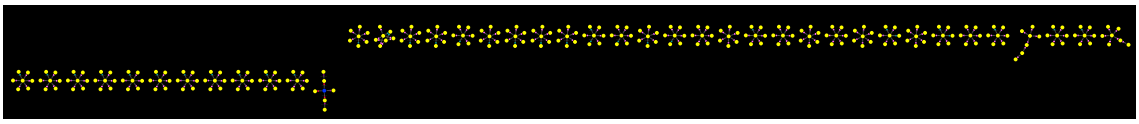


Figure IV.7: Depiction of the stratification based on the node degree (Degree = 6)

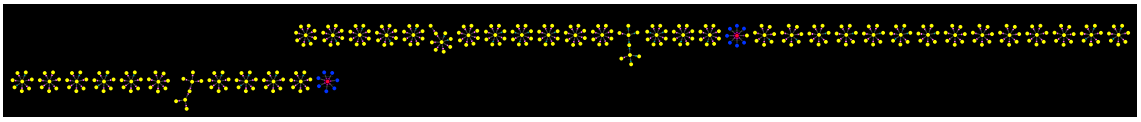


Figure IV.8: Depiction of the stratification based on the node degree (Degree = [7,8])

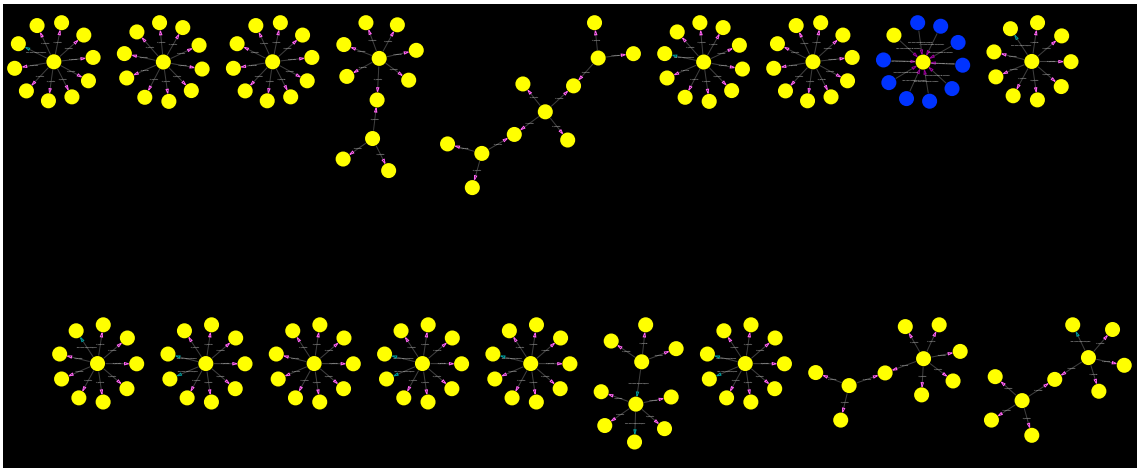


Figure IV.9: Depiction of the stratification based on the node degree (Degree = [9,10])

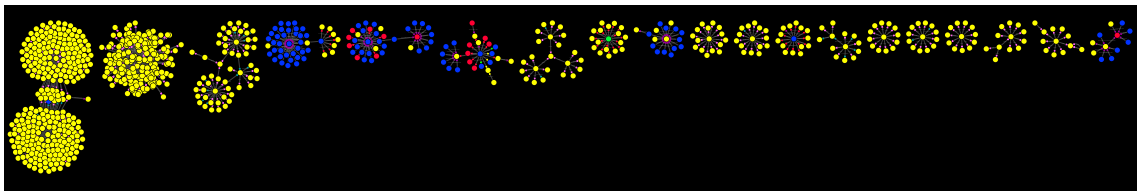


Figure IV.10: Depiction of the stratification based on the node degree (Degree > 10)