Big Data Mining For Smart Cities

Mystakidis Aristeidis

SID: 3306170007

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Computer Science – Mobile and Web Computing

DECEMBER 2019

THESSALONIKI – GREECE

INTERNATIONAL HELLENIC UNIVERSITY

Big Data Mining For Smart Cities

Mystakidis Aristeidis

SID: 3306170007

Supervisor:	Assist. Prof. Christos Tjortjis
Supervising Committee Members:	Prof. Agamemnon Baltagiannis
	Prof. Christos Berberidis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Computer Science – Mobile and Web Computing

DECEMBER 2019

THESSALONIKI – GREECE

Acknowledgment

I would like to acknowledge Professor C. Tjortjis for his methodical and continuous guidance during both the thesis preparation and the MSc in Mobile and Web Computing and for inspiring my interest in the development of data science and engineering technologies.

I would like also to thank my current company EMISIA SA (https://www.emisia.com/) for providing access to significant data regarding the thesis development.

Last but not least, I would like to thank my family and my very close people for their support during the preparation of the dissertation. Without them, this work would be far more difficult to complete and for this, I am extremely grateful.

Abstract

This dissertation was written as a part of the MSc in Mobile and Web Computing at the International Hellenic University.

These days, the volume of data information in the globe always seems to grow continuously with no turning back point. With enormously powerful machines, computers, phones and tablets saving information that earlier would be trashed is too simple. Affordable multi-terabyte drives make it very easy to delay choices over what to do with all this information. Users and companies are just purchasing another drive and saving everything. Increasingly popular electronics document users' decisions, financial choices, market habits, trips and photos. Users can access data from all way around the world, almost every record in a system, database or framework. The Internet and social media overwhelm more and more users with data information.

Furthermore, more and more data are been stored regarding cities and urban areas. This information is critical for automatizing several procedures in these areas such as road traffic control. With urban living increased exponentially the last century, road traffic congestion has become one the most significant problems of this era.

There is no panacea, but as far as the solution for this problem is concerned, analyzing the congestion data for future traffic prediction could do a significant difference.

The current thesis is a presentation, analysis and construction of a model for predicting the traffic congestion for Tsimiski street in the city of Thessaloniki using data mining and machine learning algorithms, along with python, sql and gis technologies.

Mystakidis Aristeidis

30/12/2019

Contents

ACKNOWLEDGMENT ..... **III**

ABSTRACT ..... **IV**

CONTENTS **5**

1 INTRODUCTION **7**

2 DATA MINING **8**

2.1 GENERAL MEANINGS - DATA MINING 8

2.2 ERA OF DATA MINING 10

2.3 DATA MINING MODELS 11

2.3.1 Predictive Model *11*

2.3.2 Descriptive Model *11*

2.4 FUNCTIONALITY OF DATA MINING 11

2.4.1 Clustering *12*

2.4.2 Classification *13*

2.4.3 Association Rules *14*

2.4.4 Characterization and Discrimination *15*

2.4.5 Outlier Analysis/ Deviation Detection *15*

2.4.6 Frequent Patterns Mining *16*

2.5 BIG DATA 17

2.5.1 Big Data meaning *17*

2.5.2 Big Data Vs *18*

3 SMART CITY **20**

3.1 SMART CITY BASIC TERM BACKGROUND 21

3.2 SMART CITY'S BENEFITS AND OPPORTUNITIES 23

3.3 TRAFFIC CONGESTION MANAGEMENT 25

3.3.1 Smart traffic lights *26*

3.3.2 Traffic congestion prediction and related work *26*

4 CASE STUDY AND METHODOLOGY 29

4.1 METHODOLOGY EXPLANATION 29

4.2 DATA EXTRACTION PHASE 30

4.2.1 Sources 30

4.2.2 Storage 32

4.3 DATA PREPROCESS PHASE 35

4.3.1 Timestamp 35

4.3.2 Road length and sequence 36

5 MACHINE LEARNING ALGORITHM 37

5.1.1 Selected data 37

5.1.2 Initial Prediction Algorithm 40

5.1.3 Improving Prediction Algorithms 45

5.1.4 Decision tree vs Logistic regression algorithms 49

6 CONCLUSIONS 54

6.1.1 Summary and results 54

6.1.2 Challenges 54

6.1.3 Future work 55

7 BIBLIOGRAPHY 56

APPENDIX 64

1 Introduction

It is predicted that by 2050 more than 67 percent of overall population will live in urban areas, according to latest United Nations report (2018). The report also indicates that from 1950 to 2018, urban living exponentially increased from 751 million people to 4.2 billion people.

This overpopulation with the large amount of data which record procedures and functionalities of smart cities, broaden the horizon of Data science for smart cities. This data can be analyzed for the sake of each city's procedures and functionalities' optimization. One these procedures that needs to be optimized is the traffic flow.

The existing dissertation is a demonstration, study and development of a prototype model to forecast traffic congestion in Thessaloniki City with Tsimiski Street as an example to be analyzed.

Before the case is described, a literature part is preceded and several major key words like *'data mining', 'machine learning', 'smart city', 'traffic congestion'* etc. as well as similar problem were explained and analyzed.

After that, the case is explained, data origin, technologies that were used and several difficulties are presented with a comprehensive preprocess phase. Moreover, the selected data mining algorithm for the traffic forecasting is demonstrated with Tsimiski Street as a mockup example (blueprint) and an evaluation.

Finally, an evaluation of the algorithm is conducted comparing the result with other algorithms.

2 Data mining

2.1 General meanings - Data mining

The readiness of sufficient data in nearly every sector and the eagerness to extract helpful knowledge and information from it have been substantiated as the primary motivation that has pulled scientists' eyes towards data mining and machine learning in past few years. For apps ranging from simple business management to complex engineering architecture to science exploration, the understanding of the information extracted can be unthinkably beneficial. Data mining is the study and examination of enormous datasets, with the goal of uncovering important trends and rules that have not been found before. The main goal is to utilize the computer's data processing capacity with the human's brain's ability to detect patterns (Han & Kamber 2001).

Tan, Steinbach, Karpatne and Kumar (2018) also stated that data mining is the process of extracting valuable information in large datasets automatically. Data mining algorithms are used to scan large amounts of data to discover new and useful trends that may otherwise remain hidden. These algorithms also provide the ability to foresee future observation outcomes.

The most widely accepted terminology of "data mining" is the discovery of "models" for data, based on Rajaraman, Leskovec & Ullman (2014). A"model" may be one of the following things.

- Statistical Modeling
- Machine Learning
- Computational Approaches to Modeling
- Summarization
- Feature Extraction

There are different types of algorithms that can categorize the data, either automatically or semi-automatically, and seek for the aforementioned patterns within the data (Agrawal & Srikant, 1994). Such patterns are used to provide multiple sets of rules. The discovered patterns should have meaning so that they can open the way to several advantages such as economic growth, decision making, marketing research, project

management, etc. Also, considerable amounts of data are necessary to get all these meaningful trends. Data mining takes advantage of developed models from machine learning and statistics to deal with this enormous data. Data mining offers perspectives, data interpretation, and expertise. It also offers the ability to forecast future results.

Data mining relies on the side of three different segments. First, statistics, then artificial intelligence and finally, machine learning (Zhou,2003). Statistics serves as the basis for multiple data mining methodologies, such as variance, standard deviation, regression, standard distribution, discriminatory analysis, confidence intervals and cluster analysis and so on. Artificial intelligence is based on heuristics, as it attempts to use statistical utilities as a human way of thinking. Thousands of high-end industry applications use numerous artificial intelligence methods, such as using query optimization algorithms for relational databases. Data and their relationships are analyzed using these (Wu, 2004). Machine learning (ML) aggregates artificial intelligence and statistics (Michalski, et al., 1998). Machine learning focuses on designing algorithms that can train themselves and evolve as new types of data are encountered. It also uses many methods for statistical analysis. Using these methods, individuals can make different decisions depending on the data's dominance.

According to Bishop (2006), machine learning (ML) is the scientific research of statistical models, algorithms and mathematical methods used by electronic systems to execute a particular task without using explicit instructions, focusing instead on trends, patterns and inferences. It is regarded as an artificial intelligence branch. ML algorithms develop a sample data-based mathematical model, referred as "training data," to make forecasts or decisions without been explicitly developed to do the task.

Data mining is also effective to summarize the underlying relation in data in addition to predicting future observations. Data mining can process and collect data from various data storage systems such as txt, xlsx, xml, json and csv files, databases, servers, data warehouse, transactional data, multimedia data, lists, internet, stream, time series, graphical, spatiotemporal, charts, social and knowledge networks, etc.

2.2 Era of data mining

Today, data mining is dominant among data analysts and statisticians, and other science communities.

The era of data mining implementations initiated near 1980 primarily by science-driven tools embedded in solo chore (Piatetsky-Shapiro 2000).

Term invention arises in the 1990s. As mentioned, the source of data mining on the side of artificial intelligence, machine learning and statistics.

Piatetsky-Shapiro invented the term 'knowledge discovery in database' (KDD) during the first KDD workshop in 1989. Recognition of data mining and machine learning should not be surprising, due to the scale of data gathered from multiple available sources, the data obtained are very complex to be analyzed manually, and several times automated data analysis assisted by classical statistics and machine learning could be of concern once the process is comprehensive but the knowledge gathered is made up of problematic entities. The saved, huge data volume gathered from multiple sources and held in vast and varied archives.

Data mining algorithms were developed primarily for numeral data initially, but it was expanded to all types of data such as multimedia, text, storage, image and internet etc. Originally, data mining started with individual database analysis, and progressively data mining techniques have been developed for conventional and relational databases, flat files and data warehouses. Subsequently, various algorithms evolved to process organized and unorganized data with the combination of machine learning techniques and statistics. Because of its tremendous achievement in terms of application scope, scientific progress and understanding, the field of data mining has been increasing continuously. The ever-increasing complexities in many industries and new technologies have introduced new challenges for data mining. The various areas of interest include heterogeneous data formats, networking development, computing power, fields of scientific research and business development demands, etc. Fayyad at 1996, mentioned that KDD will continue to develop in various sectors such as databases, artificial intelligence, machine learning, software discovery, scientific discovery and information retrieval, etc. (Fayyad et al., 1996). The different tools from all these areas are used throughout the cycle of knowledge discovery.

2.3 Data Mining Models

As per convention, data mining model are mainly of two types i.e. predictive model and descriptive model.

2.3.1 Predictive Model

The concept behind such models is to develop a process using the results of the existing data and to predict the impact of new and unknown data sets (Han, Kamber & Pei, 2012) (Witten, Frank, Hall & Pal, 2017). A financial institution, for instance, has the required data on previously issued loans. Independent variables in this data are the attributes of the consumers to whom the loan has been given and the dependent variable would be if the loan is returned or not. In this manner, the framework generated by this data would help to decide if or not the loan should be issued to the consumer. Classification and deviation detection are used for predictive data mining regression applications.

2.3.2 Descriptive Model

Such models' main objective is to determine patterns (correlation, trends) that describe the data relationship under laying. Descriptive data mining is usually used for generating correlation, cross tabulation and frequency (Witten, Frank, Hall & Pal, 2017). Descriptive model can also be illustrated to bring out important data trends, identify previously undiscovered patterns, and identify informative data subcategories. To classify, for instance, webpages that are visited by the type of user. Underneath the descriptive model, clustering, association rules, summarization and sequential pattern mining are used.

2.4 Functionality of Data Mining

Data mining's role is to extract the information and useful functionalities from the data. There are plenty of features available to detect patterns. Data mining is specializing in useful data trends. Such patterns are basically unknown at the initial stages but can potentially be used. Data mining provides a number of functions. Depending on the domain field and the type of information to be obtained, a specific type of feature can be assigned. These functionalities can be used to exploit different kinds of information

such as association rules, clustering, classification, characterization, rule discrimination, deviation and predictive analysis, etc. The utility of data mining is rich and extensive; it can support many purposes and areas (Tan, Steinbach, Karpatne & Kumar, 2018)

2.4.1 Clustering

The role of clustering (Figure 1) is to categorize a collection of items so that related objects are held in the same clusters (Stutz & Cheeseman, 1996) (Ester & Kriegel, 1995) (Ng & Han, 1994). It is an important data mining tool that is widely used to analyse statistical data like pattern recognition, machine learning, information retrieval, biostatistics and image analysis. Essentially, in clustering, different types of partitions are generated (Witten, Frank, Hall & Pal, 2017). then participated variables are held in those partitions on the base level of similarity that is based on certain metric.

Clustering follows the unsupervised methodology; so, categories / classes / groups are not already established in this method. As far as unsupervised methodology is concerned, grouping of entities is performed on the basis of the proximity or similarity of collection of records. In this learning, the model itself will determine the classes, picking, for example, an attribute based on the data provided and will classify the data on the basis of that. It then selects one other attribute for partitioning the data and so on. Many objects are represented in clusters that are mutually exclusive.

Clustering is a high effective tool in similarity terms. In the quantitative measure, it has the ability to interpret any instinctive measure of a similar nature (Zhang, Ramakrishan & Livny 1996). There are plenty of prospects to create clusters. One is to establish rules for participation in the same group based on the level of similarity among members. Another prospect is to develop a set of functions to calculate partition belongings as the method of partition parameter. Figure 1 illustrates an example of clustering.

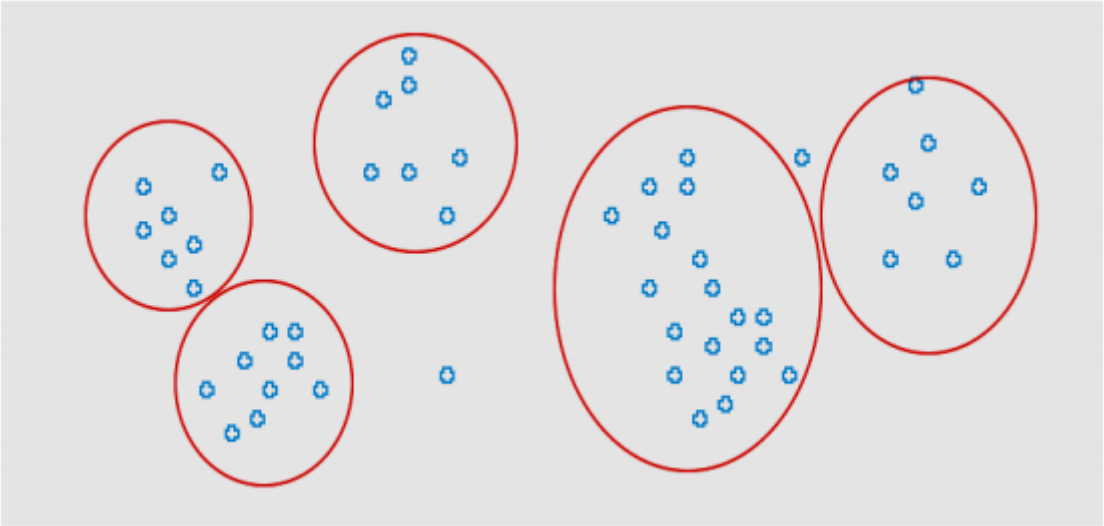

Figure 1: Clustering

2.4.2 Classification

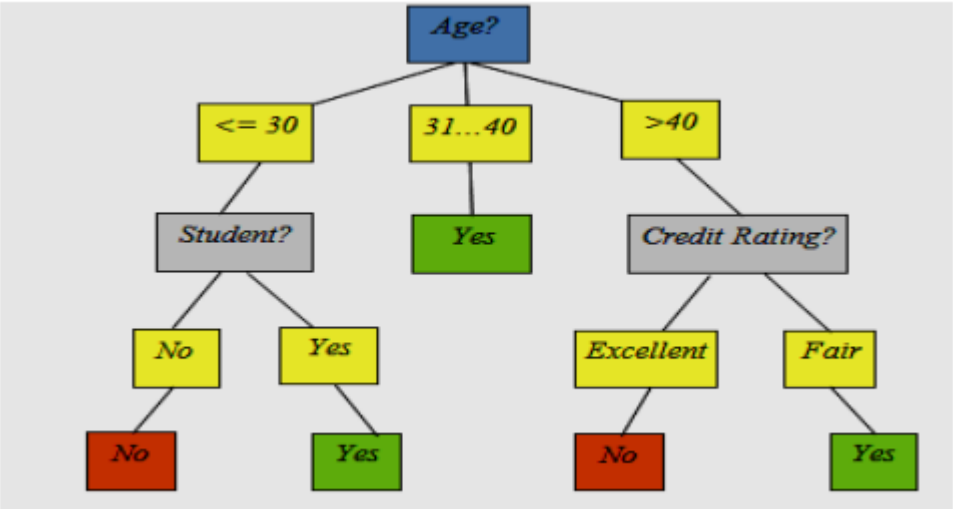

Figure 2: Classification

The data mining classification method is capable of processing a large amount of data according to Han, Kamber & Pei (2012). Classification assigns objects to desired categories in a dataset. In each record in the dataset, classification recognizes the targeted category. Classification adds a category tag to a group of unclassified instances. This phase is called supervised learning because the classification tag is given to all the training data. Classification is used to recategorize data objects into specific predefined groups (Weiss & Kulikowski, 1991). First, the training samples are given in

this type of project. A model is developed based on these training datasets that works for values of new other attributes.

The classification method is used as the identification of trends in financial markets by information disclosure frameworks and thus immediately identifies the interesting pattern of large databases. The strategies of classification deduce a model from the dataset. The data consists of various types of attributes that indicate any tuple's specific category and these attributes are titled predicted. In addition to all these attributes, there are remaining ones that considered as the predicting attributes. A variable aggregation defines a class for the desired attribute.

Firstly, regarding the learning process of the classification rules, the analyst must identify the criteria for all classes. The program determines the class based on these requirements. The data mining program then creates the specifications for these categories. Originally, the prerequisite of a process is a tuple or case with definite established attribute values so that it can forecast the case-related category. After identifying the class, The model becomes capable of predicting the patterns that define the category, thus becomes successful in finding the description of each class.

In this scenario, the description will apply to the training set's attributes that are beneficial to forecast because it will consider similar properties that satisfy the description and disregard the others. Figure provides an example of the decision tree for classification.

Several data mining classification methods exist such as Decision Tree (Figure 2), Nearest Neighbor, Naïve Bayes, Logistic regression, Neural Networks, Rule-based, Bayesian Belief Networks, Ensemble Met and Support Vector Machines.

2.4.3 Association Rules

Association rule mining is a method of discovering interesting relationships in databases between records (Agrawal & Srikant, 1994). This distinguishes specific rules from datasets utilizing different types of metrics. There are two elements to an association rule, the one is antecedent and the other is consequent. Antecedent is the item detected in the dataset and consequent is the item identified with the antecedent in the aggregation. Association rules establishment is done through examination of the data for repeated trends and then utilization of the principle of support and confidence to assess the most important relations.

Support is the number of records that exist in the database, and trust relies on support, it is the percentage of the transaction that includes support record and their dependent record. For instance 'if 80 percent of all records containing products A also include products B'. So, A plays the role of antecedent and B is the consequent. The value of B depends on A. Support is the individual number of products A and B. The value of B relies on A and its confidence value is 80 percent.

Association rules can be categorized into two classes.

1) Single-level rules of association

2) Multi-level rules of association

2.4.4 Characterization and Discrimination

Data characterization is a description or abstract type of a specific data class' general attributes (Witten, Frank, Hall & Pal, 2017). The abstraction takes place, regarding data characterization, upon request of the users' special requirement. Normally by using a query the data could be retrieved.

Summarization is the process of finding a short description for a section of data (Fayyad, et al., 1996). There are many advanced summarization methods and they are commonly used to conduct data processing and to support in the production of automated documents.

As far as data discrimination is concerned, class data target objects are contrasted with objects of one or many different classes in terms of specific generalized characteristics (Pitt & Nayak, 2007) (Dash & Liu, 1997).

2.4.5 Outlier Analysis/ Deviation Detection

Outliers are objects that do not comply with the general model or data behavior (Han, Kamber & Pei, 2012). When outliers are identified in the database, other data mining procedures are used to throw them away before processing

Outliers generally reflect exceptions or noise. Figure 3 shows outer analyzes, with R representing outlier data. Deviation detection is used to detect the major changes in data from previous normalized or calculated values (Fayyad, et al., 1996).

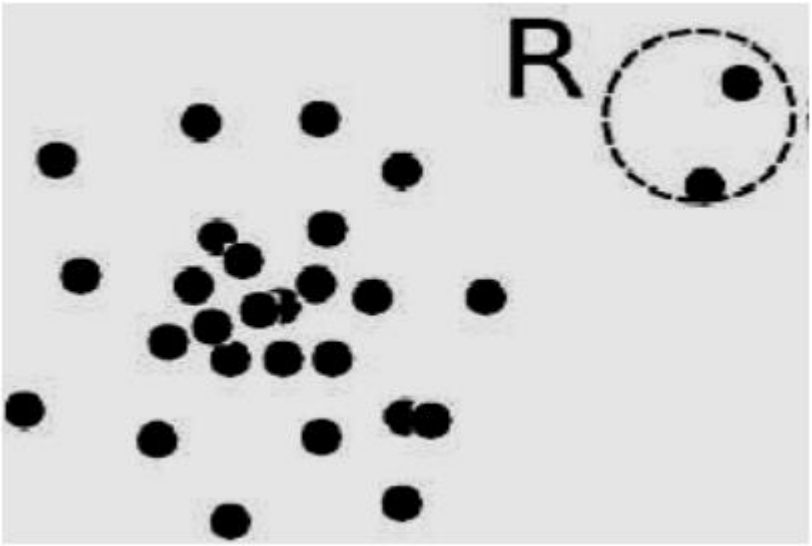

Figure 3: Outlier Analysis / Deviation Detection

2.4.6 Frequent Patterns Mining

Frequent patterns are defined as patterns that occur repeatedly in the data (Agrawal & Srikant, 1994). Patterns can be assumed to be the itemsets, sequences and subsequences. An itemset that reaches the minimum support criteria is either a frequent pattern or large-itemset. An item's support is the amount of that item appearing in all transactions. For instance, if items A, B and C exist simultaneously in seven out of ten transactions, the meaning is that itemset {A, B, C} has 70 percent support. Discovering these frequent patterns plays an important role in the relationship between the mining association link and many other useful data relationships. Consequently, frequent pattern mining has become a significant task in data mining and reflected a lot on data mining research. Finding frequent patterns with various real-world applications is a rather significant data mining concern. In many business decision-making processes like catalog layout, cross marketing, consumer purchasing activity etc., the identification of frequent pattern makes a difference. Using frequent itemsets, association rules are created. Data mining functionality covers a variety of apps and allows different types of information to be discovered and abstracted at various levels.

2.5 Big Data

Big data, business analytics, 'smart' working and 'smart' living are becoming increasingly important in past couple of years. Although these discussions are primarily technically oriented, organizations are researching new ways to effectively deliver large amount of data to create and extract value for users, companies, societies, and governments (McKinsey Global Institute, 2011). Whether this is data mining or machine learning to forecast individual action, customer behaviour, sports analytics, traffic flow or pandemic outbreaks, Big Data is increasingly becoming a method that not only investigates patterns but can also provide the forecasting probability of an occurrence.

Groups and individuals have stepped onto this path to use continuously increasing data, mostly in storage capacity of terabytes and petabytes, in order to accurately estimate results with higher accuracy. Organizations like United Nations are developing projects that use new digital data resources, like phone communications and online payment systems, with actual-time data collection and data mining to boost development efforts and identify potential threats through underdeveloped economies.

Although Big Data seems to have become increasingly common as a commercial term, hardly any published scholarship is available that addresses the issues of using such tools.

2.5.1 Big Data meaning

Big data refers to databases, measured in terabytes and above, that are too large and complex to be used effectively on conventional systems. (Kubick, 2012)

Big data is formed by a growing variety of sources such as website page views, smartphone transaction processing, customer-generated content and social networking sites, as well as intentionally created content via sensor networks or business activities such as sales queries and purchases. However, biotechnology, medical services, operations research, engineering, environmental media, gaming industry, finance and other sciences contribute to the pervasiveness of big data. Such data involve the use of powerful statistical techniques to uncover patterns and trends among these extremely large socioeconomic repositories as well as between them.

Important insights learned from this data exploitation can usefully help define official statistics, surveys and archival data sources which continue to remain static, adding

depth and perspective and while processing in in real time, reducing time gaps. It is said that the key is in big data's 'largeness' that always gets the attention of scientists to the size of the dataset. However, there is a rapidly growing dialog between specialists that 'big' is no longer the defining parameter, but how 'smart' it is, for example the insights that the amount of data can provide sufficiently.

In any case, either large or smart, the use of big-scale data to predict and determine behaviour and results is earning currency in the practice of corporate and government policies and in scientific fields in which social and physical sciences converge, recently referred to as social physics (Pentland, 2014).

2.5.2 Big Data Vs

Also, certain big data features and characteristics are labelled as big data management Vs as show in Figure 4.

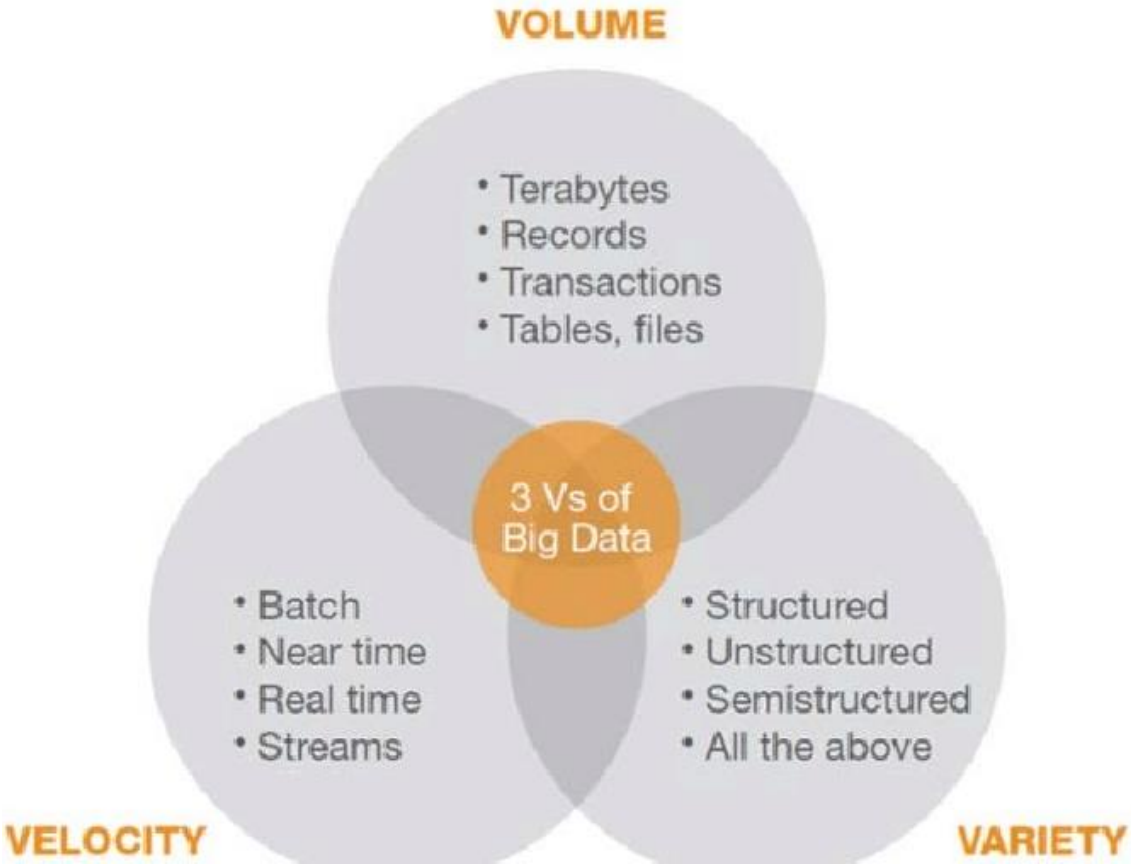

Figure 4: Original Big Data Vs

There are three (1,2,3) primary (Figure 4) and two (4,5) additional Vs according to Fan and Bifet (2013):

1. Volume: applies to the data size generated by all resources.
2. Velocity: applies to the velocity of data production, storage, analysis and processing. Recently, a focus is being set up to assist real-time analysis of big data.
3. Variety: references to the various kinds of data produced. It has become frequent for most data not to be structured and not feasibly categorized or analysed.
4. Variability: applies on how data structure and data meaning are continuously evolving, particularly while facing, for instance, with data collected by natural language analysis.
5. Value: applies for the potential benefits of utilizing big data technologies to an industry, depending on the way data collection, strategic planning and methodology of big data are utilized.

Figure 5: Big Data Vs

Some also note several extra Vs of big data representing a number of more topics (Figure 5). For instance, volatility, that applies to the formal data retention policy implemented from various sources. In addition, there is validity, that applies to software reliability, accuracy, and validation. Furthermore, there is veracity, that alludes to the precision and trustworthiness of the data recorded and the significance of the results produced for specific issues from the data. All big data's different features show the huge potential for gains and advances.

3 Smart City

Undoubtedly, the main strength of the big data concept is the high influence it will have on numerous aspects of a smart city and consequently on people's lives (Koutroumpis & Aija, 2013). Big data is growing rapidly, currently at a projected rate of 40 % growth in the amount of global data generated per year versus only 5 % growth in global IT spending. Around 90 % of the world's digitized data was captured over just the past years. As a result, many governments have started to utilize big data to support the development and sustainability of smart cities around the world.

That allowed cities to maintain standards, principles, and requirements of the applications of smart city through realizing the main smart city characteristics. These characteristics include sustainability, resilience, governance, enhanced quality of life, and intelligent management of natural resources and city facilities. There are well-defined components of the smart city, such as mobility, governance, environment, and people as well as its applications and services such as healthcare, transportation, smart education, and energy (Khan, Anjum & Kiani, 2013). To facilitate such applications and services large computational and storage facilities are needed.

A method to produce this kind of technologies is to depend on cloud computing and use the several benefits of utilizing cloud services to support big data management and apps in smart cities.

Continuously increasing working projects and studies in this sector have produced several publications that emphasized the significance of big data in assisting smart city apps and services. However, several studies explored a few of the concerns of using big data in smart cities (Kitchin, 2013), (Townsend, 2013), (Batty, 2013) (Llosa, Martinez, Domingo-Prieto, Angles & Vilajosana, 2013).

3.1 Smart City Basic Term Background

The description of the smart city has dissimilar meanings from the people's point of view and technology's point of view. That's evident as governments establish new smart cities projects and they create divergent perspectives around the smart city. Despite the fact that smart city trends are widespread worldwide, the meaning is elusive. In other words, a distributed concept of a smart city is still not available, and a generic international connotation is hard to identify. Nevertheless, most meanings outline common elements, characteristics and components that could define smart cities' points of view.

Definitions involve improving the quality of life for a specific sector–residents–by using hardware equipment, software applications, ICT networks and information on various urban sectors and services for information systems. It might also include various elements of the city such as transport, education, natural resources, facilities, energy, universal healthcare, state department and public safety and security.

According to Neirotti, De Marco, Cagliano, Mangano & Scorrano (2014), "The concept of Smart City (SC) as a means to enhance the life quality of citizen has been gaining increasing importance in the agendas of policy makers. However, a shared definition of SC is not available and it is hard to identify common global trends".

Moreover, Khan, Anjum & Kiani (2013) mentioned that "A smart city is a city which invests in ICT enhanced governance and participatory processes to define appropriate public service and transportation investments that can ensure sustainable socio-economic development, enhanced quality-of-life, and intelligent management of natural resources".

Also, according to Aguilera, Galan, Campos & Rodríguez (2013) ''Smart city is a very broad concept, which includes not only physical infrastructure but also human and social factors".

What is more, Kitchin (2014) defined that smart city is "A city that monitors and integrates conditions of all of its critical infrastructures, including roads, bridges, tunnels, rails, subways, airports, seaports, communications, water, power, even major buildings, can better optimize its resources, plan its preventive maintenance activities, and monitor security aspects while maximizing services to its citizens".

From all the meanings provided, depending on the author's point of view, the smart city can be described as an integrated living solution that brings together several aspects of life such as power, transportation, social factors and construction in a smart and efficient way to enhance the quality for city's residents. However, the descriptions also rely on the future by highlighting the value of generations to come regarding the development of resources and applications. It can be noted that these aspects concentrate on each smart city plan irrespective of the size, location and resources available.

Generally, governments worldwide are most often worried about the price of developing a smart city because of the different economical resources and lack of natural or human resources. One of the obstacles of developing and operating a smart city is the accessibility and volume of these resources and their capabilities. A further obstacle is the regulatory frameworks which might have a significant impact on the odds of success. What is more, there are also technical challenges that demand highly advanced innovative solutions to top of all that. New innovative technological advances, on the other hand, should assist reshape these challenges into opportunities.

Data are produced from numerous sources, which contributes to the creation of the aforementioned big data. Information sources are everywhere, mobile devices, game consoles, laptops, weather detectors, images, or even humans. Over the past decade, numerous technologies such as social networking websites, electronic images and videos, financial transactions, marketing apps, gaming, sports and many others have assisted boosting data cloud generation (Michalik, Stofa & Zolotova, 2014), (Khan, Anjum & Kiani, 2013).

In addition, several possible applications for big data to address issues straight from the source are available as well as analysis for deeper insights through data mining, data analytics and machine learning. In order to facilitate this increasing demand for resources to assist big data analytics, the Cloud jumped in and gave an innovative and effective solution. Cloud is an effective system for extremely intensive-resource apps for successful inter-applications coordination.

All of these matches really good for smart city technologies requirements and may assist to solve several of the possible obstacles. Smart cities have higher chances of being smarter than ever before via these innovative applications and achieving their targets being both efficient and effective.

Smart city apps produce vast quantities of data while big data technologies use this data to give proper information to improve smart city apps. Big data platforms can effectively archive, manage, and analyse data about smart cities apps and generate insights to improve various smart city utilities. Furthermore, big data may also assist decision-makers manage any improvement in smart city infrastructure, resources or regions.

3.2 Smart City's benefits and opportunities

Today, most cities are vying to be smart cities in financial, environmental and social terms, hoping to extract some of the benefits. As a consequence, they may capture opportunities created by big data analytics in smart city apps. There several advantages and opportunities following that could help to make the decision to transform or upgrade a city into a smart city. By making such decision, increased levels of stability, durability, and governance can be accomplished. Besides that, there is the improved quality of life of the resident by implementing smart infrastructure and natural resource management (Khan, Anjum & Kiani, 2013). Several of the advantages of having a smart city include:

1. Effective use of resources: with several resources being either sparse or quite costly, it is crucial to implement strategies in order to use these resources more efficiently and more controlled. It could be helpful to begin with technological systems like Geo Information System (GIS) and Enterprise Resource Planning (ERP) (Al-Hader & Rodzi, 2009). Through monitoring systems at operation, identifying failure spots and efficiently distributing resources will be simpler whilst managing costs, increasing power and use natural resources. Furthermore, one crucial element of smart city implementations is that they are engineered for interconnectivity and data gathering which can also facilitate better collaborated apps and services.
2. Higher life quality: Smart city residents can have a higher life quality with better services, more optimal living and working models, as well as less waste (in both time and resources). This would be the outcome of higher organising of working / living spaces and facilities, more effective transport infrastructure,

faster and better utilities, and the accessibility of adequate data to take better decisions.

3. Greater standards of openness and transparency: the desire for better supervision and management of the various aspects and technologies of smart cities would result to increased levels of openness and interoperability. The trend will be information exchange and resource sharing. This will also increase the accessibility of data for everybody who is interested. This should motivate among the parties cooperation and communication and create more utilities and apps that further promote the smart city idea. For instance, the United states government which, in the name of transparency and openness, gathered and published a broad range of data, papers and information.

To accomplish these benefits, higher standards of complexity, participations are needed in aspects of applications, resources and individuals involved. There are ways to accomplish such advantages; but there is a need for innovation in more software, improved development strategies, and efficient utilization of big data. Moreover, it is necessary to establish policies to guarantee data reliability, higher levels of quality, security, confidentiality and data control, and to utilize data documentation principles to establish guidelines on the content and use of the datasets (Bertot & Choi, 2013).

Additionally, technological advances could be quite effective when taking into account energy resources, infrastructure management and safety, as well as natural resources with the primary objective of raising sustainability (Kramers, Höjer, Lövehagen & Wangel, 2014). Big data technologies in a smart city have the capability to support multiple sectors (Fan & Bifet, 2013). It allows generate greater user experiences and utilities that help businesses accomplish greater performance (e.g. bigger profits or greater market shares). They also help improve health services by developing programs for preventive care, methods for treatment, diagnosis and rehabilitation, maintenance of medical records and quality of patient care.

Big data technologies could greatly be beneficial to transportation infrastructure to automate and improve routes and schedules, satisfy various requirements and become ecofriendlier. The implementation of big data technologies involves the assistance of a great network for ICT (information and communication technologies). ICT promotes smart cities technologies because it really gives useful alternatives and completely

unique approaches that could not be reached without its use. An instance in this case, would be the effective transportation management, by providing better methods to manage their operations from various sectors / areas in order to minimize travel costs (Kramers, Höjer, Lövehagen & Wangel, 2014).

Many instances involve efficient water resource and waste disposal management through the implementation of technologies to operate these kinds of systems efficiently. Waste disposal management, for instance, includes activities like collect, dispose, recycle and recover (Neirotti, De Marco, Cagliano, Mangano & Scorrano, 2014), that could all be handled effectively utilizing ICT strategies. Further instances cover latest engineering and construction technologies for maintaining good building condition and safer environment, risk assessment, security, dust emissions, employee's health and effective energy consumption.

In overall, by using information/communication technology and big data in most of these apps and services, a smart city could be more intelligent. Implementing information/communication technologies, big data and cloud computing technologies can help overcome a number of challenges such as offering tools for processing and analysis. This will also help to achieve great levels of innovation (Khan, Anjum & Kiani, 2013) and promote cooperation and interaction among smart city's various stakeholders. This could be achieved by establishing big data communities to operate as single organization to promote innovative and practical approaches for applications in fields such as education, healthcare, energy, law, research and development, climate, and safety/security

3.3 Traffic Congestion management

Big data mining and machine learning for smart cities utilization assists to solve production, transport, and traffic management problems in real-time approaches using frameworks and systems that are incorporated and provide data transfer efficiently through apps and stakeholders (Bertot & Choi, 2013). There are several cases of smart cities supporting big data mining technologies like Smart education (West, 2012) and Smart grid (U.S. Department of Energy, 2015), however a very important aspect is the Smart

traffic lights (Aguilera, Galan, Campos & Rodríguez, 2013) and the Traffic congestion management and prediction (Thianniwet, Phosaard & Pattara-Atikom 2009).

3.3.1 Smart traffic lights

A key feature of a smart city is efficient traffic flow management throughout the city, that could boost transportation networks flow and optimize traffic conditions for people and cities in general (Aguilera, Galan, Campos & Rodríguez, 2013). As the population grows, there are traffic issues, increased emissions, and environmental and economic issues. Because of the above, the utilization of smart traffic lights is among the most relevant strategies used by smart cities to come face to face regarding increasing traffic congestion problems. A strategy in order to provide enough data on traffic patterns, smart traffic lights and signals must be integrated throughout traffic grids. Every sensor could measure a particular traffic flow variable like vehicle velocity, traffic density, lights waiting time, distances, etc.

Based on machine learning algorithms, the model could make decisions depending on these variables and provide the lights and signals with the suitable specifications. Therefore, more information that this framework has, the more intelligent decisions it could make. As a consequence, it would be better to collect and analyse from all traffic signals around the city and develop smart decision mechanisms utilizing this data to deliver the best potential solutions in smart traffic lights. It includes the usage of Big Data analytics and Machine learning in real time. For instance, the introduction of smart traffic lights and signals developed by the Traffic21 project in Pittsburgh, Pennsylvania, USA, produced important results, reducing traffic delays and travel times, leading in emissions reduction by more than 20% (Kandappu, Koh, Daratan, Jaiman, 2018).

3.3.2 Traffic congestion prediction and related work

Real- or future-time traffic jam knowledge would be very important for congested and overpopulated locations. Intelligent or smart transportation system (ITS) could help develop certain congestion data reports via sophisticated prediction algorithms. Many programs were developed and introduced both by corporate and state agencies to collect

traffic information to supply ITS systems. According to Thianniwet, Phosaard & Pattara-Atikom (2009), many energies are based on selective setup of static detectors like loop coils and smart video processing recorders. That being said, considering the cost of the hardware, plus the setup and maintenance, the expenses of such technologies seem to be quite heavy.

In addition, such static detectors have always been exposed in most locations to severe weather events. What is more, it is hardly cost effective or technically feasible to deploy fixed detectors to cover all streets in major urban areas. Therefore, an alternate method of collecting traffic information and predicting traffic congestion with greater coverage at a reduced cost is required.

Estimation methodologies for the congestion rate can differ according on the characteristics of the data gathered. There are two different kinds of detectors that can indeed automatically collect traffic data: a fixed sensor and a mobile detector. Street cameras, speedometers etc., considered to be fixed sensors/detectors while mobile phones, vehicle GPS considered to be mobile sensors/detectors. The research of Pattara-Atikom & Peachavanish (2007) utilized the algorithm of the neural network from the mobile phone data gathered. this study utilized Cell Dwell Time / CDT, the moment a mobile phone connects to a mobile phone service antenna, offering a rough travel velocity. Thianniwet, Phosaard and Pattara-Atikom's research (2009) used another methodology of machine learning that was better suited to the data morphology.

GPS measurements could offer further reliable traffic information rather than the CDT results. This study implemented decision tree (J48) approach using a decision tree Classification algorithm on mobile sensors to identify road traffic congestion rates from GPS data. Getting data via mobile sensors could monitor much broader areas of traffic. The machine learning algorithm would indeed learn about vehicle's motion patterns. Fixed window sliding methodology was also utilized. The studies of Pongpaibool, Tangamchit, & Noodwong (2007) and Porikli & Li (2004) predicted the rate of traffic jams by implementing fuzzy logic and hidden Markov method, respectively, utilizing traffic camera information.

In addition, the studies of Lu & Cao (2003), Krause & von Altrock (1996) and Alessandri & Repetto (2003) explored numerous alternate approaches related to traffic congestion research. For instance, in several countries, as shown in the survey of Lomax, Tuner, Shunk, Levinson, Pratt, Bay & Douglas (1997) and Bertini (2005), the key criteria

utilized to describe traffic congestion rates are duration, velocity, size, quality of service, and the traffic signal periods that drivers need to stop for.

With that being mentioned, it is clear that a cost-effective way to deal with traffic congestion management is to utilize traffic congestion prediction methods with machine learning algorithms.

The next section described the methodology of the research.

4 Case study and Methodology

4.1 Methodology explanation

The purpose of this thesis was to deal with traffic congestion management problems regarding the city of Thessaloniki. The main strategy was to develop data mining/machine learning techniques for traffic congestion prediction in order to predict the traffic depending the street, the specific day and hour etc.

This would cause the driver to avoid high traffic on the street and to choose routes with better flow. Another advantage of utilizing this strategy could be that in a city, like Thessaloniki, where the many transportation problems occur the use of intelligent or smart transportation system based on the traffic prediction algorithms could make a difference. What is more, based on these algorithms, smart traffic lights mechanisms could be utilized on future studies.

The idea was to extract data from a specific api, get data from json array, use several technologies and techniques for preprocess and data mining – machine learning algorithms to forecast the result.

For this overall project three main tasks have been combined.

- Data extraction phase using python 3.6 programming language with Spyder interface development environment
- Data preprocess phase using Oracle database development and SQL programming language.
- Classification machine learning algorithm for traffic congestion forecasting using python's 3.6 scikit learn library with Jupyter Notebook interactive development environment

.

4.2 Data extraction phase

4.2.1 Sources

There were 3 different sources regarding the data extraction process.

1. As for the traffic congestion data, they were originated from http://opendata.imet.gr/dataset/network-congestion. This is a website that contains traffic data regarding speed, congestion, road links, travel times, historical data etc. about the city of Thessaloniki (Figure 6). The data were gathered from various sources (Mitsakis, Salanova, Chrysohoou, Aifadopoulou, 2015) (Mitsakis, Stamos, Salanova, Chrysohoou, Aifadopoulou, 2013) (Salanova, Chaniotakis, Mitsakis, Aifandopoulou, Bischoff, 2016).

In the downloadable json file for congestion, the provided data were Link_id;"Link_Direction";"Timestamp";"Congestion";

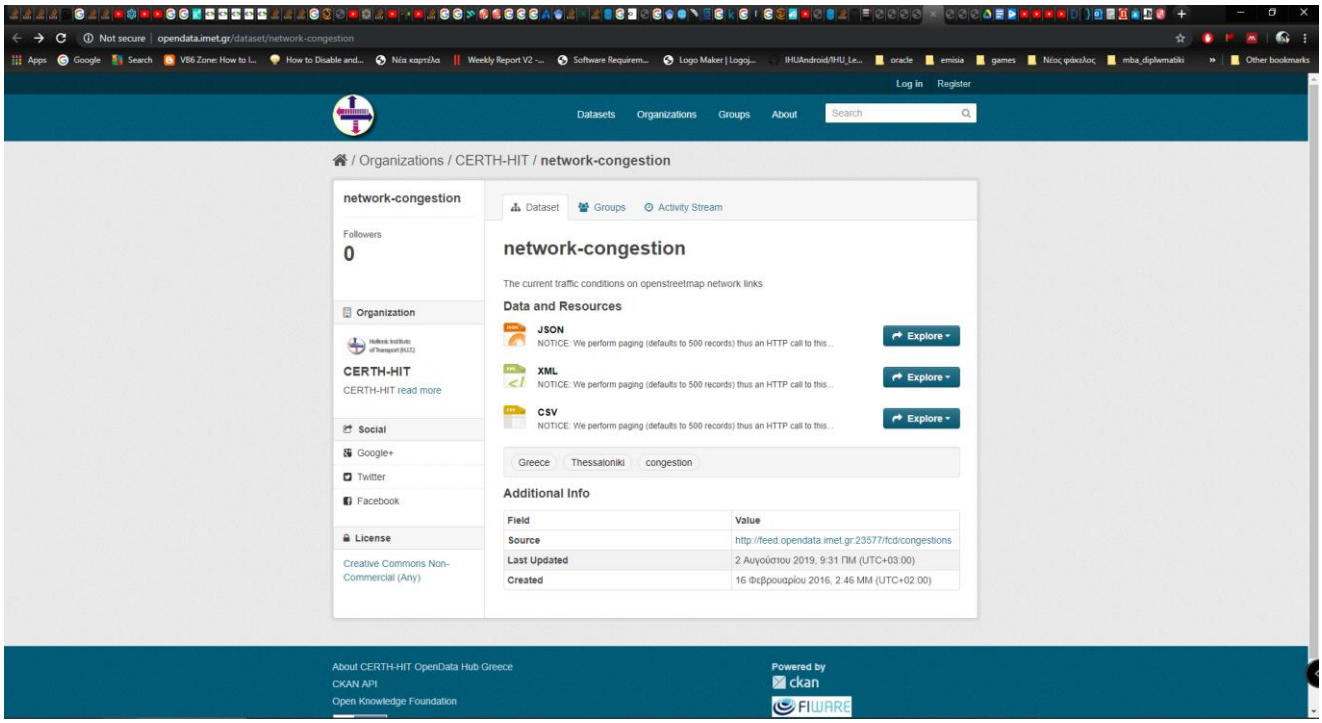

Figure 6: Website that contains traffic data

2. Emisia Database: This postgress database, provided by Emisia S.A. company, was created for an application called Wiseride. This App used to provide real time data from osmosis database (database that contains data from taxi drivers and other sources). Although these real time data were no longer accessible, the database con-

tains data about the Link_id‘s of part 1 and it could be very useful to have an idea about the street names of the top speed of each road.

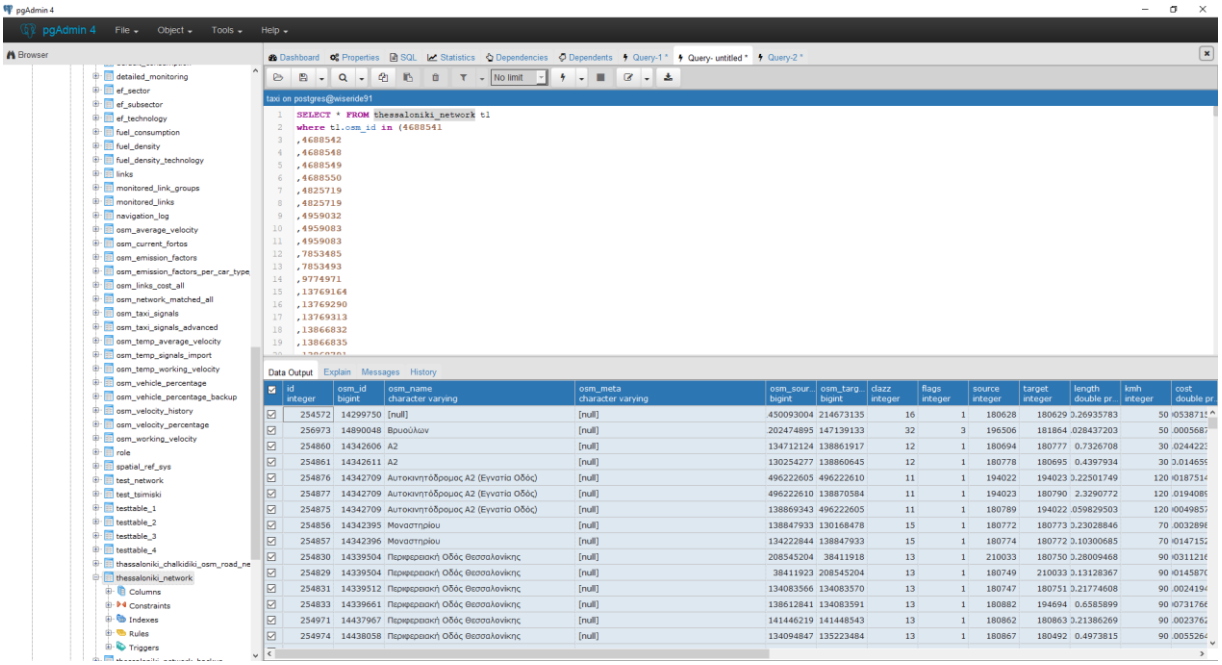

Figure 7: Emisia Wiseride Database

3. Openstreetmaps and Postgis database.

The data of part two can be visualized by QGIS 2.18 app in cooperation with open source opensteetmap.org.

OpenStreetMap (OSM) is a community project involving a world map that is public open available to access. The data generated by the project is regarded to be its main output instead of the map itself. OSM's creation and growth has been driven by restrictions on the use or accessibility of map data throughout much of the world, as well as the advent of affordable portable satellite navigation devices. OSM is regarded to be a notable example of voluntary information on geography.

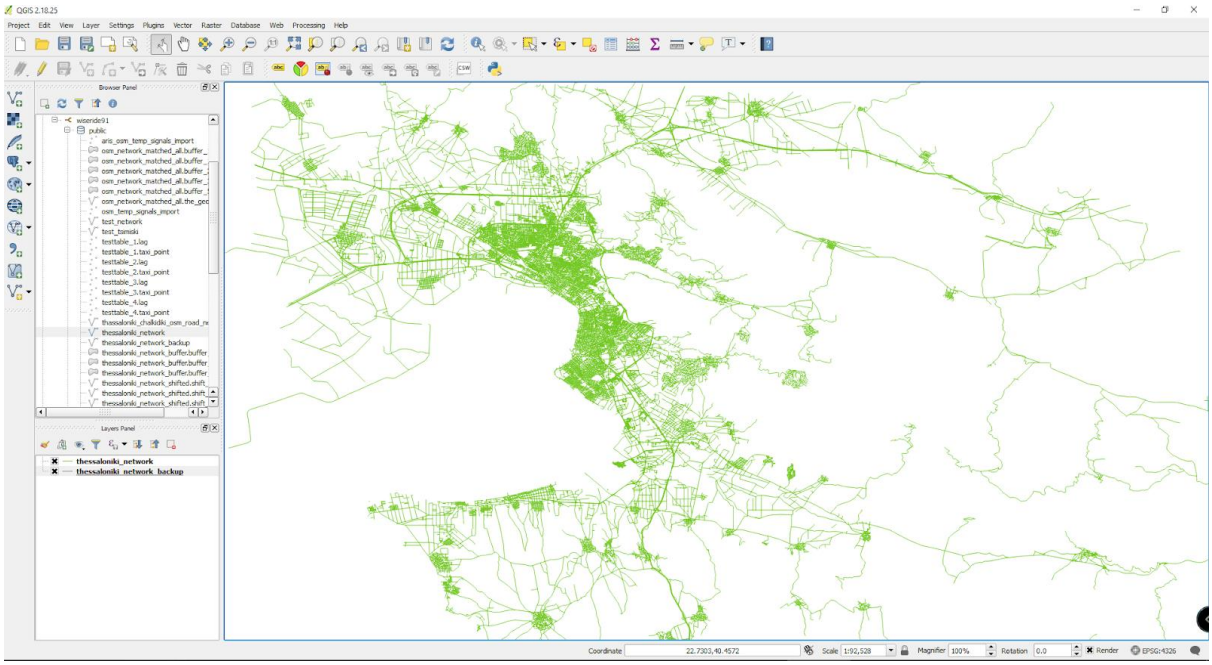

Figure 8: Thessaloniki via QGIS

The Link_id's of parts 1 and 2 were visualized in QGIS app as it can been seen in Figure 8.

4.2.2 Storage

For the purpose of this thesis, a localhost Oracle database has been developed in order to save the data.

Initially, a Thessaloniki table has been created containing data about the streets such as street names, the allowed top speed, the road type (main, secondary road etc.) as shown in Table 1. This table's data was originated from Emisia Wiseride application and contained several information, some of them unknown.

Table 1: Thessaloniki's important parameters detailed description.

Variable	Type	Description
id	NUMBER(15) UNIQUE NOT NULL	Id of the part of the road of the wiseride app
osm_id	NUMBER(15) UNIQUE NOT NULL	Id of the open street map part of the road. This matches with the traffic data, therefore this is the part of the road for the traffic con-

		gestion prediction.
osmname	VARCHAR2(100)	Name of the road
osm_source_id	NUMBER(15)	The first osm_id of the road
osm_target_id	NUMBER(15)	The last osm_id of the road
clazz	NUMBER(5)	Unknown
flags	NUMBER(1)	Unknown
source	NUMBER(15)	Unknown
Target	NUMBER(15)	unknown
Length	FLOAT	The length of each Id
Kmh	NUMBER(3)	Top speed
Cost	FLOAT	unknown
reverse_cost	FLOAT	unknown
x1	FLOAT	Map coordinates in X axis, where the Id starts
y1	FLOAT	Map coordinates in Y axis, where the Id starts
x2	FLOAT	Map coordinates in X axis, where the Id ends
y2	FLOAT	Map coordinates in Y axis, where the Id ends
category	VARCHAR2(10)	Type of the road

There were several important notes to be mentioned.

- An osm_id included several id’s
- In most of the cases an osm_id was the part of the road that describes the distance from one traffic light to the next traffic light
- An Id described the road for one block.
- There were several street categories as presented in the following Table 2.

Table 2: Street categories

KS_2K	Main collector road	2-way without median strip
DA_2K	secondary arterial road	2-way without median strip
KS_1K	Main collector road	One-way or two-way with median strip
KT600	unknown	unknown
KT100	unknown	unknown
DS_XX	Secondary collector road	unknown
DA_1K	secondary arterial road	One-way or two-way with median strip
KA_1K	main arterial road	One-way or two-way with median strip
KA_2K	main arterial road	2-way without median strip

Also, a very important table was the traffic congestion table (Table 3). This table has the following variables

Table 3: Traffic congestion table

Variable	Type	Description
LINK_TIMESTAMP_ID	VARCHAR2(100) UNIQUE NOT NULL	Unique Id of the open street map part of the road and timestamp
LINK_ID	NUMBER(10) NOT NULL	Id of the open street map part of the road. This matches with the osm_id from Thessaloniki table
LINK_DIRECTION	NUMBER(1) NOT NULL	If it has 2 directions, it is a two way road
TIME_STAMP	TIMESTAMP	Exact time of the response data. The provided data renewed every 15 minutes
CONGESTION	VARCHAR2(10)	Low, Medium or High

In order for this table to be filled, two data extracting methods were developed.

- In early stages of the project, the data were downloaded from http://opendata.imet.gr/dataset/network-congestion to csv file or copied to a txt file as Json array. Later, all these files were parsed using python 3.6 program-

ming language with Spyder IDE. As mentioned, each record of the json array contains information about Link_id, Link_Direction, Timestamp and Congestion. The Link_id, Link_Direction and Timestamp provided the unique LINK_TIMESTAMP_ID of this table, while the other values were extracted as they are.

- In order to have the best possible algorithm for traffic prediction, the amount of data needed were vast. To achieve this, a python algorithm has been developed in order to extract the traffic data from the aforementioned api and instantly insert them to the table without having duplicate records. The link was renewed almost every 15 minutes, so the algorithm could extract data every quarter of an hour. In total, 828897 records have been inserted.

The code that was developed for this phase can be viewed in Appendix. The overall extraction phase was done during the months August, September, October of 2019.

4.3 Data preprocess phase

Data preprocess phase was one of the most difficult parts of this project. In order to develop the classification for the forecast of traffic congestion all of these data were to have a necessary preprocess phase.

The idea was to combine parts 1,2,3 of the previous paragraph and turn continuous values to non-continuous ones. For example, timestamp was to be divided to different time categories like per 15 minutes or day of the week or shopping hours, as it can been seen in the next table.

One other problem had to do with the allowed max speed of each road. New parameters had to be created that would give non-continuous values providing details about the road, for example road length and sequence.

4.3.1 Timestamp

After interviewing, several employees and freelancers about the store and office hours and the importance on traffic congestion, an empirical evaluation had been done about shopping - office hours and traffic congestion.

The 24-hour cycle was divided into 4 segments. The segmentation would be different every day and would depend on store opening hours.

- From 09:00 to 20:59 in Tuesday, Friday and Thursday the stores were 'open'

- From 09:00' to 17:59 in Monday, Wednesday and Saturday the stores were 'open'
- From 07:30 to 08:59 in all days except Sunday the stores were 'opening'
- From 21:00 to 22:00 in Tuesday, Friday and Thursday the stores were 'closing'
- From 18:00 to 19:00 in Monday, Wednesday and Saturday the stores were 'closing'
- In all the other cases the stores were 'closed'

Another important segmentation done, was based on the timestamp providing the day of the week. The prediction was modeled based on a weekly cycle model.

4.3.2 Road length and sequence

As already mentioned, an osm_id or link_id (they were the same object) contained several simple ids (Table 1 - Table 3). It was easy to identify what ids contained a link_id using the database. However, the difficulty occurred when the sequence of osm_id was needed. To achieve that, the starting and the end point of each osm_id needed to be identified.

Also, the sequence and the location of simple ids was known via x1, x2, y1 and y2 (Table 1), so the end of each link_id was similar to where the spot of the end of id was while the link_id was not the same. A similar procedure was done to find the overall road length of link_id.

Also, as far as the road length is concerned, the average link_id road length was measured almost 200 meters, as it can been seen in the next image (Figure 9).

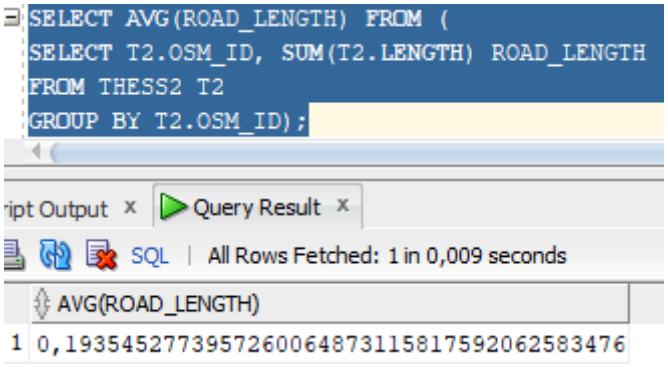

Figure 9: Average road length

For this reason, a new categorical value was created and charactizes if the link_id is more or less than 200 meters.

5 Machine Learning Algorithm

5.1.1 Selected data

As it can been seen in Figure 13 and Figure 14 the data selected were:

- Time
- Day
- Stores
- Congestion
- Osm_id
- Road larger than 200 meters
- Max kilometers allowed on the road
- Road category

The traffic prediction was done for one of the most important streets in Thessaloniki, the Tsimiski street, in osm_id that was between Venizelou street and Dragoumi street. The specific part of this street is generally not that long and very few traffic records of this osm_id were extracted.

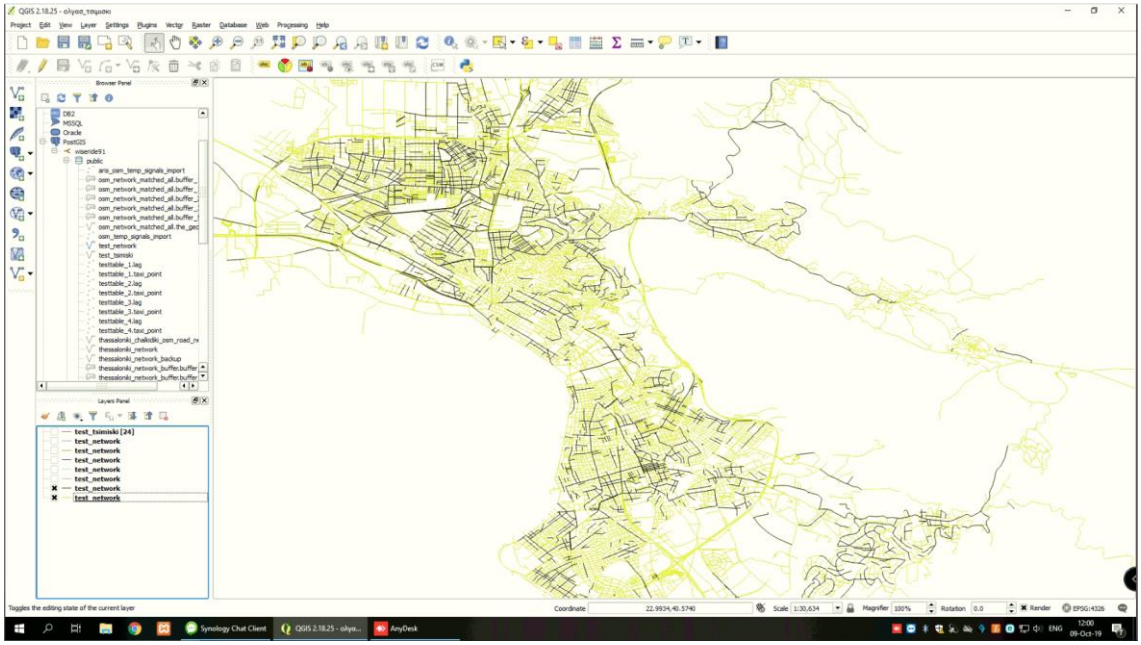

Figure 10: All OMS_ID’s vs OMS_ID’s that traffic data exists

Generally, this was a common issue regarding the extracted data for all the city of Thessaloniki. In the previous image (Figure 10), green color describes all the osm_id that Thessaloniki has, while brown color describes the osm_id that contained more than 10 traffic congestion timestamp records. As it can been seen there were several streets that did not contain enough data.

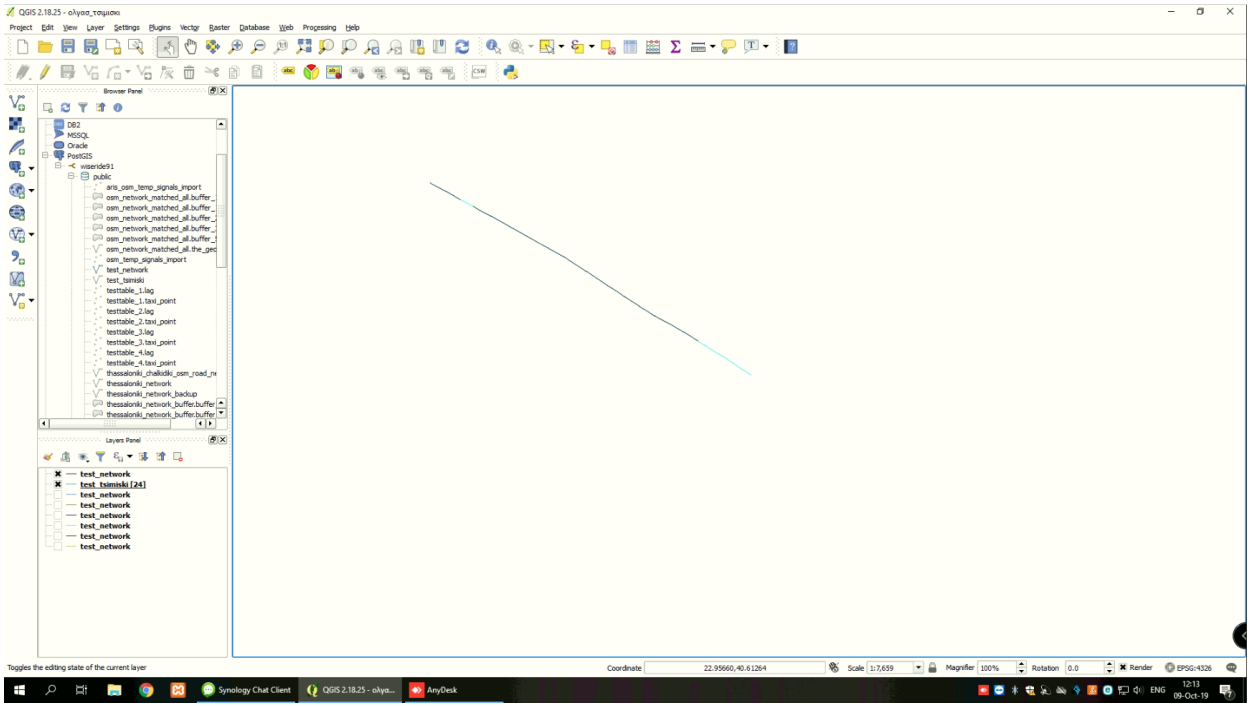

Figure 11: Tsimiski street

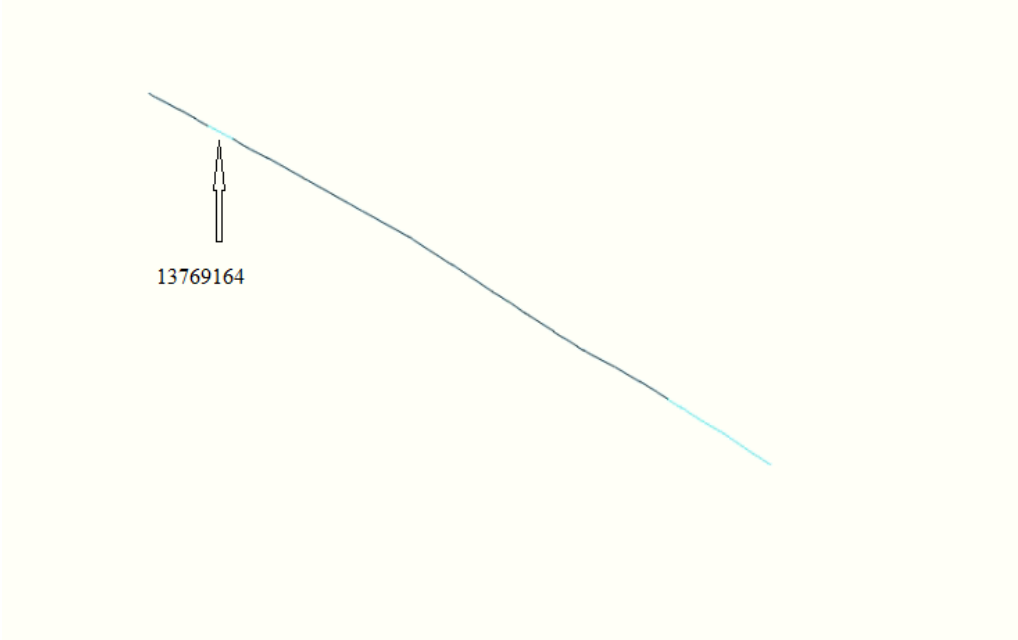

Figure 12: 13769164 Osm_id in Tsimiski street

Similarly, the traffic congestion data about the 13769164 osm_id were not enough as it can be viewed in light blue color in Figure 11 and Figure 12. Dark blue describes osm_ids of Tsimiski with sufficient data.

For this reason, the selected data would cover two exact previous osm_id's and one exact next. These 3 extra osm_id's were 197107696,176665188 and 174019380 (dark blue in Figure 12). The selection can be viewed in the next images (Figure 13 and Figure 14).

```
--select for Decision tree
SELECT
TO_CHAR(T1.time_stamp, 'HH24:MI') TIME,
TO_CHAR(T1.time_stamp, 'DAY') day,
CASE
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '09:00' AND '20:59') AND (TO_CHAR(T1.time_stamp, 'DAY')
IN ('TUESDAY  ','FRIDAY   ','THURSDAY ')) THEN 'OPEN'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '09:00' AND '17:59') AND (TO_CHAR(T1.time_stamp, 'DAY')
IN ('MONDAY   ','WEDNESDAY','SATURDAY ')) THEN 'OPEN'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '07:30' AND '08:59') AND (TO_CHAR(T1.time_stamp, 'DAY')
NOT IN ('SUNDAY   ')) THEN 'OPENING'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '21:00' AND '22:00') AND (TO_CHAR(T1.time_stamp, 'DAY')
IN ('TUESDAY  ','FRIDAY   ','THURSDAY ')) THEN 'CLOSING'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '18:00' AND '19:00') AND (TO_CHAR(T1.time_stamp, 'DAY')
IN ('MONDAY   ','WEDNESDAY','SATURDAY ')) THEN 'CLOSING'
ELSE 'CLOSED'
END AS STORES,
CASE
WHEN T1.CONGESTION IN ('Medium') THEN '1'
WHEN T1.CONGESTION IN ('High') THEN '2'
ELSE '0'
END AS CONGESTION,
t2.osm_id OSM_ID,
CASE
WHEN T3.ROAD_LENGTH*1000 > 200 THEN 'YES'
ELSE 'NO'
END AS LARGER_THAN_200M,
t2.kmh KMH,
T4.CATEGORY CATEGORY FROM traffic_congestion T1
INNER JOIN THESS2 T2 ON T2.OSM_ID = t1.link_id
INNER JOIN (
SELECT T2.OSM_ID, SUM(T2.LENGTH) ROAD_LENGTH
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
197107696,176665188,
13769164,
174019380))
GROUP BY T2.OSM_ID) T3 ON T3.OSM_ID = t1.link_id
INNER JOIN
((SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
```

Figure 13: Selected data

As it can been seen, for low congestion output was 0, for medium output was 1 and for high congestion output was 2. Also, there was a chance that each osm_id contained sev-

eral different street categories (Table 2). This may occured because each osm_id described several ids and some of them could be in different road categories.

There is no panacea, but as far as this issue was concerned, the osm_id gets the road category with the most occurrences for this specific osm_id. This can be also seen in the next image.

The export of the selection was a csv file.

```
WHERE (T2.OSM_ID IN (
197107696))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)
UNION ALL
(SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
176665188))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)
UNION ALL
(SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
13769164))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)
UNION ALL
(SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
174019380))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)) T4 ON T4.OSM_ID = t1.link_id
WHERE (T2.OSM_ID IN (
197107696,
176665188,
13769164,
174019380))
GROUP BY T1.link_timestamp_id, TO_CHAR(T1.time_stamp, 'HH24:MI'), TO_CHAR(T1.time_stamp, 'DAY'), t1.congestion, t2.osm_id,
t2.osm_name, T3.ROAD_LENGTH, t2.kmh, T4.CATEGORY;
```

Figure 14: Selected data

5.1.2 Initial Prediction Algorithm

In general, categorical data work well with Decision Trees[1] and all the provided parameters of the traffic dataset were categorized. As a result, for this task, a decision tree Classification machine learning algorithm was developed for traffic congestion forecasting using python's 3.6 Scikit learn library with Jupyter Notebook. Initially the exported

[1] https://dzone.com/articles/logistic-regression-vs-decision-tree

csv had been read using Pandas library (Figure 15). The total records of the extracted were 2155.

```
In [2]: import pandas as pd
        df = pd.read_csv('firstnormaldata20191017_DecisionTree.csv')
        df.head()
```

Out[2]:

	TIME	DAY	STORES	CONGESTION	OSM_ID	LARGER_THAN_200M	KMH	CATEGORY
0	17:15	MONDAY	OPEN	0	176665188	YES	50	KA_1K
1	22:00	MONDAY	CLOSED	0	176665188	YES	50	KA_1K
2	18:30	TUESDAY	OPEN	0	176665188	YES	50	KA_1K
3	14:30	WEDNESDAY	OPEN	0	176665188	YES	50	KA_1K
4	20:45	WEDNESDAY	CLOSED	0	176665188	YES	50	KA_1K

Figure 15: Reading csv data

After that, removal and targeting of congestion column was done (Figure 16).

```
In [3]: inputs = df.drop('CONGESTION',axis='columns')

In [4]: target = df['CONGESTION']
```

Figure 16: Congestion targeting

The next step was to use label encoder in order to categorize the values time, day, open stores, osm_id, road leght greater than 200 meters and road category. This process is presented in Figure 17.

Generally, Scikit learn library can more easily read categorical values 0,1,2 etc. This occurs because sometimes there can be text values that may be full numeric, and this may complex the algorithm and lead to bugging issues.

```
In [6]: from sklearn.preprocessing import LabelEncoder
        le_TIME = LabelEncoder()
        le_DAY = LabelEncoder()
        le_STORES = LabelEncoder()
        le_OSM_ID = LabelEncoder()
        le_LARGER_THAN_200M = LabelEncoder()
        le_KMH = LabelEncoder()
        le_CATEGORY = LabelEncoder()

In [7]: inputs['TIME_n'] = le_TIME.fit_transform(inputs['TIME'])
        inputs['DAY_n'] = le_DAY.fit_transform(inputs['DAY'])
        inputs['STORES_n'] = le_STORES.fit_transform(inputs['STORES'])
        inputs['OSM_ID_n'] = le_OSM_ID.fit_transform(inputs['OSM_ID'])
        inputs['LARGER_THAN_200M_n'] = le_LARGER_THAN_200M.fit_transform(inputs['LARGER_THAN_200M'])
        inputs['KMH_n'] = le_KMH.fit_transform(inputs['KMH'])
        inputs['CATEGORY_n'] = le_CATEGORY.fit_transform(inputs['CATEGORY'])
```

Figure 17: Using label encoder to categorize the values

The updated dataset is presented in Figure 18:

```
In [8]: inputs
```

Out[8]:

	TIME	DAY	STORES	OSM_ID	LARGER_THAN_200M	KMH	CATEGORY	TIME_n	DAY_n	STORES_n	OSM_ID_n	LARGER_THAN_200M_n	KM
0	17:15	MONDAY	OPEN	176665188	YES	50	KA_1K	69	1	2	2	1	
1	22:00	MONDAY	CLOSED	176665188	YES	50	KA_1K	88	1	0	2	1	
2	18:30	TUESDAY	OPEN	176665188	YES	50	KA_1K	74	5	2	2	1	
3	14:30	WEDNESDAY	OPEN	176665188	YES	50	KA_1K	58	6	2	2	1	
4	20:45	WEDNESDAY	CLOSED	176665188	YES	50	KA_1K	83	6	0	2	1	

Figure 18: Updated dataset

After this, the text variables had removed and the dataset then contained only categorical ones, as it can been seen in Figure 19,

```
In [9]: inputs_n = inputs.drop(['TIME','DAY','STORES','OSM_ID','LARGER_THAN_200M','KMH','CATEGORY'],axis='columns')

In [10]: inputs_n
```

Out[10]:

	TIME_n	DAY_n	STORES_n	OSM_ID_n	LARGER_THAN_200M_n	KMH_n	CATEGORY_n
0	69	1	2	2	1	0	0
1	88	1	0	2	1	0	0
2	74	5	2	2	1	0	0
3	58	6	2	2	1	0	0
4	83	6	0	2	1	0	0

Figure 19: Text variables removed

while the targeted congestion transformed like in the Figure 20's view.

```
In [11]: target

Out[11]: 0    0
         1    0
         2    0
         3    0
         4    0
         5    0
```

Figure 20: Targeted congestion

The next step of this process was to train the algorithm, using a 80 percent of the dataset for training and 20 percent for testing. This split procedure can be seen in Figure 21

```
In [12]: #feature_cols = ['TIME_n', 'DAY_n', 'STORES_n', 'OSM_ID_n','LARGER_THAN_200M_n','KMH_n','CATEGORY_n']
         from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(inputs_n, target, test_size=0.2, random_state=1)

In [13]: X_test

Out[13]:
```

	TIME_n	DAY_n	STORES_n	OSM_ID_n	LARGER_THAN_200M_n	KMH_n	CATEGORY_n
725	72	5	2	1	0	1	1
1564	86	1	0	1	0	1	1
1422	15	4	0	2	1	0	0
1145	47	0	2	2	1	0	0
1084	34	5	3	3	0	1	1
65	66	1	2	2	1	0	0

Figure 21: Spliting and training the algorithm

After spliting the algorithm, tree from sklearn library was used and a decission tree classifier model was created. This model was trained from the train set, as it can been seen in Figure 22. Moreover, the score of the model was very high, skyrocketing to 0.93.

```
In [21]: from sklearn import tree
         model = tree.DecisionTreeClassifier()
         model.fit(X_train, y_train)

Out[21]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                     max_features=None, max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                     splitter='best')

In [22]: model.score(X_train,y_train)

Out[22]: 0.9323181049069373
```

Figure 22: Creating a decission tree classifier

After that, it was time to check the model. For example, it is almost sure that at 04.00 in the morning during a week day like Wednesday that traffic congestion is low. The categorical value for 04.00 was 16, while for Wednesday was 6. Also the categorical osm_id for 13769164 was 0, the kmh parameter 1 and the road category 1 as it can been seeen in Figure 23.

TIM	DAY	STORES	OSM_ID	LAR	KM	CATEGO	TIME	DAY	STORES	OSM_ID	LARG	KMH	CATEGORY
4:00	WEDNESDAY	CLOSED	13769164	NO	70	KS_2K	16	6	0	0	0	1	1

Figure 23: Example for osm_id 13769164

The next image shows that the model's response was 0 (Figure 24). That means that the traffic congestion at the aforementioned case was low, which was expected.

```
In [23]: model.predict([[16,6,0,0,0,1,1]])
Out[23]: array([0], dtype=int64)
```

Figure 24: Model's response

After this prediction, it was time to check the model for a more uncertain different case. The model was tested for 20.30 in the afternoon during Wednesday. The categorical value for 20.30 was 82, while for Wednesday was 6. Also the categorical osm_id for 13769164 was 0, the kmh parameter 1 and the road category 1.

```
In [24]: model.predict([[82,6,0,0,0,1,1]])
Out[24]: array([1], dtype=int64)
```

Figure 25: Model's response

The Figure 25 shows that the model's response was 1. That means that the traffic congestion at the aforementioned case was medium, which was also expected.

As far as the accuracy is concerned, Figure 26 shows that after importing the metrics, the model's accuracy score was 0.678.

```
In [28]: y_pred = model.predict(X_test)

In [29]: from sklearn import metrics
         print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.6779279279279279
```

Figure 26: Model's accuracy

5.1.3 Improving Prediction Algorithms

In this stage, several efforts were made for accuracy improvement.

First, as far as criterion is concerned, the default was “gini”. This feature provides the user the ability to use different attribute selection measure. One possible attempt to improve accuracy was to choose a different attribute selection measure (Figure 27). Two supported criteria are “gini” for the Gini index and “entropy” for the information gain.

```
In [68]: from sklearn import tree
         model = tree.DecisionTreeClassifier(criterion='entropy')
         model.fit(X_train, y_train)

Out[68]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=None,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort=False,
                                random_state=None, splitter='best')
```

Figure 27: Choosing information gain

After this change the model score remained similar (Figure 28).

```
In [69]: model.score(X_train,y_train)

Out[69]: 0.9323181049069373
```

Figure 28: Model score

As for the predictions, the Figure 29 shows that the results remained the same.

```
In [71]: model.predict([[16,6,0,0,0,1,1]])

Out[71]: array([0], dtype=int64)

In [72]: model.predict([[82,6,0,0,0,1,1]])

Out[72]: array([1], dtype=int64)
```

Figure 29: Predictions

However, the acuracy score droped to 0.675 from 0.678. As it seems, utilizing information gain did not help the algorithm’s performance (Figure 30).

```
In [75]: from sklearn import metrics
         print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.6756756756756757
```

Figure 30: Acuracy while utilizing information gain

For optimizing decision tree performance, another attempt was to change the split strategy of the algorithm. This feature gives the user the ability to select the split strategy. Two supported strategies were “best” to select the best split and “random” to select the best random split. The split strategy was changed from “best” to “random” (Figure 31)

```
In [87]: from sklearn import tree
         model = tree.DecisionTreeClassifier(criterion='entropy' , splitter='random')
         model.fit(X_train, y_train)

Out[87]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=None,
                                 max_features=None, max_leaf_nodes=None,
                                 min_impurity_decrease=0.0, min_impurity_split=None,
                                 min_samples_leaf=1, min_samples_split=2,
                                 min_weight_fraction_leaf=0.0, presort=False,
                                 random_state=None, splitter='random')
```

Figure 31: Changing the split strategy

After this change the model score remained similar (Figure 32).

```
In [88]: model.score(X_train,y_train)

Out[88]: 0.9323181049069373
```

Figure 32: Model score

As for the predictions, the results did not remained the same (Figure 33).

```
In [90]: model.predict([[16,6,0,0,0,1,1]])
Out[90]: array([0], dtype=int64)

In [91]: model.predict([[82,6,0,0,0,1,1]])
Out[91]: array([0], dtype=int64)

In [92]: y_pred = model.predict(X_test)

In [93]: y_pred = model.predict(X_test)

In [94]: from sklearn import metrics
         print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
         Accuracy: 0.6711711711711712
```

Figure 33: Prediction results and acuracy

However, the acuracy score droped to 0.671. As it can been seen in Figure 33, utilizing random split did not improve the algorithm's performance.

That being noted, implementing the decision tree algorithm with Gini index and best split was the best performed algorithm.

An extra effort for improving the accuracy would be also to extract more data. For these reasons, an extra extraction stage was implemented gathering a total number of 4355 records, almost twice than the original extracted registrations (Figure 34).

```
In [25]: inputs
Out[25]:
```

	TIME	DAY	STORES	OSM_ID	LARGER_THAN_200M	KMH	CATEGORY	TIME_n	DAY_n	STORES_n	OSM_ID_n	LARGER_THAN_200M_n
0	17:15	MONDAY	OPEN	176665188	YES	50	KA_1K	69	1	2	2	1
1	22:00	MONDAY	CLOSED	176665188	YES	50	KA_1K	88	1	0	2	1
2	18:30	TUESDAY	OPEN	176665188	YES	50	KA_1K	74	5	2	2	1
3	14:30	WEDNESDAY	OPEN	176665188	YES	50	KA_1K	58	6	2	2	1
4	20:45	WEDNESDAY	CLOSED	176665188	YES	50	KA_1K	83	6	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...
4350	00:15	SATURDAY	CLOSED	197107696	NO	70	KS_2K	1	2	0	3	0
4351	05:00	SATURDAY	CLOSED	197107696	NO	70	KS_2K	20	2	0	3	0
4352	05:45	SATURDAY	CLOSED	197107696	NO	70	KS_2K	23	2	0	3	0
4353	07:45	SATURDAY	OPENING	197107696	NO	70	KS_2K	31	2	3	3	0
4354	09:45	SATURDAY	OPEN	197107696	NO	70	KS_2K	39	2	2	3	0

4355 rows × 14 columns

Figure 34: Using the algorithm with more data

The initial decision tree algorithm with Gini index and best split was the best performed algorithm and selected for this attempt (Figure 35).

```
In [31]: from sklearn import tree
         model = tree.DecisionTreeClassifier()
         model.fit(X_train, y_train)

Out[31]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort=False,
                                random_state=None, splitter='best')
```

Figure 35: Decision tree algorithm with Gini index and best split

After this change the model score decreased to 0.867 (Figure 36).

```
In [32]: model.score(X_train,y_train)

Out[32]: 0.8668197474167624
```

Figure 36: Model score

As for the predictions, the prediction results remained the same (Figure 37).

```
In [34]: model.predict([[16,6,0,0,0,1,1]])

Out[34]: array([0], dtype=int64)

In [35]: model.predict([[82,6,0,0,0,1,1]])

Out[35]: array([1], dtype=int64)
```

Figure 37: Predictions

However, the acuracy score increased almost to 0.7 according to Figure 38. As it seems, gathering, extracting and selecting more data and utilizing more information increased the algorithm's performance.

```
In [36]: y_pred = model.predict(X_test)

In [37]: from sklearn import metrics
         print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.6980482204362801
```

Figure 38: Acuracy score

5.1.4 Decision tree vs Logistic regression algorithms

In this phase, the so far developed Decision tree algorithm was compared to Logistic regression algorithm based on their accuracy score.

As far as the selection was concerned, the only difference was to road length as it is presented

in

```
--Logistic Regression (Binary Classification)
SELECT
--t1.link_timestamp_id LINK_TIMESTAMP_ID,
TO_CHAR(T1.time_stamp, 'HH24:MI') TIME,
TO_CHAR(T1.time_stamp, 'DAY') day,
CASE
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '09:00' AND '20:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('TUESDAY  ','FRIDAY   ','THURSDAY ')) THEN 'OPEN'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '09:00' AND '17:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('MONDAY   ','WEDNESDAY','SATURDAY ')) THEN 'OPEN'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '08:00' AND '08:59') AND (TO_CHAR(T1.time_stamp, 'DAY') NOT IN ('SUNDAY   ')) THEN 'OPENING'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '21:00' AND '21:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('TUESDAY  ','FRIDAY   ','THURSDAY ')) THEN 'CLOSING'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '18:00' AND '18:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('MONDAY   ','WEDNESDAY','SATURDAY ')) THEN 'CLOSING'
ELSE 'CLOSED'
END AS STORES,
--t1.congestion CONGESTION,
CASE
WHEN T1.CONGESTION IN ('Medium') THEN '1'
WHEN T1.CONGESTION IN ('High') THEN '2'
ELSE '0'
END AS CONGESTION,
t2.osm_id OSM_ID,
--t2.osm_name OSM_NAME,
--t2.clazz CLAZZ,
--t2.flags FLAGS,
TRUNC((T3.ROAD_LENGTH*1000), 0) ROAD_LENGTH_M,
--CASE
--WHEN T3.ROAD_LENGTH*1000 > 200 THEN 'YES'
--ELSE 'NO'
--END AS LARGER_THAN_200M,
t2.kmh KMH,
T4.CATEGORY CATEGORY FROM traffic_congestion T1
INNER JOIN THESS2 T2 ON T2.OSM_ID = t1.link_id
INNER JOIN (
SELECT T2.OSM_ID, SUM(T2.LENGTH) ROAD_LENGTH
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
197107696,176665188,
13769164,
174019380))
GROUP BY T2.OSM_ID) T3 ON T3.OSM_ID = t1.link_id
INNER JOIN
((SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
```

Figure 39 and Figure 40.

```
--Logistic Regression (Binary Classification)
SELECT
--t1.link_timestamp_id LINK_TIMESTAMP_ID,
TO_CHAR(T1.time_stamp, 'HH24:MI') TIME,
TO_CHAR(T1.time_stamp, 'DAY') day,
CASE
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '09:00' AND '20:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('TUESDAY  ','FRIDAY   ','THURSDAY ')) THEN 'OPEN'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '09:00' AND '17:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('MONDAY   ','WEDNESDAY','SATURDAY ')) THEN 'OPEN'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '08:00' AND '08:59') AND (TO_CHAR(T1.time_stamp, 'DAY') NOT IN ('SUNDAY   ')) THEN 'OPENING'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '21:00' AND '21:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('TUESDAY  ','FRIDAY   ','THURSDAY ')) THEN 'CLOSING'
WHEN (TO_CHAR(T1.time_stamp, 'HH24:MI') BETWEEN '18:00' AND '18:59') AND (TO_CHAR(T1.time_stamp, 'DAY') IN ('MONDAY   ','WEDNESDAY','SATURDAY ')) THEN 'CLOSING'
ELSE 'CLOSED'
END AS STORES,
--t1.congestion CONGESTION,
CASE
WHEN T1.CONGESTION IN ('Medium') THEN '1'
WHEN T1.CONGESTION IN ('High') THEN '2'
ELSE '0'
END AS CONGESTION,
t2.osm_id OSM_ID,
--t2.osm_name OSM_NAME,
--t2.clazz CLAZZ,
--t2.flags FLAGS,
TRUNC((T3.ROAD_LENGTH*1000), 0) ROAD_LENGTH_M,
--CASE
--WHEN T3.ROAD_LENGTH*1000 > 200 THEN 'YES'
--ELSE 'NO'
--END AS LARGER_THAN_200M,
t2.kmh KMH,
T4.CATEGORY CATEGORY FROM traffic_congestion T1
INNER JOIN THESS2 T2 ON T2.OSM_ID = t1.link_id
INNER JOIN (
SELECT T2.OSM_ID, SUM(T2.LENGTH) ROAD_LENGTH
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
197107696,176665188,
13769164,
174019380))
GROUP BY T2.OSM_ID) T3 ON T3.OSM_ID = t1.link_id
INNER JOIN
((SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
```

Figure 39: Data selection for logistic regression algorithm

```
INNER JOIN
((SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
197107696))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)
UNION ALL
(SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
176665188))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)
UNION ALL
(SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
13769164))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)
UNION ALL
(SELECT T2.OSM_ID, T2.CATEGORY, COUNT(T2.CATEGORY) CNT_CATEGORY
FROM THESS2 T2
WHERE (T2.OSM_ID IN (
174019380))
GROUP BY T2.OSM_ID, T2.CATEGORY
ORDER BY COUNT(T2.CATEGORY) DESC
FETCH FIRST 1 ROWS ONLY)) T4 ON T4.OSM_ID = t1.link_id
WHERE (T2.OSM_ID IN (
--20420010,
--197107695,
197107696,
176665188,
13769164,
174019380))
GROUP BY T1.link_timestamp_id, TO_CHAR(T1.time_stamp, 'HH24:MI'), TO_CHAR(T1.time_stamp, 'DAY'), t1.congestion, t2.osm_id,
t2.osm_name, T3.ROAD_LENGTH, t2.kmh, T4.CATEGORY;
```

Figure 40: Data selection for logistic regression algorithm

After the data were imported via csv parsing (Figure 41), the label encoder was utilized so the data to be categorized (Figure 42).

```
In [22]: import pandas as pd
         from matplotlib import pyplot as plt
         %matplotlib inline

In [23]: df = pd.read_csv("firstnormaldata20191026_logistic.csv")
         df.head()
```

Out[23]:

	TIME	DAY	STORES	CONGESTION	OSM_ID	ROAD_LENGTH_M	KMH	CATEGORY
0	17:15	MONDAY	OPEN	0	176665188	985	50	KA_1K
1	22:00	MONDAY	CLOSED	0	176665188	985	50	KA_1K
2	18:30	TUESDAY	OPEN	0	176665188	985	50	KA_1K
3	14:30	WEDNESDAY	OPEN	0	176665188	985	50	KA_1K
4	20:45	WEDNESDAY	CLOSED	0	176665188	985	50	KA_1K

```
In [24]: inputs = df.drop('CONGESTION',axis='columns')

In [25]: target = df['CONGESTION']
```

Figure 41: Reading csv data and congestion targeting

```
In [26]: from sklearn.preprocessing import LabelEncoder
         le_TIME = LabelEncoder()
         le_DAY = LabelEncoder()
         le_STORES = LabelEncoder()
         le_OSM_ID = LabelEncoder()
         le_CATEGORY = LabelEncoder()

In [27]: inputs['TIME_n'] = le_TIME.fit_transform(inputs['TIME'])
         inputs['DAY_n'] = le_DAY.fit_transform(inputs['DAY'])
         inputs['STORES_n'] = le_STORES.fit_transform(inputs['STORES'])
         inputs['OSM_ID_n'] = le_OSM_ID.fit_transform(inputs['OSM_ID'])
         inputs['CATEGORY_n'] = le_CATEGORY.fit_transform(inputs['CATEGORY'])

In [28]: inputs
```

Figure 42: Using label encoder to categorize the values

Before spliting the algorithm, LogisticRegression from sklearn library was used and a Logistic regression classifier model was created. This model was trained from the train set, as it can been seen in Figure 43.

```
In [32]: from sklearn.linear_model import LogisticRegression
         model = LogisticRegression()

In [33]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(inputs_n, target, test_size=0.2, random_state=1)

In [34]: model.fit(X_train, y_train)

Out[34]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
                   penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
                   verbose=0, warm_start=False)
```

Figure 43: Creating a Logistic regression classifier and spliting the data

Moreover, the score of the model was low, close to 0.66. (Figure 44).

```
In [35]: model.score(X_train,y_train)

Out[35]: 0.662743972445465
```

Figure 44: Model score

As for the predictions, the Figure 45 shows that the results did not remained the same.

```
In [37]: model.predict([[71,70,16,6,0,0,1]])

Out[37]: array([1], dtype=int64)

In [38]: model.predict([[71,70,82,6,0,0,1]])

Out[38]: array([1], dtype=int64)
```

Figure 45: Predictions

What is more, the acuracy score droped to 0.674 compared to 0.7 that Decission Tree scored. As it seems, utilizing a Logistic regression algorithm was not the best option while using the specific dataset (Figure 46).

```
In [39]: y_pred = model.predict(X_test)

In [40]: from sklearn import metrics
         print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.6739380022962113
```

Figure 46: Acuracy while utilizing Logistic regression

6 Conclusions

This final chapter summarizes the results, the challenges and the suggested future work that would come after the end of this study.

6.1.1 Summary and results

This final chapter summarizes the results. This research was conducted for Thessaloniki city to deal traffic congestion issues. The example presented was for one of the most main streets in the city, Tsimiski street. After collecting, storing and preprocessing the data from an api for 3 months (August, September, October of 2019) using python, sql and gis technologies, a model for predicting the traffic congestion for Tsimiski street was developed using decision tree machine learning algorithm and python with scikit learn and pandas libraries.

The model was used to solve a mockup example providing results for specific day and hour in a part of Tsimiski street. While the outcomes were reasonable, an effort was done for better accuracy score. For this reason, several tests were implemented with different strategies proposed and applied for the same algorithm.

Finally, the result of these attempts shown that the most important part for improving the algorithm was the data. A second attempt was done utilizing the original algorithm, which was also had the best accuracy score, combined with more data extracted and as a consequence more data to be trained and the outcome showed that the accuracy score was higher than before. This accuracy score was also higher compared to Logistic regression algorithm’s accuracy score.

6.1.2 Challenges

Although a specific data sets was used, the data extraction and preprocess phase was challenging.

The first challenge was about finding the qualified data. The goal was to retrieve both sufficient and enough information that would also be originated and describe traffic in the city of Thessaloniki. After managing with this issue, the best suited method for data storage was needed.

A decision for utilizing Oracle 12C database was done. This occurred because Oracle is a database delivers excellent performance when challenged with demanding tasks and the sql query selection for providing the necessary data was complex.

After that very important decision had to be done in order to select the most appropriate machine algorithm for traffic prediction. Generally, categorical data work well with Decision Tree algorithms, while continuous data work well with Logistic Regression algorithms. All the provided parameters of the traffic dataset were categorized and as a result a Decision Tree algorithm was selected.

6.1.3 Future work

Such work and research process require time-consuming analysis and different approaches. It is never possible to explore every aspect, especially when the time of delivery of each project being limited.

However, there are several ideas that could be implemented combing knowledge via this thesis. As mentioned, the concept of Smart City was created to enhance the life quality of citizen so an idea would be to automate the prediction procedure utilizing an application to provide traffic prediction to residents.

What is more, it is already mentioned in 3.3.1 paragraph that a key feature of a smart city is efficient traffic flow management throughout the city, that could boost transportation networks flow and optimize traffic conditions for people and cities in general. As the population grows, there are traffic issues, increased emissions, and environmental and economic issues. Because of the above, the utilization of smart traffic lights is among the most relevant strategies used by smart cities to come face to face regarding increasing traffic congestion problems. An idea would be to use the prediction results utilizing a smart traffic lights framework that would operate dynamically based on the predicted traffic congestion.

7 Bibliography

Aguilera, G., Galan, J.L., Campos, J.C. & Rodríguez, P. (2013). *An Accelerated-Time Simulation for Traffic Flow in a Smart City*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377042713006523

Agrawal, R. and Srikant, R. (1994). *Fast Algorithms for Mining Association*. Retrieved from http://www.vldb.org/conf/1994/P487.PDF

Alessandri, R. B. A. & M. Repetto. (2003). *Estimating of freeway traffic variables using information from mobile phones.* Retrieved from https://ieeexplore.ieee.org/abstract/document/1240476

Al-Hader, M. & Rodzi, A. (2009). *The smart city infrastructure development & monitoring.* Retrieved from https://www.researchgate.net/publication/46567764_The_smart_city_infrastructure_development_monitoring

Batty, M. (2013, December 10). Big data, smart cities and city planning. *Dialogues in Human Geography, 3* (3), 274–9. Retrieved from https://journals.sagepub.com/doi/full/10.1177/2043820613513390

Bertini, R. L. (2005). *Congestion and Its Extent.*
Retrieved from https://www.researchgate.net/publication/239571443_Congestion_and_Its_Extent

Bertot, J.C. & Choi, H., (2013). *Big data and e-government: issues, policies, and recommendations.* Retrieved from

https://www.semanticscholar.org/paper/Big-data%2C-open-government-and-e-government%3A-Issues%2C-Bertot-Gorham/8702bd42ed3234bbb705ed5f1d0672dc26a0a8fc

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Retrieved from http://users.isr.ist.utl.pt/~wurmd/Livros/school/Bishop%20-%20Pattern%20Recognition%20And%20Machine%20Learning%20-%20Springer%20%202006.pdf

Fan, W. & Bifet, A. (2013). *Mining big data: current status, and forecast to the future*. Retrieved from https://www.kdd.org/exploration_files/V14-02-01-Fan.pdf

Fayyad, U., Piatetsky-Shapiro G. & Smyth, P. (1996). *From Data Mining to Knowledge Discovery in Databases.* Retrieved from https://www.academia.edu/1556427/From_Data_Mining_to_Knowledge_Discovery_in_Databases

Han, J. & Kamber, M. (2001). *Data Mining: Concepts and Techniques.* Retrieved from https://cs.wmich.edu/~yang/teach/cs595/han/ch01.pdf

Kamber, M. & Pei, J. S. *(2012). Data Mining Concepts and Techniques*. Retrieved from http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf

Kandappu, T., Misra, A., Koh, M.H.D. (X.M.), Daratan, R.T. & Jaiman, N. (2018). *A feasibility study on crowdsourcing to monitor municipal resources in smart cities.* Retrieved from https://ink.library.smu.edu.sg/cgi/viewcontent.cgi?article=5085&context=sis_research

Khan, Z., Anjum, A. & Kiani SL. (2013). *Cloud Based Big Data Analytics for Smart Future Cities.* Retrieved from

https://dl.acm.org/doi/10.1109/UCC.2013.77

Kitchin, R., (2014, July 3). The real-time city? Big data and smart urbanism. *GeoJournal, 79* (1),1-14. Retrieved from

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2289141

Kramers, A., Höjer, M., Lövehagen, N. & Wangel, J. (2014). *Smart sustainable cities – Exploring ICT solutions for reduced energy use in cities.* Retrieved from

https://www.researchgate.net/publication/262570299_Smart_sustainable_cities_-_Exploring_ICT_solutions_for_reduced_energy_use_in_cities

Krause, B. & von Altrock, C. (1996). *Intelligent highway by fuzzy logic: Congestion detection and traffic control on multi-lane roads with variable road signs.* Retrieved from

https://ieeexplore.ieee.org/document/552649

Kubick, W. R. (2012). *Applied Clinical Trials*. Retrieved from

https://search.proquest.com/openview/b168144876659e85e8890280d10a8020/1?pq-origsite=gscholar&cbl=44052

Koutroumpis, P., Aija L. (2013). *Understanding the value of (big) data.* Retrieved from

https://www.researchgate.net/publication/261314325_Understanding_the_value_of_big_data

Leskovec, J., Rajaraman, A. & Ullman, J. D. (2014). *Mining of Massive Datasets.* Retrieved from

https://books.google.gr/books?hl=el&lr=&id=16YaBQAAQBAJ&oi=fnd&pg=PR1&dq=Rajaraman,+Leskovec+%26+Ullman+(2014).+&ots=Jge62YtPdi&sig=UYfCA6daH

WK7dGk20HSXKF-y8Vo&redir_esc=y#v=onepage&q=Rajaraman%2C%20Leskovec%20%26%20Ullman%20(2014).&f=false

Lomax, T., Turner, S., Shunk, G., Levinson, H.S., Pratt, R. H., Bay, P. N. & Douglas B. B. (1997). *Quantifying Congestion. Volume 1: Final Report.*

Retrieved from

https://trid.trb.org/view/475257

Lu, J. & Cao, L. (2003) *Congestion evaluation from traffic flow information based on fuzzy logic.* Retrieved from

https://ieeexplore.ieee.org/abstract/document/1251919

Michalik, P., Stofa, J. & Zolotova, I. (2014). *Concept definition for Big Data architecture in the education system.* Retrieved from

https://ieeexplore.ieee.org/document/6822433

Michalski, R. S., Bratko, I. & Kubat, M. (1998). *Machine Learning and Data Mining: Methods and Applications.* Retrieved from https://www.wiley.com/en-us/Machine+Learning+and+Data+Mining%3A+Methods+and+Applications-p-9780471971993

Mitsakis, E., Salanova, J. M., Chrysohoou, E. & Aifadopoulou G. (2015). *A robust method for real time estimation of travel times for dense urban road networks using point-to-point detectors*. Transport 30(3) 2015, pp. 264-272. Special Issue on Smart and Sustainable Transport. [DOI:10.3846/16484142.2015.1078845]

Retrieved from

http://opendata.imet.gr/about

Mitsakis, E., Stamos, I., Salanova, J.M. G., Chrysohoou, E. & Aifadopoulou, G. (2013). *Urban Mobility Indicators for Thessaloniki.* Journal of Traffic and Logistics Engineer-

ing. (JTLE) (ISSN: 2301-3680), Vol. 1 No. 2, June 2013. pp. 148 – 152. [DOI:10.12720/jtle.1.2.148-152]

Retrieved from

http://opendata.imet.gr/about

Neirotti, P., De Marco, A., Cagliano, A.C., Mangano, G. & Scorrano, F. (2014). *Current trends in Smart City initiatives: Some stylised facts.* Retrieved from

https://www.researchgate.net/publication/260015335_Current_trends_in_Smart_City_initiatives_Some_stylised_facts

Pattara-Atikom, W. & Peachavanish, R. (2007). *Estimating Road Traffic Congestion from Cell Dwell Time using Neural Network.* Retrieved from

https://www.researchgate.net/publication/4272120_Estimating_Road_Traffic_Congestion_from_Cell_Dwell_Time_using_Neural_Network

Pattara-Atikom, W., Pongpaibool, P. & Thajchayapong, S. (2006) *Estimating Road Traffic Congestion using Vehicle Velocity.*

Retrieved from

https://www.researchgate.net/publication/224761073_Estimating_Road_Traffic_Congestion_using_Vehicle_Velocity

Piatetsky-Shapiro, G. (2000). *Knowledge Discovery in Databases: 10 years after.*

Retrieved from:

https://www.researchgate.net/publication/220520095_Knowledge_Discovery_in_Databases_Ten_years_after

Pongpaibool, P., Tangamchit, P. & Noodwong, K. (2007). *Evaluation of Road Traffic Congestion Using Fuzzy Techniques.* Retrieved from

https://www.researchgate.net/publication/224302537_Evaluation_of_road_traffic_congestion_using_fuzzy_techniques

Porikli, F. & Li, X. (2004). *Traffic congestion estimation using hmm models without vehicle tracking.* Retrieved from

https://www.researchgate.net/publication/4092555_Traffic_congestion_estimation_using_HMM_models_without_vehicle_tracking

Salanova, J. M., Chaniotakis, E., Mitsakis, E., Aifandopoulou, G. & Bischoff J. (2016). *Mobile data for transportation*. Mobile Data, Geography, LBS, 29/06 - 01/07 Tartu, Estonia

Retrieved from

http://opendata.imet.gr/about

Stutz, J., Cheeseman, P. (1996). *Autoclass — A Bayesian Approach to Classification.* Retrieved from https://link.springer.com/chapter/10.1007/978-94-009-0107-0_13

Tan, P. N., Steinbach, M., Karpatne, A. & Kumar, V. (2018). *Introduction to Data Mining*. Retrieved from https://www-users.cs.umn.edu/~kumar001/dmbook/index.php

Thianniwet, T., Phosaard, S. & Pattara-Atikom, W. (2009). *Classification of Road Traffic Congestion Levels from GPS Data using a Decision Tree Algorithm and Sliding Windows*. Retrieved from

https://pdfs.semanticscholar.org/9b1f/3d653fe09fa0bb337af23e7e0401441e6e5a.pdf

Townsend, A.M. (2013). *Smart cities: big data, civic hackers, and the quest for a new utopia*. Retrieved from

https://ssir.org/books/excerpts/entry/smart_cities_big_data_civic_hackers_and_the_quest_for_a_new_utopia

United Nations. World Urbanization Prospects: The 2018 Revision. 2018.

Retrieved from: https://esa.un.org/unpd/wup/Publications/Files/WUP2018-KeyFacts.pdf

U.S. Department of Energy, Smart Grid / Department of Energy,

Retrieved Sep. 29, 2019 from

http:// energy.gov/oe/technology-development/smart-grid

Vilajosana, I., Llosa, J., Martinez, B., Domingo-Prieto, M., Angles, A., Vilajosana, X., (2013, June 10). Bootstrapping smart cities through a self-sustainable model based on big data flows. *IEEE Communications Magazine, 51* (6), 128–34. Retrieved from

https://ieeexplore.ieee.org/document/6525605?tp=&arnumber=6525605

Weiss, S. M., Kulikowski, C. A. (1991). *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Retrieved from

https://books.google.gr/books/about/Computer_Systems_that_Learn.html?id=hEoPAQAAMAAJ&redir_esc=y

West, D.M. (2012). Big Data for Education: Data Mining, Data Analytics, and Web Dashboards. Retrieved from

https://library.educause.edu/resources/2012/9/big-data-for-education-data-mining-data-analytics-and-web-dashboards

Witten, I. H., Frank, E., Hall, M. A., Pal, C. J., (2017). *Data Mining Practical Machine Learning Tools and Techniques*. Retrieved from http://bookslibrary.in/engineering/Data%20Mining%20Practical%20Machine%20Learning%20Tools%20and%20Techniques%204th%20Edition.pdf

Wu, X. (2004). *Data mining: artificial intelligence in data analysis.* Retrieved from https://ieeexplore.ieee.org/abstract/document/1342916

Zhang, T., Ramakrishnan, R. & Livny, M. (1996). *An efficient data clustering method for very large databases*. Retrieved from https://dl.acm.org/citation.cfm?id=233324

Zhou, Z. H. (2003). *Three Perspectives of Data Mining, Artificial Intelligence*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0004370202003570

Appendix

Below one can find some code for data extraction, gathering, and database importing Besides that, all samples scripts were written in Python.

```
"""
Created on Wed Sep 25 17:52:48 2019

@author: Mystakidis Aristeidis
"""



import urllib.request, json
import cx_Oracle
import os, json
import pandas as pd
from glob import glob
import time
from datetime import datetime, timedelta
```

```
class JsonOracle():

    def JsonToOracle():
        dsn_tns = cx_Oracle.makedsn('localhost', '1521', service_name='orcl3') #if needed, place an 'r' before any parameter in order to address any special character such
        conn = cx_Oracle.connect(user='system', password='26282628', dsn=dsn_tns) #if needed, place an 'r' before any parameter in order to address any special character su

        c = conn.cursor()
        c.execute('select LINK_TIMESTAMP_ID from TRAFFIC_CONGESTION') # use triple quotes if you want to spread your query across multiple lines
        LINK_TIMESTAMP_ID_LIST = []
        for row in c:
            LINK_TIMESTAMP_ID_LIST.append(row[0])


        print('LINK_TIMESTAMP_ID_LIST IS ', len(LINK_TIMESTAMP_ID_LIST))

        with urllib.request.urlopen("http://feed.opendata.imet.gr:23577/fcd/congestions.json?offset=3000&limit=3000") as url:
            data = json.loads(url.read().decode())

            print(data[0])
            print("is of type", type(data[0]))
    #       data[1].split(',')[2]
    #       print(data[1].split(',')[2])
        #item_dict = json.l oads(json_data)
            jsonlength = len(data)
            print('jsonlength is ',jsonlength)
            #print(data[0]['Timestamp'])
            #c2 = conn.cursor()
            for x in range(jsonlength):
                #print(x)
                timest = str(data[x]['Timestamp'].replace('.000', ''))
                LINK_TIMESTAMP_ID = str(data[x]['Link_id'])+'_'+str(data[x]['Link_Direction'])+'_'+timest
                #print(LINK_TIMESTAMP_ID)
                if LINK_TIMESTAMP_ID not in LINK_TIMESTAMP_ID_LIST :
                    timestamp = 'TO_TIMESTAMP('+timest+''', 'YYYY-MM-DD HH:MI:SS')'''
                    c.execute('''INSERT INTO TRAFFIC_CONGESTION
                              (LINK_TIMESTAMP_ID, LINK_ID, LINK_DIRECTION,CONGESTION, TIME_STAMP)
                              VALUES
                              (:1,:2,:3,:4,TO_TIMESTAMP(:5, 'YYYY-MM-DD HH24:MI:SS'))''',
                              {"1" : LINK_TIMESTAMP_ID, "2" : str(data[x]['Link_id']), "3" : str(data[x]['Link_Direction']),"4" : str(data[x]['Congestion']),"5" : timest})

            c.execute('COMMIT')

            print('done')
```

```
def JsonFileToOracle():
    dsn_tns = cx_Oracle.makedsn('localhost', '1521', service_name='orcl3') #if needed, place an 'r' before any parameter in order to address any special character such as '\
    conn = cx_Oracle.connect(user='system', password='26282628', dsn=dsn_tns) #if needed, place an 'r' before any parameter in order to address any special character such as

    c = conn.cursor()
    c.execute('select LINK_TIMESTAMP_ID from TRAFFIC_CONGESTION') # use triple quotes if you want to spread your query across multiple lines
#     LINK_TIMESTAMP_ID_LIST = []
#     for row in c:
#         LINK_TIMESTAMP_ID_LIST.append(row[0])

    path_to_json = 'C:/Users/Mangekyo/Documents/Python Scripts/Json_Tests/jsonfiles/'
    json_files = [pos_json for pos_json in os.listdir(path_to_json) if pos_json.endswith('.txt')]
    print(json_files[1])  # for me this prints ['foo.json']

    for f_name in json_files:
        with open(f_name) as json_file:
            data = json.load(json_file)
            print(data[0])
            c.execute('select LINK_TIMESTAMP_ID from TRAFFIC_CONGESTION') # use triple quotes if you want to spread your query across multiple lines
            LINK_TIMESTAMP_ID_LIST = []
            for row in c:
                LINK_TIMESTAMP_ID_LIST.append(row[0])

            print('LINK_TIMESTAMP_ID_LIST IS ', len(LINK_TIMESTAMP_ID_LIST))

            print(data[0])
            print("is of type", type(data[0]))
#           data[1].split(',')[2]
#           print(data[1].    (',')[2])
        #item_dict = json.l oads(json_data)
            jsonlength = len(data)
            print('jsonlength is ',jsonlength)
            #print(data[0]['Timestamp'])
            #c2 = conn.cursor()
            for x in range(jsonlength):
                #print(x)
                timest = str(data[x]['Timestamp'].replace('.000', ''))
                LINK_TIMESTAMP_ID = str(data[x]['Link_id'])+'_'+str(data[x]['Link_Direction'])+'_'+timest
                #print(LINK_TIMESTAMP_ID)
                if LINK_TIMESTAMP_ID not in LINK_TIMESTAMP_ID_LIST :
                    #timestamp = 'TO_TIMESTAMP('+timest+''', 'YYYY-MM-DD HH:MI:SS')'''
                    c.execute('''INSERT INTO TRAFFIC_CONGESTION
                              (LINK_TIMESTAMP_ID, LINK_ID, LINK_DIRECTION,CONGESTION, TIME_STAMP)
                              VALUES
                              (:1,:2,:3,:4,TO_TIMESTAMP(:5, 'YYYY-MM-DD HH24:MI:SS'))''',
                              {"1" : LINK_TIMESTAMP_ID, "2" : str(data[x]['Link_id']), "3" : str(data[x]['Link_Direction']),"4" : str(data[x]['Congestion']),"5" : timest})

            c.execute('COMMIT')

            print(datetime.datetime.now().strftime("%Y-%m-%d %H:%M"))
            print('done')
#jsontake()
```

```
def executeSomething():
    try:
        print(datetime.now())
        JsonOracle.JsonToOracle()
        time.sleep(850)
    except:
        print("An exception occurred")



while True:
    executeSomething()

```