

Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Statistics



Diploma Thesis

Modelling and classification of large textual data

Vyacheslav Gatin

© 2018 CULS Prague

DIPLOMA THESIS ASSIGNMENT

Bc. Vyacheslav Gatin

Economics and Management

Thesis title

Modelling and classification of large textual data

Objectives of thesis

Diploma thesis is based on the problem of processing and classifying large amounts of textual data gathered from truck selling websites. Objective of the thesis is to explore different classification techniques and then attempt to build our own classifier for the task.

Methodology

Assessment is based on exploration of several classification models for textual data. Among others, JavaScript programming language is mainly used to model the data. Diploma thesis also implements machine learning techniques available in the field of natural language processing.

The proposed extent of the thesis

60 – 80 pages

Keywords

Big data, unstructured textual data, machine learning, natural language processing, word2vec, data modelling

Recommended information sources

Abbott, D. Applied Predictive Analytics : Principles and Techniques for the Professional Data Analyst. Wiley, March 2014. ISBN 9781118727935

Azzalini, A., Scarpa, B. Data Analysis and Data Mining : An Introduction. Oxford University Press, USA, March 2012. ISBN 9780199909285

Foreman, J. W. Data Smart : Using Data Science to Transform Information into Insight. Wiley, October 2013. ISBN 9781118839867

Franks, B. Analytics Revolution : How to Improve Your Business By Making Analytics Operational In The Big Data Era. Wiley, September 2014. ISBN 9781118976753

Hurwitz, J., Nugent, A., Halper, F., Kaufman, M. Big data for dummies. For Dummies, 1 edition, April 15, 2013. ISBN 9781118644010

RUD, O P. *Data mining : praktický průvodce dolováním dat pro efektivní prodej, cílený marketing a podporu zákazníků (CRM)*..

Expected date of thesis defence

2018/19 WS – FEM (February 2019)

The Diploma Thesis Supervisor

Ing. Tomáš Hlavsa, Ph.D.

Supervising department

Department of Statistics

Electronic approval: 25. 11. 2016

prof. Ing. Libuše Svatošová, CSc.

Head of department

Electronic approval: 25. 11. 2016

Ing. Martin Pelikán, Ph.D.

Dean

Prague on 29. 11. 2018

Declaration

I declare that I have worked on my diploma thesis titled "Modelling and classification of large textual data" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the diploma thesis, I declare that the thesis does not break copyrights of any other person.

In Prague on 29.11.2018

Acknowledgement

I would like to thank my supervisor Tomáš Hlavsa and Hypercharge team, especially Ivo Anastácio and my friend Maxime Goossens for their advice and continuous support during my work on this thesis.

Last, but not least, I would like to thank my love, Evgeniia Tetenova for supporting me the most during these hard times.

Modelling and classification of large textual data

Abstract

The amount of data that the world creates has been increasing in geometrical progression (IBM, 2016), while textual data represents a big part of it. Manipulation with and classification of textual data are dealt with Natural Language Processing field of machine learning. This thesis explores the paradigm of machine learning, as being a key technique in modern data analysis and manipulation. Research introduces the theory behind machine learning by defining its structure and processes, problems that it deals with, and its subfields. Following the purposes of the study, the research then continues to define specific machine learning algorithms best suited for Natural Language Processing tasks.

During the practical part of the work, author deals with a semi-structured dataset of more than 260 000 objects, which is a collection of professional vehicles placed on an online platform for remarketing purposes. Practical work consists of creation of appropriate classification algorithm and further improvement of its quality through exploration of unclassified data using JavaScript programming language and word2vec deep learning technique. Author has achieved the initial coverage of 80% while keeping the accuracy close to 100%. Upon improvement, the coverage of the model increased to 85%, without significant drops to accuracy. Recommendations for further improvement of the model are then provided.

Keywords: Big data, unstructured textual data, machine learning, deep learning, natural language processing, word2vec, data modelling

Modelování a klasifikace velkých textových dat

Abstrakt

Množství dat, které svět vytváří, vzrůstá geometrickou řadou (IBM, 2016) a textová data z celkového objemu reprezentují velkou část. Manipulace a klasifikace textových dat je řešena pomocí zpracování přirozeného jazyka, což je oblast strojového učení. Tato diplomová práce zkoumá paradigma strojového učení, jakožto klíčové techniky moderní analýzy a manipulace dat. Výzkum představuje teorii stojící za strojovým učním, definováním jeho struktury a procesů, problémy, kterými se zabývá, a jeho podobory. Sledující účel studie, výzkum poté pokračuje definováním konkrétních algoritmů strojového učení vhodných pro úkoly zpracování přirozeného jazyka. V průběhu praktické části práce se autor zabývá polostrukturovanými sety dat o více než 260 000 objektech, které jsou tvořeny kolekcí nákladních vozidel umístěných na online platformě pro účely remarketingu. Praktická část práce se skládá z tvorby vhodného klasifikačního algoritmu a dále zlepšování jeho kvalit skrze prozkoumávání neklasifikovaných dat pomocí programovacího jazyka JavaScript a techniky hlubokého učení word2vec. Autor dosáhl výchozího pokrytí 80%, zatímco přesnost se drží blízko 100%. Po vylepšení vzrostlo pokrytí modelu na 85% bez výraznějšího poklesu přesnosti. Poté byla vytvořena doporučení pro další vylepšení modelu.

Klíčová slova: Velká data, nestrukturovaná textová data, strojové učení, hluboké učení, zpracování přirozeného jazyka, word2vec, datové modelování

Contents

1	Introduction	11
2	Objectives and Methodology of Thesis.....	13
2.1	Objectives.....	13
2.2	Methodology	14
3	Theoretical part.....	15
3.1	Concept of Big Data	15
3.1.1	History.....	15
3.1.2	Definition	16
3.1.3	How Big Data entails the use of Machine Learning	19
3.2	Machine Learning.....	21
3.2.1	Definition	21
3.2.2	Structure and process	23
3.2.3	Problem categories	24
3.2.4	Learning subfields.....	27
3.2.5	Performance measures (model evaluation).....	32
3.3	Machine learning approaches to classification.....	39
3.3.1	Statistical approaches	40
3.3.2	Methods for solving text processing problems	41
3.3.3	Machine Learning Methods for Natural Language Processing Tasks	43
3.3.4	Deep learning technologies for natural language processing tasks.....	47
3.3.5	Overview of the algorithms.....	50
4	Practical part	51
4.1	Definition of the problem.....	51
4.1.1	Data and its context.....	51
4.1.2	The goal of classification	52
4.2	Structuring the classifier.....	52
4.2.1	Constructing the set of rules.....	54
4.2.2	Improvement of the results	56
4.3	Data manipulation for information extraction.....	56
4.3.1	Formatting and filtering a JSON file.....	57
4.3.2	Using word2vec technology to extract patterns from our data	65
4.3.3	Appliance of the new rules.....	80
5	Results and Discussion	82
5.1	Assessment of results	82
5.2	Future recommendations	84
6	Conclusion.....	85

List of figures

Figure 1: "Big Data" google search query over time	15
Figure 2: Excerpt from Meta group report on data from 2001 concerning Big Data	16
Figure 3 Definition of the big data.....	17
Figure 4: overfitting and generalization of models.....	28
Figure 5: labeled and unlabeled data	29
Figure 6: deep neural network structure - multiple hidden layers	31
Figure 7: Machine learning categories by algorithm	32
Figure 8: Spam emails confusion matrix	34
Figure 9: holdout cross-validation	37
Figure 10: 10-fold cross-validation.....	39
Figure 11: words' representation models	41
Figure 12: Continuous Bag of Words and Skip-Gram algorithms	48
Figure 13: word2vec toolkit results' example	49
Figure 14: example of one object	52
Figure 15: classifier scheme	53
Figure 16: sample from the rules for "Volvo"	55
Figure 17: synchronous reading and parsing of the file.....	59
Figure 18: asynchronous reading and parsing of the file.....	60
Figure 19: example of the JavaScript object.....	61
Figure 20: mapping and parsing the attributes.....	61
Figure 21: parsed attributes.....	62
Figure 22: reducing the mapped array	63
Figure 23: final code snippet	63
Figure 24: results of JavaScript modelling	64
Figure 25: pie chart of "problematic" brands.....	65
Figure 26: example of the JSON file	66
Figure 27: code for clearing the text	67
Figure 28: impact of new rules	81
Figure 29: example of "null" error.....	83
Figure 30: example of type I error	83

1 Introduction

We live in amazing time. Humanity is on the edge of very prominent and rapid changes, in fact they are happening every day and every hour as we breath. The pace of the progress is already hard to keep up to, and it`s only increasing. The level of technological advancement that people are at now is quite remarkable, especially considering that it is truly an ongoing process. Science is showing us the complexity of reality, shining a light on things that were beyond human understanding only a decade ago. With every tiny step made on a way of research, no matter which field of life it is concerned with, people race stretches its understanding of how the universe functions and hence, what is our place in it.

The progress was made possible not by anything else but our own creativity and intelligence: we created computers. Human brain`s capacity for calculations and analysis is tremendous, but is nothing compared to what machines, created specifically for that purpose, can do. Started off as simple calculators, computers now send rockets to space, estimate the distance between stars and galaxies, perform open hearth surgeries, forecast weather, or simply notify me about a new message while lying in my pocket. While computers have been becoming more and more sophisticated and powerful, the amount of data that humanity generates has been increasing in geometrical proportion. Now it would not only be inefficient for people to work with that scale of data, but it would not be simply possible anymore. As of 2016 and according to IBM, every day we create 2.5 quintillion bytes of data.¹ Moreover, by 2016, 90 percent of the world`s data had been created only during 12 prior that date months, and many digital analysts predict its volume to be 40 times bigger by 2020.²

Computing has disconnected itself from the stereotypic claws of being just personal computer, also known as workstations and servers. Most of today`s data is coming from the gadgets that we hardly name computers. Mobile and wearable devices, sensors,

¹ BELFIORE, M. *How 10 industries are using big data to win big*. 2016.

² IBM. *10 Key Marketing Trends for 2017 and Ideas for Exceeding Customer Expectations*. 2017.

cameras, microphones, cars and airplanes, nearly everything created by people now are changing the way data is created, assembled, consumed, and analyzed.

People went further on that road during the last decades: we taught machines how to learn. We in fact made them so smart, that now they are making us look stupid. It sounds scary, and it even may appear so if we look on how increasingly more and more people are losing their jobs, because their job activity is being replaced by simple algorithms, performed by the program. In every industry companies and governments are now concerned with implementing the sharpest algorithms that would perform normal people's tasks. But fear not, it is a good thing. Self-learning machines and their massive usage all over the world free up people's time. Instead of monotonously clicking over the same patterns, filling the same old forms, writing the same emails – overall, doing roughly speaking non-demanding and brain killing things – people now have the time and the motivation to be (or become) truly creative. In the end of the day – is it not we are good at - creativity, imagination, and brining these two together to follow our dreams? Outcome would only benefit us and the world around us.

Our goal is to get in touch with machine learning paradigm, because we want to be a part of the change and we see numerous ways how computers can be used for the greater benefit of the humanity. We decided to start the acquaintance by gradual introduction to the filed of machine learning, and then focusing the research down to identifying best techniques for text classification. The problem of text classification and mechanisms that enable us working with natural language are of a great importance today, since there are enormous amounts of data being created and stored around the globe in textual format.

2 Objectives and Methodology of Thesis

2.1 Objectives

The **aims of the thesis** could be divided into two parts:

1. Theoretical exploration. To analyze existing techniques in the field of unstructured text classification, distinguish those techniques in regard to their implications and identify the most suitable one for the given task.
2. Classifying the given data in a most appropriate way, potentially implementing machine learning techniques to perform the most accurate classification of the given unstructured data set

Named research questions would be:

- What are the methods/techniques for classifying/mapping/clustering hierarchical data based on an unstructured textual description?
- How unstructured textual data can be manipulated in order to extract information from it?
- What is the best way to classify the data given the context of truck industry?

We have set the following hypotheses:

H₁: implication of **machine learning technology** in classification of truck industry related data should support the classification task and **will positively affect the coverage or the accuracy of the model.**

H₀: in the context field of the data, **machine learning technology will cause no effect on the results of the classification.**

2.2 Methodology

In this thesis, implemented methods are to be split among two groups: methods for theoretical part and methods used for practical part.

Theoretical line of work implied analytical and comparative approach. The results of the research on different approaches to machine learning techniques are analyzed. Theory is presented broadly in the beginning to give a proper introduction to the field, whereas closer to the end it focuses more on analysis of natural language processing problematic, as being a key to the thesis. Research includes analysis of scientific literature studying this topic, practical tutorials and code examples.

Practical piece of used methodology would consist of data collection through internet browsing (for prior construction and then finalization of the classifier' rules), data manipulation through programming, logical and deductive thinking for sketching the structure of the model, model creation based on the chosen algorithm.

In the course of the work, the following technology will be used to manipulate and classify the data: JavaScript programming language will be used for data manipulation and "word2vector" machine learning toolkit will be used to withdraw meaningful insights from the data to assist the classification task.

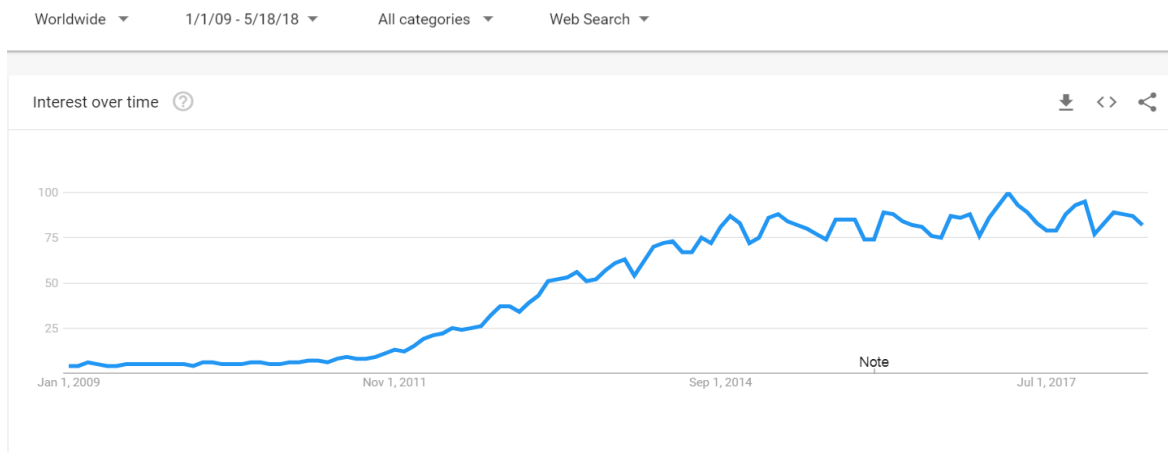
3 Theoretical part

3.1 Concept of Big Data

3.1.1 History

The research has shown that one of the first emergence of the term “Big Data” has been brought to public by John R. Mashey (Chief Scientist, SGI) in his document dated April 1998³. Taking from there, other early publications on the topic refer to Mr. Roger Magoulas from O’Reilly company⁴. Both writings, even though having a little more than a decade time difference, serve as detailed description of the subject and highlight its importance both in current and future time frame. Since 2009, use frequency of these two words put together have been increasing tremendously. The figure below (fig illustrates such frequency from year 2009 onwards. Numbers represent search interest relative to the highest point on the chart for the given region and time. A value of 100 is the peak popularity for the term. A value of 50 means that the term is half as popular. A score of 0 means there was not enough data for this term.

Figure 1: "Big Data" google search query over time



Source: Google Trends, 2018.

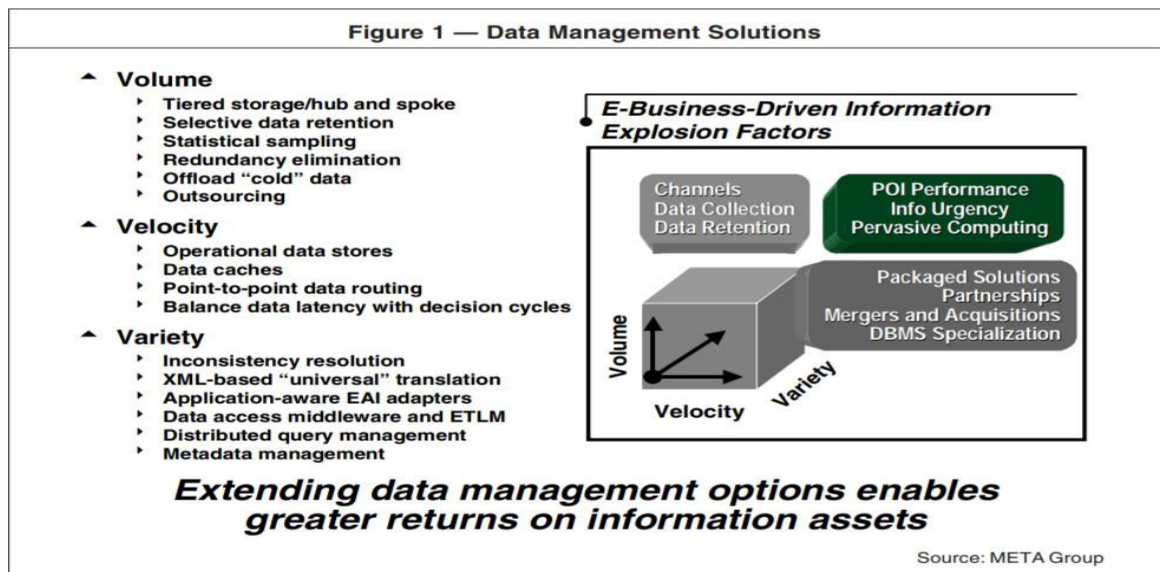
³ MASHEY, J. *Big Data and the Next Wave of InfraStress*. 1998

⁴ MAGOULAS, R., LORICA, B. *Big Data: Technologies and Techniques for Large Scale Data*. 2009.

3.1.2 Definition

As for the definition of the term, it has been successfully coined in the Gartner report in 2001, and later expanded by META Group's analyst during the same year⁵. Once it was formulated there, it is now almost impossible to find definition of the subject that would exclude three V-words: Volume, Velocity, and Variety.

Figure 2: Excerpt from Meta group report on data from 2001 concerning Big Data

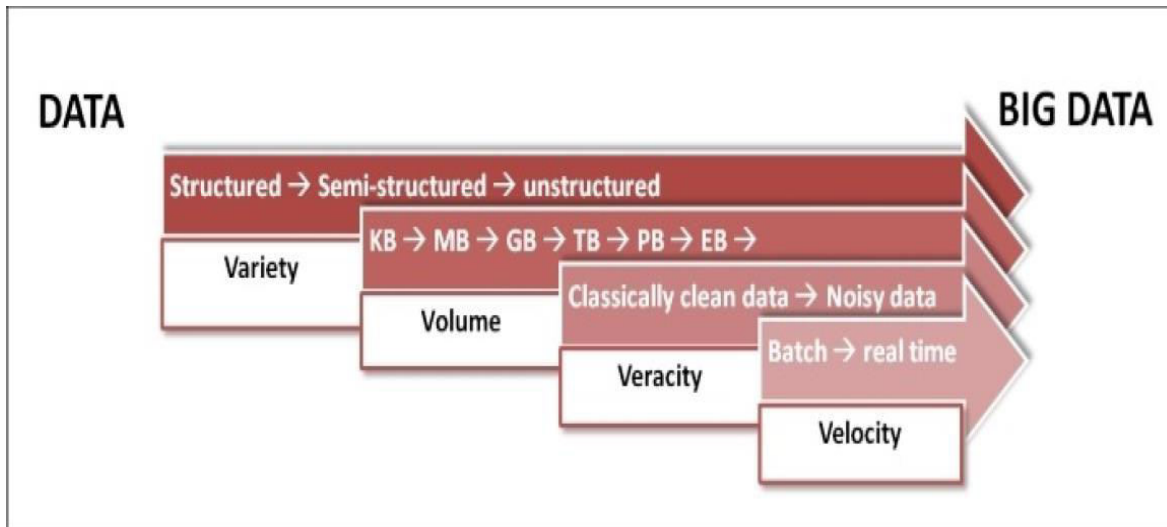


Source: Laney, 2001

Based on the figure above, it can be identified that report has mentioned quite specific data problems associated with its volume, velocity and variety. However, today's definitions of big data that consist of three V try to explain the concept in more general terms. For example, S. Gollapudi defines it in the following manner:

⁵ LANEY, D. *3D Data Management: Controlling Data Volume, Velocity, and Variety*. 2001.

Figure 3 Definition of the big data



Source: Golladupi, 2016

As the picture illustrates, each main characteristic of big data represents increased quantity or complexity over the traditional data features. It means, to be identified as “Big”, data should have certain combination of volume, velocity and variety (sometimes “Veracity” is added as an indication of increased noise in the data). Data can come to process in plain, simple format (e.g. integer numbers from 1 to 10) but at extremely high volumes, or it can be relatively small volume of very complex, unstructured data; it could be considered big only when processing and manipulation of such data is hard to perform with conventional tools.

Now let’s focus on each of the features in more detail.

1. Volume

Big volumes of data are such that typically cannot be easily handled by traditional database engines and can go from scale of terabytes, petabytes, exabytes, or higher.⁶

2. Velocity

Second main characteristic of big data is its increased velocity, in other words: how fast data is processed. For example, in case of troubleshooting in networks, this has been an

⁶ GOLLADUPI, S. *Practical Machine Learning*, Packt Publishing Ltd. 2016.

open question in academic and industrial fields, considering uses from traditional faults in the hardware components like routers or links down to virtual machines.⁷ For some other companies in different industries real-time processing of data is also of a crucial importance: real-time trend analytics in investments finance⁸, real-time video capturing and analyzing for sports statistics⁹, and so on.

3. Variety

Third main characteristic, variety, represents the wide range of how data can look like nowadays. Generally, we can divide data types into three: structured, unstructured and semi structured. To shortly explain all three types, let's recap using L. Pierson formulations¹⁰:

Structured Data: Data is stored, processed, and manipulated in a traditional relational database management system (RDBMS). Examples include both computer (machine) and human-generated data, such as sensors data (e.g. Global Positioning Systems or radio frequency ID tags for containers` tracking), point-of-sale data (that happens when the cashiers around the globe swaps the cards through POS machines) or click-stream data, occurring when humans click on any links around the web.

Unstructured Data: Data that is commonly generated from human activities and doesn't fit into a structured database format. It commonly represents 80% of any organization's data.¹¹ Such data types include satellite images, meteorological data, oceanographic seismic activity data or, when it comes to human-generated types, we can talk about social media data, websites' contents and so on.

Semi-structured: Data doesn't fit into a structured database system, but is nonetheless structured by tags that are useful for creating a form of order and hierarchy in the data. To these types fall EDI, SWIFT, and XML. Usually, semi-structured data comes in simple key/value pairs, e.g. "family" = Smith, "mother" = Nicole, and "son" = John.

⁷ WU, Y., HU, F., MIN, G., Y. ZOMAYA, A. *Big Data and Computational Intelligence in Networking*. 2017

⁸ GARG, A. et al. *Analytics in banking: Time to realize the value*. 2017

⁹ SAS Inc. *Finding the next football star with artificial intelligence*. 2018.

¹⁰ PIERSON, L. *Data Science For Dummies*. 2017, p. 18.

¹¹ HURWITZ, J. et al. *Big Data For Dummies*. 2013, p. 29.

4. Veracity

Fourth, often used feature of big data means reliability, trustworthiness of the data. For example, if someone obtains the data from a weather satellite, it does not indicate a truthful information about the actual weather in that specific region. It often must be necessary to get this same data from different satellites and then correlate this data with, let's say, with social reaction in that region by taking Twitter posts containing weather observations. If such observations align with satellite data, the veracity of weather is set.

The amount and the variability of the available data that has been recorded over the last years has increased so tremendously, that it is no wonder that brand-new ways of using such data have followed in. Some of the ways even put at question usage of conventional programming tools for withdrawing insights from such data. It became more evident that it would be more efficient to teach machines how to learn, and then use that to derive knowledge from all this new data.

3.1.3 How Big Data entails the use of Machine Learning

Scientific progress enhanced our ability to observe reality through allowing creation of more and more new ways of recording data, which influences the volume and the richness of observations. Today's sensors that now can record visual, audial, kinesthetic and even gustative¹² data inputs. There are now databases with large volumes of data about virtually everything, from weather temperature and pressure data, traffic lights and sidewalks surveillance video recordings to transactions, friend requests and chat messages. Large and unique datasets are now accessible only a google query away. Such wealth of all sorts of data has the potential to be analyzed and made sense of.

¹² LAKSHMI, P. K., REWANTHWAR, S. L. *Electronic tongue: An analytical gustatory tool. J Advanced Pharmaceutical Technology Research*. 2012, pp. 3–8.

Scientific field that is concerned in development of computer self-learning algorithms to transform data into knowledge is known as **machine learning**. It evolved as a combination of programming and statistics under a condition of rich volumes of variative data. We will touch the definition of this field more closely in later chapters.

Machine learning works the best when it helps the specialist that is using it, rather than working autonomously in the specialist's field. Medical doctors are using ML (*machine learning – now and further*) on the edge of researches to beat cancer, engineers combine efforts with programmers to construct smart homes and cars, make better cameras and smartphones. ML is being used around the globe in numerous companies, labs, hospitals, schools, military and governmental organizations, not to mention tech industry and its giants. Basically, any institute that generates or stores data is very likely to make use of some ML algorithms to help work with that data.

Here are some bright examples of what has been and what is being achieved implementing machine learning techniques on the big data:

- Spam emails identification¹³
- Customer segmentation (targeting)¹⁴
- Development of algorithms for auto-piloting drones and self-driving cars¹⁵
- Weather and climate short-term and long-term fluctuations¹⁶
- Optimization of energy use in homes and office buildings¹⁷
- Forecasts of election results (even assistance in controlling of public opinion towards election candidates – see Cambridge Analytica)¹⁸
- Juridical decision making in trials¹⁹

¹³ BUZDAR, A. *How To Build a Simple Spam-Detecting Machine Learning Classifier*. 2017.

¹⁴ VIVEK, W. *Clustering algorithms for customer segmentation*. 2018.

¹⁵ SAVARAM, R. *The Machine Learning Algorithms Used in Self-Driving Cars*. 2017.

¹⁶ AMYX, S. *Machine Learning Brings Accuracy to Climate Forecasts*. 2017.

¹⁷ FAYAZ, M., DOHYEUN, K. *Energy Consumption Optimization and User Comfort Management in Residential Buildings Using a Bat Algorithm and Fuzzy Logic*. 2017

¹⁸ CADWALLADR, C., GRAHAM-HARRISON, E. *Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach*. 2018.

- Projection of areas where criminal activity is most likely – predictive police²⁰
- Discovery of genetic sequences linked to diseases²¹

Even though taught machines reach such outstanding milestones, they are still relatively restrained in their capacity to deeply understand a problem. A machine is more capable than a human in discovering hidden patterns and correlations in huge databases, but it still requires homo sapiens to guide the analysis and shift the output into sensible action. Computers can't ask questions; they are unable to know what questions to ask. Instead, they are way better at answering them, taken the question is asked in a comprehensible for machine way.

3.2 Machine Learning

In this part of the thesis we will start with definition, then gradually go through machine learning lifecycle and architecture, identify the main problem categories, differentiate among machine learning sub-fields and define model performance measures.

3.2.1 Definition

This chapter will go through existing definitions of machine learning with a purpose of composing a clear understanding of the concept. Also, a historical background of the term will be shown.

Machine learning lacks one distinctive definition as something that emerged on the edge of several other sciences like statistics, computer science and mathematics. ML is now taught in many universities worldwide as an academic discipline, as well as being widely

¹⁹ CEVRIZ, J. *The AI judge: could we be sentenced to jail by a computer algorithm?* 2017.

²⁰ BENNETT MOSES, L., CHAN, J. *Algorithmic prediction in policing: assumptions, evaluation, and accountability.* 2018.

²¹ LIBBRECHT, M. W., NOBLE, W. S. *Machine learning applications in genetics and genomics.* 2015, pp. 321-32.

promoted and adopted for online courses programs, educational tutorials, trainings and so on. It lays on the intersection of three sciences and encompasses the ideas of artificial intelligence, learning theory, pattern recognition and optimization.

There are enough ways, both functional and technical, in which machine learning can be described. Let us conduce some of them:

"A computer program is said to learn from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E ."

Tom M. Mitchell, Machine Learning, 1997

“Pre-solved data and the resulting output are fed to the computer. These two inputs are used to create a program. This program then can do the job of traditional programming.”

William Sullivan, Machine Learning: Beginners Guide Algorithms, 2017

“Machine learning is a subset of artificial intelligence in the field of computer science that often uses statistical techniques to give computers the ability to "learn" (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed.”

Wikipedia

Among these definitions we notice a core principle, which is basically giving a program something to learn so it can teach itself based on that and then improve its results. We also found another part of the definition quite important: pattern recognition.

Learning here refers to the possibility of machine becoming intelligent enough to withdraw patterns from the provided data. Pattern recognition or pattern search is a field of studies concerned with how machines understand the environment, separate target behavior from the rest and then decide the category of such behavior. The aim is to facilitate speed and accuracy of those decisions, while solving practical and specific problems in as wide as possible range of applications. Hence, the third compartment of the definition for us is the reusability of the machine learning implication: the model should be general to work with any new data or problem.

To differentiate machine learning from classical programming we can state that machine learning is about training machines to kind of program itself by building models using observational data, instead of directly writing strict code that define the model for the data to address a specific classification or prediction. Model means system and is similar to algorithm. If the initial input data changes, the system also adapts to it for the next level of training on the new data. In turn, it indicates that all such a program needs is the ability to process big-scale data, while classic program would require a skilled programmer to rewrite the conditions, which still implies human factor (erroneous) and cannot be guaranteed to be ideal.

3.2.2 Structure and process

Machine learning process can be successfully described in several steps. Some authors shrink the details and put the whole cycle into three steps²²:

1. **Training Phase** – when this phase is complete, it is expected to obtain the learning model on which the upcoming validation will be based. Training data is used during this stage to train such a model.
2. **Validation Phase** – in this phase the model is tested against statistical characteristics (such as error measurement) on how well it has been trained. As a result, initial model is being updated to more complex and reliable version.
3. **Application Phase** – real data is fed into the model with a purpose of extracting meaningful answers.

Some other expert in the field divide learning process into broader five steps, expanding the classical training-validation-application by including initial data gathering and preparation milestones²³:

1. **Data collection** – it includes assembling the material that will be used by the future model, preferably single sourced

²² GOLLAPUDI, S. *Practical Machine Learning*. 2016, p. 65.

²³ LANTZ, B. *Machine Learning with R, second edition*. 2015, p. 39.

2. **Data exploration and preparation** – cleaning data to be ready for the learning process, deleting unneeded data, and precoding it to meet the learner’s expectations and needs
3. **Model training** – machine learning task at hand will imply the use of specific algorithm which will form the data into a model
4. **Model evaluation** – no matter the chosen algorithm (model), it is important to assess its accuracy through testing techniques of choice (e.g. cross-validation)
5. **Model improvement** – in 3 steps approach this part is included in validation step. Again, the purpose is to tweak the model parameters to improve the performance.

Another, more universal way to describe the logic of learning is presented in four steps and given below²⁴:

1. **Data storage** utilizes observation, memory, and recall to provide a factual basis for further reasoning
2. **Abstraction** involves the translation of stored data into broader representations and concepts
3. **Generalization** uses abstracted data to create knowledge and inferences that drive action in new contexts
4. **Evaluation** provides a feedback mechanism to measure the utility of learned knowledge and inform potential improvements.

3.2.3 Problem categories

There are widely differentiated problems that machine learning is faced with, but it is possible to safely categorize them into the following main groups:

- Classification

²⁴ BALI, R., DIPANJAN, S., LANTZ, B., LESMEISTER, C. R: *Unleash Machine Learning Techniques*. 2016.

“Classification is a systematic arrangement in groups or categories according to established criteria”.

Merriam-Webster Dictionary

In machine learning context, it is fair to say that classification is a grouping method or discrimination mechanism in which data entry is being granted a certain class depending on its value. Learning here is based on existing labeled and (ideally) structured dataset: learner use what is known about such set to train machine to recognize the classify future data entries, which helps defining data patterns of any sort.

For instance, take metrics used in nowadays marketing to identify whether it is worth the effort to retain old or obtain new customers in business. Imagine that company has analyzed the customer data through clustering techniques and found out, that it is statistically very likely that customer will stay with a company if it spent more than, for example, 1000\$ per one purchase. Then it is possible to build a classifier that will automatically define new customers as “likely to stay” and “unlikely to stay” based on discovered pattern. This knowledge in turn can be wisely used in the company.

Note, that classification is not necessarily a binary problem: there can be as many classes as needed for the sake of research.

- Clustering (also sometimes referred as pattern discovery)

Every so often, the knowledge is hidden in the data and waits to be unraveled. The fact that it is hidden makes it possible only to make educated guesses about specifics of such knowledge. Today, one of the best ways to prove the educated guess is to build a machine learning clustering algorithm that will unearth unusual pattern or insightful correlation from the unexplored (and often unstructured) data. At the essence, clusters are combinations of unlabeled data inputs with somewhat similar characteristics (or values) while clustering algorithms divide the data into such combinations based on its similarity.

For example, clustering is effectively used to detect customer churn patterns. Customer churn is when a customer decides to finish his or her interactions with a company (e.g. end

a contract). Clustering methods can reveal what matters the most for leaving customers, whether it is having paperless billing, the amount charged to the customer monthly or term of the contract. Clustering is also implemented for segmentation analysis that defines groups of people with similar demographics, behavior or other information, so that ad campaigns could be created for specific audiences.

- Regression or Prediction / Forecasting

Prediction task aims at discovering a relationship between target (predicted, explained) variable and other variables of the set (also called “explanatory”). Despite the literal meaning of the word, prediction not always used to forecast future events: it can be focused on the past or present as well. For example, doctors can determine the conception date of the baby by looking at mothers’ hormones levels.²⁵ Real time application of predictive models would be traffic lights controls in the rush hour, or navigation applications as Waze. Discovery is happening based on the past experience (e.g. time series data); in some cases, there is not enough data and regression is needed to attempt forecasting the future, as regression models quantify in exact terms the relationship between inputs and the target, including the magnitude and uncertainty of such relationship. Results of predictive models always need to be verified and statistically justified and are presented with some degree of uncertainty.

Other tasks for machine learning can be also **simulation** and **optimization**.

Simulation occurs when it is needed to estimate several different outcomes of the future alternative events. Example can be drug design problems, or project management simulations.

Optimization is essentially a process of making something better. In machine learning context, the task can be for example finding the most efficient way to distribute work among the machines or people.

²⁵ LANTZ, B. *Machine Learning with R, second edition*. 2015, p. 42.

3.2.4 Learning subfields

Generally, authors divide machine learning types into several sub-fields. The main differentiation point is relation between human and the machine: the extent to which human is involved in the process. There are different learning styles distinguished within the ML society, and here we will touch the most important and most used ones.

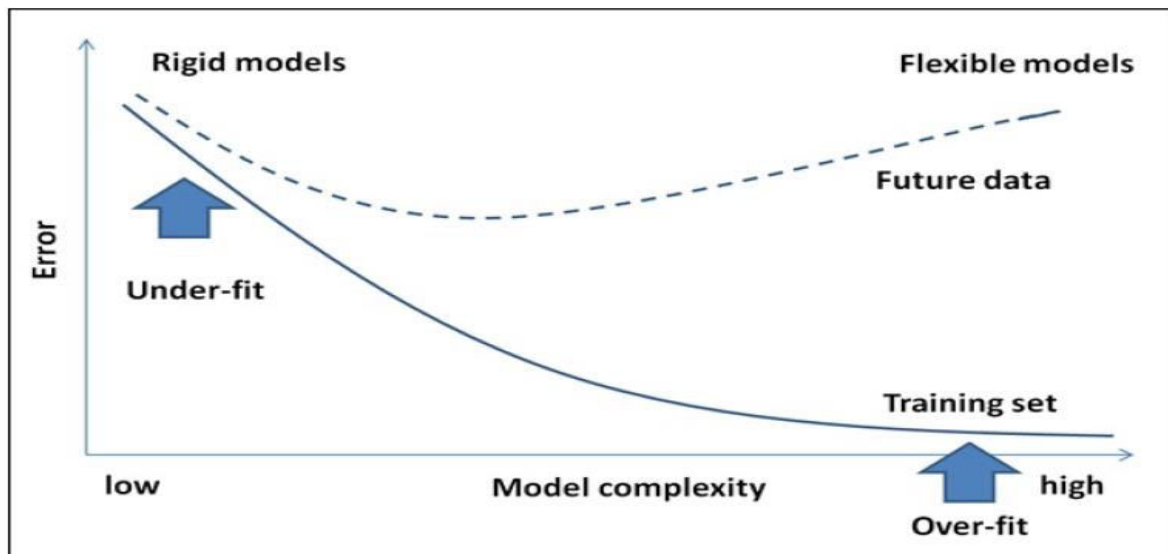
Supervised Learning

In this kind of learning, chosen algorithms are being trained on specifically and carefully constructed set of data, better termed as a training data set. Training data set (sometimes also called “labeled”) usually features a pair of two elements, with input as a string of data (vector)²⁶ and output being already known and desired result or value. Supervised algorithm then uses the training data set to “get acquainted” with the data and form a function. It is worth noting that training process gets repeated until the model is “trained enough” – meaning the model has achieved a desired level of accuracy. The inferred function is then used to properly classify new data inputs, also known as test data. Algorithms classified under this category focus on establishing a relationship between the input and output attributes and use this relationship speculatively to generate an output for new input data points. Well learnt algorithm will correctly identify yet unseen data in a reasonable way.

Supervised learning brings in two main problems: overfitting and generalization. Overfitting issue is met when the algorithm is trained in the way that it is “shaped” too perfectly for the training data, so when it is faced with new data, it can generate big errors. Generalization (sometimes can be referred as underfitting) does the opposite: trained model is too wide and broad, so it handles the new data all right, but with results being not too reliable. Both concepts work around random errors or noise in the input data, while overfitting tries to fit all the noise in, generalization attempts to reduce its effect.

²⁶ BALI, R., DIPANJAN, S. *Machine Learning by Example*. 2016.

Figure 4: overfitting and generalization of models



Source: Gollapudi, 2016

There are two main clusters of algorithms under the category of supervised learning: regression based, and classification based.

First type of supervised algorithms tries to solve the questions of how many/ how much. Such algorithms forecast the output value for new data based on the model, which is in turn based on training data set. Examples include linear regression, multivariate regression, regression trees.

The latter type involves attempts to answer yes-or-no questions (or objective questions), e.g. “is this customer is going to leave?” or “can this drug treat pain?”. Such algorithms predict the class labels for the new data. Examples include: Support Vector Machines (SVM), decision trees, random forests, neural networks, K-Nearest Neighbors.

Unsupervised learning

In this type of learning, it is fair to say that machine learns by its own. It occurs when researcher/s is faced with a learning problem when there are no labeled predefined training set. Machine then processes the data to unleash any hidden patterns, rules, summarize the

data points, discover relationships. The goal is to let algorithm derive meaningful insights from unlabeled (and often unstructured) data.

Similarly to the supervised type, let us divide unsupervised learning into two main fields: association rule-based machine learning and clustering-based machine learning.

First type channels the input data to look for rules and patterns. All in all, those potential rules are desired to reveal interesting connections / logical chains that can occur in the data.

Examples include: Apriori, FP-Growth.

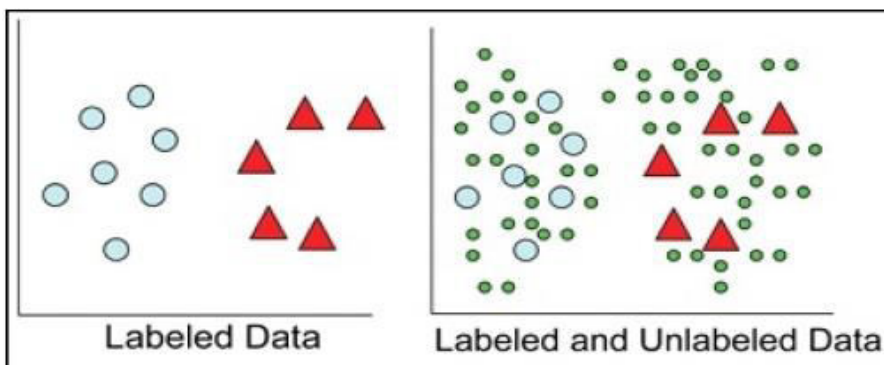
Second type of learning is similar to classification-based problems described in the supervised part. The main goal of such algorithms is to cluster the input data into the different categories operating just with the data alone. The key difference from classification algorithms is that in clustering, labels are not known to the algorithm.

Examples include k-means, k-medoids, hierarchical clustering.

Semi-supervised learning

This type of learning is concerned with using both labeled and unlabeled data for training purposes.

Figure 5: labeled and unlabeled data



Source: Gollapudi, 2016.

In such scenario it is crucial to have right, appropriate assumptions for unlabeled data.

For instance: labeled data is used to help discover that there are specific patterns existing in the input data. The algorithm is then trained on unlabeled data to identify the boundaries of those patterns and might even define new types of patterns that were unspecified in the initial human-entered labels.

There are other types of learning that are worth mentioning.

Reinforcement learning.

One of the more formal ways to define reinforcement learning is the following:

“Reinforcement Learning is defined as a way of programming agents by reward and punishment without needing to specify how the task is to be achieved.”

—Kaelbling, Littman, & Moore, 96

It is a type of learning that emphasize the concept of rewarding algorithm upon reaching the desired result. For instance, when training a dog to give paw, we can give it a little tasty treat every time it successfully performs the task. As a goal, the model should produce a sequence of steps – decisions, that will lead to expected result in the most efficient way possible. It is achieved by programming the algorithm in a way, that it considers data properly and measure trade-offs of every potential decision in a timely manner. Examples of problems that require such method include:

- Chess game: chess players decide on their next moves by predicting what is the set of counter-moves from the opponent. Hence, every decision leading to the result (victory) varies upon the potential responses from the environment.²⁷
- Network package routing: in such case defining a routing policy for dynamically changing networks is performed. Q-learning techniques are used to identify which adjacent node the packet should be routed to.²⁸
- Elevator Scheduling: for building with many floors and several elevators, the task is to schedule elevators in the most reasonable way. Input here being combination of buttons pressed in and out of the lift, combination of floors and number of elevators. In such case, reward would be the least possible waiting time for the user (people).²⁹

²⁷ MANNEN, H. *Learning to play chess using reinforcement learning with database games*. 2003.

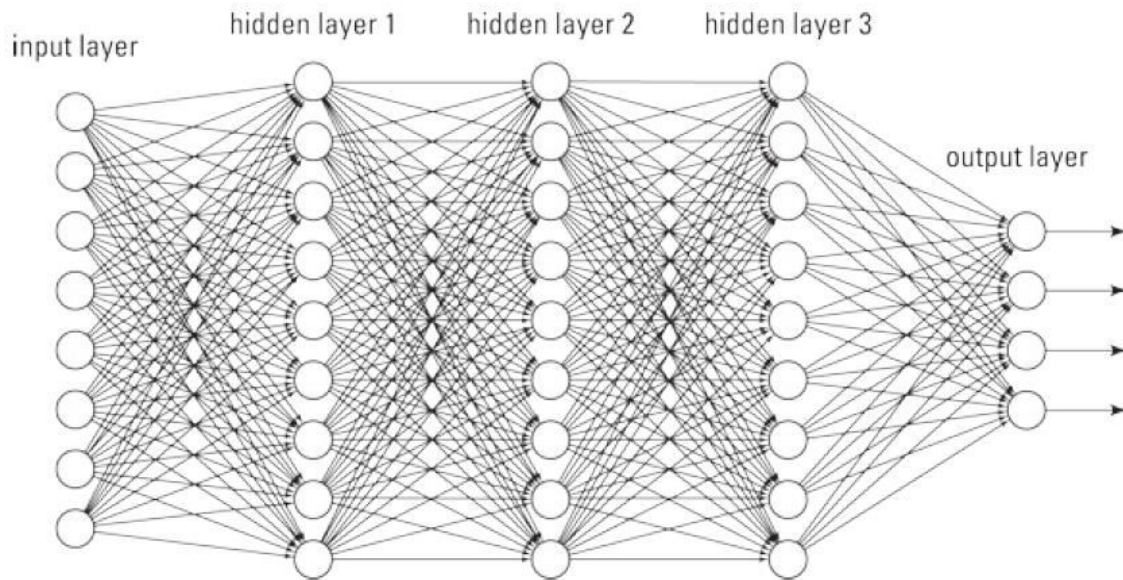
²⁸ BOYAN, J., LITTMAN, M. *Packet Routing in dynamically changing networks: a reinforcement learning approach*. 1993, pp. 671-678

²⁹ JANSSON, A., LINGVALL, K.U. *Elevator Control Using Reinforcement Learning to Select Strategy*. 2015.

Deep learning

Deep learning is human brain-work inspired method of learning that encompasses usage of non-biological neural networks to study and learn the aspects of data and have more than one hidden layer of processing units for named feature extraction.

Figure 6: deep neural network structure - multiple hidden layers



Source: Pierson, 2017

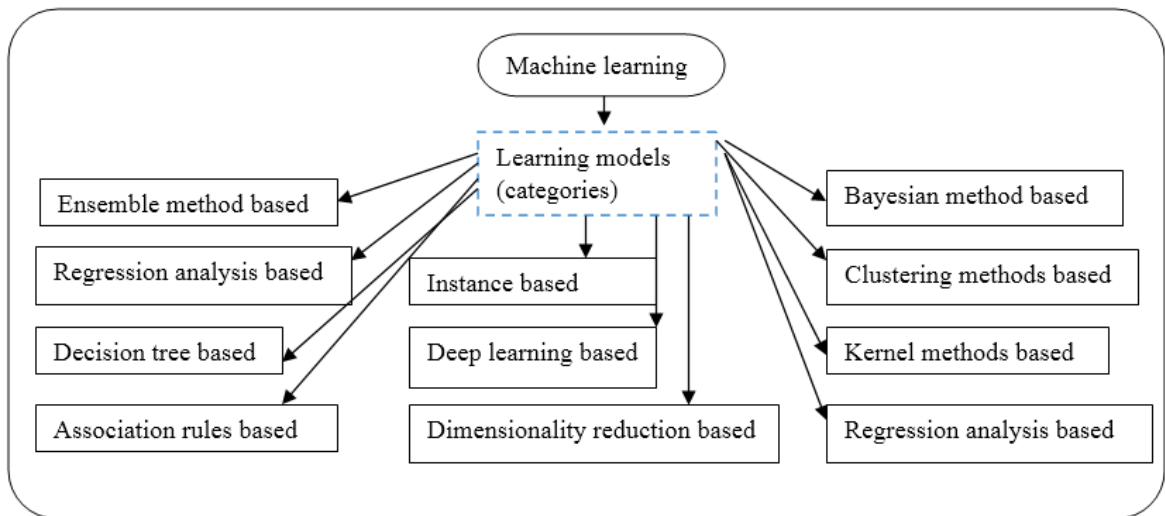
Those processing layers are located between the input and output, while on the in the input such model homes the network on nodes that simulate the neurons of our brain (each node represents a mathematical function) and on the result on the output. Mechanism tries to eliminate the need to create a feature extractor for every category of data. The architectures of deep belief system, deep learning system, recurrent neural networks as well as deep neural networks all are used and applied in many fields including computer vision, speech recognition, machine translation, social network filtering, natural language processing, bioinformatics, sound recognition, etc.³⁰ For example, as deep learning is used to construct networks used for face recognition, deep layers include such face features as color of eyes, length of the eyebrows, distance between

³⁰ SULLIVAN, W. *Machine Learning: Beginners Guide Algorithms*. 2017

the nose and the mouth and so on. Deep learning can be supervised, unsupervised or semi-supervised.

Machine learning algorithms can be categorized not only by the type of learning used (supervised/unsupervised/semi-supervised) or problem it is supposed to face (classification, regression, clustering, optimization), but also in more specific and narrowed way, e.g. by type of algorithm involved:

Figure 7: Machine learning categories by algorithm



Source: author

Describing all existing machine learning categories does not correspond to the aim of this thesis and hence is unnecessary. However, in the following chapters we will look more closely on the algorithms designed to work with our specific problem: unstructured text classification. As for now, let's continue with the main compartments of any machine learning task, specifically moving to performance evaluation techniques.

3.2.5 Performance measures (model evaluation)

Depending on a type of problem (and hence, the type of algorithm being used) there exist different approaches how to assess the model's performance and accuracy. If the algorithm attempts to forecast a certain value (being a continuous variable), the model performance indicators differ from those that used in classification problems.

In the first case scenario, the metrics are received through comparing the predicted values of the algorithm with the real values of the target variables and calculating the average error. Here are the examples:

- **Mean Squared Error (MSE)** – sum of all squares of the difference between the actual and predicted values divided by the number of records (averaged). Take the predicted value of the i^{th} record is P_i and the actual value is A_i , then:

$$MSE = \frac{\sum_{i=1}^n (P_i - A_i)^2}{n} \quad (1)$$

Also, a squared root of MSE is often used and called **root mean square error (RMSE)**.

- **Mean absolute error (MAE)** – sum of all absolute differences between the actual and predicted values divided by the number of records (averaged):

$$MAE = \frac{\sum_{i=1}^n |P_i - A_i|}{n} \quad (2)$$

Studying MAE and MSE (often even RMSE) provides with extra information regarding the distribution of the errors. For example, in regression, if MAE is close to RMSE it indicates that the model makes many relatively small errors. On the other hand, if RMSE is close to MAE^2 , algorithm creates few but large errors.³¹





In case of classification problems, there are various assessment metrics, however, many of them are based on a single very useful tool: confusion matrix.

A confusion matrix is a table that split predictions based on whether they matched the actual value. On the principal diagonal, the predictions matched the reality (True Positive and True Negative), while on the other cells of the matrix algorithm failed to predict

³¹ GOLLAPUDI, S. *Practical Machine Learning*, 2016, p. 83

correctly (False Positive and False Negative). Let us imagine the spam classification problem where the algorithm's results were depicted in a form of 2x2 confusion matrix:

Figure 8: Spam emails confusion matrix

		Predicted to be Spam	
		no	yes
Actually Spam	no	<div style="text-align: center;">  TN True Negative </div>	<div style="text-align: center;">  FP False Positive </div>
	yes	<div style="text-align: center;">  FN False Negative </div>	<div style="text-align: center;">  TP True Positive </div>

Source: Lantz, 2015

Categories can be explained in the following manner:

- True Positive (TP): correctly assigned as the target class
- True Negative (TN): correctly assigned as not the target class
- False Positive (FP): Incorrectly assigned as the target class
- False Negative (FN): Incorrectly assigned as not the target class

Note, that target class is known as *positive* and all the other classed as *negative*.

Using confusion matrix, we can extract such metrics as **model accuracy, error, sensitivity, specificity, precision and recall**. Those metrics can take value from 0 to 1, 1 being the most desired outcome and indicating 100% result.

The **accuracy** is calculated as the sum of correct predictions divided by the total number of predictions, and the **error** can be understood as the sum of all false predictions divided by the number of total predictions, respectively:

$$ACC = \frac{TP + TN}{FP + FN + TP + TN} \quad (3)$$

$$ERR = \frac{FP + FN}{FP + FN + TP + TN} \quad (4)$$

Also, accuracy can be calculated from the error:

$$ACC = 1 - ERR \quad (5)$$

The **sensitivity** (6) of an algorithm (also called the true positive rate) indicates the proportion of positive examples that were rightfully categorized classified, by dividing the number of true positives by the total number of positives (both TP and FN). On the contrary, the **specificity** (7) of an algorithm (also called the true negative rate) shows the proportion of negative examples that were correctly classified. To calculate it, divide the number of true negatives by the total number of negatives (TN + FP):

$$Sensitivity (TPR) = \frac{TP}{TP + FN} \quad (6)$$

$$Specificity (TNR) = \frac{TN}{TN + FP} \quad (7)$$

Precision and **recall** are interconnected performance metrics to those of sensitivity and specificity. They relate to compromised made during classification and attempt to show how relevant the results of classification are, or whether these results are weakened by senseless noise.

Precision is an indicator of how often the algorithm is correct when predicting positive class. Logically, precise algorithm will only predict a positive class when it is most likely to be truly positive.

$$PRE = \frac{TP}{TP + FP} \quad (8)$$

Recall metric, in turn, is denoted in the same way as Specificity or True Positive Rate, however, recall bares a slightly different meaning. It indicates how complete the results of an algorithm are, its spread. A model with high recall would capture a large part of positive examples.

$$REC = \frac{TP}{TP + FN} \quad (9)$$

Often, a combined metric is used – so called **F-measure** (sometimes F1 score, F-score):

$$F_{measure} = \frac{2 \times PRE \times REC}{REC + PRE} = \frac{2 \times TP}{2 \times TP + FP + FN} \quad (10)$$

F-measure encompasses the use of harmonic mean³², since both precision and recall can be viewed as rates from 0 to 1. It can also be useful when comparing multiple models to each other, since F-measure is just one number. However, this assumes that equal weight should be assigned to precision and recall, which is not always the case.

3.2.5.1 Cross validation

Key aim of machine learning model evaluation is to understand how well the model behave with unseen data. Such estimate is called generalization error, and there are two interconnected techniques that help obtaining it: **holdout method** and **k-fold cross validation**. They both in fact are parts of cross validation technique that has been around for more than 50 years now, with early papers mentioning it date back to 1968³³ and 1969³⁴.

- Holdout method

Widely adopted standard for generalization error estimation is a technique of splitting dataset into two parts: training set and test set, while first is being used to train the model and second used for estimation of performance. Split is volatile and can vary from 60/40 (training/testing) to 90/10, usually being one third for testing and two thirds for training, or at least with a condition of test set being smaller than the training set. Holdout method is the simplest kind of cross validation.³⁵ Being easy to implement in its basic simplest format (train set + test set division), holdout method however does not prevent overfitting during the model selection process (we will show the example of parameters tuning later in the work). It happens because researchers or specialists are also interested in improving the

³² Harmonic mean. Retrieved from: https://en.wikipedia.org/wiki/Harmonic_mean. 2018.

³³ LACHENBRUCH, P. A., MICKEY, M. R. *Estimation of error rates in discriminant analysis*. 1968, pp. 1-12.

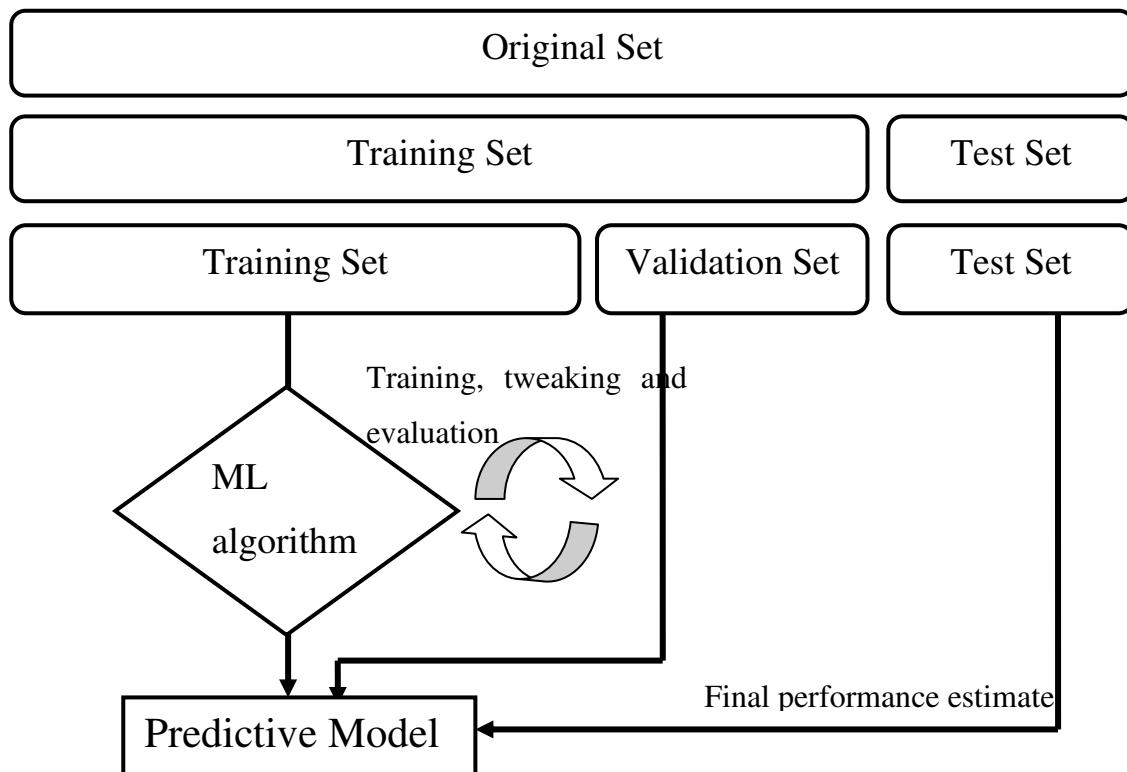
³⁴ LUNTZ, A., BRAILOVSKY, V. *On estimation of characters obtained in statistical procedure of recognition (in Russian)*. 1969.

³⁵ SCHNEIDER, J., MOORE, A.W. *A locally weighted learning tutorial using Vizier 1.0. Tutorial*. 1997.

model by tuning and modifying its parameters (also referred as “hyperparameters”)³⁶. Doing so in the same test dataset repeatedly will eventually turn this set into a part of the training data thus the model most likely become overfitted.

The solution to this issue is creation of additional set of data – validation dataset. Any refining and iterating of the model are then supposed to be tested on such validation dataset, without touching the test dataset – leaving it for the final estimation. A usual division between training, validation, and test would be 50 percent, 25 percent, and 25 percent, respectively.³⁷ Doing so we ensure that the algorithm will not see the final test set until we are done tweaking its parameters, hence the final estimation of its ability to generalize to new data will be less biased. The below figure depicts the concept of holdout cross-validation:

Figure 9: holdout cross-validation



Source: author

³⁶ RASCHKA, S. *Python Machine Learning*. 2016, p. 173.

³⁷ HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. *The Elements of Statistical Learning 2nd edition*. 2009.

It is natural to assume that final performance of the model will be based on how the training set is divided: different samples of data possibly can represent data differently as well. Luckily, there is another, more robust cross-validation technique.

- K-fold cross-validation

K-fold CV randomly splits the data into k separate random partitions named folds, while $k-1$ folds are used for the model training and one fold is used for testing. Such process is repeated k times to receive k models and performance estimates. Number of folds is arbitrary and depends on the size of the set, when small size of k may reduce your variance by not taking all data into account, while large k implies more data per each iteration (use of more training data), hence lowering the bias and increasing the variance. The extreme case of increasing the size of k is so called **leave-one-out cross-validation**, when $k=n$. In that scenario only one training sample is used for testing during each run. LOO CV (leave-one-out cross-validation) is recommended for implementation with small datasets.³⁸ On the opposite, if we are handling large datasets, evidence suggests to pick a smaller size of k (e.g. $k=5$) and still yield an accurate assessment of the algorithm.³⁹

Overall, 5- or 10-fold cross-validation is recommended as a good compromise.⁴⁰ Following figure visualize the process in a clear manner, where $E(\lambda) = \frac{1}{10} \sum_{i=1}^{10} E_i(\lambda)$ represents **cross-validation error** (CV) for chosen tuning parameter λ (for example, subset size⁴¹). It is done for many values of λ and then the value of λ that makes CV (λ) smallest is chosen.

Figure 10: 10-fold cross-validation

³⁸ RASCHKA, S. Python Machine Learning. 2016, p. 177.

³⁹ HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. The Elements of Statistical Learning 2nd edition. 2009, p. 262.

⁴⁰ BREIMAN, L., SPECTOR, P. Submodel Selection and Evaluation in Regression. The X-Random Case. 1992.

⁴¹ HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. The Elements of Statistical Learning: Data Mining, Inference and Prediction. 2009



Source: Raschka, 2016

It also can be considered as efficient to combine k-fold cross-validation with so called grid search for parameters' tuning. Grid search is a “brute-force exhaustive search paradigm where we specify a list of values for different hyperparameters, and the computer evaluates the model performance for each combination of those to obtain the optimal set”⁴².

3.3 Machine learning approaches to classification

Such approaches can be divided into statistical extraction and rules-based⁴³, although quite often these two classes are used together (see, ⁴⁴ ⁴⁵). Most methods for extracting information at any stage require human participation, except for unsupervised learning implications, analyzing only the structure of the text⁴⁶.

⁴² RASCHKA, S. *Python Machine Learning*. 2016, p. 186.

⁴³ CIMIANO, P., HANDSCHUH, S., STAAB, S. *Towards the Self-Annotating Web*. 2004, pp. 462-471.

⁴⁴ PANTEL, P., PENNACCHIOTTI, M. *Automatically harvesting and ontologizing semantic relations*. 2008, pp. 171–195.

⁴⁵ SCHUTZ, A., BUITELAAR, P. *RelExt: A Tool for Relation Extraction from Text in Ontology Extension*. 2005, pp. 593-606.

⁴⁶ CRESCENZI, V. et. al. *Roadrunner: Towards automatic data extraction from large web sites*. 2001, pp. 109–118.

3.3.1 Statistical approaches

Statistical approaches are aimed at learning statistical models or classifiers. Most often, such an approach is based on a machine learning algorithm, such as the Bayes classifier⁴⁷, hidden Markov models⁴⁸, maximum entropy⁴⁹, conditional random fields⁵⁰ or deep learning algorithms^{51 52}.

The basic approach common to most methods consists of 3 stages: preprocessing of data to obtain vectors of documents, training or implementation of classifiers, and finally, post-processing of results.

However, there are differences in the methods of preprocessing and data representation. For example, the deep learning algorithm called *word2vec* is based on the training of neural networks and practically does not require any preprocessing, since the text is converted into a representation of a vector space called embeddings⁵³. Although this is a relatively new approach, it turned out to be so effective that it has already become a classic in the field of word processing using neural networks.

⁴⁷ MAHALAKSHMI, G.S., ANTONY, B.J., KUMAR, A., ROSHINI, B.S. *Domain Based Named Entity Recognition using Naive Bayes Classification*. 2016, pp. 234-239.

⁴⁸ AGEISHI, R., MIURA, T. *Named entity recognition based on a Hidden Markov Model in part-of-speech tagging*. 2008, pp. 397-402.

⁴⁹ CHIEU, H.L., TOUNG, H. *Named Entity Recognition: A Maximum Entropy Approach Using Global Information*. 2003.

⁵⁰ SOBHANA, N.V., PABITRA, M., GHOSH, S.K. *Conditional Random Field Based Named Entity Recognition in Geological Text*. 2010, pp. 119-122.

⁵¹ WANG, C., CHEN, W., XU, B. *Named Entity Recognition with Gated Convolutional Neural Networks*. 2017, pp. 110-121.

⁵² LYU, C., CHEN, B., REN, Y., JI, D. *Long short-term memory RNN for biomedical named entity recognition*. 2017, pp. 462-473.

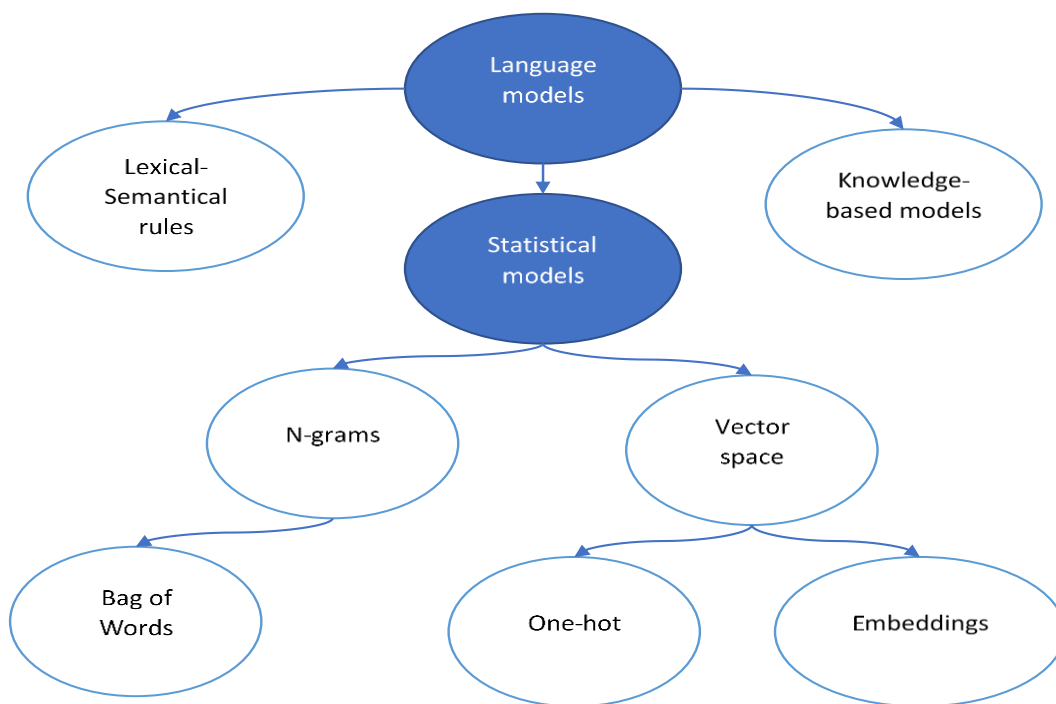
⁵³ MIKOLOV, T. et al. *Efficient Estimation of Word Representations in Vector Space*. 2013

3.3.2 Methods for solving text processing problems

3.3.2.1 Data representation

Before applying the methods of text analysis, it is necessary to determine the way the text is presented. For different approaches to NLP exist corresponding modes of representing words (see fig. 17)

Figure 11: words' representation models



Source: author

The simplest approach to building language models includes various statistical models based mainly on the distribution of words in a document or a set of documents, in other words, on distributional semantics.

Distributional semantics approaches determine semantic proximity between two linguistic units, based on the distribution of words in text boxes. No specialized knowledge of the lexical or grammatical meanings of these units is used. One of the fundamental

assumptions used in distributional semantics is the Harris distribution hypothesis⁵⁴, according to which words with similar meanings tend to occur in similar contexts. Another necessary condition is the formal representation of textual units (usually words) in the form of numerical vectors, for example, using vector models (or embeddings). To represent words in this format, documents are divided into n-grams - sequences of words (in a general case - characters) that are n long.

A special case in the n-gram model occurs when n equals 1. Such a model can be called a Unigram, but more often it is called a bag of words (BoW). This model ignores the connections between words, except for the statistics of occurrence of words. It can be represented as a matrix, where each row represents a separate document or text, and each column is a word found in the corpus. The values at the intersection reflect the number of occurrences of the word in the corresponding document.

One of the main problems of such a model is called a semantic gap. The essence of the problem lies in the formation of sparse word vectors, since the ratio of unique words encountered in the corpus considerably exceeds the number of unique words encountered in a single document. Approaches of distributional semantics try to solve this problem by reducing the dimension of the vector space by the methods of singular decomposition, principal components and others.

Another statistical model - the continuous space model, also helps to cope with the problem of semantic gap. This model presents the text as a continuous stream of words perceived through a context window, which includes a certain number of words to the left and to the right of the considered one. To calculate the vector representation of each word, this type of model uses specialized transformations of character sequences for embeddings. Whereas previous models were words from a collection of documents in the form of a sparse vector space, embeddings are the result of transformation of words into a numerical representation based on the context around these words. Particularly these vectors reflect the probability to meet a certain word in the corresponding environment in the real text.

⁵⁴ HARRIS, Z.S. *Mathematical structures of language*. 1968, p. 230.

This is usually done with the help of various approaches to machine learning, mainly such as neural networks.

3.3.3 Machine Learning Methods for Natural Language Processing Tasks

Currently, machine learning approaches have gained high popularity in the field of NLP, since they have several significant advantages over systems, using manually created linguistic rules for analysis:

- ML methods use statistical analysis to produce models that are resistant to erroneous and unfamiliar input data. At the same time, to teach algorithms with coded logic to make soft, probabilistic decisions is extremely difficult, error-prone and time-consuming.
- The cost of using ML methods consists of machine processing time and time for marking time of the training set (provided that supervised learning is used, and the appropriate training set is not available). Rule-based systems take expert time to develop resources that meet the requirements of the task. As computing power increases, the time required for the application of machine learning methods gradually decreases, while the experts' labor costs remain constant.
- ML methods are easier to scale. To expand a trained model, it is only necessary to add additional examples to the training set, which may well occur in automatic or semi-automatic modes. Programmed logic approaches will require the addition of new logic to the algorithm.

The field of machine learning is extensive and contains many algorithms applicable to many classes of problems. For NLP problems, algorithms based on Bayesian classifiers, logistic regression, hidden Markov fields and many others are used.

3.3.3.1 Naïve Bayes classifier

To use a naive Bayes classifier, you must provide all the words of the input data in the form of a bag of words. When this is done, the Bayesian classifier problem is reduced to finding such a class c_j for which the probability $P(c_j|D)$ is maximum, where $P(D|c_j)$ is

the likelihood or probability of the document D under the condition of the class c_j and $P(c_j)$ is the a priori probability of the class. In addition, in this problem $P(D)$ - the prior probability of the document can be omitted, since the documents are not duplicated, hence this value is a constant.

$$\hat{c} = \operatorname{argmax}_{c_j} \frac{P(c_j) \cdot P(D|c_j)}{P(D)} \quad (11)$$

Document D is represented as a set of attributes $(f_1, f_2, \dots, f_n | c_j) = P(f_1|c_j) \dots P(f_n|c_j)$, where each attribute is an element of the bag of words $w_1, w_2, \dots, w_{|D|}$.

$$\hat{c} = \operatorname{argmax}_{c_j} P(c_j) \cdot \prod_{i=1}^{|D|} P(w_i|c_j) \quad (12)$$

A training set consists of a set of N_{doc} documents, each of which is assigned one or several unique classes with $c \in C$. N_c - is the number of documents belonging to the class c . $C(w, c)$ means the number of occurrences of the word w in a class document c . Then:

$$\hat{P}(c) = \frac{N_c}{N_{doc}} \quad (13)$$

$$\hat{P}(w|c) = \frac{C(w, c)}{\sum_{w'} C(w', c)} \quad (14)$$

The disadvantage of the method is its low adaptability to unknown input data:

- Missing combinations of w and c in the training set.
- Words not found in the training set

Nevertheless, this method is one of the classical machine learning algorithms successfully used in practice. Despite the simplicity of the method, the naive Bayes classifier is

successfully used for the tasks of document classification⁵⁵ and even for analysis of tonality⁵⁶.

3.3.3.2 Maximum entropy model (logistic regression)

Like the naive Bayes classifier, during the learning phase, the maximum entropy model (MaxEnt) is trained based on the weighting table, which reflects the frequency of classifications within the classes - one parameter for each pair of feature class. However, unlike the Bayesian classifier, it is a discriminative classifier trying to model the distribution $P(D|c_j)$ directly, based on the training set.

A typical algorithm for using maximum entropy for NLP problems consists of the following steps:

- First, a set of attributes is generated based on data from the training set.
- Then, based on the training set, the optimal parameters of the scales for the signs are selected.
- After selection of parameters, the formula maximizing the probability of classes is applied:

$$\hat{c} = \underset{c_j}{\operatorname{argmax}} \hat{P}(c|x) = \frac{e^{\sum_{i=0}^k w_i f_i(c,x)}}{\sum_{c'} e^{\sum_{i=0}^k w_i f_i(c',x)}} \quad (15)$$

The most difficult step is the choice of optimal weights w_i . There are several methods for approximating the weight vector. One of them is the maximization of the likelihood function using numerical optimization methods such as gradient descent or generalized iterative ascent⁵⁷. The maximum entropy - based classifier is a very flexible one. It can be

⁵⁵ SUMATHI, S., MOHANA, R. *Document classification using Multinomial Naïve Bayesian Classifier*. 2014, pp. 1557-1563.

⁵⁶ DEY, L. et al. *Sentiment Analysis of Review Datasets Using Naive Bayes and K-NN*. 2015.

⁵⁷ RATNAPARKHI, A. *A Simple Introduction to Maximum Entropy Models for Natural Language Processing*. 1997.

adapted to solve a wide class of NLP problems, in particular, to markup parts of speech⁵⁸, extract dependencies⁵⁹, and others.

3.3.3.3 Hidden Markov models

Hidden Markov models or HMM are finite automata in which transitions between states are weighted by the probabilities of these transitions. HMM is useful when one wants to calculate the probability of a sequence of observed events.

In NLP, the main use of the HMM is related to the task of marking parts of speech and recognizing named entities. A winning criterion for models that use purely statistical approaches is that the SMM focuses on modeling long sequences. This means that this model considers not only statistical data on the words encountered, but also the context in which they occur. In particular, the HMM method implies the conversion of all input data into a single line with a length of T .

3.3.3.4 Conditional random fields

Hidden Markov models, being a generative classifier, calculate probability based on joint distribution. This task is so complex that to solve it, it is necessary to simplify the model, making it “naive”, namely, to introduce a strict restriction on the independence of attributes.

Being a discriminant one, the model of conditional random fields (CRF) has two significant advantages over the HMM:

- It is no longer necessary to calculate the joint distribution. Accordingly, strict restrictions on the independence of attributes are no longer needed.

⁵⁸ RATNAPARKHI, A. *Maximum entropy models for natural language ambiguity resolution*. 1998.

⁵⁹ HALL, K. *K-best spanning tree parsing*. 2007, pp. 392–399.

- Character weights are no longer probabilities, i.e. may take values greater than 1 and less than 0. Among other things, this method, in contrast to the Markov model of maximum entropy (HEMM is another type of Markov model for analyzing texts), does not have the label bias problem⁶⁰.

3.3.4 Deep learning technologies for natural language processing tasks.

3.3.4.1 Algorithms of learning

For the training of neural networks, a vector representation of features is necessary, and embeddings are one of the methods for representing words in a vector space. There are many ways to form embeddings, from simple statistical models, such as a bag of words and a matrix of joint occurrence, to those created using neural networks: word2vec, GloVe, fastText, etc.

Perhaps the most famous technology for teaching the language model using neural networks is word2vec — the subject of a study of two works by the group of Tomáš Mikolov^{61 62} The technology works according to the following principle: a corpus is fed to the input, each word in which is converted into embedding. The resulting word vectors reflect not only the word itself, but also its context, and, therefore, the meaning, if we take into account the Harris hypothesis. This feature of n-gram-based representations is perhaps the most significant, since now, when measuring proximity of vectors, not only statistics but also semantics are considered. It is for this reason that this class of representations is so well suited for learning neural networks.

In his works, Mikolov offers two basic learning algorithms: CBOW (Continuous Bag of Words) and Skip-gram. The CBOW algorithm is aimed at predicting the current word,

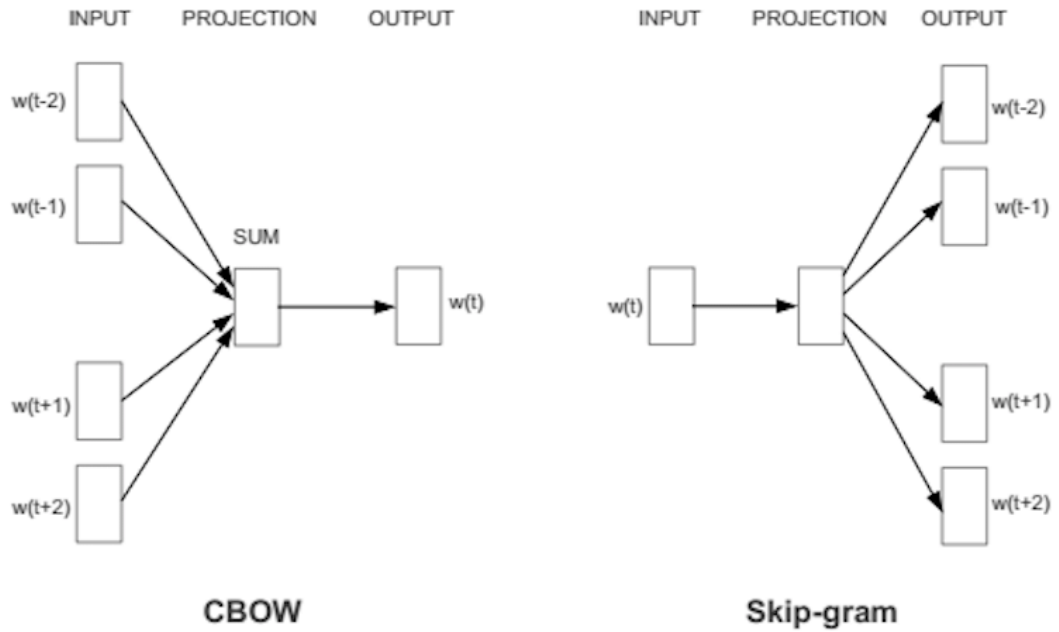
⁶⁰ LAFFERTY, J., MCCALLUM, A., PEREIRA, F. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. 2001, pp. 282-289.

⁶¹ MIKOLOV, T. et al. *Efficient Estimation of Word Representations in Vector Space*. 2013

⁶² MIKOLOV, T. et al. *Distributed representations of words and phrases and their compositionality*. 2013, b. 2, pp. 3111-3119.

based on its context window, while The Skip-gram algorithm is opposite to CBOW - based on the current word, it tries to determine its context (see fig. 18).

Figure 12: Continuous Bag of Words and Skip-Gram algorithms



Source: Camacho-Collados, Pilehvar, 2018.

Respectively, the CBOW algorithm function is:

$$J_{\theta} = \frac{1}{T} \sum_{t=1}^T \log p(w_t | w_{t-n}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+n}) \quad (16)$$

While Skip-Gram function looks like that:

$$J_{\theta} = \frac{1}{T} \sum_{t=1}^T \sum_{-n \leq j \leq n, j \neq 0} \log p(w_{t+j} | w_t) \quad (17)$$

It is important to note, that Skip-gram model has been shown to produce better word-embeddings than CBOW⁶³.

⁶³ MIKOLOV, T. et al. *Distributed representations of words and phrases and their compositionality*. 2013, b. 2, pp. 3111-3119.

Full implementation of word2vec model requires a creating of neural network in order to obtain the weights (vectors) of words which are stored in the middle layer of the network once it has been trained. However, the authors of the model prepared a useful toolkit, that allows exploring the power of the technology without coding a neural network. Such tool is available for free use at <https://code.google.com/archive/p/word2vec/>, and here we will cite the author to explain how it works.

“The word2vec tool takes a text corpus as input and produces the word vectors as output. It first constructs a vocabulary from the training text data and then learns vector representation of words. The resulting word vector file can be used as features in many natural language processing and machine learning applications.

A simple way to investigate the learned representations is to find the closest words for a user-specified word. The distance tool serves that purpose. For example, if you enter 'france', distance will display the most similar words and their distances to 'france', which should look like:

Figure 13: word2vec toolkit results' example

Word Cosine distance

spain	0.678515
belgium	0.665923
netherlands	0.652428
italy	0.633130
switzerland	0.622323
luxembourg	0.610033
portugal	0.577154
russia	0.571507
germany	0.563291
catalonia	0.534176

Source: Mikolov, 2013

3.3.5 Overview of the algorithms

The key factor when deciding upon an algorithm is the need of the algorithm for the training set and, of course, the presence of this set. Because, despite the theoretical independence of the language, the quality of the algorithm can be closely interrelated with the quality of the training set. If, among other things, the algorithm requires a labeled

learning set, then it is highly likely that for highly specialized subject areas such a set will not exist. Thus, the criterion of adaptability is no less significant than the criterion of language support. According to our research, machine learning algorithms (including deep learning and clustering methods) are in a better position, with the exception of the methods of the naive Bayes classifier and conditional random fields that require retraining to analyze words that are absent in the training set, while deep learning algorithms are the most stable and accurate.

In the case of lexical, hardcoded rules, it is necessary to understand that it is quite possible to create a set of universal rules, but the completeness of the results obtained will be unacceptable for use in practice. To obtain a satisfactory result, one will most likely have to supplement the general set of rules with specialized rules for each subject area, which negates the indicator of adaptability of this class of approaches.

Based on the results of this analysis, it can be said that the most suitable approach to our specific practical task of classification is to combine both lexical rules and the deep machine learning algorithms (namely, word2vec tool). Former – as a main classifier, for the reason of unavailability of the training set and complexity of creating one in such specific field with so many variations among truck models. Latter – as a helpful instrument, for the reason of best performance among other ML methods, and relative ease of implementation.

4 Practical part

4.1 Definition of the problem

In this thesis, the task at hand is to prepare a classifier that will process textual data and will be able to correctly identify brand and model of the vehicle.

Hence, before we approach the problem, it is necessary to explain what kind of data we are working with.

4.1.1 Data and its context

Data for the problem has been received from the Lisbon, Portugal based company called Hypercharge⁶⁴. The company has been established in the year 2016 with focus on providing modern automotive software, particularly for dealership management and B2B vehicle trading. As a result of collaboration with this company, author of the thesis has agreed on receiving the dump from the database of vehicles, 260000 data entries on every vehicle that was entered into their system. This dataset represents a collection of (mostly used) professional vehicles placed on an online platform for remarketing purposes. The data is semi-structured, features like year, euronorm and make have its own field and restricted set of possible values, while model is usually encapsulated in the advertisement title. The dataset was obtained through a daily web scraping process over a period of time (about 6 months), so that vehicles entering the market would be captured by the dataset. Technically speaking, the API of mobile.de was used to query the data. Mobile.de is the biggest platform within Europe of that kind. The way Portuguese team has stored it is a PostgreSQL database (relational) for which the JSON object representing each vehicle was stored. Data has been received in a form of a JSON file, which is a textual file in JavaScript Object Notation format⁶⁵. This file included information on more than 270 thousand vehicles, each one of them presented in a following way: it has the id, the title, the attributes and the description. Each attribute is in turn a collection of “key”:”value”

⁶⁴ Startup Lisboa. *Hypercharge*. 2016.

⁶⁵ Json.org. *Introducing JSON*. 2018.

pairs, representing the metadata on every vehicle, including “brand”, “year”, “emission type” etc.

Here is the example of one vehicle from the original file:

Figure 14: example of one object

```
id: '259442950',
title: 'Iveco ad410t45w Bordmatic',
description: 'allrad fahrzeug top zustand mit vielen extras Ausgestattet',
attributes:
  '{"make": "iveco", "year": 2013, "cabin": "localTraffic", "country":
"de", "damaged": false, "gearbox": "manual", "options": ["cruiseControl",
"fourWheelDrive"], "bodywork": "threesidedTipper", "category":
"truckOver7500", "euronorm": "euro5", "location": {"lat": 48.04715, "lon":
7.884985}, "kilometers": 101000, "sellerName": "meder", "sellerType":
"commsbo", "definitionId": "vehicle", "configuration": "8x8",
"firstRegistration": "12/2013"}' },
```

JavaScript ▾

Source: author

4.1.2 The goal of classification

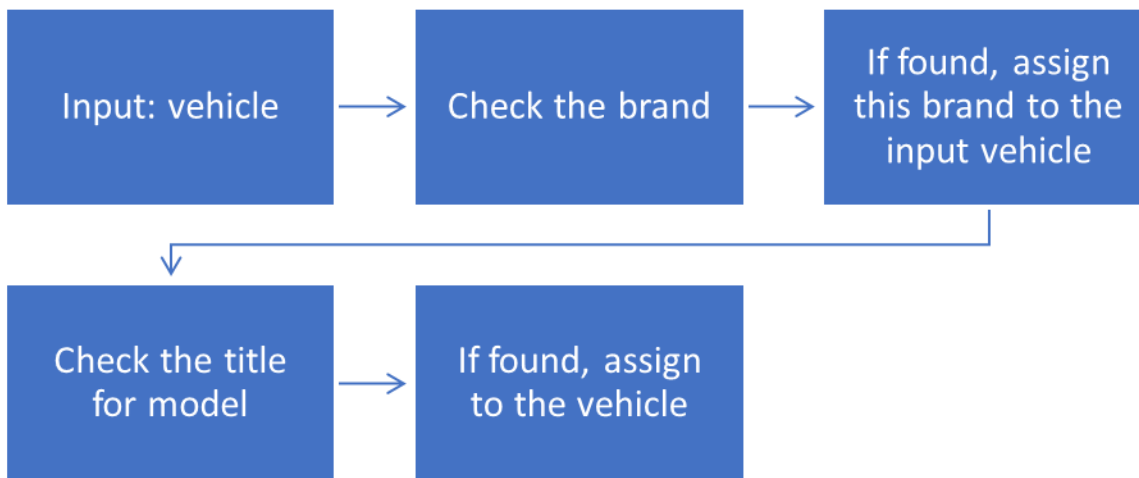
In compliance to the purposes of the Hypercharge, the future classifier must be able to sort through the constantly updated database and correctly identify brand and model of every vehicle.

4.2 Structuring the classifier

While pointing to a correct brand seems straightforward at the first glance because that information in majority of cases is already predefined in the metadata (attributes. make), the main complication is assumed to be in identifying the correct model. Sellers and buyers are not required to enter a model of the vehicle in the specific field, therefore they put it in the title of their advertisement when listing it online. Sometimes their description does not indicate the correct, full denomination of the model, but rather just a part of it. This in turn means that our classifier should be able iterate through attributes to find brand, and then iterate through titles of advertisements and withdraw the model name from there, while correlating it with the specific brand.

Such algorithm can be described in the following scheme:

Figure 15: classifier scheme



Source: author

Based on described above scheme, we assume that our classifier should know every truck brand and model within every brand, so it can find matches in the data when running. To achieve that there can be two possible scenarios:

1. Manually prescribe the behavior of the classifier

That option essentially means writing down all the “rules” by which the algorithm should proceed when dealing with data. It means that supervisor should research on and enlist all the possible truck brands and all the possible truck models for each brand. Such task can be viewed somewhat cumbersome due to the huge variation within truck models’ range throughout the years. To face and define that variation one should carefully inspect the whole lineage of specific car’s manufacturer, and such data might be either not available (for old series) or overlapping in denominations with current models (more examples on that below).

2. Implement a machine learning approach

This way of dealing with a task is desirable, yet hard to make use of. The reason for that is hidden in the process of machine learning itself. In order to teach machine how to distinguish among various encounters of trucks, we first need to help it learn such distinctions using the appropriate training set. Such training set should consist of numerous correct examples of every possible model of every brand within truck production industry.

The feasibility of such task is questioned due to a lot of dependencies and sub rules one should keep in mind while working with a truck data, e.g. some series of models for some manufacturers are being released year after year with the same names, however with changed emission type and horse power capacities. This information is often entered wrongly or in many various ways in description of the vehicle. In other words, vehicle can be assigned a proper model only when certain conditions are met, but, obtaining clear and correct examples of advertisements is too hard of a task. Machine learning algorithms aimed at classifying textual data should be able to review many correctly named entries in order to “remember” it. Meanwhile, while our own research shows that there are no existing ready-made training sets for fields as narrow as “truck industry”, compiling the correct training set with many examples of how specific models can be written down by human encompasses a scale of work beyond this thesis.

Considering challenges of both approaches, it was decided that first approach is more applicable and practical for the task. Nevertheless, this thesis’ theoretical part was focused on machine learning, hence, we will implement machine learning techniques to help ourselves build a proper classifier.

4.2.1 **Constructing the set of rules**

The matching heuristic for determining the vehicle model based on its data (make, year, euronorm, title) was put together to work as follows:

1. Remove blacklisted words from title (e.g., "t/m", "4x4");
2. Based on the knowledge base of Hypercharge and extensive internet research, we have checked which words usually identify a model (e.g., a DAF XF 95 can be specified using XF, XF 95 or just 95) and count how many variations of each model appear in the title. The model with most keywords from its possible variations is chosen.
3. If multiple models tie in keyword count, we validate its euronorm and year properties. For instance, the DAF XF 95 and DAF 95 XF have the same keywords but if the advertisement says its a Euro 4 vehicle, we know it cannot be the 95 XF.
4. If after the validation step we only have 1 possible model, that is the chosen answer, otherwise we emit no answer.

To assist us in the second step, various sources have been explored, including internet catalogues of vehicles (<https://www.bisonparts.co.uk/>, <https://www.trucksplanet.com/>) and official websites of the trucks' manufacturers (<http://tools.mercedes-benz.co.uk/current/trucks/specification-sheets/>, <https://www.iveco.com/czech/press-room/kit/Documents>, <http://www.daf.co.uk/en-gb/trucks/specsheets-search-page>, <https://www.truck.man.eu/konfigurator/#/quickconfig>, <https://www.scania.com/global/en/home/products-and-services/trucks/our-range/r-series/specification.html>).

According to the above-mentioned scheme and considering the querying mechanism, the set of rules have been constructed in the following manner (for example, "Volvo"):

Figure 16: sample from the rules for "Volvo"

```
[
  {
    "make": "volvo",
    "models": [
      { "id": "f7_10_12_16", "names": ["F7", "F 10", "F 12", "F 16"] },
      { "id": "fe", "names": ["FE"] },
      { "id": "fh", "names": ["FH 12", "FH 13", "FH 16", "FH"] },
      { "id": "fs7_10", "names": ["FS 7", "FS 10"] },
      { "id": "flc", "names": ["FLC"] },
      { "id": "fl", "names": ["FL"] },
      { "id": "fm", "names": ["FM"] },
      { "id": "fmx", "names": ["FMX"] },
      { "id": "n7_10_12", "names": ["N 7", "N 10", "N 12"] },
      { "id": "nl10_12", "names": ["NL 10", "NL 12"] }
    ]
  },
]
```

JavaScript ▾

Source: author

A JavaScript application that loads each advertisement's data from a PostgreSQL database and applies the above-mentioned rules was developed.

Heuristics is applied to every input-object of our dataset. When this request is processed, we receive a dump of entries, that has been sorted out and assigned a pair model-brand. Our set of rules has been able to correctly classify roughly 80% of all values. The percentage was assessed by simple division of unclassified data to the initial set, which represented 20% of the data. It is hard to apply verification techniques as Holdout method, since we don't really have a consecutive episodes of training and testing here, or F-measure, since the true negatives or false positives are impossible to measure and even if they were possible to measure, that would only indicate the influence of human factor (incorrectly (FP) or incompletely (TN) entered names of the vehicles). However, we did a thorough check of correctly classified trucks and we struggled to find a mistake. Logically, mistake is possible only for cases when owner/seller of the vehicle mistyped the name of the vehicle completely.

4.2.2 Improvement of the results

After the initial classification was done, it appeared that we can proceed in several ways in order to increase the coverage of the rules, to improve the model.

The most obvious way would be to go into the web and manually check all the websites of truck manufacturers, Wikipedia, trying to find old brochures and journals, digging the truck-lovers forums and do all other sorts of online investigation. Such approach is possible, but time-consuming, ineffective and overall infeasible. In other words, it can be compared to looking for something, somewhere, while not being sure what and where.

However, we could assist ourselves with such impractical task by narrowing the range of our search. First and most important step would be understanding, which brands have caused the most trouble for our rules.

4.3 Data manipulation for information extraction

At this stage of the practical part of the thesis the main task is help us narrow down the scope of the future search and investigation. To do so, we need to study and understand the unclassified 20% of the data, which was obtained in a form of database dump. How to approach such a task?

It became immediately apparent that even though our new object of research is much smaller than the original dataset and consists of 51678 objects as opposed to ~260000. The newly constructed 20% dataset is a file of JSON data format, which means it's structured in a way that is familiar to Java Script, however being language-independent. JSON (Java Script Object Notation) is an open-standard file format that uses human-readable text to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value). It is a very common data format used for asynchronous browser–server communication.

4.3.1 Formatting and filtering a JSON file

There are several ways how to deal with JSON file. Among them one of the most popular would be “jq”, a lightweight and flexible command-line JSON processor⁶⁶. Other options include Python, Pearl, php, Java and other languages⁶⁷. However, I went to a different path by attempting to convert json file into a JavaScript object which would then let me manipulate it in various ways.

To refresh, our task is to filter the file in a way, that we could see the most frequent “makes” of trucks in a form of a list. Let us analyze the problem and showcase the milestones:

1. Converting JSON file into a JavaScript object

⁶⁶ Github.io. *Jq*. 2018.

⁶⁷ Unix & Linux Stack Exchange. *How to extract data from a JSON file*. 2016.

2. Mapping the file in a way so we have only “makes” left for all of the objects within a file
3. Counting each “make” and creating a tally of all makes

4.3.1.1 Prerequisites

Before we could begin working with the file, additional software had to be installed on the computer. Namely, we required IDE (integrated development environment), which is essentially a software application that provides facilities for software development. An IDE normally consists of a source code editor, build automation tools, and a debugger. Basically, IDE is a program that let anybody write and test other programs. Following advices from Portuguese team and generally internet-community opinion⁶⁸, we have decided to go with Visual Studio Code (VS Code). It is a modern - cross-platform editor from Microsoft. It supports almost all programming languages and it is integrated with GIT- a most commonly used version control system. Also, running JavaScript code requires a runtime, environment where it can be performed: turn the JavaScript code into machine code that the computer can understand⁶⁹. In case of JavaScript, as being historically and essentially a browser programming language (the web is based on JavaScript as a behavioral engine for HTML blocks stylized with CSS), we could just test it in Google Chrome or Mozilla Firefox web browsers, while Chrome would be most preferable.⁷⁰ Nevertheless, we decided to proceed with implementing code within a server environment, hereby achieving the associated speed and possibilities of server side (meaning you could run it on your machine as a standalone application). To emulate the backend server environment, there exist various software tools as NodeJS, Go, Elixir, Clojure, Haskell, Java/Kotlin/Scala, PHP, Python, Pharo. However, the most feasible choice would be NodeJS due to its simplicity, non-blocking (asynchronous) character and outstanding community support and spread. According to the stackoverflow.com (the biggest website for developers to exchange knowledge) 2017 survey which have

⁶⁸ Quora. *What is the best JavaScript IDE?* 2018

⁶⁹ UTTARIELLO, J. *The Javascript Runtime Environment*. 2018.

⁷⁰ Stack Overflow. *Javascript Engines Advantages*. 2015.

undergone more than 68 000 developers worldwide, Node.js was considered as the most commonly used technology.⁷¹ After installing Node.js it was also necessary to install NPM, the package manager for the Node JavaScript platform. It puts modules in place so that node can find them, and manages dependency conflicts intelligently. It is extremely configurable to support a wide variety of use cases. Most commonly, it is used to publish, discover, install, and develop node programs.⁷² After all that was done, we could start by creating a new .js file in VS Code.

4.3.1.2 Converting JSON file into a JavaScript object.

This step consisted of two actions:

1. Reading/getting the content of our json file. Textual content of the file remains uninterpretable for computer unless it is loaded into its memory in a form of a string. This process is called input and can be performed by any of the input/output techniques.
2. When the content of the file is read as a string, it then can be converted to a JavaScript object through parsing.

Here are how these two actions can be run:

Figure 17: synchronous reading and parsing of the file

```
var fs = require('fs');  
var obj = JSON.parse(fs.readFileSync('/home/rainzo/Desktop  
/modelshard.json', 'utf8'));  
console.log(obj)
```

JavaScript ▾

Source: author

1st line assigns variable **fs** a function, that requires “fs” package, which is a library with various commands designed for data manipulation. We in particular need “readFileSync” command, that will read our file into a system.

⁷¹ Stack Overflow. *Developer Survey Results*. 2018.

⁷² SCHLUETER, I. Z. *NPM Javascript Package Manager*. 2018.

2nd line then assigns variable **obj** a result of the already mentioned parsing function `JSON.parse`. The parsing function is performed on the content of our file, which we read by the function inside the parenthesis (`fs.readFileSync("location of our file", encoding format)`).

3rd line calls `logs` everything into a console of VS Code program, so we can see the result.

Even though the function was working, it was taking a lot of time to process our file. It was also loading processor of the computer significantly. This was not the most desirable outcome, so we continued to work on the code to improve the performance. It has soon become apparent that we should make use of asynchronous capabilities of `node.js` by writing the code using in a different manner. To explain what is meant by synchronous and asynchronous code, let us use popular examples.

Synchronous: you are in a queue to get a theater ticket. You cannot get one until everybody in front of you gets one, and the same applies to the people queued behind you.

Asynchronous: you are in a restaurant with many other people. You order your food. Other people can also order their food, they don't have to wait for your food to be cooked and served to you before they can order. In the kitchen restaurant workers are continuously cooking, serving, and taking orders. People will get their food served as soon as it is cooked.

Asynchronous code ensures faster speed and performance in cases where many I/O requests are done. Also, it is highly efficient when working with large data.⁷³ Thorough research indicated, that it would be better off using asynchronous “`async/await`” functions build on “`promises`”.^{74 75 76 77}

As a result, we have got the following code, that was running smoothly and quickly:

Figure 18: asynchronous reading and parsing of the file

⁷³ STRINGFELLOW, A. *When to Use (and Not to Use) Asynchronous Programming: 20 Pros Reveal the Best Use Cases*. 2017

⁷⁴ COSSET, D. *Asynchronous code with `async/await`*. 2017.

⁷⁵ ORENDORFF, A. *ES6 In Depth: Arrow functions*. 2015

⁷⁶ MDN Web Docs. *Standard built-in objects: Promise*. 2018

⁷⁷ GUNASEKARA, B. *Node.JS Concurrency with `Async/Await` and `Promises!`*. 2018

```

const fs = require('fs') //requiring textual node module called "fs"
const FILE = '/home/rainzo/Desktop/modelshard.json' //assigning our JSON
file to variable "FILE"
async function read () { //creating asynchronous function "read" that
returns new Promise
  return new Promise((resolve, reject) => { //which resolves to
    fs.readFile(FILE, 'utf8', (err, data) => { //asynchronous reading of
the file
      if (err) reject(err) //if it does not work, it stops the operation
      else resolve(JSON.parse(data)) // function results with parsing the
file
    })
  })
}
read() //that calls the function

```

JavaScript ▾

Source: author

Text after “//” is a commentary to the code and is not part of the code itself.

4.3.1.3 Mapping the file in appropriate way

Upon completion of the first part, we obtained JavaScript object array that consists of 51678 objects, each of them being an individual truck’s description in the following form:

Figure 19: example of the JavaScript object

```

id: '259442950',
title: 'Iveco ad410t45w Bordmatic',
description: 'allrad fahrzeug top zustand mit vielen extras Ausgestattet',
attributes:
  '{"make": "iveco", "year": 2013, "cabin": "localTraffic", "country":
"de", "damaged": false, "gearbox": "manual", "options": ["cruiseControl",
"fourWheelDrive"], "bodywork": "threesidedTipper", "category":
"truckOver7500", "euronorm": "euro5", "location": {"lat": 48.04715, "lon":
7.884985}, "kilometers": 101000, "sellerName": "meder", "sellerType":
"commsbo", "definitionId": "vehicle", "configuration": "8x8",
"firstRegistration": "12/2013"}' },

```

JavaScript ▾

Source: author

Only now we can manipulate the objects and access their values. We are interested in property “attributes”, key “make”. However, we have noticed, that property “attributes” of

the objects is present in a form of a string i.e. line of text. In other words, even though the whole file was parsed, individual elements (attributes) of objects remained unparsed, inaccessible for JavaScript. Hence, to map the whole array to show only brands of trucks involved, we need to parse the “attribute” property of every object, to present attributes as an object, so each of its sub-properties become accessible to us. Then, parsed attributes should be put into a new array for us to continue working with that. We continue the code:

Figure 20: mapping and parsing the attributes

```
async function main () { //begins a new function called "main" in which we
will call the previous function (read)
const originalContent = await read() //it is a part of async/await
functionality. This command assigns to variable "originalContent" a
function, that tells the computer to wait until "read" is performed,
because we need its results
const newArray = originalContent.map(obj => JSON.parse(obj.attributes))
//creates a new array(collection of parsed attributes) that consist of
parsed "attributes" of original objects
```

JavaScript ▾

Source: author

Then, if we log the result into a console, we will see that our new array consists of 55000 objects in a following accessible format:

Figure 21: parsed attributes

```
{ make: 'iveco',
  year: 2013,
  cabin: 'localTraffic',
  country: 'de',
  damaged: false,
  gearbox: 'manual',
  options: [ 'cruiseControl', 'fourWheelDrive' ],
  bodywork: 'threesidedTipper',
  category: 'truckOver7500',
  euronorm: 'euro5',
  location: { lat: 48.04715, lon: 7.884985 },
  kilometers: 101000,
  sellerName: 'meder',
  sellerType: 'commsbo',
  definitionId: 'vehicle',
  configuration: '8x8',
  firstRegistration: '12/2013' }
```

JavaScript ▾

Source: author

4.3.1.4 Counting each “make” and creating a tally of all makes

The next step is to count occurrences of the values for the key "make", which means counting how often each brand of car is met in our "unclassified" set of data. This is achieved through `array.reduce()` method⁷⁸:

Figure 22: reducing the mapped array

⁷⁸ PITZALIS, J. *A Guide to the Reduce Method in Javascript*. 2017.

```

const makes = newArray.reduce(function (tally, value) { // reduce method
  gives us two parameters to operate in a function - first (tally) is a
  total sum that changes after each iteration. Second parameter (value) is
  an iterator that checks every value of our array.
  tally[value.make] = (tally[value.make] || 0) + 1 // here is the
  function: it counts how often each value of make is met, whilst every
  first value is assigned to be 1, and then when it is met again it is
  assigned to become +1.
  return tally //returns result

```

JavaScript ▾

Source: author

In the end, our final code looked like that:

Figure 23: final code snippet

```

const fs = require('fs')
const FILE = '/home/rainzo/Desktop/modelshard.json'

async function read () {
  return new Promise((resolve, reject) => {
    fs.readFile(FILE, 'utf8', (err, data) => {
      if (err) reject(err)
      else resolve(JSON.parse(data))
    })
  })
}

async function main () {
  const originalContent = await read()
  const newArray = originalContent.map(obj => JSON.parse(obj.attributes))
  const makes = newArray.reduce(function (tally, value)
    tally[value.make] = (tally[value.make] || 0) + 1
    return tally
  }, {})
  console.log(makes)
}

main()

```

JavaScript ▾

Source: final.js – attachments to the thesis

And when run using Node.js, it logged into VS Code's console the following results:

Figure 24: results of JavaScript modelling

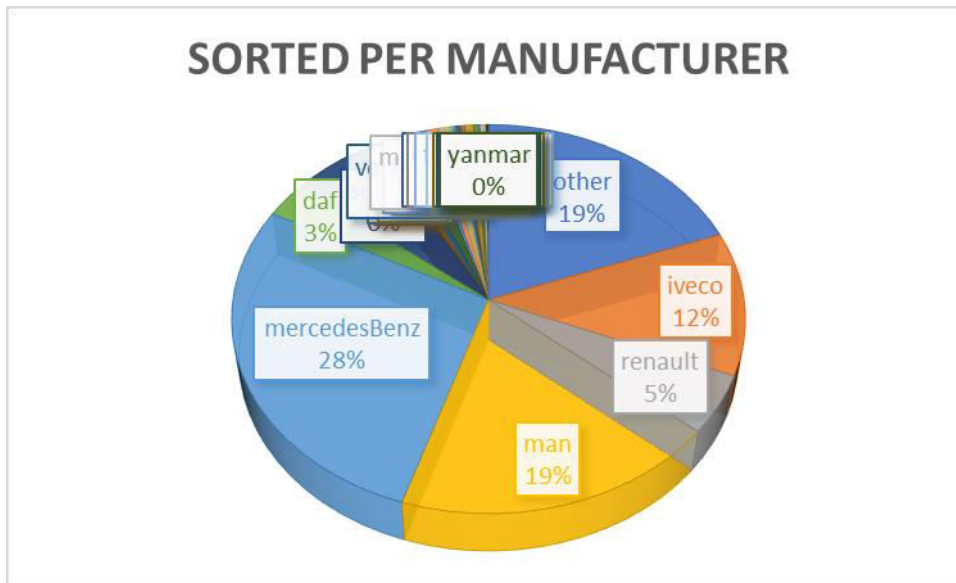
```
{ other: 9965,  
  iveco: 6163,  
  renault: 2511,  
  man: 9763,  
  mercedesBenz: 14359,  
  daf: 1576,  
  scania: 3140,  
  tatra: 202,  
  nissan: 68,  
  ginaf: 213,  
  volkswagen: 473,  
  volvo: 774,  
  palfinger: 246,  
  opel: 420,  
  magirusDeutz: 256,  
  faun: 98,  
  liebherr: 186,  
  demag: 53,  
  undefined: 171,  
  meiller: 95,  
  mitsubishi: 101,  
  schmidt: 306,  
  ruthmann: 7,  
  steyr: 182,  
  freightliner: 53,  
  grove: 64,  
  kamaz: 71,  
  hako: 18,  
  kenworth: 28,  
  mack: 33,  
  peugeot: 39,  
  skoda: 22,  
  toyota: 6,  
  peterbilt: 12,  
  daewoo: 3,  
  yanmar: 1 }
```

JavaScript ▾

[Source: author](#)

Here is the same data put through Microsoft Excel and presented as pie chart:

Figure 25: pie chart of "problematic" brands



Source: author

As we can see, if we just take first 5 most frequent brands (Mercedes-Benz, MAN, Iveco, Renault, DAF) + “other” (which meant the seller of the vehicle entered its brand as “other”), it will make 86% of the all the “problematic” brands.

It is a valuable information which can allow us to proceed and begin research on these 5 brands. However, we can help ourselves even more by implementing machine learning in the process, namely, “word2vec”.

4.3.2 Using word2vec technology to extract patterns from our data

The idea behind implementing w2v technology in our research boils down to applying w2v toolkit to our data in attempt for discovery of meaningful connections within our data. Since w2v represents word as vectors and allows exploring the closest vectors as potentially words with close meanings or interconnected words, we have decided to **apply such mechanism on our unclassified data in attempt to discover exact unidentified models for each of our most problematic brand.**

Such task involved four steps:

1. Obtaining needed data (narrowing down existing data)
2. Preprocessing data (clearing data, getting rid of meaningless symbols)

3. Training w2v model on our data
4. Exploring

Prerequisites

Following the advice of Portuguese team, and after researching the web, it was decided to perform all the operation with w2v using Linux distributive, Ubuntu 16.04, as being more responsive and appropriate to use with such technology.

1. Obtaining necessary data

After querying database using our first set of rules, we have received two datasets: successfully classified 80% and unclassified, problematic 20%. We are interested in the second subset. However, the content of the set is represented in JSON format and each unit (vehicle/object) comprises out of 4 properties, not every one of which is of our interest. We can only be interested in either property “title” or property “description”, both entered manually as text. It was decided that for the purposes of our work, property “title” is the most valuable, since it represents a compressed and laconic description of the vehicle, whereas “description” is often too long and does not always mentions the model of the vehicle. **Sellers always put brand and model name in the title of the advertisement.**

For us to get the titles of all unclassified vehicles, we had to request querying (SQL Query) of the database with that certain condition: to display only title per each vehicle/object. This operation was done in Portugal by Hypercharge team, and as a result we received JSON file in the following format (this file can be found in attachments to the thesis:

Figure 26: example of the JSON file

```
"Other Syst me porte container DTM"  
"Iveco ad410t45w Bordmatic"  
"Renault Renault pulmino 9 posti "  
"Iveco Magirus 120-19"  
"MAN 27.414 6x6 Blatt Blatt "  
"Mercedes-Benz Meiler 3 Seitenkipper f r Abroller"  
"Iveco 190E330 euro 5"
```

JSON ▾

Source: Hypercharge, 2018

The file contained 51679 lines, each of each represented the title of the ad.

2. Preprocessing data (clearing data, getting rid of meaningless symbols)

Before we could train w2v model on our data, we had to get rid of all possible meaningless symbols, following recommendation from the author of the toolkit, Tomáš Mikulov: “Recommended text pre-processing is to keep one sentence / paragraph per line, and possibly discard punctuation and special symbols.”⁷⁹

This was ensured with simple command in Linux terminal⁸⁰:

Figure 27: code for clearing the text

```
tr -d '"/\','<hard-titles.txt >hard-titles-noquotes.txt
```

Bash ▾

Source: author

Where after “-d” we listed all the unwanted symbols, then after “<” indicated original file and after “>” stated where the new content will load into. Command line creates a new file “hard-titles-noquotes.txt” that contains the same text but without symbols.

Newly created file appears then in the same folder as the old one. We also renamed it to be just “thes.txt” for simplicity reasons.

3. Training w2v model on our new, clean data

In implementation we were following the recommendations given by the authors of the model⁸¹ and other materials found on the topic⁸².

After downloading and compiling a toolkit, one should open a terminal in the BIN folder of word2vec directory and there type: “./word2vec”.

Upon completion, the instructions will appear in the terminal:

WORD VECTOR estimation toolkit v 0.1c

Options: Parameters for training:

-train <file> Use text data from <file> to train the model **-output <file>** Use <file> to save the resulting word vectors / word clusters

-size <int> Set size of word vectors; default is 100

⁷⁹ Google Groups. *What Preprocessing Should I Do to a Text Corpus?* 2017.

⁸⁰ Unix & Linux Stack Exchange. *How to Remove the Double Quotes in a CSV.* 2018.

⁸¹ Google Code Archive. *word2vec.* 2013

⁸² MCCORMICK, C. *Word2Vec Tutorial Part 2 - Negative Sampling.* 2017

-window <int> Set max skip length between words; default is 5

-sample <float> Set threshold for occurrence of words. Those that appear with higher frequency in the training data will be randomly down-sampled; default is 1e-3, useful range is (0, 1e-5)

-hs <int> Use Hierarchical Softmax; default is 0 (not used)

-negative <int> Number of negative examples; default is 5, common values are 3 - 10 (0 = not used)

-threads <int> Use <int> threads (default 12)

-iter <int> Run more training iterations (default 5)

-min-count <int> This will discard words that appear less than <int> times; default is 5

-alpha <float> Set the starting learning rate; default is 0.025 for skip-gram and 0.05 for CBOW

-classes <int> Output word classes rather than word vectors; default number of classes is 0 (vectors are written)

-debug <int> Set the debug mode (default = 2 = more info during training)

-binary <int> Save the resulting vectors in binary mode; default is 0 (off)

-save-vocab <file> The vocabulary will be saved to <file>

-read-vocab <file> The vocabulary will be read from <file>, not constructed from the training data

-cbow <int> Use the continuous bag of words model; default is 1 (use 0 for skip-gram model)

Our choice of parameters is presented in the following line of command:

```
rainzo@rainzoY580:~/word2vec/bin$ ./word2vec -train thes.txt -output vec.txt -size 100 -window 7 -negative 10 -cbow 0 -binary 1
```

We have chosen the above parameters for several reasons:

- **-window 7** – even though window around 10 is recommended for small Skip-grams, titles of advertisements are not usually exceeding 5-7 words.

- -negative 10 – negative sampling is chosen over hierarchical SoftMax due to its better performance. Also, k=10 was chosen following the recommendation of using k from 5 to 20 when dealing with small samples.⁸³
- -cbow 0 - skip-gram architecture was proven to be slower but better for infrequent words vs CBOW
- -binary 1 – we need vectors to be written in binary mode, because we plan to apply ./distance tool of w2v on our results to withdraw meaning
- Other parameters were appropriate to leave as default

Model was trained in a matter of seconds and we were ready to apply “./distance” tool to explore our vectors. Also, we have trained a model one more time with the same parameters but with **-binary = 0** and **-output vec2.txt** (different name) so we can have for our disposal txt file with vectors, which can be viewed as display of the frequency of all words⁸⁴. By checking that file, we indicate that Mercedes-Benz is the most frequent word combination in the text. It correlates with our data perfectly.

4. Exploring

We are making an assumption that the meaning of a word can be inferred by the company it keeps. Combining both our insights on frequency of problematic brands, we could now explore those brands using freshly trained w2v model. The task is to run the main 5 brands (words) that encompasses 86% of all problematic brands through our model to see what the closest words to those brands are. Those close words are assumed to be problematic models, not included in the original set of rules. Next step is to check those words one by one to verify if they are really truck models, and if yes, to add them to original rules. When above described task is repeated for all 5 brands and the rules are accordingly updated, we shall test whether the coverage of our model has improved or not. In this task, we follow the descending order of brands occurrences, starting from Mercedes-Benz and finishing

⁸³ MIKOLOV, T. et al. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. pp. 4-5.

⁸⁴ Google Groups. *Word Frequency*. 2015.

with DAF. Source of the following tables is a results of using “./distance” command on vectors file, created as a result of w2v toolkit training.

4.3.2.1 Mercedes-Benz

Word: Mercedes-Benz Position in vocabulary: 1

Word	Cosine distance
2644	0.705737
EPS	0.705234
K	0.689709
MP3	0.686205
Actros	0.675467
MB	0.668553
L	0.653894
V8,	0.653095
LS	0.642835
MP2	0.641210
3353	0.635994
Retarder	0.632401
Mega	0.624795
MegaSpace	0.618184
1213	0.618046
(keine	0.614038
1844	0.606343
2641	0.600720
Kupplungspedal	0.596098
Megaspace	0.593091
V8	0.592499
Kipphydraulik/	0.585551
1838	0.584593
1840	0.584261
1846	0.583071
SCHALTER	0.581651
ACTROS	0.580179
1848	0.579176
1827	0.578884
kein	0.578831
1841	0.577955
keine	0.576297
2646	0.573383
Blatt/Luft	0.572639
2628	0.571699
3328	0.570587
Actross	0.560626

1. 2644 - found it to be Actros 2644 Mercedes⁸⁵ ⁸⁶. Added to the rules under "actros" id.
2. EPS means electronic power shift, a computer-aided shift support⁸⁷. This feature is a part of a big range of models hence is not relevant to the classification task.
3. MP3 - added as Actros MP3. Same for MP2, MP1 and MegaSpace, Megaspace, BigSpace, Bigspace, ACTROS.
4. 3353 = actros⁸⁸. Rule added.
5. LS being a common spec for both actros and axor, while both have overlapping years of production and euro norm. Hence, LS was not added to the rules despite being a close word connection to Mercedes-Benz⁸⁹. However, we decided to check LS in w2v:

Word: LS Position in vocabulary: 13

Word	Cosine distance
Standklima	0.890054
1848	0.869933
1840	0.862432
Megaspace	0.857100
LSNR	0.853580
BIGSPACE	0.840992
1845	0.833045
1843	0.832048
1842	0.827151
NAVi	0.822531
Retarder	0.821472
1844	0.815947
MP4	0.811488
Retarder	0.809168
SAFETY	0.805018
ESP	0.802160
MEGASPACE	0.800919
MP3	0.798963
Standklima	0.794478
PPC	0.792921
1841	0.791708

⁸⁵ Mercedes-Benz UK Ltd. *Specification Sheets / Actros 6x4 Tractor*. 2012

⁸⁶ Mercedes-benz.com. *Actros 2644: Big in Malaysia*. 2018.

⁸⁷ *Elektropneumatische Schaltung*. Retrieved from:

https://de.wikipedia.org/wiki/Elektropneumatische_Schaltung. 2018.

⁸⁸ Bindertrucks.com. *Mercedes Benz Actros 3353 6x 6*. 2018

⁸⁹ *List of Mercedes-Benz Trucks*. Retrieved from: https://en.wikipedia.org/wiki/List_of_Mercedes-Benz_trucks#1990s. 2018.

SAFETY	0.789483
Kipphydraulik/	0.787639
1851	0.785655
MegaSpace	0.784315
BiXENON	0.782164
LSNRL	0.780017
1846	0.779936
Lowliner	0.779291

All following words have been checked and found to be related to Actros Model:

6. **1848, 1842, 1844, 1851, 1841, 1848, 1855, 1860, 2532, 2536, 2541, 2544, 2546, 2548, 2551, 2555, 2560, 2663** based on specifications document.⁹⁰

Then we decided to run through w2v other possible words, related to Mercedes-Benz. We have checked both “Mercedes-benz” and “Mercedes”. For the first it was found that the closest meaningful words are K35/XZ and A380T, which both were found to be one single Iveco model of the Trakker range and added to the Iveco rules. “Mercedes” word revealed no valuable connections.

4.3.2.2 MAN

Word: MAN, Position in vocabulary: 3

Word	Cosine distance
BL	0.696332
FLLC	0.675893
18.430	0.660371
FK	0.656488
Schalter	0.644102
Silent	0.643271
26.440	0.642681
Assmann	0.634771
1.Hand	0.631311
FLS	0.631132
XLX	0.624071
Kran	0.621188
FAK	0.614834
LLLC	0.613995
9.150	0.613131
10.224	0.610369
Steinzange	0.607091

⁹⁰ Daimler AG. *Mercedes-Benz. Actros. Axor.* 2008.

Pferdetransporter	0.606574
26.480	0.604473
18.232	0.604284
TG	0.602282
18.280	0.595417
Hydrodrive	0.593723
26.500	0.593454
Meiller	0.591747
Euro4	0.590719
26.540	0.590585
10.163	0.588129
LLC	0.585889
17.232	0.585806
BL,	0.585658
*Palfinger	0.582722

1. **BL** means leaf suspension on front axle(s), air suspension on rear axle(s).⁹¹
2. **FLLC** was not found.
3. We were using <https://www.truck.man.eu/konfigurator/#/quickconfig> - official MAN vehicle Configurator to verify the other results.
4. After exploring all the current options, it has been found no **18.430** configuration. Checking on mobile.de website gave us the idea, that 18.430 is the only possible combination for discontinued range of MAN TG-A models, hence it was added to the rules.
5. Searching for **FK** on <https://www.bisonparts.co.uk/man> <https://www.trucksplanet.com/catalog/index.php?id=42> and mobile.de did not yield any result. Running FK through w2v did not get us much either:

Word: FK Position in vocabulary: 1279

Word	Cosine distance
10.224	0.781976
*Palfinger	0.767397
Hakenlift	0.753649
3-S	0.729829
19.314	0.728327
17.232	0.727302
2m=3t.	0.721681
9501	0.712814
5.+6.Steuerkreis	0.704734
FB	0.699599
Kran	0.698494

6. **Schalter** - means little window if translated from German.

⁹¹ MAN Truck & Bus AG. *Guidelines to fitting bodies*. 2012.

7. **Silent** - can relate to any model
8. **26.440** - official Man configurator showed no match among current models of MAN. Search on Mobile.de identified different possible combinations for 26.440 model: being both TGA and TGX. Hence, rules are not updated.
9. Further words had no results for us, except of 9.150, which was found to be MAN G-90. Added to the rules (6.100 - 10.150) with years (1987 - 1993)⁹².
10. Running "Man" and "man" through w2v did not yield meaningful results.

4.3.2.3 Iveco

Word: Iveco, Position in vocabulary: 4

Word	Cosine distance
CURSOR	0.676974
ML	0.671107
AT	0.646320
AD	0.628685
190	0.622664
120	0.620383
120E18	0.620085
IVECO	0.609918
120E28/P	0.606986
Trakker	0.600946
80	0.597087
E42	0.583443
ML120E25/P	0.582501
E	0.578953
80E15	0.575628
Cursor	0.572934
260	0.572105
ML120	0.571884
42	0.569929
48	0.567496
140E25	0.564966
18	0.562298
AS440T/P	0.561180
180E30	0.558697
E22	0.556008
MAGIRUS	0.554551
100E18	0.553598
150E25	0.552767
cursor	0.551773
Stralis	0.545836
75	0.543788
EL	0.539479

⁹² Trucksplanet.com. *MAN G90 (MAN-VW)*. 2018.

25/P	0.538575
80E21	0.536972
HI-WAY,	0.536880
AS440	0.535639
E18	0.533293
S	0.531787
ML180E28	0.530144
AS440S45T/P	0.529996

1. Based on internet exploration (apart from <http://mobile.de>, and <http://bisonparts.co.uk>, <https://www.iveco.com/czech/produkty/pages/novy-trakker-cursor-8.aspx>, came in handy) it was identified that Cursor is a type of engine, that can be installed on Trakker, Eurotrakker, EuroTech and Stralis trucks alike. Hence, rules were not modified.
2. **ML** range explained at page 14. ML is a line of EuroCargo models⁹³. Rule added.
3. **AT** - <https://www.iveco.com/ckd/en-us/vehicleslineup/pages/stralis.aspx> and https://www.iveco.com/en-us/press-room/kit/Pages/Trakker_ar_in_brief.aspx was found to be Active Time configuration for both Trakker and for some Stralis trucks. Hence AT is not added to the rules as it is, but only together with the follow-up model number.
4. **AD (active Day)** - was found to be only belonging to Trakker range. Added to rules
5. **120 + 120E18** was found only in ML range of Eurocargo model with ML reference.
6. **AS and AS440** added to Stralis rules.
7. <https://www.iveco.com/ckd/en-us/vehicleslineup/pages/eurocargo.aspx> - Medium range models for Eurocargo added to the rules.
8. Word: iveco Position in vocabulary: 2046. Did not add any valuable insight.

⁹³ Iveco Press Office. *The new Eurocargo in brief*. 2008

4.3.2.4 Scania

Word: Scania, Position in vocabulary: 6

Word	Cosine distance
GB	0.859765
LB	0.854156
R	0.847811
SCANIA	0.840552
420	0.813919
580	0.798531
124	0.774924
NA	0.770580
380	0.761285
R124	0.759344
LA4X2NA	0.758513
GA	0.757855
6x2*4	0.752726
R143	0.748553
R114	0.745417
P	0.744702
400	0.737831
470	0.728646
360	0.726327
164	0.724858
480	0.718341
G	0.715549
Torpedo	0.706801
R164	0.704754
LA4x2NA	0.702778
CB	0.701949
124L	0.699787
340	0.698620
Opticruise	0.698044
113	0.696768
manual	0.694437
8X2	0.692817
8x2	0.689838
Topline	0.686771
124G	0.684467
LA	0.683014
490	0.680811
R164LB6X2NB480	0.679287
OPENING+MANUAL	0.673419
G420	0.669028

1. **GB** means Medium Forward Control Cab + truck bodywork (basic)⁹⁴
2. **LB** means Low-end forward control cab + truck bodywork (basic)

⁹⁴ Scania PRT Range. Retrieved from: https://en.wikipedia.org/wiki/Scania_PRT-range. 2018.

3. **R** has been present to the rules.
4. 420, and 124 (already in the rules) can relate to any of the PGRT range of Scania trucks.
5. **580** is found to be R model of Scania. Added to the rules together with **520, 730** (HP). Based on comparison between <https://www.scania.com/global/en/home/products-and-services/trucks/our-range/r-series/specification.html> and <https://www.scania.com/global/en/home/products-and-services/trucks/our-range/p-series/specification.html> and also same specification sheets for G and L models indicated, that R models have higher horse power models (520, 580 and 730). The rest are similar, hence unfeasible to add to the rules.
6. **164** added as a new rule and represent a model from Scania 4-Series⁹⁵.
7. **R164, R124, R114 and R143** are added based on w2v.
8. Discontinued from 2005 T-series of Scania are produced by Vlastuin company since then⁹⁶. **Torpedo** and **torpedo** added to the rules for T models.
9. **G420** is added to the G model rules.

4.3.2.5 Renault

Word: Renault, Position in vocabulary: 8

Word	Cosine distance
DXI	0.745136
Premium	0.729270
270	0.719411
DCI	0.700137
Midlum	0.698985
PREMIUM	0.697468
480.18T	0.660622
440.19	0.657914
RVI	0.656428
T	0.653243
Dxi	0.650159

⁹⁵ Scania 4 series. Retrieved from: https://en.wikipedia.org/wiki/Scania_4-series. 2018.

⁹⁶ Vlastuin Truckopbouw. *Scania Torpedo*. 2018.

MIDLUM	0.647868
440.19T	0.647661
410.19	0.638056
Maxter	0.636388
lames	0.631840
Premiun	0.628084
dCi	0.627852
440-19T	0.621801
dxi	0.619581
460.19	0.615645
220	0.615457
DEALER	0.608234
440-19	0.605569
Premum	0.598959
TI	0.597142
HR	0.589268
330	0.588687
DXi	0.588627
AE	0.587979
P450	0.587553
380.19	0.586178
Major	0.585766
480.18	0.582924
LANDER	0.579579
dci	0.578962
430-19T	0.578768
FOR	0.577001
Manager	0.575075
460dxi	0.574082

1. **DXI** = Renault Engine, that can be installed in any of their vehicles.
<https://www.youtube.com/watch?v=CQ92Uvm2c3s>
2. **DCI** = diesel Common-rail injection
3. Renault **G** (Manager/Maxter) and Renault **R** (Major) are added⁹⁷
4. **480.18T** was found to be Magnum. The other models (440, 460, 400) can relate to Premium and T, so they were not added to the rules.
5. **460** and **460.18/19T** was found to be able to relate to both Premium and T series.
Left unadded.

4.3.2.6 DAF

Word: DAF, Position in vocabulary: 12

⁹⁷ *Renault Trucks*. Retrieved from: https://en.wikipedia.org/wiki/Renault_Trucks. 2018.

Word	Cosine distance
CF	0.870194
FT	0.779714
FAN	0.763653
430	0.741017
FAR	0.736828
85	0.735754
510	0.735168
85.430	0.733224
460	0.732497
95	0.729931
320	0.698109
250	0.695798
210	0.691096
290	0.689273
85.410	0.687896
Daf	0.686326
XF95	0.684575
SSC	0.680921
XF	0.680884
FAD	0.663906
ATE	0.662149
SC	0.655421
105.460	0.652322
XF460	0.651979
Day	0.650099
cf	0.649854
310	0.647218
Cf	0.645001
95XF	0.641422
FA	0.641414
LF	0.639019
95.480	0.634733
Super	0.628547
XF460FT	0.608954
xf	0.605953
ATI	0.603654
CF85	0.595145
cf310	0.594898
75.310	0.592456
Globe	0.587256

1. **CF** range has been carefully described and categorized
2. **FT** means "tractor" axles configuration rather than rigid (FA) one⁹⁸
3. **FAN** means specific type of axles (as well as FAR and FAD).
4. **85.430** and **95.430** → added to 85 and 95 respectively
5. **510** = either CF 85.510 or XF105.510. Both rules added

⁹⁸ Daf.co.uk. *Specification Sheets*. 2018.

6. **460** = also both 85 and 105 range. **106** range also added.
7. In general, to the CF 85 range there were added: **360, 410, 460, 510**.
8. **210** relates to LF range. "45.210", "210" added to 45lf model rules.
9. However, 320 and 250 were found to correspond both to LF and CF range.
10. 290 correspond to CF series of E6 emission. This led to thorough put through of all E6 CF and XF models, hence the "E6 only" rule would cut off all unrelated models. Also, between these two new CF and XF, the models do not overlap. Hence we added: for CF (**cfEuro6**) - "**290**", "**230**", "**300**", "**370**", "**410**", for XF (**xfEuro6**) - "**450**", "**530**". Same check was performed for LF range (**lfEuro6**) - we added "**150**", "**180**", "**230**" to the rules.
11. Trying "Daf" instead of "DAF" gave us CF85 as the closest match. Combination added to the CF 85 rule:

Word: Daf, Position in vocabulary: 1035

Word	Cosine distance
CF85	0.731954
ssc	0.717045
DAF	0.686326
ft	0.683652

4.3.3 Appliance of the new rules

When we have explored the problematic brands through w2v and updated the rules with new entries, rules were sent to Hypercharge team and requested to be applied to the database. This process usually lasts for several hours, when the application is processing the database. The impact totaled in a 5% increase in coverage, hereby decreasing the number of unclassified vehicles to 38759 down from 51678. However, the accuracy of the model has been negatively impacted (from what we could see when we explored the results), indicating the insufficient research during the w2v phase of the work. Here is the example of changes that new rules have caused to the classification, where the data is presented as follows: previous, new | id, title:

Figure 28: impact of new rules

```
1 null, eurocargo | 0004da48098fddb3021127e974ba2ab4 Iveco ML 80 E18 Do/Ka Pritsche Klima Nebenantrieb E5
2 null, stralis | 0005a9a92e18072b8bf37bbee0842616 Iveco AT 260S33 Y/FS-D - LIFT-LENKACHSE
3 null, stralis | 000d998fd50d571638e684a757fad288 Iveco Hi-Road AT440S46T/P_EEV_Alufelgen
4 null, eurocargo | 001820113951de578d702157352255d3 Iveco Iveco 140e22p cube
5 null, actros | 001ecfaa37c8539b9a49e66ea7b1f815 Mercedes-Benz 2548 L 6x2 Lenk Retarder Klima Meiller Tele E 5
6 null, eurocargo | 00214785ba4138fdae6b0d52b1850c Iveco 120E250
7 null, actros | 002d85fca2d41b96ce30da796d07ba76 Mercedes-Benz 2546L KEIN 2544L 6x2 MEGASPACE EURO 5 FIN:L092..
8 null, actros | 0034c72d9d61d886757bfff0ad46eafe Mercedes-Benz 1845 LS BIGSPACE *ALUS* SAFETY *2 Tanks*
9 null, actros | 0036cd0e322a73b6e10e34cc6822abcc Mercedes-Benz DB1841 Rolfo Pegasus Bj07 2008
10 null, actros | 0044f13c687140946cbe4ed5799a9af0 Mercedes-Benz 1846 Mega Space
11 null, eurocargo | 00493ffee434f79284f024a28d9e02f3 Iveco 120E25/FP EEV KLIMA LBW
12 null, eurocargo | 0049da4527518878454608d8690e1218 Iveco ML80E22/P Ladebordwand
13 tgl, g90 | 004elf2dbc573e15620bbab0c6cdb875 MAN TGL 8.150 4x2 BL Luftfederung Koffer 6,10m LBW
14 null, lfEuro6 | 00562e30b24b95b4b1b471b47c4472bd DAF LF 180
15 null, actros | 005aabc274d4f6ebb56ac67c677106b92 Mercedes-Benz 1841---RETARDER
16 null, eurocargo | 005cefe9738be4d34f815fb05d8b7745 Iveco ML180 E25
17 null, xf95 | 005f9422f9b67e1b65da4c1349b3d535 DAF XF95 430.Manual.Euro 4.Steel Air
18 null, eurocargo | 005fd979f1c7929e8f9cc1706916027 Iveco 160E25 (E5+EEV) Kipper+Kran Hiab XS 088, Drehser
19 null, r | 0065f484b59730c64b5caf2376336ef1 Scania Scania CV R 124
20 null, actros | 006a81eb67d5a38827b8aace0eb50cbf Mercedes-Benz 1944 / 1844 mit Retarder !!!!
21 null, eurocargo | 006f728574a5ad45ec675149a2a1387a Iveco ML 135 E
22 null, actros | 00710a9a4c988f0a7d2a8ed46e8ce9bd Mercedes-Benz 2551 6x2 Containerbil m/løft
23 null, actros | 00813f35a1630c08c90cd58a0f0f127c Mercedes-Benz Actros 2551 V8
24 null, p | 0083ac4997eac9725218e78675d3a0d1 Scania SCANTIA P124 LA(ID10557)
25 null, p | 008413b449fcb23a3404c2b84fcd39da Scania P124 CB 6x4 HZ
26 null, actros | 00874c86a891e065ecfad8dd55a161cf Mercedes-Benz 2532 L, Haller Abfallsammelaufbau M 21
27 null, actros | 0089142eb578ba78671f95df8ac5fbdd Mercedes-Benz 2546
28 null, actros | 00897cec32e050657c3b1168f0926bc8 Mercedes-Benz Mega Space
29 null, actros | 008b9dc2534877d7288f6b9026f7a982 Mercedes-Benz 2544 E5 6x2 Rohr 23 4701, 2 Krammer, 108 230 km
30 null, tga | 00902485ca1492547455396ba3506ada MAN 18.430
31 null, eurocargo | 009c0ff826dcc770d1441108acdde05b Iveco Iveco ML 80 E Meiler 3-Seitenkipper
32 null, eurocargo | 00a233cb1e7852787b7647fbf6f2a322 Iveco ML 120E15
33 null, cfEuro6 | 00a47c8408ad6aab7de4c13982dffe3c DAF CF 410
34 null, actros | 00ae47c8233f90b16f1ccc2cd25ec5bf Mercedes-Benz 1841 LS
35 null, actros | 00b0ccb047d399314d8680a84d33ca8a Mercedes-Benz 1841Megaspace Retarder, deutsches Fahrzeug
36 null, eurocargo | 00d5c109524b06e21f0a914035024a94 Iveco ML80E22P EEV Vollausstattung,Tischer Aufbau,Top!
37 null, actros | 00dc27aca174f410bbe0a4777df13fe8 Mercedes-Benz 1844 Mega Retarder blatt/luft Kipphydraulik
38 null, actros | 00e416580b8a03806ad3e2b3fdee4e7b Mercedes-Benz 1841 Teleskop-Absetzkipper
39 stralis, null | 00eb3ed441b3ad9de304ec429c8316ab Iveco STRALIS M/AD190S31/FP-D PRITSCH&LBW
```

Source: author

As we can see, many of previously unclassified vehicles now has a model assigned to them!

5 Results and Discussion

5.1 Assessment of results

In a course of practical work, we have faced several tasks, such as: creating a classifier based on system of manually prescribed rules, modelling unclassified data to withdraw insights, and applying word2vec algorithm to increase the quality of our classifier.

Henceforth, the results can be divided into three sections:

1. Building initial classifier
2. Modelling the unclassified data
3. Improving the classifier using w2v model

We can assess the results in the following way:

1. Logic behind classifier heuristics was described in previous chapter (4.2). We have tried to include all possible heavy vehicles' models without creating potential ambiguations, but we were limited in that task. Such database does not exist and our research was constricted by what is possible to find online. However, it was a good point of beginning, and the classifier was able to identify 80% of trucks, which, taking in account its low tolerance to error, is a very good starting result.
2. To approach this task, we have decided to use JavaScript programming language. It has yielded a very good result: allowing us to see which brands were the most problematic for the initial classification. The capabilities of the JavaScript allowed us to know exactly the frequency of each brand as well.
3. Implication of w2v algorithm toolkit upon our unclassified data deepen down our research and eventually uncovered models, which, upon inclusion to the classifier, resulted in increased coverage of the model by 5%, rounding up to 85% coverage. However, it has been identified, that some new additions created ambiguity and stopped the classifier from assigning model to the vehicle, or even created a situation, when classifier reassigned the wrong model to initially correctly classified vehicles (also known as type I error in statistics).

Example of the first case scenario is the following:

Figure 29: example of "null" error

```
79 r, null | 01b6d4504af7ed7e64e22a6a47b61c97 Scania R 440 LB 6x2 Abroller/L-Fhs/Ret/Lenkachse/AHK/E5
```

Source: author

Where “r” is what was assigned before and “null” was the newly assigned value, while the rest is ID + title. It means that the classifier faced an ambiguity when trying to process the title. Namely, we assume that the problem occurred because the title of that specific ad included references both to models “R” and “L”, according to our rules and we don’t have any other references for the classifier to look at in Scania section in general (meaning, no “year” or “euronorm” parameter is available).

Example of the second scenario which is classical type I error, can be the following line:

Figure 30: example of type I error

```
51 95xf, xf95 | 0112109431b17943fed58bac6cdf6692 DAF XF 95.430 - 4515
```

Source: author

This example shows that our rules for DAF range of 95XF and XF95 requires further research and modification.

Research questions of this thesis were:

- What are the methods/techniques for classifying/mapping/clustering hierarchical data based on an unstructured textual description?
- How unstructured textual data can be manipulated in order to extract information from it?
- What is the best way to classify the data given the context of truck industry?

We can safely say that we have thoroughly studied and analysed the current approaches to classification of the data, considering the nature of today’s data (Volume, Velocity, Variety and Veracity) and focusing on working with textual data (Natural Language Processing). The most potent approaches are discovered to lay within machine learning paradigm, more specifically – deep learning approaches based on neural networks. We have chosen to implement in our work one of such approaches, word2vec model, as being the easiest to

use in practice by somebody, who practically has from none to little machine learning background, as author of the thesis.

For the second question, we have chosen to proceed with JavaScript programming language, which definitely has proven to be very flexible and powerful tool to work with data and allowed us to derive meaningful information from it.

Third research question can be answered as follows: given a narrow context of the data, it is safe to say that rule-based algorithm is a very good suit to the problem.

As for the thesis hypothesis, we have proved our alternative hypothesis:

“H1: implication of machine learning technology in classification of truck industry related data should support the classification task and will positively affect the coverage or the accuracy of the model.”

Indeed, we have managed to increase the coverage of the model by 5% after we implemented machine learning paradigm, though indirectly.

5.2 Future recommendations

After we updated the rules, we have managed to test them against the database only once, because this action should be performed at Hypercharge side, in Portugal, and their team is very busy. However, only after one repetition, it yielded quite interesting results. Hence, we think it is plausible to continue the improvement of the rules by analysing these results: check when the classifier failed, investigate, correct the rules, then repeat. We also assume, that it is very feasible to try tweaking the parameters of word2vec model and training it on our newly unclassified data in order to map potential additions to the classifier.

Another assumption is that a matcher that was probabilistic and could look at the ad description would probably be more resilient to manual maintenance and increase coverage. However, given the need for very low false positives, it remains to be seen if such an approach would be accurate enough.

6 Conclusion

The reality of modern world is ever so rapidly changing and demanding. Human civilization has pushed technological progress to the extent, when it is no longer possible using old techniques and approaches to make sense of what is going around us, to keep up with volumes and variety of information that we constantly produce, to effectively and timely make decisions not only in scientific or corporate spheres, but in everyday life of individuals. What is socially adapted as “success” and its reach implies dealing with enormous competition; one that is striving for the success, should be able process information and withdraw knowledge from it faster and better, than his or her competitors. To assist us in such task we now use machines, we create programmes to analyse data, but also, we teach machines, so they can teach us. It’s called machine learning. This thesis was not prepared to assess whether contemporary humanity is on a good track with its constant pushing and running and achieving. However, this thesis does research the methods that we use help us doing all the above.

During the work on this thesis, author has studied the concept of big data, as it is today almost the only data that we have. As a matter of course, thesis has also put light on the paradigm of machine learning, its structure and process, problems that it faces, its subfields. Then the research goes deeper into exploring machine learning methods that deal with text processing, and forms an idea of what those methods are, and how they can be implemented.

Following the practical purposes of this work, the author has chosen and created an appropriate classification mechanism that can deal with natural language in a form of a text. Such model required additional research in order to be improved. Thesis has used programming language of choice to model the data and narrow down the future research. Additionally, machine learning technique was used in practice to assist the task of classification by further narrowing of needed research. The model has been improved on a basis of insights, that author has received from the data by above methods.

Even though the author was not able to fully reach desired outcome, being 100% coverage and accuracy of the classification, there has been done significant steps towards that goal.

The author believes, that should further research and modelling continue, the named classification algorithm should be able to become something of a practical use in various fields of life.

As for the moment, this work can be used as decent introductory tool for understating a machine learning paradigm and also as a practical guide for particular tasks of textual data manipulation and classification.

7 References

AGEISHI R., MIURA T. (2008). *Named entity recognition based on a Hidden Markov Model in part-of-speech tagging*. In Proceedings of First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT). pp. 397-402.

AMYX, S. (2017). *Machine Learning Brings Accuracy to Climate Forecasts*. [online] Medium Inc. Available at: <https://medium.com/@ScottAmyx/machine-learning-brings-accuracy-to-climate-forecasts-8d9b287a3a50> [Accessed at 14.08.2018].

- BALI, R., DIPANJAN, S., LANTZ, B., LESMEISTER, C. (2016). R: *Unleash Machine Learning Techniques*. Packt Publishing. ISBN-13: 978-1787127340.
- BALI, R., DIPANJAN, S. (March 2016). *R Machine Learning by Example*. Packt Publishing. ISBN-13: 978-1784390846.
- BELFIORE, M. (2016). *How 10 industries are using big data to win big*. [online] Available at: <https://www.ibm.com/blogs/watson/2016/07/10-industries-using-big-data-win-big/>. [Accessed 8 May 2018].
- BENNETT MOSES, L., CHAN, J. (2018). *Algorithmic prediction in policing: assumptions, evaluation, and accountability*. *Policing and Society*, 28:7, 806-822, DOI: 10.1080/10439463.2016.1253695.
- Bindertrucks.com. (2018). *Mercedes Benz Actros 3353 6x6 | Lastwagen*. [online] Available at: <https://bindertrucks.com/mercedes-benz-actros-3353-6x6/> [Accessed 28 Nov. 2018].
- BOYAN, J., LITTMAN, M. (December 1993). *Packet Routing in dynamically changing networks: a reinforcement learning approach*. NIPS'93 Proceedings of the 6th International Conference on Neural Information Processing Systems.
- BREIMAN, L., SPECTOR, P. (1992). *Submodel Selection and Evaluation in Regression. The X-Random Case*. *International Statistical Review / Revue Internationale de Statistique*. Vol. 60, no. 3, pp. 291-319.
- BUZDAR, A. (2017). *How to Build a Simple Spam-Detecting Machine Learning Classifier*. [online] Medium Inc. Available at: <https://hackernoon.com/how-to-build-a-simple-spam-detecting-machine-learning-classifier-4471fe6b816e> [Accessed 14 Aug. 2018].
- CADWALLADR, C., GRAHAM-HARRISON, E. (2018). *Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach*. [online] The Guardian. Available at: <https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election> [Accessed 14 Aug. 2018].
- CEVRIZ, J. (2017). *The AI judge: could we be sentenced to jail by a computer algorithm?* [online] University of Leuven. [online] Available at: <https://www.law.kuleuven.be/citip/blog/the-ai-judge-could-we-be-sentenced-to-jail-by-a-computer-algorithm> [Accessed 1 Sept. 2018].
- CHIEU H.L., TOU NG H. (2003). *Named Entity Recognition: A Maximum Entropy Approach Using Global Information*. Proceedings of the seventh conference on Natural language learning at HLT-NAACL.
- CIMIANO P., HANDSCHUH S., STAAB S. (2004). *Towards the Self-Annotating Web*. In Proceedings of the 13th international conference on World Wide Web (WWW'04). New York. pp. 462-471.

- COSSET, D. (2017). *Asynchronous code with async/await*. [online] The Practical Dev. Available at: <https://dev.to/damcosset/asynchronous-code-with-asyncawait-7cd> [Accessed 28 Nov. 2018].
- CRESCENZI V., MECCA G., Merialdo P., et. al. (2001). *Roadrunner: Towards automatic data extraction from large web sites*. In Proceedings of the international conference on very large data bases. pp. 109–118.
- Daf.co.uk. (2018). *Specification Sheets*. [online] Available at: <http://www.daf.co.uk/en-gb/trucks/specsheets-search-page> [Accessed 28 Nov. 2018].
- Daimler AG. (2008). *Mercedes-Benz. Actros. Axor*. [online] Available at: https://www.trucksplanet.com/photo/mercedes/actros_mp3/actros_mp3_k1.pdf [Accessed 28 Nov. 2018].
- DEY L., CHAKRABORTY S., BISWAS A., BOSE B., TIWARI S. (2015). *Sentiment Analysis of Review Datasets Using Naive Bayes and K-NN Classifier*. Cornell University Library, Computer Science, Vol. 8, no. 4, pp. 54-62.
- FAYAZ, M., DOHYEUN K. (2018). *Energy Consumption Optimization and User Comfort Management in Residential Buildings Using a Bat Algorithm and Fuzzy Logic*. MDPI, Open Access Journal, vol. 11, pp. 1-22.
- GARG, A., GRANDE, D., MACÍAS-LIZASO, M., SPORLEDER, C., WINDHAGEN, E. (2017). *Analytics in banking: Time to realize the value*. McKinsey Global Banking Annual Review. McKinsey.
- Github.io. (2018). *Jq*. [online] Available at: <https://stedolan.github.io/jq/> [Accessed 2 Nov. 2018].
- GOLLADUPI, S. (January 30, 2016). *Practical Machine Learning*. Packt Publishing Ltd. ISBN-13: 978-1784399689.
- Google Code Archive. (July 30, 2013). *word2vec*. [online] Available at: <https://code.google.com/archive/p/word2vec/> [Accessed 28 Nov. 2018].
- Google Groups. (2015). *Word Frequency*. [online] Available at: <https://groups.google.com/forum/#!msg/word2vec-toolkit/UK29Ke1Rzpg/eBE2qhmmyAkJ;context-place=forum/word2vec-toolkit> [Accessed 28 Nov. 2018].
- Google Groups. (2017). *What Preprocessing Should I Do to a Text Corpus?* [online] Available at: <https://groups.google.com/forum/#!searchin/word2vec-toolkit/preprocessing%7Csort:date/word2vec-toolkit/TI-TQC-b53w/8iPbiEJSAAAJ> [Accessed 28 Nov. 2018].
- GUNASEKARA, B. (February 4, 2018). *Node.JS Concurrency with Async/Await and Promises!*. [online] Medium. Available at: <https://medium.com/platformer-blog/node-js-concurrency-with-async-await-and-promises-b4c4ae8f4510> [Accessed 8 Nov. 2018]

HALL K. (2007). *K-best spanning tree parsing // In Proceedings of the 45th annual meeting of the association of computational linguistics*. pp. 392–399.

Harmonic mean (2018, October 10). Retrieved from:
https://en.wikipedia.org/wiki/Harmonic_mean [Accessed 28 Nov. 2018].

HARRIS Z.S. (1968). *Mathematical structures of language*. Interscience Publishers. ISBN-13: 978-0882759586.

HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. (2009). *The Elements of Statistical Learning 2nd edition*. Springer-Verlag. ISSN: 0172-7397.

HURWITZ, J. et al. (2013). *Big Data for Dummies*. John Wiley & Sons. ISBN-13: 978-1118504222.

IBM (International Business Machines) (2017). *10 Key Marketing Trends for 2017 and Ideas for Exceeding Customer Expectations*. [online]. Available at:
<https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wr112345usen/watson-customer-engagement-watson-marketing-wr-other-papers-and-reports-wr112345usen-20170719.pdf>. [Accessed 8 May 2018].

Iveco Press Office (2008) *The new Eurocargo in brief*. [online] Iveco.com. Available at:
https://www.iveco.com/czech/press-room/kit/Documents/Eurocargo/eurocargo_press.pdf [Accessed 28 Nov. 2018].

JANSSON, A., LINGVALL, K.U. (2015). *Elevator Control Using Reinforcement Learning to Select Strategy*. KTH Royal institute of technology.

Json.org. (2018). *Introducing JSON*. [online] Available at: <https://www.json.org> [Accessed 2 Nov. 2018].

LACHENBRUCH, P. A., MICKEY, M. R. (February 1968). *Estimation of error rates in discriminant analysis*. *Technometrics*, vol. 10, no. 1, pp. 1–12.

LAFFERTY J., MCCALLUM A. AND PEREIRA F. (2001). *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML-2001)*. pp. 282-289.

LAKSHMI, P. K., REWANTHWAR, S. L. (2012). *Electronic tongue: An analytical gustatory tool*. *Advanced Pharmaceutical Technology Research*. pp. 3–8. DOI: 10.4103/2231-4040.93556.

LANEY, D. (6 February 2001). *3D Data Management: Controlling Data Volume, Velocity, and Variety*. META Group. [online] Available at: <https://blogs.gartner.com/douglaney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf>. [Accessed 16 May 2018].

Lantz, B. (July 2015). *Machine Learning with R, second edition*. Packt Publishing Ltd. ISBN 978-1-78439-390-8.

LIBBRECHT, M. W., NOBLE, W. S. (2015). *Machine learning applications in genetics and genomics*. *Nature reviews. Genetics*, Vol. 16, pp. 321-332.

- LUNTZ, A., BRAILOVSKY, V. (1969). *On estimation of characters obtained in statistical procedure of recognition* (in Russian). *Techicheskaya Kibernetica*, vol. 3.
- LYU C., CHEN B., REN Y., JI D. (2017). *Long short-term memory RNN for biomedical named entity recognition*. *BMC Bioinformatics*, Vol. 18. pp. 462-473.
- MAGOULAS, R., LORICA, B. (February 2009). *Big Data: Technologies and Techniques for Large Scale Data*. [online] O'Reilly Media, Inc. Available at: <http://cdn.oreillystatic.com/radar/r2/r2.0.11excerpt.pdf>. [Accessed 18 May 2018].
- MAHALAKSHMI G.S., ANTONY B.J., KUMAR A., ROSHINI B. S. (2016). *Domain Based Named Entity Recognition using Naive Bayes Classification*. *Australian Journal of Basic and Applied Sciences*, Vol. 10, no. 2, pp. 234-239.
- MAN Truck & Bus AG (2012). *Guidelines to fitting bodies. TRUCKNOLOGY® GENERATION A (TGA)*. [online] Available at: https://www.manted.de/manted/aufbaurichtlinien/_pdf/tga_gb.pdf [Accessed 10 Oct. 2018].
- MANNEN, H. (October 2003). *Learning to play chess using reinforcement learning with database games*. Utrecht university.
- MASHEY, J. (1998). *Big Data and the Next Wave of InfraStress*. [online] Available at: https://www.usenix.org/legacy/publications/library/proceedings/usenix99/invited_talks/mashey.pdf. [Accessed 15 Nov. 2018].
- MCCORMICK, C. (January 11, 2017). *Word2Vec Tutorial Part 2 - Negative Sampling*. [online] McCormickml.com. Available at: <http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/> [Accessed 28 Nov. 2018].
- MDN Web Docs. (2018). *Standard built-in objects: Promise*. [online] Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise [Accessed 28 Nov. 2018].
- Mercedes-Benz UK Ltd. (November 2012). *Specification Sheets / Actros 6x4 Tractor*. [online] Available at: <http://tools.mercedes-benz.co.uk/current/trucks/specification-sheets/actros/actros-6x4-tractor-2644-2655.pdf> [Accessed 28 Nov. 2018].
- Mercedes-benz.com. (2018). *Actros 2644: Big in Malaysia*. [online] Available at: <https://www.mercedes-benz.com/en/mercedes-benz/vehicles/trucks/actros-2644-big-in-malaysia> [Accessed 28 Nov. 2018].
- MIKOLOV, T. et al. (2013). *Distributed Representations of Words and Phrases and their Compositionality*. NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2. [online] Available at: <https://arxiv.org/pdf/1310.4546.pdf> [Accessed 1 Nov. 2018].

- MIKOLOV, T. et al. (2013). *Efficient Estimation of Word Representations in Vector Space*. Cornell University Library Computer Science. [online] Available at: <https://arxiv.org/pdf/1301.3781.pdf>. [Accessed 25 Aug. 2018].
- ORENDORFF, A. (2015). *ES6 In Depth: Arrow functions*. [online] Mozilla Hacks – the Web developer blog. Available at: <https://hacks.mozilla.org/2015/06/es6-in-depth-arrow-functions/> [Accessed 28 Oct. 2018].
- PANTEL P., PENNACCHIOTTI M. (2008). *Automatically harvesting and ontologizing semantic relations*. In Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge. pp. 171–195.
- PIERSON, L. (2017). *Data Science For Dummies*. John Wiley & Sons. ISBN-13: 978-1119327639.
- PITZALIS, J. (February 10, 2017). *A Guide to the Reduce Method in Javascript*. [online] freeCodeCamp.org. Available at: <https://medium.freecodecamp.org/reduce-f47a7da511a9> [Accessed 28 Nov. 2018].
- Quora. (2018). *What is the best JavaScript IDE?* [online] Available at: <https://www.quora.com/What-is-the-best-JavaScript-IDE>. [Accessed 28 Nov. 2018].
- RASCHKA, S. (2016). *Python Machine Learning*. Packt Publishing. ISBN-13: 978-1783555130.
- RATNAPARKHI A. (1997). *A Simple Introduction to Maximum Entropy Models for Natural Language Processing*, University of Pennsylvania, Philadelphia, IRCS Report № 97-08.
- RATNAPARKHI A. (1998). *Maximum entropy models for natural language ambiguity resolution*. (Doctoral Dissertation) University of Pennsylvania, Philadelphia.
- SAS Inc. (2018). *Finding the next football star with artificial intelligence*. [online] Available at: https://www.sas.com/en_us/customers/scisports.html. [Accessed 8 Aug. 2018].
- SAVARAM, R. (2017). *The Machine Learning Algorithms Used in Self-Driving Cars*. [online] KDnuggets. Available from: <https://www.kdnuggets.com/2017/06/machine-learning-algorithms-used-self-driving-cars.html> [Accessed 14 Aug 2018].
- SCHLUETER, I. Z. (2018). *NMP Javascript Package Manager*. [online] CLI Documentation. Available at: <https://docs.npmjs.com/cli/npm> [Accessed 28 Nov. 2018].
- SCHNEIDER, J., MOORE, A.W. (February 1997). *A locally weighted learning tutorial using Vizier 1.0. Tutorial*. [online] Available at: <https://www.cs.cmu.edu/~schneide/tut5/node42.html>. [Accessed 29 Aug. 2018].
- SCHUTZ A., BUITELAAR P. (2005). *RelExt: A Tool for Relation Extraction from Text in Ontology Extension*. In Proceedings of the 4th international conference on The Semantic Web (ISWC'05). pp. 593-606.

- SOBHANA N.V, PABITRA M., GHOSH S.K. (2010). *Conditional Random Field Based Named Entity Recognition in Geological Text*. International Journal of Computer Applications, Vol. 1, no. 3, pp. 119-122.
- Stack Overflow. (2015). *Javascript Engines Advantages*. [online] Available at: <https://stackoverflow.com/questions/2137320/javascript-engines-advantages> [Accessed 28 Nov. 2018].
- Stack Overflow. (2018). *Developer Survey Results 2017*. [online] Available at: <https://insights.stackoverflow.com/survey/2017> [Accessed 28 Nov. 2018].
- Startup Lisboa. (June 3, 2016). *Hypercharge*. [online] Available at: <https://www.startuplisboa.com/portfolio-startups/hypercharge> [Accessed 28 Nov. 2018].
- STRINGFELLOW, A. (2017). *When to Use (and Not to Use) Asynchronous Programming: 20 Pros Reveal the Best Use Cases*. [online] Stackify. Available at: <https://stackify.com/when-to-use-asynchronous-programming/> [Accessed 28 Nov. 2018].
- SULLIVAN, W. (2017). *Machine Learning: Beginners Guide Algorithms: Supervised & Unsupervised learning, Decision Tree & Random Forest Introduction*. CreateSpace Independent Publishing Platform, ISBN-13: 978-1975632328.
- SUMATHI S., MOHANA R. (2014). *Document classification using Multinomial Naïve Bayesian Classifier*. International Journal of Engineering and Technology, Vol. 3, no. 5, pp. 1557-1563.
- Trucksplanet.com. (2018). *MAN G90 (MAN-VW) (Commercial vehicles)*. [online] Available at: <https://www.trucksplanet.com/catalog/model.php?id=590> [Accessed 28 Nov. 2018].
- Unix & Linux Stack Exchange. (2016). *How to extract data from a JSON file*. [online] Available at: <https://unix.stackexchange.com/questions/243428/how-to-extract-data-from-a-json-file> [Accessed 28 Nov. 2018].
- Unix & Linux Stack Exchange. (2018). *How to Remove the Double Quotes in a CSV*. [online] Available at: <https://unix.stackexchange.com/questions/338524/how-to-remove-the-double-quotes-in-a-csv> [Accessed 28 Nov. 2018].
- UTTARIELLO, J. (2018). *The Javascript Runtime Environment*. [online] Medium. Available at: <https://medium.com/@olinations/the-javascript-runtime-environment-d58fa2e60dd0> [Accessed 28 Nov. 2018]. <https://medium.com/@olinations/the-javascript-runtime-environment-d58fa2e60dd0>
- VIVEK, W. (2018). *Clustering algorithms for customer segmentation*. [online] Medium Inc. Available at: <https://towardsdatascience.com/clustering-algorithms-for-customer-segmentation-af637c6830ac> [Accessed 14 Aug. 2018]
- Vlastuin Truckopbouw (2018). *Scania Torpedo*. [online] Vlastuin Truckopbouw. Available at: <https://www.vlastuin-truckopbouw.nl/en/scania-torpedo/> [Accessed 28 Nov. 2018].

WANG C., CHEN W., XU B. (2017). *Named Entity Recognition with Gated Convolutional Neural Networks*. In Proceedings of 15th China National Conference of Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data. (CLL and NLP-NABD 2017). pp. 110-121.

WU, Y., HU, F., MIN, G., Y. ZOMAYA, A. (2017). *Big Data and Computational Intelligence in Networking*. CRC Press. ISBN: 9781498784870.