# Text analysis of handwritten production deviations

Master of Science Thesis
University of Turku
Department of Computing
Data analytics
2021
Kaarina Kangas

UNIVERSITY OF TURKU
Department of Computing

Kaarina Kangas: Text analysis of handwritten production deviations

Master of Science Thesis, 52 p., 6 app. p.
Data analytics
May 2021

---

Companies want to understand the latest trends and summarize product status or public opinion based on social media data. Because data is rich and very diverse, there has been a need to create automated and real-time opinion polling and data mining. This need has contributed to the huge popularity of text analysis and at the same time the development and use of it is being applied to more and more industries. Not just for evaluating consumer feedback, for example.
Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence which is focused to enable computers to understand and interpret human language. Its goal and strength is specifically to program computers to process and analyze large amounts of natural language. NLP technology can extract data accurately from text and classify and organize data. Using machine learning methods makes text analysis much faster and more efficient than manual word processing. The methods can be used to reduce labor costs and speed up the processing of texts without compromising on quality.
The main focus of the thesis is to study the textual material received from the client and to develop a prediction model based on it using natural language processing (NLP) techniques. As a research strategy has been used a case study. The obtained text data, sentences about 9000, are from the period 2016/11-2018/9 from the production deviations observed in the welding and assembly process. Text sentences, i.e. user comments, were available at all stages from the detection of a deviation to its solution. This study has focused on the first observational comment written on the deviation. Based on them, a predictive model has been trained that can predict based on the given first comment, what can be the root cause of the deviation.
The research material has been analyzed using both traditional machine learning methods and more advanced deep learning methods, pre-trained FinBERT and multilingual BERT. The accuracy of the model has been a key measure of the superiority of the model. The result was a reliable prediction model that can be used to predict when a deviation falls into class 100 (missing part) or class 200 (other deviations). The best accuracy of the traditional machine learning model was 85.7 % and of the transformer model was 82.6 %. The most common word in the all Finnish sentences was "puuttua" in different forms.

Keywords: NLP, text analysis, machine learning, transformer

# Contents

**1 Introduction** **1**

**2 NLP and Text classification** **4**

2.1 Use cases of text analytics . . . . . 5

2.2 Text analysis and classification tasks . . . . . 6

2.3 Natural language processing, NLP . . . . . 7

2.4 Common NLP tasks . . . . . 8

**3 Approaches** **10**

3.1 Machine Learning and text analysis . . . . . 10

3.2 Traditional Machine Learning classification algorithms . . . . . 11

3.3 Deep Learning and text analysis . . . . . 12

3.4 Transformer models . . . . . 14

3.4.1 BERT . . . . . 16

3.4.2 Finnish BERT model . . . . . 17

3.4.3 Development of transformers . . . . . 18

3.5 Performance metrics . . . . . 19

3.5.1 Classification report . . . . . 20

3.5.2 Machine Learning metrics . . . . . 24

3.5.3 Transformers and metrics . . . . . 26

3.6 Hyperparameters . . . . . 28

3.7 CRISP-DM Methodology . . . . . 28

**4 Case study description and the data** **31**

4.1 Starting point . . . . . 31

4.2 Planning and analysis methods . . . . . 31

4.3 Data introduction . . . . . 34

4.4 Simple tactics to combat uneven training data . . . . . 35

4.5 Visualization of textual data . . . . . 36

4.6 Text analysis workflow . . . . . 37

4.7 Deep learning approach and FinBERT . . . . . 41

4.8 Implementation and challenges . . . . . 42

**5 Results** **43**

5.1 Traditional machine learning methods . . . . . 45

5.2 Transformers FinBert and M-BERT performance . . . . . 46

**6 Discussion and Future** **50**

**References** **53**

**Appendices**

**A Original classes** **A-1**

**B New classes** **B-1**

**C FN and FP values** **C-1**

# 1 Introduction

There is a lot of literature and research on the use of text analysis in the analysis of customer experiences, but less if any, on quality control in industry. Automation and robotics have long played an important role in the industry. In quality control the "eye" of the machine is tireless and is able to distinguish in great detail the different characteristics of the object. Various maintenance and service functions are increasingly moving towards proactive maintenance and predictability of production system processes. The purpose of proactivity is to avoid idling and urgency, which is likely to cause quality deviations. With the help of data analytics, maintenance and service activities can be shifted to more proactive maintenance and the predictability of production systems can be used to try to avoid idle runs and rushes. [1]

Various methods of Natural Language Processing (NLP) have been popular and many papers have been published that solve various tasks in the field, such as text classification [2], named entity recognition [3] or summarization [4]. A paper of the use of traditional machine learning methods and Transformer in text classification has been carried out e.g. Santiago González-Carvajal, Eduardo C. Garrido-Merchán [5]. They have experimentally tested different scenarios of BERT's behavior against the traditional TF-IDF vocabulary fed to machine learning algorithms.

As a research strategy in this thesis has been used a case study. Its purpose is to study in depth only one or a few objects or phenomena. The analysis presented in the work was made from natural language comments concerning one stage of the metal

company's production process, welding and assembly. The aim was to investigate employees' handwritten deviation comments in the quality control system. When the installer detects a deviation, he writes a description of it to the system and selects a code to describe the deviation.

The problem was to find out if the installers set the comments in the right categories and at the same time examine whether the written comments contain other information that could benefit the quality of the production process. Also was asked to explore the possibility of creating a forecast model based on the anomalies observed in the production process, a kind of traffic light model. Data have been collected from a production deviation monitoring system called MES (Manufacturing Excecution System) for the period 2016/11-2018/9. The entries had been made by about 60 people and there are about 9000 comment lines to be handled. The CRISP-DM methodology was used as a tool to perform the steps of the analysis process where applicable.

First, Chapter 2 introduces key concepts such as NLP (Natural Language Processing) and text analysis, and their common functions. The concepts of machine learning and deep learning transformers are shown in Chapter 3. The same chapter discusses performance metrics and key hyperparameters and their impact to analysis. These aspects are not explored in more detail in these studies, but only their central thought pattern and mutual differences are presented. At the end of the chapter is briefly discussed CRISP-DM Methodology and its main steps.

After the concepts, the actual analysis case is described in chapter 4. The chapter reviews how the research began, the design and analysis of the methods, and introduces the data and its visualization. The methods used, such as machine learning and its different stages, as well as the pre-trained deep learning transformers, are presented. The obtained results are reviewed using various methods of analysis in chapter 5. Chapter 6 suggests possible future actions to monitor both anomalies

and thus quality, and how the current model can be utilized and further refined. The final section also discusses other findings and possible actions to improve production quality monitoring.

# 2 NLP and Text classification

Text classification, also known as text categorization, is a classic problem in natural language processing (NLP). Its purpose is to define tags for sentences, queries, paragraphs and documents. It has a lot of applications such as answering questions, spam detection, opinion analysis, newsgrouping, classification of user intentions and content control. Text information can be compiled from various sources, such as online information, emails, chats, social media, tickets, insurance claims, user reviews, and questions and answers from customer service. As a source of information, the text is very broad, but making useful data discoveries from text can be challenging and time consuming due to its unstructured nature. [6]

The actual study of natural language processing began in the 1950s. In 1950, Alan Turing published an article, which introduced the Turing test [7]. It is now considered as a criterion of machine intelligence. The Turing test measures the humanity of artificial intelligence by setting a machine to talk to a human. If a test person is unable to tell if the debater is a human or a machine, the computer passes the Turing test when it manages to at least show thinking like a human. The test is performed on a text-based basis. [8]

Text analysis, text mining and text extraction can be said to be the same thing. They describe the same process, i.e., extracting meaning from data by observing similar patterns. It is good to note, however, that text analysis and text analytics are slightly different things.Text analysis works with concepts, the meaning of text.

Text analysis can be used to answer, for example, the question: is the review positive or negative? Text analytics studies patterns and the results can be presented as a graph, scheme or spreadsheet. Text analysis provides qualitative results and text analytics quantitative results. [9]

## 2.1 Use cases of text analytics

In recent years, unstructured, textual format information has increased a lot around the world due to continuous growth of Information and Communication Technology (ICT) and social media platforms. Text data, if correctly identified and extracted, are useful in informed decision-making and can support wide range of business. Increased need to mine useful information from text data sources has led to developments in text mining techniques that can be seen as a continuation of classical data mining (knowledge discovery in databases) for traditional structured and unstructured non-textual data. Text mining is becoming increasingly important in industry in the coming years and will be used to gain a competitive advantage.

Several well-known companies have introduced text analytics to support and improve its core business. *Netflix* strongly uses text analytics techniques to analyze feedback and comments posted by its customers on various platforms, including social media networks. *The Bank of England* has introduced analytics techniques for a many potential applications at the Central Bank. In the example case, they look for tweets that contain phrases or terms that could indicate that the depositors are preparing to withdraw their money from Scottish financial institutions immediately following the Scottish Independence Referendum in September 2014. *IBM Watson Question-Answering (QA) system* responds to a natural language input by searching for or justifying an answer based on information gathered from various sources.[10]

## 2.2 Text analysis and classification tasks

Typical text analysis problems can be divided into the following tasks:

1. Sentiment analysis or other words opinion mining, identifies and examines the emotional states perceived in the text. In understanding texts, it is important to identify the author's feelings. Sentiment analysis allows you to rate the opinions of a new product from bad to good or to evaluate the reputation of a brand. It is also suitable for analyzing reviews, surveys and social media posts.

2. Topic analysis or topic modeling classifies texts by subject. It allows you to find books in the library, goods in the store, and find customer support tickets in CRM. Text classifiers can be customized to your needs.

3. Content tagging. Professionals in various fields who process huge amounts of unstructured data on a daily basis make it easier to do the work by marking and categorizing texts.

4. Meaning extraction. Text analysis makes it possible to extract keywords, prices, features and other important information from the text. For example. the seller can perform a competitor analysis and find out everything about prices and special offers easily.

5. Keyword extraction. It is technique that helps to identify keywords. Measuring words amount is useful for summarizing the content of texts, creating index data and creating word clouds.

6. Entity recognition. Entities are people, companies or places mentioned in the text. Such an ability is useful in machine translation so that the program does not translate last names or brand names. For example a Finnish food company, VALIO, that mainly processes and markets dairy products would be CHAMPION after the brand name translation. In addition, recognition of entities is essential for market and competitor analysis in business.[9]

## 2.3 Natural language processing, NLP

NLP was born in the 1950s as a combination of artificial intelligence and linguistics. Noam Chomsky's Syntactic Structures [11], in 1957, was groundbreaking. In it, he suggested that the structure of a sentence should be changed for a computer to understand the language. On this basis, Chomsky created a grammar style called Phase-Structure Grammar, which translated natural language sentences into a form suitable for computer processing. The overall goal was to create a computer capable of imitating the human brain, in terms of in thinking and communicating.

After Chomsky's results, John McCarthy published the programming language LISP (Locator / Identifier Separation Protocol), in 1958 [12]. It is still the computer language in use. In 1964, ELIZA [13], commenting and response machine designed to mimic psychiatry, was developed at the MIT Artificial Intelligence Laboratory by Joseph Weizenbaum [14]. He is considered one of the fathers of modern artificial intelligence . ELIZA answered questions by rearranging sentences and following relatively simple rules of grammar. Following these successes, however, in 1966, research into artificial intelligence and natural language processing (NLP) was considered a dead end.

In the late 1980s, NLP underwent a revolution.This development was made possible by both a steady increase in computing power and a shift to machine learning algorithms.Throughout the 1980s, IBM was a major player in the development of several successful and complex statistical models. The growth of NLP systems was even stronger with the 1990s and 2000s.

In 1997, "Long Shor-Term Memory" or LSTM for short [15], recurrent neural network models were introduced. Recurrent Neural Network (RNN) can be seen as parent of LSTM. The reason for use of RNNs in Natural Language Processing is their ability to take advantage of data sequencing. LSTM is needed in RNN as a memory unit that can remember word context from the beginning of the input.

RNNs and LSTM found their special area of expertise in speech and text processing in 2007.

In 2001, Yoshio Bengio and his team introduced the first “language model” that used a feed-forward neural network. In 2011, Apple Siri became one of the world’s first successful NLP/AI assistants to be used by ordinary general consumers. [16] These steps outlined above beginning from Turing test, have made it possible what can be achieved today using NLP methods.

## 2.4 Common NLP tasks

NLP tasks can be divided into several groups [17]. The tasks are in order according to their complexity.

1. Structured Prediction - Focused on low level syntactic aspects of a language and such as Parts-Of-Speech (POS) where POS ID tags identifies single words whether it is a verb or a noun and Named Entity Recognition (NER) tasks. NER is also known as (named) entity identification, entity chunking, and entity extraction, is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured text into pre-defined categories such as person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages. [18]

2. Sentence Classification: Refers to a range of tasks that require a model to categorise sentences into a series of linguistic buckets. There can be classification tasks where classified something as spam or classified reviews of a restaurant or product as being positive or negative or classifying articles into buckets of topics such as news or sport.

3. Sentence Retrieval: These tasks try to find semantically similar sentences. For example, are the phrases "I like pizza" and "I don’t like pizza" similar? Somewhat yes, but if you work in a restaurant and your chat bot direct a person who said that

"I don't like pizza", on the pizza selection menu, the customer isn't happy. Semantic Search or Semantic Text Similarity (STS) tasks work by having multiple sentences associated with a label. The label can be used to identify whether they are similar or not, or whether they are duplicates of each other.

4. Question Answering: These are more complex than similarity tasks because the questions are not structurally similar to the answers and must take into account the context. There may be simple questions like "what is the capital of Finland?". There is an unequivocal answer to this. When asked "what is the largest city in Finland?", You come across a number of options, such as, what do you mean by the largest, population or land?

5. Higher level tasks: To this group can be added multi-state dialogue systems, single domain chatbots, open domain chatbots. [17]

Tasks complexity increases from structured prediction to higher level tasks. When you select pretrained model you have to fine-tune it according on tasks. That's why is relevant to understand your dataset and select correct way to handle it. Once you have a model that is able to read and process text you have to find out how well it does perform current task and fine-tune its hyperparameters like batch size, learning rate and weight decay.

# 3 Approaches

Deep learning models are said to have surpassed the classical machine learning approach in different text classifications tasks. This is certainly true when there is a lot of data to process. This study shows as shown in Figure 3.1. that the deep learning method is only slightly better than the machine learn methods. It can therefore be said that deep learning methods can be used to continue where ML methods end. Both have their place in the field of text analysis.

## 3.1 Machine Learning and text analysis

Machine Learning is as roughly said the science of prediction. Given certain known features, you wish to predict some unknowns targets. The unknown can be structured like numeric or unstructured e.g. a video, audio or image. In text analysis, the classical machine learning based model consists mainly of two steps. First, some for example hand-crafted features have been extracted from the text and second step, these features are entered into the classifier to make a prediction.[6] Classification is the process by which all new data points are assigned to a specific category based on the mapping function. This function is derived from data set for which the category of each observation is known. There are three different types of classifiers; binary, multi-class and multi-label.

1. Binary: only two mutually - exclusive possible outcomes e.g. Cat or Not
2. Multi-class: many mutually - exclusive possible outcomes e.g. animal, veg-

etable, OR mineral

3. Multi-label: many overlapping possible outcomes - a single document may cover several subject areas, such as sports, economics and politics. [19]

When comparing the metric results given by several models with the nature of the data and the expected results, the best algorithm is most likely to be selected. In this study will be used the evaluation metrics for multi-class classification.The reason for this is that the phrases to be classified in the material, generally contain features that are suitable for one certain class.

## 3.2 Traditional Machine Learning classification algorithms

In this study, four traditional machine learning classifiers have been used in the classification. One of the most common algorithm is k Nearest Neighbor (kNN) [20]. It uses so called nearest neighbor decision rule. It gives the unclassified sample point the closest rating among the previously classified points. Similar values are grouped into k numbers of sets.

The Naive Bayes, proposed by Minsky (1961), is in machine learning a collection of classification algorithms, based on Bayes Theorem. It is named after the English mathematician and priest Thomas Bayes (1761). Bayesian inference has been known since the work of Bayes (1763) [21] and was first applied to text classification by Mosteller and Wallace (1964). Naive Bayes algorithm uses conditional probability to classify the sample to be of a certain class according to prior knowledge. It bases on an assumption that all the features are independent of each other. The name Naive comes from the nature of the Bayesian classifier, because it makes a simplistic (naive) assumption about the interaction of features. [22]

For example, a cat can be considered to be a cat because it has four legs and

a tail. According to this assumption Naive Bayes classifier will classify as a cat all figures that have four legs and a tail. That is why it is called 'Naive'. In NLP, Naive Bayes will count each word occurrence in the text. Word classifications are taught to the model, and when new text is submitted for classification, the algorithm classifies the text based on where it is likely to belong based on the previously learned word classification. The algorithm is used, for example, in an e-mail spam filter.

Support Vector Machine (SVM) was proposed in 1990's [23]. It's machine for two-group classification problems so tasks with more than two classes are not directly supported. Multi-class classification problem can be solved by using methods, One-vs-the-Rest or One-vs-one, to divide data into several binary classification data sets. [24][25]

In SVM, for very high dimension feature space will be constructed linear decision surface, hyper plane. The plane has been set between two sets of samples such that the distance (empty space without any sample) between sets is as large as possible. The hyperplane can be linear or nonlinear.SVM and linear SVM are basically similar, but to increase flexibility they are implemented differently. SVM multi-class mode is implemented using the One-vs-one model, while Linear SVM uses One-vs-the-Rest [26].

## 3.3 Deep Learning and text analysis

Deep learning is sub field of machine learning where data is learned hierarchically. First, the simplest components are emerged, followed by more complex components based on simpler ones. In general, this results in a simple layered hierarchy structure. The main difference between deep learning and traditional machine learning is its performance as the amount of data increases. When the amount of data is low, deep learning algorithms do not work well. This is because deep learning algorithms need a large amount of data to fully understand and learn it.

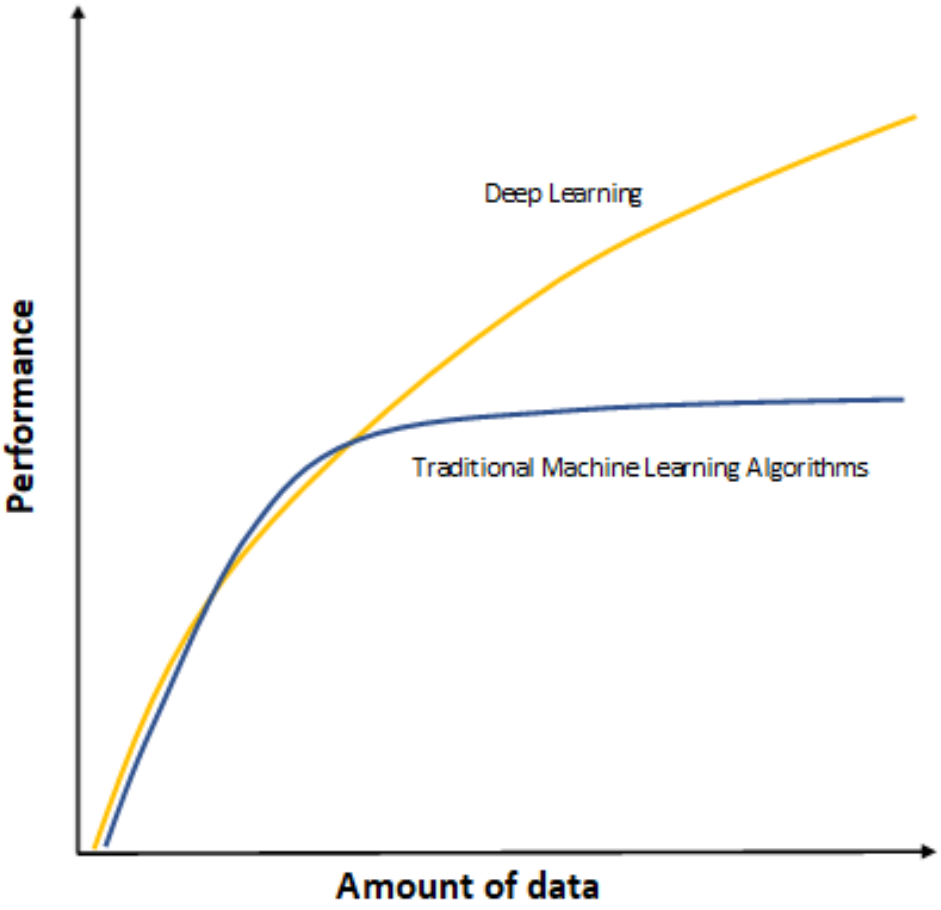


Figure 3.1: How do data science techniques scale with amount of data [27]

Figure 3.1. describes the importance of the amount of data when dealing with traditional machine learning methods and then the latest deep learning methods. The purpose of the figure is only to illustrate the effect of the data on the prediction performance, and thus does not describe any actual data or model. Second essential difference is feature engineering. Feature engineering is a process by which domain knowledge is transferred to features creation to reduce data complexity and increase the visibility of patterns for learning algorithms to work. This process is time consuming and expensive.[27]

Deep learning algorithms problem solving approach is to try to learn high-level features from data and to solve the problem. This is a very essential part of deep learning and a significant step ahead of traditional machine learning. The traditional machine learning algorithm solves the problem by first dividing it into parts and then solving the parts one by one and combining them to get the result. There is also a difference in the use of time, as the training of deep learning algorithms takes a long time. This is because the deep learning algorithm has so many parameters that it takes longer than usual to train them. Testing takes much less time from the deep learning algorithm than from the machine learning algorithm.[27]

There is a one thing that makes using deep learning algorithm particularly challenging compared to machine learning algorithm. That is interpretability. For example we use deep learning to give automatic scoring to essays. The results are good in that the performance given in the scoring is quite excellent and close to human performance, but it does not tell you why it gave that score. Mathematically, you can find out which nodes in the deep neural network were activated, but we don't know what neurons are needed for modeling and what these layers of neurons did together [27]. Deep learning models have become more popular than the classical machine learning approach e.g. when dealing with different text classifications tasks such as analyzing opinions, news grouping, answering questions, and reasoning natural language [6].

## 3.4 Transformer models

Network architecture, the Transformer, has been proposed in 2017 [28]. Transformer was the first reliable transduction model which was able to self-compute input and output representations using parallel encoder-decoder structure to learn contextual relations between words, without sequence-aligned RNNs or convolution [28]. Encoder reads the text input and generates a vector for each word and a decoder

produces the translated text from it. The model was initially created as a language translator and with it the concept of TLM (Translation language model) became established. Today transformer models have created a whole new dimension to natural language processing by developing the field by big leaps. New and better models appear almost monthly, bringing the performance benchmarks of forecast models to a new level in a wide range of tasks. [29]

The most significant reason why a lot of pre-trained models are used today, is the cost of running the algorithms on the hardware. Pre-trained models are models that can be applied with small modifications to your own analysis problem. In many natural language processing (NLP) tasks tasks such as sentence classification, sequence tagging and question answering, pre-trained transformer-based language models such as BERT and GPT have shown great improvement.Even larger and more accurate models such as GPT2 and Megatron are available. These more advanced models shows the growing popularity of large pre-trained Transformer models. Using these large models in production environments is a complex task that requires a lot of computing, memory, and power supplies. [30]

Pre-trained models are trained for very large corpora and result to a great number of parameters. The most advanced of all models require a lot of memory, computation, and wide bandwidth during the inference. In addition, the most advanced models must guarantee a high-quality customer experience with a low execution delay. These features mentioned above pose challenges for the wider adoption of models in the production environments. That is why it is crucial develop energy - efficient and minimum - cost methods to use these models in production. [30]

In this NLP study, I have used pre-trained BERT models, FinBERT and multilingual BERT and compared their results with each other, as well as the results obtained in traditional machine learning methods. FinBERT models, useful tools, and resources are available under open licenses at https://turkunlp.org/finbert.

I have used FinBERT version Release 1.0 (November 25,2019) and for multilingual BERT I have used version BERT-Base, Multilingual Cased (New) from site https://github.com/google-research/bert/blob/master/multilingual.md.

### 3.4.1 BERT

In 2018, BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers), created by Jacob Devlin and his colleagues at Google, arrived on the scene. BERT is pre-trained with large unlabelled raw textdata from Wikipedia containing up to 2,500 million words and with the Book Corpus comprising 800 million words. As the name suggests, the transformer architecture is the base of BERT. BERT differs from the first Transformer model when it uses masking. Model masks (drops) some words in Transformer's encoder side when training to learn the language model and then try to predict them so that it can use the whole context, both the left and right of the masked word. Masking is not present in the encoder side. [31]

It is a "deeply bidirectional" model. Bidirectional refers to BERT's ability to learn information from both the left and right sides of the word during the training phase. Thanks to that ability, the model is able to understand the real, meaning of words in its context. For example sentences "We went to river bank" and "I need to go to bank to make a deposit" gives for word "bank" different meaning due to its different context. [32]

If we now try to predict the nature of the word "bank" solely on the basis of the content of either the left or right sentence, our model incorrectly predicts at least one of these examples. To avoid such an error, both the left and right contexts must be considered before making a prediction. The greatness of BERT lies in its ease of application to all textual information. All you have to do is fine-tune the model by adding a couple of additional output layers to create top models for many NLP tasks. [32]

The BERT model available for multiple languages is a trained multilingual model (M-BERT) that combines data from top 104 languages. Although most of the pre-trained BERT models has focused on high - resource languages, in particular English has recently introduced multilingual models that can be fine - tuned to suit tasks in a variety of languages. The language-specific model has been presented systematically and clearly better than multilingual, but challengers can be found, like cross lingual transformer model XLM-R.

### 3.4.2 Finnish BERT model

Finnish BERT (FinBERT) is created by TurkuNLP Group. Group consists of researchers at the University of Turku and the UTU graduate school(UTUGS). Its main focus is on biological, biomedical and clinical texts and methods and resources for the analysis of the Finnish language. More detailed information about the group can be found at site turkunlp.org. FinBERT contains a comprehensive vocabulary of 50,000 words and it has been pre-trained for 1 million steps on over 3 billion tokens (24B characters) of Finnish text. Pretraining data has been collected from news, online discussions and an internet crawl. For comparison, multilingual BERT has been trained with Wikipedia texts, of which the Finnish Wikipedia text corresponds to approximately 3 % of all Finnish texts used in FinBERT's training. [33]

Traditional text analysis tasks such as POS tagging, where POS stands for words **p**art **o**f **s**peech and **n**amed **e**ntity **r**ecognition (NER) have been used to evaluate the performance of the models and the results have been compared with multilingual BERT. Third evaluated task has been dependency parsing. Researcher group presents based on tests new state-of-the art results that for lower-resourced language such as Finnish, is possible to create a language-specific BERT model, which can outperform multilingual BERT in tasks mentioned above in Finnish data.

### 3.4.3 Development of transformers

XLM-R is a new model presented Facebook AI, where letter 'R' stands for Roberta. Roberta has been introduced at Facebook and originally was written in format RoBERTa. It is a heavily optimized BERT trained with an improved training method with up to 1000 % more data and computing power. For pre-training, RoBERTa uses 160 GB of text:

- 16 GB of Books Corpus and English Wikipedia used in BERT
- 76 GB Common Crawl News dataset (63 million articles)
- 36 GB Web text corpus
- 31 GB Stories from Common Crawl.

This huge amount of the data combined with 1024 V100 Tesla GPU's running for a day, creates a pre-trained Roberta.[34] XLM stands for **cross**-lingual **l**anguage **m**odel and it approaches multiple languages. State-of-art results on cross-lingual classification, unsupervised and supervised machine translation, was presented 2019 by Guillaume Lample and Alexis Conneau in paper Cross-lingual Language Model Pretraining. [35]

XLM requires parallel examples, which causes problems as they can be difficult to obtain sufficiently, while XLM-R uses self-supervised technique. One of the biggest updates XLM-R offers as opposed to XML is the increase in the amount of training data. XLM-R has been trained in 100 languages, using 2.5TB of newly created clean Common Crawl data. It outperforms multi-lingual models like mBERT and XLM on many tasks like classification, sequence labeling and question answering. XLM-R is a challenger related to FinBERT because it has a huge 54.3GB of training data. [35] FinBERT has been trained using custom 50,000 wordpiece vocabulary [33].

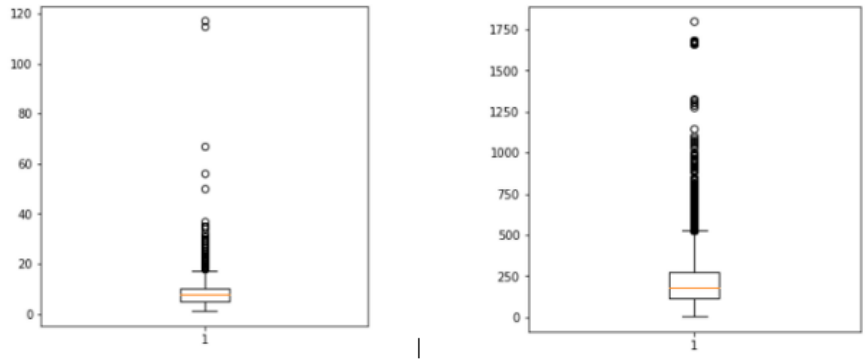


Figure 3.2: Boxplots about data's words length on left and sentence's length on right side

## 3.5 Performance metrics

When evaluating the performance of classifiers, it is essential to choose the right performance metric. Accuracy is one of the most commonly used metrics used to report the percentages of correctly classified test cases and is the primary measure of evaluating the performance of a classifier. It is very common for a classifier to work well with one performance meter, but poorly with others. For example, boosted trees and SVM classifiers achieve good performance in classification accuracy, while their mean square error produces poor performance. [36] Accuracy is the mix of precision and recall. Classification report provide these metrics. Precision equation is TP / TP+FP and Recall's equation is TP / TP+FN.

Boxplot is a method for graphically depicting groups of numerical data through their quartiles. Figure 3.2. the sizes of the text data words and sentences in the case study are presented in boxplots. Figure 3.3 illustrates the normal distribution of the boxplot. The green line reflect outliers. The detection of outliers in text data has become significant due to the need to find anomalies in countless text data sources.The text datas high feature dimensionality and document collections

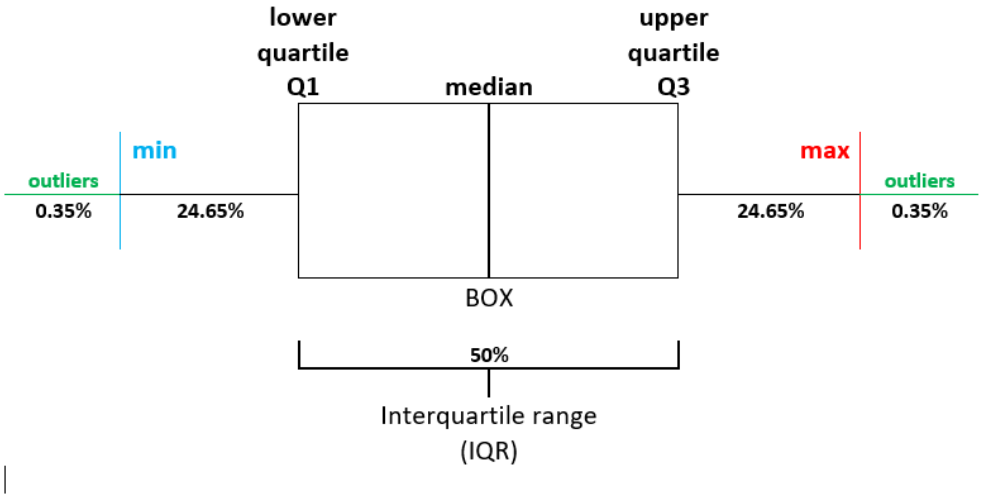


Figure 3.3: Boxplot and Normal Distribution [37]

large size, requires the development of accurate, highly efficient outlier identification methods. [38]

In this study, the treatment of outliers has been omitted because I did not see it as necessary for this textual data. Data preprocessing makes the words consistent and eliminates enough outliers so that they do not affect the outcome. However, when I used FinBERT model, I cannot help thinking how much the English words, dialects, the industry's own slang words in the sentences affect the accuracy of the prediction. Does the model treat them as outlier?

### 3.5.1 Classification report

Accuracy is the mix of two main indicators, called precision and recall. Classification report provide these two metrics, that gives a lot of information. Precision tells us the accuracy of the positive predictions. It's equation is TP / TP+FP. Precision is usually used along with another metric, recall. It's also called sensitivity or true positive rate (TPR). Recall's equation is TP / TP+FN. Metrics precision and

recall are often combined into a single metric called the F1-score. It's simple way to compare two classifier, because it's actually mean of precision and recall and equation is thus F1 = 2 * (precision * recall / precision+recall). It depends on your data which one metric you should prefer, because both you cannot have, when increasing precision reduces recall, and vice versa. In nutshell, precision tells what percent of your predictions were correct and recall tells what percent of all positive instance were correct predicted. A higher precision means less false positives, while a lower precision means more false positives. Higher recall means less false negatives, while lower recall means more false negatives. It depends on our task which value is preferred, high recall (less FN) or high precision (less FP).Precision: When it predicts yes, how often is it correct? To that question will answer precision. Table 3.1 shows the classification report of the study data when attempted to divide the data into three categories.

In text analysis, recall measures how many times the algorithm is able to identify a specific topic from a text and precision measures how many times the algorithm has classified a topic correctly, amongst all the times that topic has been mentioned in text. In binary classification, recall is called sensitivity. In Table 3.1 has been shown how in this study recall values are higher than precision values when algorithm classifies to class 100 but to other classes are precision values higher. This difference can be explained with support value. Class 100 has more values than classes 400 and 500 so data is some unbalanced. [39] Recall is more important in cases where False Negatives are more costly than False Positive. The focus in these problems is finding the positive cases. For example Covid-19 virus diagnose. It's more important to find all positive cases even some false cases would be included. On the otherhand if you let this value be too high due to aggressive false positives detection, it can lead to unnecessarily load for helth care. There has to find balance between these values and it is called the precision/recall tradeoff.

Table 3.1: Classification report

| Classifier | Class | Precision | Recall | F1-score | Support |
|---|---|---|---|---|---|
| kNN | | | | | |
| | 100 | 0.61 | 0.95 | 0.75 | 995 |
| | 400 | 0.75 | 0.44 | 0.55 | 693 |
| | 500 | 0.76 | 0.44 | 0.56 | 618 |
| Accuracy | 0.660017346 | | | | |
| Linear SVM | | | | | |
| | 100 | 0.79 | 0.88 | 0.83 | 995 |
| | 400 | 0.71 | 0.70 | 0.70 | 693 |
| | 500 | 0.70 | 0.57 | 0.63 | 618 |
| Accuracy | 0.744579358 | | | | |
| Naive Bayes | | | | | |
| | 100 | 0.77 | 0.91 | 0.84 | 995 |
| | 400 | 0.71 | 0.74 | 0.73 | 693 |
| | 500 | 0.79 | 0.53 | 0.64 | 618 |
| Accuracy | 0.757588899 | | | | |
| SVM | | | | | |
| | 100 | 0.80 | 0.89 | 0.84 | 995 |
| | 400 | 0.71 | 0.74 | 0.73 | 693 |
| | 500 | 0.74 | 0.56 | 0.64 | 618 |
| Accuracy | 0.758022550 | | | | |

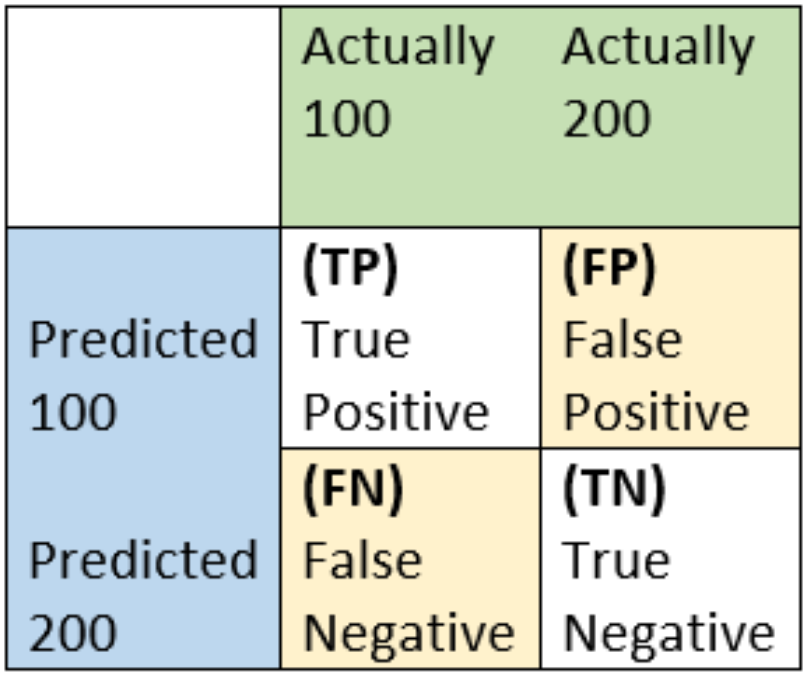


Figure 3.4: TP, FP, FN, TN

The metric called Support is the number of occurrence of the given class in your dataset. Based on that value we can see how the data is distributed among the different categories. For example, if we have 4000 occurrence of class 100 and 4000 of class 200, the dataset is really well balanced.

There are four ways, shown in Figure 3.4, to check if the predictions are right or wrong:

TP / True Positive: Model predicted positive and it's true.

FP / False Positive: Model predicted positive and it's false.

FN / False Negative: Model predicted negative and it's false.

TN / True Negative: Model predicted negative and it's true

If the dataset contains significantly more negative samples (a.k.a imbalance data set), precision as informative metric is better. That is due to Precision metric nature to focus more on the positive class than the negative class (Precision = TP /

```
SVM
[[885  61  49]
 [107 516  70]
 [121 150 347]]
              precision    recall  f1-score   support

         100       0.80      0.89      0.84       995
         400       0.71      0.74      0.73       693
         500       0.74      0.56      0.64       618

    accuracy                           0.76      2306
   macro avg       0.75      0.73      0.74      2306
weighted avg       0.76      0.76      0.75      2306

0.7580225498699046
```

```
SVM
[[1077  201]
 [ 173 1173]]
              precision    recall  f1-score   support

       100.0       0.86      0.84      0.85      1278
       200.0       0.85      0.87      0.86      1346

    accuracy                           0.86      2624
   macro avg       0.86      0.86      0.86      2624
weighted avg       0.86      0.86      0.86      2624

0.8574695121951219
```

Figure 3.5: SVM metric results for classifier to 2 and 3 classes

TP+FP). In Figure 3.5 is displayed how using Python library ELI5 ( show weights) can be visualized classifier weights. So abbreviation ELI5 doesn't stand for "explain like I'm 5", what is widely used in internet when people are asking others to explain some complex topic more simple way. On the other hand ELI5 provides nice and simply way explain classifiers predictions, so maybe they have some common. When comparing the metric results given by several models with the nature of the data and the expected results, the best algorithm is most likely to be selected. In this study will be used the evaluation metrics for multi-class classification.The reason for this is that the phrases to be classified in the material, generally contain features that are suitable for one certain class.

### 3.5.2 Machine Learning metrics

In Figure 3.5 the accuracy of the predictability of the model is presented. On the left is a confusion matrix where the material is classified into three categories 100, 200 and 300. In Appendix C, Figure C.1 illustrates how matrix values should be read. For example, Recall values (TP / TP+FN) are calculated as follows:

885/(885+61+49) = 0.889447236

516/(516+107+70) = 0.744588745

Table 3.2: Classifiers accuracy to 2 and 3 classes

| Classifier | Accuracy (3 class) | Accuracy (2 class) |
|---|---|---|
| KNN | 0.6600173460537727 | 0.826600609756097 |
| Linear SVM | 0.7445793581960104 | 0.8559451219512195 |
| Naive Bayes | 0.7575888985255854 | 0.8559451219512195 |
| SVM | 0.7580225498699046 | 0.8574695121951219 |

347/(347+121+150) = 0.561488673

The Figure 3.5 shows that the Precision value is clearly higher for class 500 than the Recall value. This indicates that the classifier does not recognize the correct member value in its own class, but rather classifies them into classes 100 or 400. A high Precision value indicates that the classifier recognizes fairly well the values in its own class. Class 100 has high Recall value 0.89 compared to other classes and it indicates that for the class 100, model is well able to target values that truly belong to the class. There are only a few FN values, but on the other hand a lot of FP values, which means that the model classifies many values from other classes into class 100. SVM Model's accuracy is 0.76. The right side of Figure 5.4 shows confusion matrix and metrics from a SVM model, that predicts values for only two classes. Its data set is well balanced when support values are almost equal (1278 vs. 1346) and also precision and recall values are quite equal. As a result, in this case we can only rely on the accuracy value and justify the choice of the most predictive classifier. For this data, it was also SVM, with an accuracy of 0.86. The accuracies of all classifiers are shown in the table 3.2.

### 3.5.3 Transformers and metrics

The deep learn model must take into account different values than in traditional machine learning, especially if the model is created from scratch by yourself. In the study has been used a pre-trained NLP models FinBERT and BERT, so there is much less adjustment of hyperparameters. I have used one Sampo Pyysalo's example task in examining how well FINbert and multilingual BERT are able to predict sentences from the material I am processing, into the right categories.

In the Pyysalo's example task, it was asked to limit the number of examples and the maximum sequence length to make training faster. In addition, it was introduced that it is necessary to check that the input length and batch size are not so large that the batch could exceed the GPU memory. The values that achieved the best accuracy within a reasonable time and using basic computer are shown below. For comparison, both transformers, FinBERT and multilingual BERT used the same values

1. Maximum number of examples to read
   - MAX EXAMPLES = 5000
2. Maximum length of input sequence in tokens
   - INPUT LENGTH = 130
3. Number of epochs to train for
   - EPOCHS = 6
4. Optimizer learning rate
   - LEARNING RATE = 0.00001
5. Training batch size

```
Train on 4500 samples, validate on 500 samples
Epoch 1/4
4500/4500 [==============================] - 343s 76ms/sample - loss: 0.4213 - sparse_categorical_accuracy: 0.8111 - val_loss: 0.3774 - val_sparse_categorical_accuracy: 0.8260
Epoch 2/4
4500/4500 [==============================] - 299s 66ms/sample - loss: 0.2802 - sparse_categorical_accuracy: 0.8791 - val_loss: 0.3915 - val_sparse_categorical_accuracy: 0.8140
Epoch 3/4
4500/4500 [==============================] - 300s 67ms/sample - loss: 0.2016 - sparse_categorical_accuracy: 0.9191 - val_loss: 0.3828 - val_sparse_categorical_accuracy: 0.8520
Epoch 4/4
4500/4500 [==============================] - 300s 67ms/sample - loss: 0.1470 - sparse_categorical_accuracy: 0.9471 - val_loss: 0.3986 - val_sparse_categorical_accuracy: 0.8540
```

Figure 3.6: Training and testing history per epoch

– BATCH SIZE = 8

In Figure 3.6 has been presented the returned "history" object that holds a record of the loss values and metric values during training. Dataset is reset at the end of each epoch, so it can be reused in the next epoch. History tells us the running time for each epoch, loss and accuracy values. In this example has been used Keras performance metric Sparse Categorical Accuracy to evaluate the accuracy. It calculates how often predictions matches integer labels. This metric can be used when classes are mutually exclusive e.g. when each sample belongs exactly to one class. Loss is the measurement result of the error made by the model, while accuracy is the measurement result of the performance of the model. If training loss (sparce categorical accuracy) is decreasing and validation one is increasing then it's likely you've overfitted the model.

Overfitting occurs when the model learns the training information well, but it does not become very common to new data. In other words, the model learns well trained, seen data, but does not know how to deal with to new, unseen data. Underfitting is the opposite of overfitting. That is, when a model has too little capacity to learn the problem, while a model with too much capacity can learn it too well and overfits the training dataset. In both cases, a model has been created that will not generalize well.

## 3.6 Hyperparameters

Hyperparameters are parameters used to control and tune the behavior of an algorithm when building a model. These parameters cannot be learned from the normal training process. They must be determined before training the model. There are a variety of automated optimization or tuning techniques and each has its own strengths and weaknesses when applied to different types of problems. Hyperparameter optimization or tuning in machine learning is the process of selecting the best combination of hyper-parameters that deliver the best performance. There are libraries for Keras, such as Talos, Kept, Hyperas and different hyperparameter search methods like Grid Search, Random Search, Bayesian optimization and many more. [40]

Deep learning neural networks are trained using the stochastic gradient descent optimization algorithm. It is an "iterative method for optimizing an objective function with suitable smoothness properties". More detailed explanation of how the algorithm works will be found from Wikipedia [41]. The learning rate is one of the most important and easiest to tune hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Choosing the right learning rate value is challenging because too low a value can lead to an unnecessarily long training process and the model can get stuck, while too high a value can lead to learning too optimal weights too fast and thus lead to sub-optimal result. [42] This work does not discuss these hyperparameters in more detail, but only adjusts the learning rate value.

## 3.7 CRISP-DM Methodology

CRISP-DM (Cross-Industry Standard Process for Data Mining) is about two decades old. According to many studies and user surveys, it remains the de facto standard

in the development data mining and knowledge discovery projects. In any case, 20 years is a long time in the IT industry and a lot has happened during that time. It has been clarified that when a project is goal-oriented and process-oriented, the process model view is still largely valid. If projects become more exploratory, project paths will vary and a more flexible model will be needed. [43]

The CRISP-DM process or the CRISP-DM method can be described in six main steps:

1. Business Understanding. The aim is to understand the goals and requirements of the project from a business perspective. Understanding is refined into a data mining problem and a preliminary plan is developed.

2. Data Understanding. From the beginning of the data collection, the data is continuously examined to identify quality problems and to find the first views of the data

3. Data Preparation. The data preparation phase includes all activities to build the final data set from the original raw data.

4. Modeling. Evaluate, select and apply appropriate modeling techniques. Some technologies, such as neural networks, have specific requirements for the format of the data, it is possible to go back to data preparation.

5. Evaluation. Select models that appear to be of high quality based on the selected loss operations. Ensure that the model is able to generalize sufficiently unknown data and ensure that the model is able to generalize sufficiently unknown data. Confirm that the model is comprehensive enough regarding key business issues. The end result is a selected best model or models.

6. Deployment. Implementation of the model code representation in the operating system.


Figure 3.7: CRISP-DM steps. Figure from [44], licensed CC BY-SA 3.0

# 4 Case study description and the data

## 4.1 Starting point

In January 2020, I was asked by a large metal company on a Finnish scale to analyze the deviations recorded in the production control system (MES) using data analysis methods. The aim was to find out what information was written into the system and whether it was classified correctly. It was assumed that the user does not always record the observed deviation in the correct category. In addition, it was wanted to find out whether it is possible to create a model based on data, which could predict what is the root cause of a particular deviation and could be used to predict future actions. Deviations automatically recorded by the system were excluded from the data and only human records were included.

## 4.2 Planning and analysis methods

Initially, a lot of time was spent getting to know the data and getting to know the topic. I had discussions with the client and data related people to understand the production process and at what stage what values are recorded in the system and by whom.

First, I tried to find out by clustering whether there are uniform data sets in the

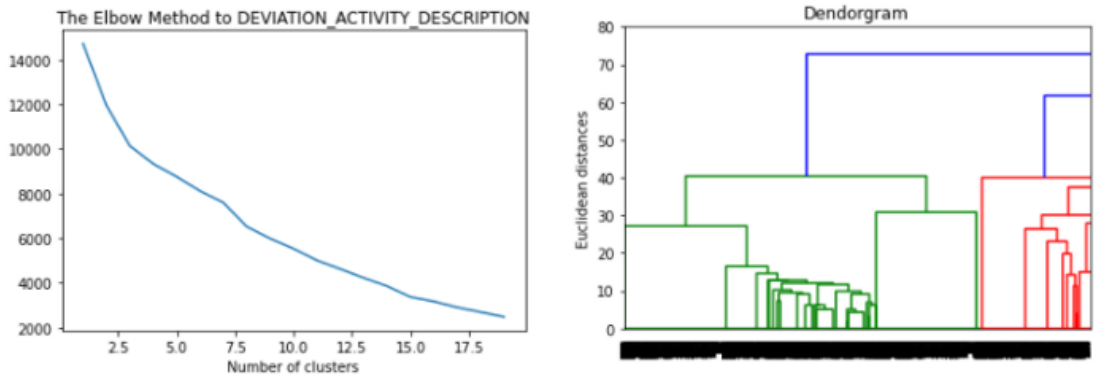


Figure 4.1: The Elbow Method and Dendorgram

data. I figured out the number of possible clusters using The Elbow Method and also Dendorgram, Figure 4.1. According to them the value 3 as number of cluster was promising. Clustering results can be seen in Figure 4.2. and they were not good. No clear clusters were visible, so I stopped unsupervised learning for this and moved to take advantage of the original classification of the data.

In the MES system, the user has a choice of 62 different categories for the classification of the deviation in the first step when the deviation is detected. It's impossible to categorize into so many categories, so I combined the categories drastically. After various experiments, I limited the number to just two categories. It allows me to judge that I get the best model that can then be modified later.

One of the most essential steps in any machine learning workflow is exploratory data analysis. There are many different tools available and you should be able to choose the most suitable and efficient ones for your case. I have selected for this case the most common and most used methods. I used NLP methods to study the data and create a prediction model by both machine learning and deep learning methods. Deep learning model is based on a pretrained BERT model, FinBERT.

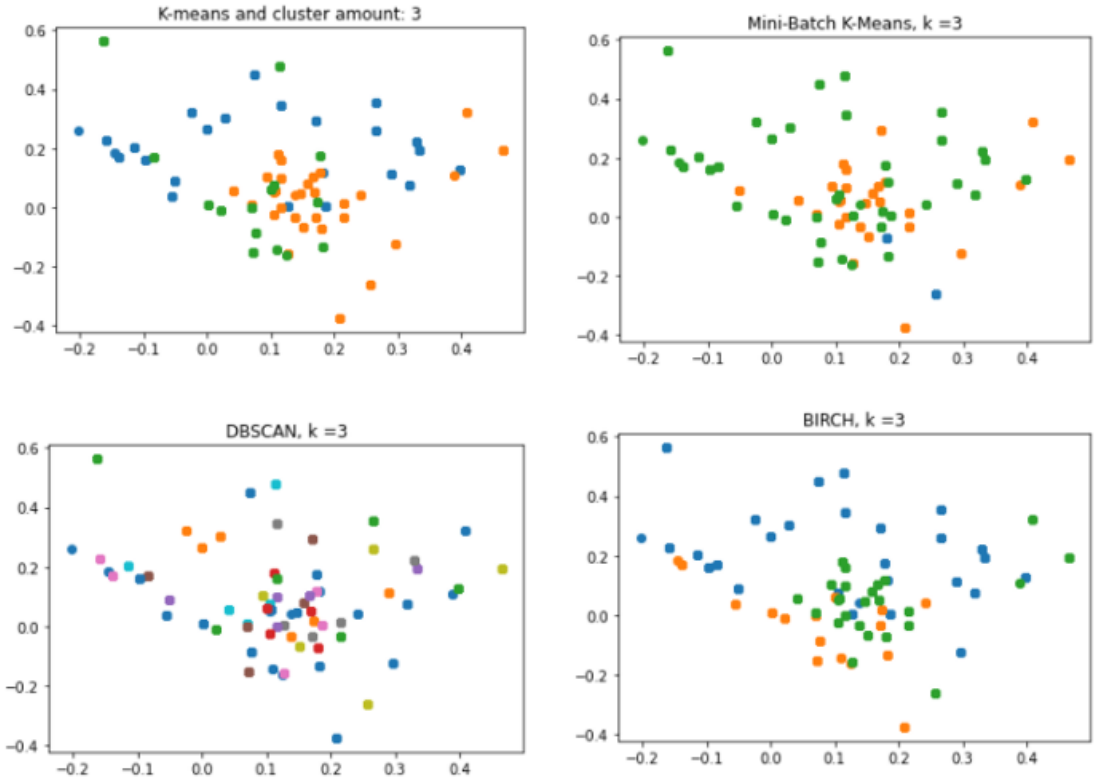


Figure 4.2: Different clustering methods

Table 4.1: Deviations in 2017

| Month | Amount | Month | Amount |
|---|---|---|---|
| January | 40 | July | 59 |
| February | 116 | August | 304 |
| March | 314 | September | 675 |
| April | 70 | October | 730 |
| May | 53 | November | 596 |
| June | 129 | December | 502 |
| | | Total | 3588 |

## 4.3 Data introduction

The data in an Excel spreadsheet was from 2016/11 to 2018/09. For example, in 2017, deviation amounts saved in the system are shown in Table 4.1. The whole data includes 8719 rows. Markings MES (Manufacturing execution system) system is made by 60 different people all the time.

At first, I went through all the text written by the user from the material. The text was collected from five different column. To the COMMENT TEXT field written by installer is filled in the overall description of deviation, what is an obvious fault. The rest of fields are filled by quality controller. DEVIATION ACTIVITY DESCRIPTION, describes where at the work stage, the deviation has occurred. The COMMENT STARTED field tells you how to react to the deviation, the first steps and the COMMENT RESOLVED field tell you the solution. In the last field COMMENT CLOSED, the findings and records of the deviation are summarized and any actions are also reported that the deviation would not happen again. The following is an example of a single line: "Roiskesuoja PAAE022875 puuttuu keräyksestä Kanban vaiheella 2 näytetty asentajalle Roiskesuoja ei tule keräilyssä vaan on

kanbannina linjalla. Löytyy paikoista 1-7 B-Alas, 2-2 B-alas, 2-5 B-alas ja 2-7 B-alas. Haetaan tarvittaessa 2-vaiheen paikasta eikä lisätä uusia paikkoja 3-vaiheelle, kun kulutus on niin vähäistä."

In system there are 62 DEVIATION REASON CODE ID classes and 192 ROOT-CAUSE REASON CODE ID classes, so the user has several options to choose the category for his deviation. Misclassification certainly happens. The data is unevenly distributed and a third of the values were in one deviation reason category named "KOMPONENTTIPUUTE". The most likely rootcause class was "SALDOMATERIAALI".

As an usage example, I created only three categories into which I grouped the existing categories. In the first class (4322 rows), the main theme is the missing part, for one reason or another. In the second class (2451 rows) , the causes of the deviation are related to the error, e.g., collection error (wrong amount) drawing error, scratch, or other. The third class (2186 rows) includes the rest. Total of rows is 8958 and without null rows 7685. The data set is imbalanced and therefore classification problem is imbalanced. Small imbalances are often not a concern, and the problem can often be treated as a common problem for predictive classification modeling. Big imbalances in classes can be challenging to model and may require the use of specialized techniques. [45]

## 4.4 Simple tactics to combat uneven training data

1. Collect more data. Larger material may reveal a different perspective on the classes. 2. Try different performance metrics. Accuracy is not suitable metric when dealing with imbalanced data set. More reliable metrics are Confusion Matrix, Precision, Recall and F-score. 3. Resampling data set. Copies of instances of the underrepresented class can be added and vice versa, removing examples from an overrepresented class is called under-sampling. Under-sampling is useful method

when we have a lot of data (tens- or hundreds of thousands of instances or more) and over-sampling when we have a lot of data (tens of thousands of records or less). 4. Different algorithms. It is not always advisable to use the same familiar algorithm for all problems, instead trying to find the most suitable one for the problem at hand. For example, using decision trees can be achieved good results. [46]

## 4.5 Visualization of textual data

One of the main strategies used in exploratory text data analysis is data visualization. Data visualization has become more common and the importance of visualization has increased as data masses grow. It is a quick and easy way to illustrate data material and allows you to easily identify trends, find anomalies, make and present interesting summaries. Common examples of data visualization are charts include different types of pie charts, histograms, word clouds, bar charts, tree maps, area charts, point distribution maps, timelines, time series, flow charts and dendograms.

Data visualization can be divided to four part; quantities -, sense -, context - and trend visualization. Quantities Visualization (QViz) means visual analysis that handles counts of textual characters, features, or sequences. Common examples this kind of analysis are wordclouds (Figure 4.3), count frequencies, tables, and pie charts. Sense Visualization (SViz) is visual analysis type that provides sense to textual characters, features or sequences. Its common examples are Sentiment analysis, Semantics and Natural Language Processing.

Context Visualization (CViz) investigates characters or sequences context in text. It can for example explore "Who, When, and Where" in a social media posts. Trend Visualization (TViz) provides analysis that apply a timeline, process, or temporality to textual characters, features or sequences. It can be done through time series, process, or evolution. One visualization tool for evolution of relationships is gestalt-matrix. Today, an increasing proportion of data visualization is largely tool-based,


Figure 4.3: Wordcloud from study case

and therefore it is essential to understand the data visualization tools environment. [47]

## 4.6 Text analysis workflow

I started doing NLP text analysis as a beginner, so I searched the web a lot for tips on how to do it. One of the clearest guidelines for the text analysis process was found by Gunjit Bedi. [48]

The steps of text analysis can be divided into 8 different steps:

1. Activate the required libraries
2. Set the random seed
3. Insert corpus
4. Data pre-processing and use of natural language processing techniques (NLP)

- delete null values, ie blank lines

- change uppercase to lowercase

- edit words to basic body format only

- remove unnecessary words, numbers, special characters

- in the English text search for the basic form of words (lemmatization)

5. Divide the data into training and test sets

6. Rename labels, Y values if needed(encoding)

7. Convert words into vectors

8. Select machine learning algorithm

Human written texts usually contain lots of noise and uninformative parts such as punctuation, special characters, numbers and uppercase and lowercase letters. In addition, on words level, many words in the text do not have an impact on the general meaning of text. Words like subjugation and prepositions can be simply removed. Reducing noise from text helps improve classifier performance and speed up the classification process. [49]

Text pre-processing is the process of cleaning up text and preparing it for classification. In general, it can be assumed that the text preprocessing creates a bag (multiset) of index terms that do not themselves have an internal structure. This presentation is also sometimes called a bag of words model. It does not matter in practice whether the search terms are words, but instead characters n-grams, word stems, word n-grams or any of the corresponding text representations. When each word in the text is treated as as one dimension, keeping those words makes the dimensionality of the problem higher and hence the classification more difficult.

I also explored the possibility of using Voikko, free linquistic software and data for Finnish. It is a collection of tools and vocabularies for language parsing, text checking and hyphenation. The core of Voikko is a library libvoikko. Voikko has been developed primarily for the needs of the Finnish language and it can be loaded here https://voikko.puimula.org/. However, installing and using the program turned out to be too complicated and I gave it up. For this study, the use of ordinary text pre-

processing libraries, provided by NLTK (Natural Language Toolkit), was sufficient.

After clearing the text, a feature can be selected. It can also be called filtering. In this study, on part traditional machine learning model creation, a feature is selected when words are converted to vectors. The most popular ones are Feature Frequency (FF), Term Frequency Inverse Document Frequency (TF-IDF) and feature presence (FP). FF is the number of occurrences in the document. In equation, N indicates the number of document and DF is the number of documents that contains this feature. In equation format feature selection TF-IDF can be said as follows:

$$\text{TF-IDF} = \text{FF} * \text{Log}\ (\text{N}/\text{DF})$$

In Figure 4.4. has been shown the words TF-IDF weights when data was divided into three classes. To illustrate the weights have been used scikit-learn, free software machine learning library for the Python programming language [50] [51].

In this study case, machine learning, text classification algorithms such as SVM, linear SVM, and Naive Bayes are used. K-Nearest neighbors algorithm is also used for comparison. Selected algorithm's advantages and disadvantages are shown in image 4.3. Several variations of the naive Bayes models have been used and have long produced the best results, mainly at the turn of the 21st century. Since then, comparisons have shown that the use of Naive Bayesian learning methods in text classification has been somewhat less favorable, yet still achieving respectable efficiency. This may be because more classification data is available in the text classification and the datasets favor algorithms that produce more complex classifiers. [52]

A simple and powerful classifier for common phrases is to represent sentences as a bag words (BoW) and train a linear classifier, e.g. a logistic regression or SVM. [23] In this study has been used both basic SVM and linear SVM. The basic model of SVMs was presented by Cortes and Vapnik in 1995. SVM and linear SVM are

| y=100 top features | | y=400 top features | | y=500 top features | |
|---|---|---|---|---|---|
| Weight[?] | Feature | Weight[?] | Feature | Weight[?] | Feature |
| +1.620 | 'riit', | +2.100 | 'runkolaaker'] | +2.249 | 'kurkistelem', |
| +1.418 | 'kierretap'] | +1.949 | 'pääty', | +2.091 | 'running', |
| +1.377 | 'etukät', | +1.688 | 'hyvåksyty'] | +1.960 | 'poistetu', |
| +1.374 | 'sourc', | +1.679 | 'kiinnnit', | +1.942 | 'ccm', |
| +1.372 | 'sami', | +1.627 | 'havaittav', | +1.864 | 'powerpack'] |
| +1.356 | 'kump', | +1.620 | 'kaasusuodatin', | +1.844 | 'aiheuttav', |
| +1.341 | 'ilmanpoistoputkisto', | +1.589 | 'kaapel'] | +1.711 | 'riviliitin', |
| +1.316 | 'nm'] | +1.579 | 'pedro', | +1.689 | 'kalibroitav', |
| +1.302 | 'saata', | +1.530 | 'mainitu'] | +1.549 | 'roisk', |
| *... 1482 more positive ...* | | +1.503 | 'kannennumero', | +1.532 | 'reuna'] |
| *... 3000 more negative ...* | | +1.493 | 'jakotu', | +1.490 | 'piirnum', |
| -1.283 | 'stock'] | +1.470 | 'ses'] | +1.486 | 'anchor', |
| -1.294 | 'maadoituskaapel', | +1.461 | 'jokaker', | +1.482 | 'riviliittim'] |
| -1.296 | 'imuputk', | +1.461 | 'ponnistelu', | +1.459 | 'pieni', |
| -1.316 | 'pik', | +1.453 | 'etsin', | +1.447 | 'pien', |
| -1.318 | 'm', | +1.452 | 'siir'] | +1.442 | 'asennustel', |
| -1.356 | 'runkolaaker'] | +1.441 | 'läpivientitulp', | +1.437 | 'pituus'] |
| -1.384 | 'kiinnnit', | +1.438 | 'kaasuvent', | +1.423 | 'kiva', |
| -1.390 | 'kalibroitav', | +1.417 | 'kuparirengas', | +1.412 | 'lämpötil', |
| -1.450 | 'it'] | +1.389 | 'lähet', | +1.388 | 'kupariprikо', |
| -1.450 | 'ilmanpoisto', | +1.382 | 'conventer', | +1.370 | 'rellatu', |
| -1.483 | 'ohjauslohko', | +1.375 | 'ongelm', | +1.364 | 'kuoku'] |
| -1.491 | 'pituut'] | *... 2309 more positive ...* | | +1.352 | 'pumpu', |
| -1.617 | 'aiheuttav', | *... 2443 more negative ...* | | *... 1972 more positive ...* | |
| -1.753 | 'ruuve', | -1.403 | 'powerpack'] | *... 2753 more negative ...* | |
| -1.803 | 'kurkistelem', | -1.406 | 'kuoku'] | -1.437 | 'riit', |
| -2.163 | 'kapa'] | -1.438 | 'koloruuv'] | -1.774 | 'kaasusuodatin', |

Figure 4.4: Features weights per class

Table 4.2: Comparison of different classifiers

| Classifier | Advantage | Disadvantage |
|---|---|---|
| KNN | Simple | Requires time and memory when data is big |
| Decision Tree | Fast and easy | Overfitting of data |
| Naive Bayesian | Easy to implement and computationally cheap | Low classification performance for large data set |
| SVM | High accuracy and no overfitting | Problems in representing document into numerical vector |

conceptually similar, but implemented in another way to add flexibility. Support Vector Machines were designed for binary classification and do not natively support classification tasks with more than two classes, thus in linear SVM the multiclass support is handled according to a one-vs-rest scheme and SVM handles multiclass support according to one-vs-one. [53]

## 4.7 Deep learning approach and FinBERT

Deep learning part of study is implemented using pretrained BERT model and especially a version of Google's BERT deep transfer learning model for Finnish. The model named FinBERT is BERT model that has been trained from scratch on Finnish and can be fine-tuned to achieve state-of-the-art results for various Finnish natural language processing tasks. FinBERT features a custom 50,000 wordpiece vocabulary that has much better coverage of Finnish words than e.g. multilingual BERT models from Google. [54]

## 4.8 Implementation and challenges

When I started working, I was excited because my assumption was that the work could be done easily. Artificial intelligence and machine learning, that's it. I was wrong. At first, I did text analysis only by clustering data and based on that I created a forecast model. My best clustering model had about 50 percent accuracy. I was disappointed and unaware of how to get ahead of this. My studies had not dealt with text analysis, so I was faced with a whole new thing.

I had to start from the very basics. I received good material from the thesis supervisor Sampo Pyysalo to get acquainted with text analysis. Gradually, the work steps became clearer and I was able to move forward. The text analysis as a field is extensive, so there is a danger of easily getting too deep into some details and the work will never be completed. You had to be able to find the most relevant things and methods for the job.

# 5 Results

The thesis investigated how text analysis is suitable for the needs of the metal industry to determine the classification of deviations recorded in production. The main research problem was to find out whether the deviations were correctly classified into the current categories. The assumption was that the classification made by the employee does not always go correctly. In addition, the purpose was to examine comments at a general level and to investigate whether it is possible to create a prediction model for handwritten deviations.

The Finnish word "puutu" is by far the most common word in the sentences. It is well seen in wordcloud in Figure 4.3. However, missing itself can be related to many different things. In other words, part is missing, collection is missing, work instructions are missing, image is missing, etc. For this reason, when creating the traditional machine learning model, is good to utilize for example the n-gram model, which slides the data into selectable word groups. I used value two as n-gram value because the word "puutu" i.e "missing" appears in so many different places and meanings and usually for a couple of missing reasons such as missing instruction (ohje puuttuu), missing part (osa puuttuu), missing from shelf (puuttuu hyllystä), missing from collection (puuttuu keräyksestä). The lengths of the words are mostly around 10 characters and the lengths of the sentences are mainly 100-250 characters. Those values are presented via boxplots, in Figure 3.2. The classification report (Table 3.1) shows how the metrics of this study behave when there are three

Table 5.1: Algorithms accuracy

| Traditional machine learning | |
|---|---|
| KNN | 0.83 |
| Linear SVM | 0.86 |
| Naive Bayes | 0.86 |
| SVM | 0.86 |
| Deep learning transformers | |
| FinBERT | 0.87 |
| Multilingual BERT | 0.86 |

categories and they are unbalanced. To get the classes the same size, the number of classes had to be reduced to two.

I supplemented the case study by making a model also utilizing deep learning transformers FinBERT and Multilingual BERT. Surprisingly, the accuracy provided by the transformers was of the same level as that of traditional machine learning models. The similarity in accuracy between FinBERT and multilingual BERT can also be considered a small surprise. The results are shown in Table 5.1.

Table 5.2: Traditional machine learning algorithms

| Accuracy of Traditional machine learning | |
|---|---|
| KNN | 0.8266006097560976 |
| Linear SVM | 0.8559451219512195 |
| Naive Bayes | 0.8559451219512195 |
| SVM | 0.8574695121951219 |

## 5.1 Traditional machine learning methods

By experimenting with different types of hyperparameter combinations, I came to the conclusion that when sentences are classified into only two categories, the best predictive accuracy is achieved. The first category named 100 has findings that some deficiency or missing part has caused the deviation, and the second category named 200 all others. A more detailed breakdown of the original categories (62) into these two classes is provided in Appendix B. By teaching the model to classify sentences into the two categories presented above, one of the best results is shown in Table 6.2. These values are good. Slightly best accuracy is achieved using SVM, 85.7 % compared to Linear SVM and Naive Bayes. K-Nearest Neighbors provides quite clearly the worst accuracy of these.

## 5.2 Transformers FinBert and M-BERT performance

It can be seen in Figure 5.1 that deep learning model starts to overfit already after first epoch when val sparse categorical accuracy is bigger than sparse categorical accuracy. After that point, it no longer makes sense to continue training the model, but it can be said that it is best to stop on it. Val loss is the value of the cost function of validation data and the loss is the value of the cost function of the training data.

Validation data is used to investigate if model overfits the data or not. On that second epoch has achieved also the best accuracy of model, 82.6 % and it's good. It shows that the model has the ability to predict the correct class for a sentence. At the same time loss is also quite high and that is not good thing. It might be caused by some outliers which distort the result or otherwise the model overfits. I will only state this observation here, and I will not go into it in more detail.

In the following tables 5.3. and 5.4. are examples of how FinBERT and Multilingual BERT classify test data. The table shows the top 10, worst 10, and middle 10 rating values. That is, the best values are in 100 class with 100 % probability according to the classifier and the worst with 100 % probability belongs not to 100 class. Based on the examples, it can be said that the classification of both transformers is very similar.

The results given by both methods, traditional and pretrained transformer models, are very similar. From this it can be deduced and as in Figure 3.1. appears, the deep learning method is maybe not yet able to demonstrate its ability with this amount of data. By adding data, it can make the result better.

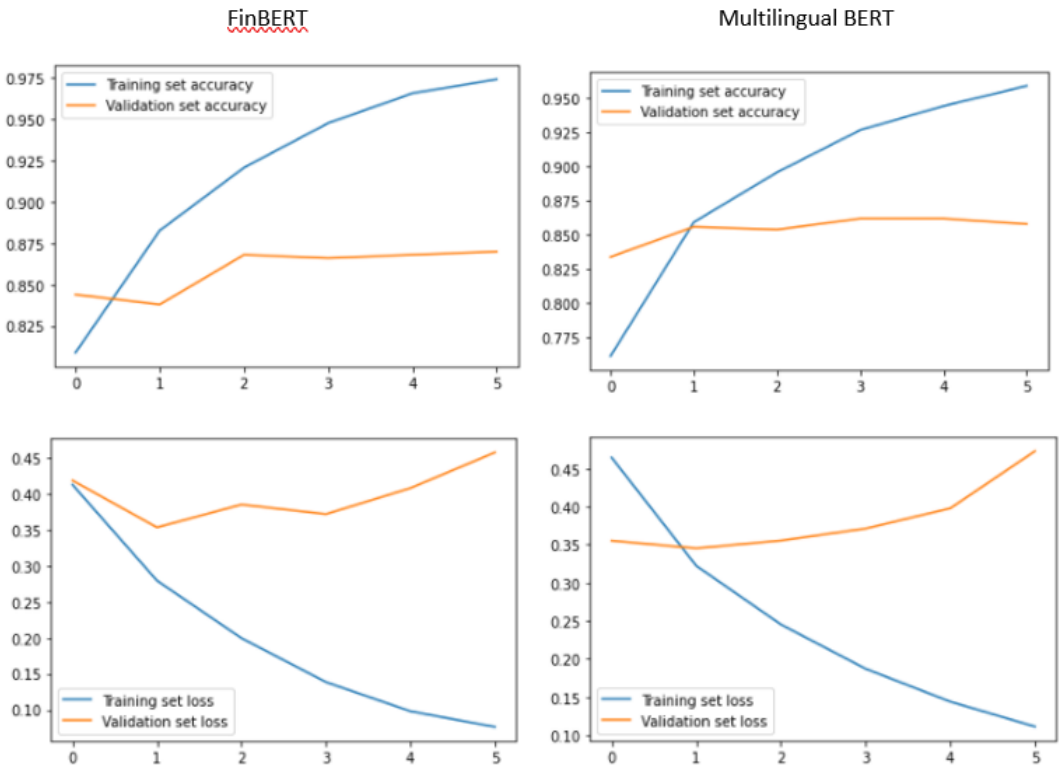


Figure 5.1: Accuracy and loss using DL method

Table 5.3: Results using FinBERT

| Class 100 | Percentage | Test Comment Sentence |
|---|---|---|
| 1 | 100.0 | KP |
| 2 | 100.0 | puutteessa . hylly tyhjä. |
| 3 | 100.0 | kannen vaarnat loppu b-puoli |
| 4 | 99.9 | 4kpl pumppuja puute |
| 5 | 99.9 | Ahtimet puuttuu,laitettu hylly ilman ahtimia. |
| 6 | 99.9 | xx INTERMEDIATA GEAR WHEEL Puuttuu 2kpl |
| 7 | 99.9 | xx fitted plate- koko lava puuttuu hyllystä |
| 8 | 99.9 | puuttuu keräyksestä 7kpl a-puoli |
| 9 | 99.9 | Pipe clamp xx puuttuu 20kpl. |
| 10 | 99.9 | Vauhtipyörä puuttuu siiretty keskeneräisenä nelosvaiheelle |
| 447 | 49.2 | keräyksestä puuttuu aluslaatta paac002088 4kpl puute on toistuva !! |
| 448 | 49.1 | PAAF140551 puuttuu B-puolelta 15kpl myös a-puolelta 24kpl. Ja A-puoleta puuttuu xx distance sleeve 24kpl. |
| 449 | 49.0 | Turbo puuttuu, ei voida hepata. |
| 450 | 48.6 | DE-boksilta moottorin B-puolelle tuleva kaapeli CV644 puuttuu kokonaan. Pitää itse lisätä. |
| 460 | 46.6 | Anturin tasku PAAF056508 puuttuu |
| 461 | 45.9 | Puuttuu suora jatkoliitin xx Löytyy kuvasta xx osa :17 |
| 991 | 45.6 | pesä xx puuttuu |
| 992 | 45.5 | A-puolen keräyksessä vanhan revision putki. (VÄÄRÄ) |
| 993 | 45.5 | ei voi vielä asentaa |
| 994 | 45.4 | ei ole tullut linjalle |
| 991 | 0.0 | reiät liian lähellä reunaa |
| 992 | 0.0 | virheellinen öljy reijän sijainti. |
| 993 | 0.0 | uii-luku ei onnistu akselista |
| 994 | 0.0 | kampiakselin uii koodi lappu puuttuu |
| 995 | 0.0 | Kasattu väärin |
| 996 | 0.0 | Johto lähtee väärään suuntaan |
| 997 | 0.0 | Ei voi suorittaa ennen turboa (Turbomoduli puutuu) |
| 998 | 0.0 | ei anna lukia lohkoa |
| 999 | 0.0 | sokanreikä liian pieni. |
| 1000 | 0.0 | poikkeama tehty jotta pääsee eteenpäin |

Table 5.4: Results using multilingual BERT

| Class 100 | Percentage | Test Comment Sentence |
|---|---|---|
| 1 | 100.0 | KP |
| 2 | 100.0 | Banduitit xx Puuttuvat hyllystä —> haettu 2.6 linjalta lisää. |
| | | Ilmoitettava alihankkijalle että tuo lisää ennekuin loppuu sieltäkin. |
| 3 | 99.9 | Putket loppu vaiheen 2-2 hyllypaikasta. xx, osa nro 1. |
| 4 | 99.9 | Vesiputket loppu hyllystä. |
| 5 | 99.9 | Ahdinmoduuli ei ole valmis |
| 6 | 99.9 | loppu |
| 7 | 99.9 | Piirnro xx osa nro4 pipe clamp. Clamppi on kaksi osainen, nyt |
| | | hyllyssä vain alaosia. Haettu 2-linjalta itse. |
| 8 | 99.9 | Tekemättä Vauhtipyörän puuttumisen johdosta |
| 9 | 99.9 | xx BRACKET OVERS.TRIP DEV.B-BANK FREE END puuttuu |
| | | vaiheelta 1/3 b-ylä |
| 10 | 99.9 | Tarvitaan 1kpl. LT-WATER PIPE mat. xx |
| 447 | 55.5 | Öljymodulista puutuu putki xx |
| 448 | 54.4 | Flywheel Cover Material ID xx oli toimitettu linjalle |
| 449 | 53.5 | xx 4kpl |
| 450 | 53.5 | puuttuu keräyksestä xx cover profile |
| 460 | 52.5 | sentry kaapelit puuttui keräyksestä |
| 461 | 48.7 | 1 kpl kateis |
| 449 | 47.9 | Tarkastaja |
| 450 | 46.6 | puuttuu xx 1kpl |
| 460 | 46.2 | xx CONNECTING PIECE FOR SLIDE-IN PIPES TO + |
| | | jäänyt taas keräämättä |
| 461 | 46.1 | puuttuu keräyksestä kyseinen putkentynkä |
| 991 | 0.0 | Jalustan kahdessa reiässä lastuja kiinni |
| 992 | 0.0 | QDMS ei toimi |
| 993 | 0.0 | reiät liian lähellä reunaa |
| 994 | 0.0 | vaiheensiirtopoikkeama |
| 995 | 0.0 | siirretty keskeneräisenä |
| 996 | 0.0 | sokanreikä liian pieni |
| 997 | 0.0 | ei anna lukia lohkoa |
| 998 | 0.0 | tehdään myöhemmin |
| 999 | 0.0 | 3 vaiheelle |
| 1000 | 0.0 | vaarnoja ei saa qdms |

# 6 Discussion and Future

An earlier (2016) thesis for a company, have been examined the frequency of detected errors in a test run. The test run is after the assembly phase, and in my study, I looked at the anomalies observed in the welding and assembly work phases, before test run. In the work done in 2016, the quality errors were mainly related to leaks and the error reports most often included words such as pipe, filter, and connection. There was seen as problematic that the text was often misspelled, incorrect headings were used and there were several error messages. Based on these findings, I see that performing text analysis would also be useful for test run error reports. In other words, text analysis has use in this company.

Based on the literature and publications, text analysis has not yet been widely used in industry in Finland, but there is certainly an order for it. This was revealed in this work as well as in the study on the utilization of Data Analytics in Satakunta. [1] Identifying deviations from production is important to be able to stay on production schedule. The study found, based on user comments, that different types of deficiencies are by far the most common root cause of the deviation. A certain part was missing for various reasons such as that part was missing from the collection, shelf, drawing, or was otherwise defective. The reason for the misclassification made by employees is probably due to the large number of categories (62) and partial overlap. Person doesn't known to which category the deviation would belong. This can be evidenced by the fact that the comment field was left blank in

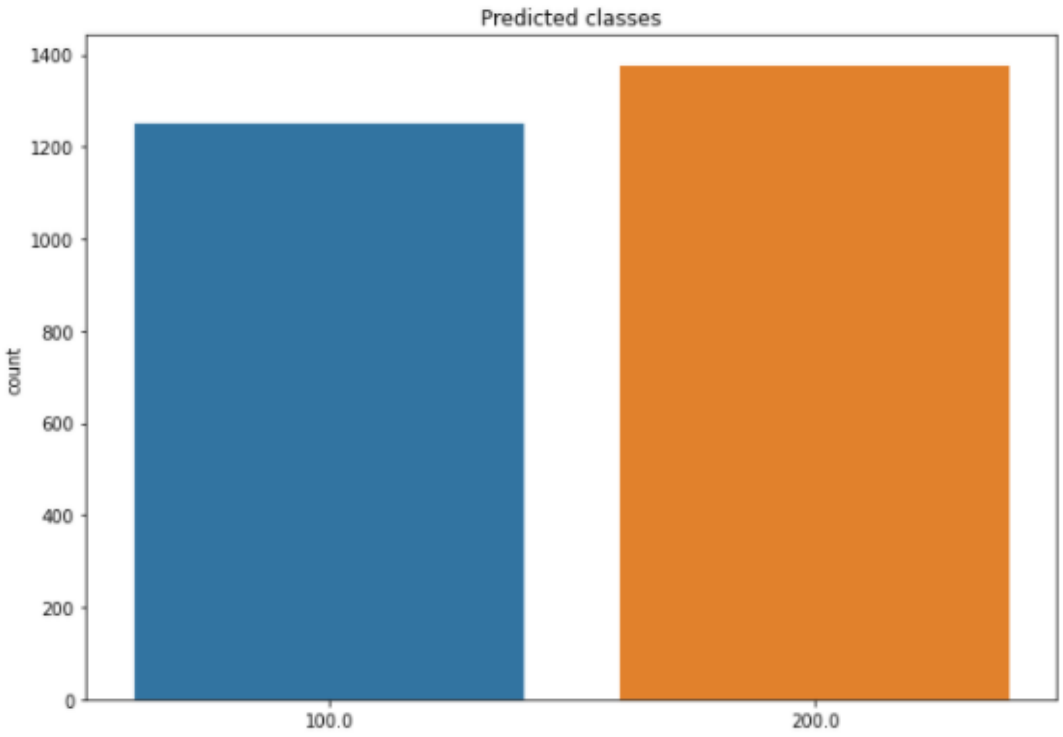


Figure 6.1: Predicted classes 100 Missing part and 200 Other

14 % of all comments and deviation that based on some lack, in 25 % of cases.

Based on the text analysis, the trend, increase and decrease of the deviations caused by the missing part, could be followed. An example of such a simple follow-up diagram is shown in Figure 6.1. Class divisions 100 and 200 can be modified as needed. However, in the classification, I consider it important to follow the deviations from the original classification, especially category 1 *. All critical deviations for the period under review were included in this category. It is also the most common category group, with 25% of findings. By teaching the classifier differently and utilizing the Multilingual BERT model, non-Finnish deviation sentences can also be studied, eg English, Swedish.

Quality monitoring as well as deviation monitoring are an essential part of quality improvement and the above support measures can improve quality through artificial

intelligence. The models used in this study can be used and fine-tuned according to the subject being investigated. If you want to find out the occurrences of smaller class in the MES data, you just need to collect more data to get the same size classes and then use machine learning methods for analysis. If, on the other hand, there is a wealth of data available, then there are deep learning methods and tools useful.

# References


[1] "Selvitys data-analytiikan nykytilasta ja data-analytiikan hyödyntämisestä satakunnassa", p. 105, (accessed 2021-003-04). [Online]. Available: http://www.datatiede.fi/wp-content/uploads/2018/03/Data-analytiikan-selvitys-Julkaisuversio-2018-03-23.pdf.

[2] C. Aggarwal Charu C.and Zhai, "A survey of text classification algorithms", in. Boston, MA: Springer US, 2012, pp. 163–222, ISBN: "978-1-4614-3223-4. DOI: 10.1007/978-1-4614-3223-4_6. [Online]. Available: https://doi.org/10.1007/978-1-4614-3223-4_6.

[3] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification", *Lingvisticae Investigationes*, vol. 30, pp. 3–26, 2007. DOI: 10.1075/li.30.1.03nad. [Online]. Available: https://nlp.cs.nyu.edu/sekine/papers/li07.pdf.

[4] C. Puente, J. A. Olivas, E. Garrido, and R. Seisdedos, "Creating a natural language summary from a compressed causal graph", in *2013 Joint IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS)*, 2013, pp. 513–518. DOI: 10.1109/IFSA-NAFIPS.2013.6608453.

[5] S. González-Carvajal and E. C. Garrido-Merchán, *Comparing bert against traditional machine learning text classification*, 2021. arXiv: 2005.13012 [cs.CL].

[6] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning based text classification: A comprehensive review", Apr. 2020. arXiv: 2004.03705 [cs.CL].

[7] A. M. Turing, "Computing machinery and intelligence", *Mind*, vol. LIX, no. 236, pp. 433–460, Oct. 1950, ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. [Online]. Available: https://doi.org/10.1093/mind/LIX.236.433.

[8] Wikipedia, *Turing test — Wikipedia, the free encyclopedia*, http://en.wikipedia.org/w/index.php?title=Turing%20test&oldid=1022418143, [Online; accessed 13-May-2021], 2021.

[9] MonkeyLearn, "Text analysis", (accessed 2021-05-14). [Online]. Available: https://monkeylearn.com/text-analysis/.

[10] A. Ittoo, L. M. Nguyen, and A. van den Bosch, "Text analytics in industry: Challenges, desiderata and trends", vol. 78, pp. 96–107, 2016.

[11] N. Chomsky, "Syntactic structures", *Mouton Co.*, p. 117, 1957. [Online]. Available: http://217.64.17.124:8080/xmlui/bitstream/handle/123456789/557/syntactic_structures%20(1).pdf?sequence=1.

[12] J. McCarthy, "Recursive functions of symbolic expressions and their computation by machine, part i", *Commun. ACM*, vol. 3, no. 4, pp. 184–195, Apr. 1960, ISSN: 0001-0782. DOI: 10.1145/367177.367199. [Online]. Available: https://doi.org/10.1145/367177.367199.

[13] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine", *Commun. ACM*, vol. 9, no. 1, pp. 36–45, Jan. 1966, ISSN: 0001-0782. DOI: 10.1145/365153.365168. [Online]. Available: https://doi.org/10.1145/365153.365168.

[14] Wikipedia, *Joseph Weizenbaum — Wikipedia, the free encyclopedia*, http://en.wikipedia.org/w/index.php?title=Joseph%20Weizenbaum&oldid=1015768051, [Online; accessed 13-May-2021], 2021.

[15] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[16] K. Foote, "A brief history of natural language processing (nlp)", (accessed 2021-03-07. [Online]. Available: https://www.dataversity.net/a-brief-history-of-natural-language-processing-nlp/.

[17] "Nlp datasets: How good is your deep learning model?", (accessed 2021-02-27). [Online]. Available: https://blog.floydhub.com/nlp-datasets-how-to-train-and-evaluate-your-deep-learning-model/#classification.

[18] Wikipedia, *Named-entity recognition — Wikipedia, the free encyclopedia*, http://en.wikipedia.org/w/index.php?title=Named-entity%20recognition&oldid=1020618016, [Online; accessed 13-May-2021], 2021.

[19] H. Goonewardana, "Evaluating multi-class classifiers", (accessed 2021-02-20). [Online]. Available: https://medium.com/apprentice-journal/evaluating-multi-class-classifiers-12b2946e755b.

[20] T. Cover and P. Hart, "Nearest neighbor pattern classification", *in IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967. DOI: 10.1109/TIT.1967.1053964.

[21] T. Bayes, "An essay toward solving a problem in the doctrine of chances", 1763. DOI: 10.1098/rstl.1763.0053.

[22] D. Jurafsky and J. H. Martin, "Naive bayes and sentiment classification", (accessed 2021-05-09). [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/4.pdf.

[23] C. Cortes and V. Vapnik, "Support-vector networks", *Machine Leaming*, vol. 20, pp. 273–297, 1995. DOI: 10.1007/BF00994018.

[24] scikit-learn, "Sklearn.multiclass.onevsrestclassifier", (accessed 2021-05-09). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html.

[25] ——, "Sklearn.multiclass.onevsoneclassifier", (accessed 2021-05-09). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsOneClassifier.html.

[26] ——, "Sklearn.svm.linearsvc", (accessed 2021-05-09). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html.

[27] F. Shaikh, "Deep learning vs. machine learning – the essential differences you need to know!", (accessed 2021-02-20). [Online]. Available: https://www.analyticsvidhya.com/blog/2017/04/comparison-between-deep-learning-machine-learning/.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need", 2017. arXiv: 1706.03762 [cs.CL].

[29] T. Rajapakse, "A hands-on guide to text classification with transformer models (xlnet, bert, xlm, roberta)", (accessed 2021-03-25). [Online]. Available: https://towardsdatascience.com/https-medium-com-chaturangarajapakshe-text-classification-with-transformer-models-d370944b50ca.

[30] O. Zafrir, G. Boudoukh, P. Izsak, N. Nikzad, and M. Wasserblat, "Q8bert: Quantized 8bit bert", Oct. 2019. arXiv: 1910.06188 [cs.CL].

[31] R. Horev, "Xlm — enhancing bert for cross-lingual language model", (accessed 2021-03-26). [Online]. Available: https://towardsdatascience.com/xlm-enhancing-bert-for-cross-lingual-language-model-5aeed9e6f14b.

[32] M. Rizvi, "Demystifying bert: A comprehensive guide to the groundbreaking nlp framework", (accessed 2021-03-25). [Online]. Available: https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/.

[33] "Turkunlp / finbert", (accessed 2021-03-25). [Online]. Available: https://github.com/TurkuNLP/FinBERT.

[34] S. Khan, "Bert, roberta, distilbert, xlnet — which one to use?", (accessed 2021-03-27). [Online]. Available: https://towardsdatascience.com/bert-roberta-distilbert-xlnet-which-one-to-use-3d5ab82ba5f8.

[35] G. Lample and A. Conneau, "Cross-lingual language model pretraining", Jan. 2019. arXiv: 1901.07291v1 [cs.CL].

[36] L. Yangguang, Z. Yangming, W. Shiting, and T. Chaogang, "A strategy on selecting performance metrics for classifier evaluation", *International Journal of Mobile Computing and Multimedia Communications*, vol. 6, pp. 20–35, 2014, (accessed 2021-05-15). [Online]. Available: https://www.igi-global.com/viewtitlesample.aspx?id=144443&ptid=91599&t=a+strategy+on+selecting+performance+metrics+for+classifier+evaluation.

[37] "Understanding boxplots", (accessed 2021-02-16). [Online]. Available: https://www.kdnuggets.com/2019/11/understanding-boxplots.html.

[38] W. A. Mohotti and R. Nayak, "Efficient outlier detection in text corpus using rare frequency and ranking", *ACM Transactions on Knowledge Discovery from Data*, vol. 14, Oct. 2020. [Online]. Available: https://doi.org/10.1145/3399712.

[39] A. Géron, *Hands-On Machine Learning with Scikit-Learn TensorFlow*, 1st ed. O'Reilly Media, 2017, pp. 86–92, ISBN: 978-1-491-96229-9.

[40] "10 hyperparameter optimization frameworks.", (accessed 2021-02-27). [Online]. Available: https://towardsdatascience.com/10-hyperparameter-optimization-frameworks-8bc87bc8b7e3.

[41] Wikipedia, *Stochastic gradient descent — Wikipedia, the free encyclopedia*, http://en.wikipedia.org/w/index.php?title=Stochastic%20gradient%20descent&oldid=1021654745, [Online; accessed 13-May-2021], 2021.

[42] "Understand the impact of learning rate on neural network performance", (accessed 2021-02-27). [Online]. Available: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/.

[43] F. Martínez-Plumed, C. Contreras-Ochando F.and Ferri, J. Orallo, M. Kull, N. Lachiche, M. Quintana, and P. A. Flach, "Crisp-dm twenty years later: From data mining processes to data science trajectories", Dec. 2019. arXiv: 1912.07076 [cs.CL].

[44] Wikipedia, "Cross-industry standard process for data mining", (accessed 2021-03-29). [Online]. Available: https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining.

[45] J. Brownlee, "A gentle introduction to imbalanced classification", (accessed 2021-02-20). [Online]. Available: https://machinelearningmastery.com/what-is-imbalanced-classification/.

[46] ——, "8 tactics to combat imbalanced classes in your machine learning dataset", (accessed 2021-02-20). [Online]. Available: https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/.

[47] C. Conner, J. Samuel, A. Kretinin, Y. Samuel, and L. Nadeau, "A picture for the words! textual visualization in big data analytics", May 2020. arXiv: 2005.07849v1 [cs.SI].

[48] B. Gunjit, "A guide to text classification(nlp) using svm and naive bayes with python", (accessed 2021-02-13). [Online]. Available: https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34.

[49] E. Haddi, X. Liu, and L. Shi, "The role of text pre-processing in sentiment analysis", (accessed 2021-02-13). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050913001385.

[50] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011, (accessed 2021-05-15). [Online]. Available: https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf?source=post_page---------.

[51] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. Vanderplas, A. Joly, B. Holt, and G. Varoquaux, "Api design for machine learning software: Experiences from the scikit-learn project", 2013. arXiv: 1309.0238 [cs.LG].

[52] D. D. Lewis, "Naive (bayes) at forty: The independence assumption in information retrieval", (accessed 2021-02-13). [Online]. Available: https://www.researchgate.net/profile/David_Lewis24/publication/2551521_Lecture_Notes_in_Computer_Science/links/561284f408ae83674f4376cd/Lecture-Notes-in-Computer-Science.pdf.

[53] scikit-learn, "Support vector classification", (accessed 2021-02-14). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html.

[54] A. Virtanen, J. Kanverva, R. Ilo, J. Luoma, J. Luotolahti, T. Salakoski, F. Ginter, and S. Pyysalo, "Multilingual is not enough: Bert for finnish", Nov. 2019. arXiv: 1912.07076 [cs.CL].

# Appendix A Original classes

Table A.1: Original deviation classes

| CODE_ID | Reason Text | lines pcs | CODE_ID | Reason Text | lines pcs |
|---|---|---|---|---|---|
| 1 | MODUULIPUUTE | 103 | 2.3. | KYTK_KANNATIN | 1 |
| 2 | KOMPONENTTIPUUTE | 347 | 2.4. | MÄNTÄ | 4 |
| 3 | JIG_KULJETALUSTA | 3 | 2.5. | PAKOPUTKI | 3 |
| 4 | KERAYSPUUTE | 350 | 2.6. | PILOTTIPUMPPU | 9 |
| 5 | KOMPONENTTI | 187 | 2.7. | POLTTOAINE | 22 |
| 6 | PAKKAUS | 19 | 2.9. | PUMPPUKOTELO | 27 |
| 7 | SUUNNITTELUVIRHE | 78 | 3.1. | YLIM_MATERIAALI | 16 |
| 8 | TYOOHJEET | 35 | 3.2. | PAKKAUS_PURKAM | 8 |
| 9 | AKTIVITEETTI | 39 | 3.3. | TOIMITUSP_VÄÄRÄ | 19 |
| 10 | MUU | 123 | 3.4. | VIRHEEL_PAKKAUS | 26 |
| 11 | TYOKALU | 8 | 3.5. | KER_VÄÄRÄ_ASETT | 13 |
| 12 | JIG_KULJETALUSTA | 4 | 4.1. | VERST_SISÄ_VIRH | 189 |
| 13 | PAINEILM_SAHK_VE | 2 | 4.2. | MUUN_OSAST_VIRH | 324 |
| 14 | TYOSTOKONE_ROBOT | 1 | 4.3. | TOIMITT_VIRHE | 818 |
| 15 | MITTALAITTEET | 2 | 4.4. | SUUN_VIRHE_NPI | 442 |
| 16 | MUU | 7 | 4.5. | SUUNN_VIRHE_MUU | 232 |
| 17 | VIRHEILMOITUS | 25 | 4.6. | ERITTELYVIRHE | 23 |
| 18 | KOULUTUS_PUUTE | 1 | 5.1. | TYÖOHJEET | 100 |
| 19 | PUUT_TOIMINNALLI | 25 | 5.2. | AKTIVITEETTI | 210 |
| 20 | PurchasingDefect | 1 | 5.3. | SETITYSVIRHE | 161 |
| 1.1. | KOMP_PUUTE | 2880 | 5.4. | MUU | 747 |
| 1.2. | JIG_KULJ_PUUT | 2 | 6.1. | TYÖKALU | 30 |
| 1.3. | KOK_KERÄYS_PUUT | 472 | 6.2. | JIG_KULJ_KIINNI | 6 |
| 2.1. | AHTOILMA | 39 | 6.3. | PAINILM_SÄHK_VE | 2 |
| 2.10. | PÄÄKAASUPUTKI | 2 | 6.4. | TYÖSTÖK_ROBOTTI | 5 |
| 2.11. | RYNT_SUOJ_SÄIL | 14 | 6.5. | MITTALAITTEET | 5 |
| 2.12. | SÄÄT_KÄYT_LAIT | 9 | 6.6. | MUU | 46 |
| 2.13. | STARTTI | 35 | 7.1. | MES | 150 |
| 2.14. | SYL_KANSI | 10 | 7.3. | QDMS | 45 |
| 2.15. | TURBO | 118 | 7.4. | MUU | 51 |
| 2.16. | VOITELUÖLJY | 29 | | | |
| 2.2. | HUKKAPORTTI | 15 | | TOTAL | 8719 |

# Appendix B New classes

Table B.1: New deviation classes

| Class | | New Class | Class | | New Class |
|---|---|---|---|---|---|
| 1 | MODUULIPUUTE | 100 | 3 | JIG_KULJETALUSTA | 200 |
| 2 | KOMPONENTTIPUUTE | 100 | 4 | KERAYSPUUTE | 200 |
| 1.1 | KOMP_PUUTE | 100 | 5 | KOMPONENTTI | 200 |
| 1.2 | JIG_KULJ_PUUT | 100 | 6 | PAKKAUS | 200 |
| 1.3 | KOK_KERÄYS_PUUT | 100 | 7 | SUUNNITTELUVIRHE | 200 |
| 2.1 | AHTOILMA | 100 | 8 | TYOOHJEET | 200 |
| 2.10 | PÄÄKAASUPUTKI | 100 | 9 | AKTIVITEETTI | 200 |
| 2.11. | RYNT_SUOJ_SÄIL | 100 | 10 | MUU | 200 |
| 2.12. | SÄÄT_KÄYT_LAIT | 100 | 11 | TYOKALU | 200 |
| 2.13. | STARTTI | 100 | 12 | JIG_KULJETALUSTA | 200 |
| 2.14. | SYL_KANSI | 100 | 13 | PAINEILM_SAHK_VE | 200 |
| 2.15. | TURBO | 100 | 14 | TYOSTOKONE_ROBOT | 200 |
| 2.16. | VOITELUÖLJY | 100 | 15 | MITTALAITTEET | 200 |
| 2.2. | HUKKARAPORTTI | 100 | 16 | MUU | 200 |
| 2.3. | KYTK_KANNATIN | 100 | 17 | VIRHEILMOITUS | 200 |
| 2.4. | MÄNTÄ | 100 | 18 | KOULUTUS_PUUTE | 200 |
| 2.5. | PAKOPUTKI | 100 | 19 | PUUT_TOIMINNALLI | 200 |
| 2.6. | PILOTTIPUMPPU | 100 | 20 | PurchasingDefect | 200 |
| 2.7. | POLTTOAINE | 100 | 3.1. | YLIM_MATERIAALI | 200 |
| 2.9. | PUMPPUKOTELO | 100 | 3.2. | PAKKAUS_PURKAM | 200 |
| | | | 3.3. | TOIMITUSP_VÄÄRÄ | 200 |
| | | | 3.5. | KER_VÄÄRÄ_ASETT | 200 |
| | | | 4.1. | VERST_SISÄ_VIRH | 200 |
| | | | 4.2. | MUUN_OSAST_VIRH | 200 |
| | | | 4.3. | TOIMITT_VIRHE | 200 |
| | | | 4.4. | SUUN_VIRHE_NPI | 200 |
| | | | 4.5. | SUUNN_VIRHE_MUU | 200 |
| | | | 4.6. | ERITTELYVIRHE | 200 |
| | | | 5.1. | TYÖOHJEET | 200 |
| | | | 5.2. | AKTIVITEETTI | 200 |
| | | | 5.3. | SETITYSVIRHE | 200 |
| | | | 5.4. | MUU | 200 |
| | | | 6.1. | TYÖKALU | 200 |
| | | | 6.2. | JIG_KULJ_KIINNI | 200 |
| | | | 6.3. | PAINILM_SÄHK_VE | 200 |
| | | | 6.4. | TYÖSTÖK_ROBOTTI | 200 |
| | | | 6.5. | MITTALAITTEET | 200 |
| | | | 6.6. | MUU | 200 |
| | | | 7.1. | MES | 200 |
| | | | 7.3. | QDMS | 200 |
| | | | 7.4. | MUU | 200 |

# Appendix C FN and FP values

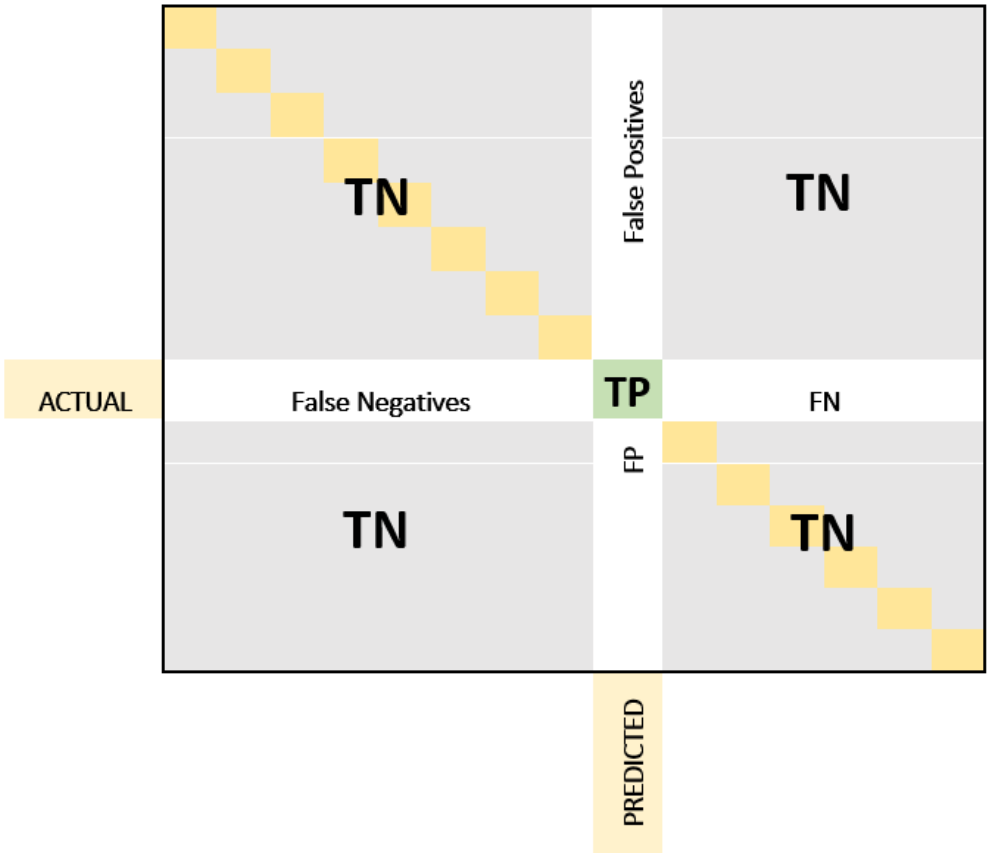


Figure C.1: How to read FN and FP values from matrix