jamk

Predicting budget adequacy

A case study for Kieku data

Author Teemu Väisänen

Master's thesis
December 2023
Artificial Intelligence and Data Analytics

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Description

Väisänen, Teemu

Predicting budget adequacy, a case study for Kieku data

Jyväskylä: Jamk University of Applied Sciences, December 2023, 69 pages.

Degree Programme in Artificial Intelligence and Data Analytics. Master's degree.

Permission for open access publication: Yes

Language of publication: English

Abstract

Machine learning (ML) has gained popularity in the recent years due to increasing availability in computing power and tools such as ChatGPT. This created more interest in ML and companies started to invest in the related technologies. As interest increased additional exploration was done to find out where ML could be implemented. This eventually led to people wanting to predict future events such as engineering malfunctions and stock prices.

In the financial field there are multiple different use cases as well but in this study the focus is on budget adequacy prediction. The goal was to use ML algorithms to predict whether the given budget covers the costs of each tracking subject to the end of the year. The data that was available was the monthly expenses, salaries, and budget for the subject. Since that was the data that was being gathered for operational usage, it was theorized that it could be used for ML to create predictions. This ML model was in turn meant to be used by the financial department as a way of keeping track of each budgeted monitoring subject so that in case some of them were in danger of going over budget. The department could then react before the budget was exceeded. The study is done to see if the data could be used to train a ML model that would be able to learn and predict budget behavior patterns and would be able to create accurate predictions based on them.

The implementation takes advantage open-source software, python libraries and Microsoft Azure cloud environment. The data comes from the government HR and financial system Kieku. It has been in use since 2016 and the data is readily available for research purposes. After evaluating multiple ML algorithms, an artificial neural network (ANN), or a random forest classifier (RFC) was found to be accurate for this purpose. In the testing phase ANN model reached 96% and the RFC model reached 99% accuracy. Further testing should be done with production data to see if the difference in production data and historical data makes a difference in the models' accuracy.

Improvements to the underlaying data quality are presented as it could improve the machine learning models accuracy and overall performance. In addition, a visualization of the predictions for end users would also bring added benefits to the process as visual representations tend to be easier to read.

Keywords/tags (subjects)

Finance, Financial data, Budget prediction, Machine Learning, Artificial Intelligence

Miscellaneous (Confidential information)

Contents

Figures

Tables

Abbreviations

AGI	Artificial General Intelligence
ANN	Artificial neural network
CRISP-DM	Cross-industry standard process for data mining
DL	Deep Learning
EDA	Exploratory data analysis
IaaS	Infrastructure as a Service
LSTM	Long Short-Term Memory
ML	Machine Learning
NLP	Natural Language Processing
PaaS	Platform as a Service
RL	Reinforcement Learning
RNN	Recurrent neural network
SaaS	Software as a Service
SVC	Support Vector Classifier
SVM	Support Vector Machine

1 Introduction

This study aims to answer the question of "Can you use budget, wages and expenses data to create accurate predictions on whether a budget will last until the end of the year?". The primary method of research is a case study. It consists of quantitative data analysis from a data set that has been gathered over many years in a financial and human resources system. Understanding of the data was grown by interviewing substance matter experts to further understand the contents and complexity of the data and to help grow knowledge on how the substance field operates. This will further enhance the data analysis portion of the study by giving insight on the data itself and to understand what works and what does not, what is faulty data and what are the pitfalls that need to be taken into consideration.

The organization that this study was done for was called KEHA-keskus. KEHA operates as development and governance center for both Centre for Economic Development, Transport and the Environment and the work and livelihood offices (Työ ja Elinkeino -toimistot). The role of KEHA among others is to offer services that benefit both organizations and as such can oversee funding on a greater scale. This in turn creates other challenges in the finance department, as the funding for each organization is similar yet they operate in slightly different ways and have slightly different projects going on. To add to the layers of complexity, multiple ministries and government offices that were also being served had their own projects and fundings.

To better understand what goes on in monthly basis and to help reduce budgets from going over, it was theorized that a machine learning (ML) model could be used to create predictions to help the financial department to pre-emptively target those budgets. These results could then be displayed in a visual format with Microsoft PowerBI to create a tool that the financial department could use on a daily basis. As an added benefit, the study could be used as to pique interest on other government offices on the possibilities of ML on the shared Kieku service. It could also serve as a starting point for other government offices that wish to implement related solutions so they would not need to start from scratch.

2 Artificial intelligence and Machine learning

Artificial intelligence (AI) is a term used to describe a field of study in computer science of which end goal is to achieve human like intelligence in machines. In their journal Haenlein & Kaplan (2019) go through how the field has been around for approximately 67 years. Although the beginning can be tracked all the way back to 1942 to the short story Runaround, the official founding of the term was done in 1956 in a workshop named *Dartmouth Summer Research Project on Artificial Intelligence* (*DSRPAI*), that was held at Dartmouth College in New Hampshire. The field has gone through multiple phases but due to lack of computing power it was in hibernation for a long time. It resurfaced as computing capabilities increased exponentially and it became feasible to run AI applications.

Haenlein & Kaplan (2019) also explain how artificial neural networks (ANN) and deep learning (DL) resurfaced to public after Googles AlphaGo beat the world champion in 2015 in a game of Go. While the technology has evolved to a more readily available state, the current AI solutions use the same principles to make intelligent solutions. Both ANN and DL are under the ML category which is a subcategory of AI. The advances in ML refueled the funding and development of different methods and frameworks to make AI solutions more readily available.

In 2023 big tech companies, such as Google (Pichai, 2023) and Microsoft (Microsoft, 2023) announced that they were making significant investments in AI technologies. Meta and Apple were also making investments in the field but are not publicly drawing as much attention to it. This in turn prompted a significant movement from investors to invest in Nvidia as they were the top competitor for AI proprietary hardware. The company's stock briefly even touched the 500 USD mark.

Figure 1. Nvidia stock price rally after start of 2023, following 2022 market decline and big tech companies announcing their AI plans (Google market summary, 2023)

For common people this was both exiting and worrisome as the technology could be used to both harm and help the general populus. Among the popular references against AI are the Chinese social credit system and George Orwell's novel 1984. Both references bring up cases of mass surveillance and control over the population by controlling what is acceptable behavior and punishing those that do not behave accordingly. In the Chinese case this is enforced with surveillance tools that have been amplified with AI tools such as facial recognition that is used to put fines instantly on jaywalkers for example (Baynes, 2018). To combat these issues and worries both the United States (The White House, 2022) and The European Union (European Commission, 2021) are working on separate AI legislations. In the EU version artificial intelligence use cases are categorized into four categories: Unacceptable risk, high risk, generative artificial intelligence, and limited risk. These categories are used to prevent certain uses completely, allow use with strict oversight, allow use with transparency requirements, and allow use based on letting the user know that they are interacting with an AI, respectively. Controlling AI tools by legislation also brings up the question if implementing and making changes to it will be fast enough to truly control the environment as it is growing very rapidly and reaching new heights day by day (Chhillar & Aquilera, 2022). Even though

the European legislation is not yet in use, based on the presented categories, this study falls on the limited risk category.

2.1 Artificial intelligence

In general conversations the term AI is often being used as Artificial General Intelligence (AGI). This means that it is generally referred as a computer that very similar things to humans (AGI Workshop, 2007). Although while services such as ChatGPT — an AI-powered large language model developed by OpenAI, capable of generating human-like text based on context and past conversations (OpenAI, n.d.) — have proven to be able to have conversations and give code examples from natural language questions would make it seem like AGI could soon be possible, the technology is not quite yet there. The journal by Du-Harpur et al. (2020) explains how currently most of the AI solutions are fundamentally ML algorithms that have been taught patterns that occur in the data. ML algorithms are being fed different kinds of information, and they generate responses based on the given information and the desired output defined in the model. While it does have intelligence to a degree, it is limited by the data it is given and cannot really think on its own outside of the data. This in turn limits the possibilities on what AI can and cannot do. For the time being data-based learning covers most of the capabilities of AI applications. This has been expanded by reinforcement learning to teach an algorithm how to reach a goal by giving it rewards and punishments.

However, as we move further into the future and with the rapid growth of AI applications it might become possible to expand the capabilities further in the future. This might require additional computing power from Quantum computing and a breakthrough from medical and/or psychological fields to further understand how thinking and combining data to create new thoughts work. Even now ethics and morals should really be taken into consideration so that AI doesn't become a malevolent tool but would instead benefit mankind. This becomes increasingly important in the future as the technology evolves. As for the reality and judging from past technological inventions, more often than not emerging technologies do include both malevolent and benevolent use cases. This tends to mimic human behavior as technology is usually just as good or bad as its user, or in software cases its designer and implementer.

2.2 Machine learning

ML is a subcategory of AI. Du-Harpur et al. (2020) explains how on the most basic level it is a way of teaching an algorithm to make conclusions from given data. Most commonly it is based on statistical formulas that are formed from a large amount of data. Using neural networks however changes the algorithms to a more human-like way of handling data by using artificial neurons. The more examples the algorithm gets the more accurate the resulting model can be. However, as life tends to be unpredictable and data can end up being skewed, incomplete or inaccurate, making it harder to produce an algorithm that would be accurate (Yang et al. 2023). Another issue that hinders the accuracy of machine learning is the problem of defining an accurate dataset. This includes selecting data that should be taken into consideration and features that should be removed from data to prevent biases or inaccuracies (Yu, 2004). When designing a dataset for a use case the implementation should also consider how to handle events that happen outside of the provided data that affects the events in the data. These can sometimes make the algorithm inaccurate outside of the testing phase.

An example of data outside the provided data would be an algorithm that evaluates the values of houses but does not have information on laws that the government could make that that fundamentally affects the prices. In their book Solow & Lo (2012) explained the events of the 2008 financial crisis. This could be used as a real-life example of an event in which outside events affect the events in the dataset. If a model existed that would have predicted house prices and the underlying data, that was being used to retrain or further train the model, updated two times a year, it might have been too slow to update the model to match the latest changes. As the data reflects the change after a delay it might be slow to retrain the algorithm depending on how much new data comes in. One could argue that the financial crisis was building up over the years but if the algorithm was not given data that takes the buildup into consideration, it would not have been possible for the algorithm to adjust accordingly. A similar effect would have been achieved if houses would have not been selling and the amount of new data remained small. This way the resulting model might still assume that the historical events would be accurate predictors of the future.

2.3 Machine learning process

Machine learning process works very similarly in many cases and follows a set of steps to reach the desired result (Sivula, 2021). While there might be some deviation in how the result is reached the required steps remain majorly the same. These steps are illustrated in figure 2.

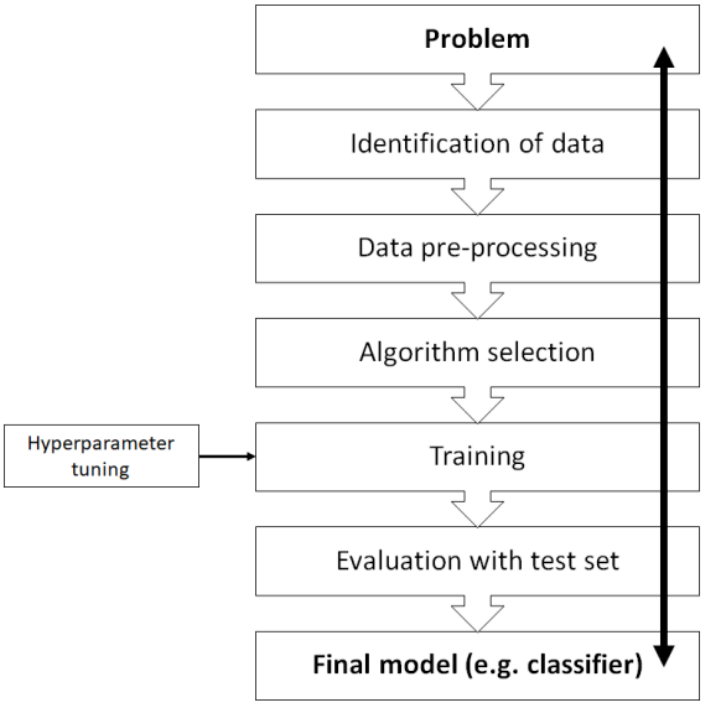

Figure 2. Machine learning algorithm development (Sivula, 2021)

The first step in most cases is identifying a problem that needs a solution (Sivula, 2021). However, this can be interchangeably used with identification of data and formulating a solvable problem from the available data. After the initial problem is identified and described the next step is to find data that could be used in finding the solution in (Sivula, 2021). Usually, the problems that are being solved through ML involve issues that are harder to solve trough conventional means or ones that require analyzing considerable amount of data. This process can be troublesome if the problem comes around before data is available. That means that either the required data does not exist, is not collected in a machine-readable format, or is very limited in quantity. Not having data leads to issues as to where to acquire the relevant data. If the problem that is being solved is a company specific issue, then the data needs to be collected and often this leads to huge delays in

the ML process. In most cases collecting data from scratch takes months, if not years, depending on the complexity of requirements.

Sivula (2021) then describes the third step after identifying a problem and gathering the data is preprocessing it. This is done to make the data usable in machine learning applications. Research by Yang et al. (2023) includes an analysis phase to the preprocessing phase. This is where errors in data are corrected, missing values and outliers are addressed, and duplicates and unnecessary features are removed. This is done to further optimize the data into a subset that fits the purpose. This process can be sped up by having substance and data specialists give insights on how the data is collected and how it is being used. Usually this means that there are people who are familiar with the data to inform the person, group or team doing the analysis about what are the shortcomings of the data, how it operates and what is the best way to make it fit the use case. In an optimal situation the substance specialists can even give insight on which fields and rows are useful and which are irrelevant. Identifying outliers and faulty data from the start is useful insight to the data as well. Having insight into the substance field is useful even if the specialists do not have insight into the data itself. By gaining domain knowledge feature usefulness and outliers can be identified and their utility determined. Some of the outliers and faults can be identified with proper data analysis methods since they tend to stand out from the rest of the data; however, it might be difficult for a person without sufficient domain knowledge to understand whether they need to be removed. Doing the preprocessing is a crucial part of the process as it can also raise issues that might prevent the next steps entirely. As data is the primary source of intelligence on the algorithm it needs to be in adequate state to make the resulting ML model viable for use beyond testing. Should the data be inadequate the resulting model will produce similar results.

The last three steps Sivula (2021) described before reaching the final model are the machine learning parts. After data has been confirmed to be in an apt state, the algorithm selection, training and tuning, and model evaluation can be executed. In the algorithm selection part computing power requirements, data amount, and accuracy are the significant factors of the decision on how to approach the problem. While cloud computing environments provide seemingly unending amounts computing power they usually come with increasing costs, which usually limits the amount that is appropriate for the use case (Microsoft Azure, n.d.-a). Data amount can be reduced in cases where it is plentifully available but on cases where data is limited it is harder to evaluate whether the

data is enough for the algorithm to generalize it (Yang et al., 2023). According to Sivula (2021) these steps culminate in the training and evaluation step where testing can be done to see if data is plentiful and if the algorithm has been chosen adequately. In this phase the parameters and hyperparameters are used to optimize the learning process. After choosing a model that has been tested, trained, and evaluated to provide reasonable results the process reaches the end of the first iteration. The process can be repeated as many times as deemed necessary. However, it is more common to and cost effective to focus on updating and training the algorithm further as new data becomes available.

2.4 Common data issues

As for the data that ML algorithm needs, it needs to be useful and accurate enough so that the patterns and events in the substance are well represented. Among the most common issues that ML faces are the lack of data itself, biases in the data, inaccurate, incomplete, or faulty data (Yang et al. 2023). In this chapter some of the common issues, that should be taken into consideration when planning a ML project, are presented. Simply having data might not be a solid base to build ML upon.

Yang et al. (2023) mentions how quantity of data will affect the quality of the ML model. The lack of data creates issues in the learning process where the provided amount is not enough to make accurate generalizations of the events that the ML algorithm is tasked to find. An example of this would be categorizing a data set of 100 rows into three categories in which categories 1 and 2 would have only a few samples, and the majority of the data would consist of category 3 samples, while categories 1 and 2 would in reality be larger portions of the data. The amount of data itself heavily affects the outcome as it is important to have enough source material so that the learning process is not limited to a set that doesn't allow the ML algorithm to learn the necessary patterns of behavior in a broad enough way. Such as if an event happens only a few times a year but the machine is learning from a set where the percentage of such events is higher than the real rate. Another possible flaw is that the data has too few examples to draw generalizations from. These can include situations in where the contents of the data vary too much to accurately correlate what affects the outcome and what does not. This can also be called interference or jitter.

Biases are systemic statistical anomalies that affects the outcome, but they should not be part of the derivation (Yang et al. 2023). An example of this was the hiring algorithm from Amazon (Lavanchy, 2018) in which they set up an algorithm to hire people in the technology field . Gender was included in the data so eventually the algorithm started to form a bias towards male applicants and recommended them over female applicants for the most promising people to hire. This was since most people in the technology field tend to be male. This created a data set that was not intended to favor the mentioned group but eventually learned to do so. Similar biases can be spotted in other fields and ML applications as well if they are not properly vetted out. Although the vetting process can be difficult as human behaviors include biases in them due to how memory works (Cacioppo, 2002). Nevertheless, vetting out as much of it as possible should be an integral part of ML model tuning.

In the journal Yang et al. (2023) also include categories of inaccurate or incomplete data that need to be taken into consideration. This refers to a data set that technically can have all the necessary data that is required for a successful ML model, but the contents are either inaccurate or incomplete. For example, you can have a database that collects all the needed fields, but the contents are empty, they contain user inputted free text that is not related to the case in question or mistakes the user has made while saving the data. Some other possibilities are logical impossibilities on the subject, for example a user has been logged swimming multiple times but are marked as person who doesn't know how to swim, or data relationship errors, for example an office worker that has no office. These tend to be spotted in the exploratory data analysis (EDA) phase. If these kinds of errors make up for most of the data, then even though technically the data exists, it is low quality and not usable for ML purposes. ML cannot be used to turn bad data into good results. The outcome is very dependent on the quality of the data.

Faulty or incompatible data can be worked or repaired to a certain extent during the EDA part, but even that has its limits (Yang et al., 2023). For example, if the data is in the wrong format, for example in pdf-, picture- or word-format, these might still be able to be converted into a machine-readable format but would require some manual labor to get it done. However, if the data is in a corrupted disk or contents of the data have been corrupted it might be impossible to get them to be in a usable format. While this is not an AI specific issue, it should be kept in mind as partial corruptions are possible as well. In their article Wang & Bu (2020) explain how corruptions can be

tackled in power systems. In partial corruption cases only a portion of the data is usable which in turn might make the amount of data too scarce or inaccurate. This varies on case-by-case basis and while in the power systems situation awareness system it was able to be handled by applying a novel aggregation of random matrix and a long short-term memory (LSTM) model to enhance the existing one, this might not always be the case.

Technically incompatible data is a category in which the data is not otherwise limited but due to how and what is being collected it might not be feasible to link given data to other data in the set. For example, if you are given three different files one consisting of cars, another about taxi routes, and the third about times and dates of which the routes have been driven but none of the files have a key that can be used as a relation. In these cases, the data itself might not be usable in conjunction with each other. This in turn limits the possible ML models that can be derived from the data.

2.5 Learning types

ML can be done in multiple ways. Currently the main three processes of applying it to the data are supervised, unsupervised and reinforcement learning (Sivula, 2021). Each of these differ based on the available resources, the behavior of the events that are being observed and the desired result. For example, events that follow linear patterns and have a clearly defined measurement can be tracked with a supervised linear regression model. While a self-driving car might be better off with a reinforcement learning model.

In the book Doshi et al. (2022) explain how supervised learning is based upon data that has a clearly defined measurable metric that can be used as a goal post. This is done through a dataset that has examples of the things that are being measured. For example, an algorithm that categorizes pictures of dogs and cats into their respective categories belongs in this category. The term supervised comes from the fact that data and the outputs are handled by a supervisor which in most cases means that a human has been arranging the data into preset categories that the algorithm is then learning to use. While this process can be very accurate and have good results, the process done to reach the result involves manual labor in categorizing the new data when the algorithm needs to be trained further. This can often be tedious especially when the output categories or raw unlabeled data increase.

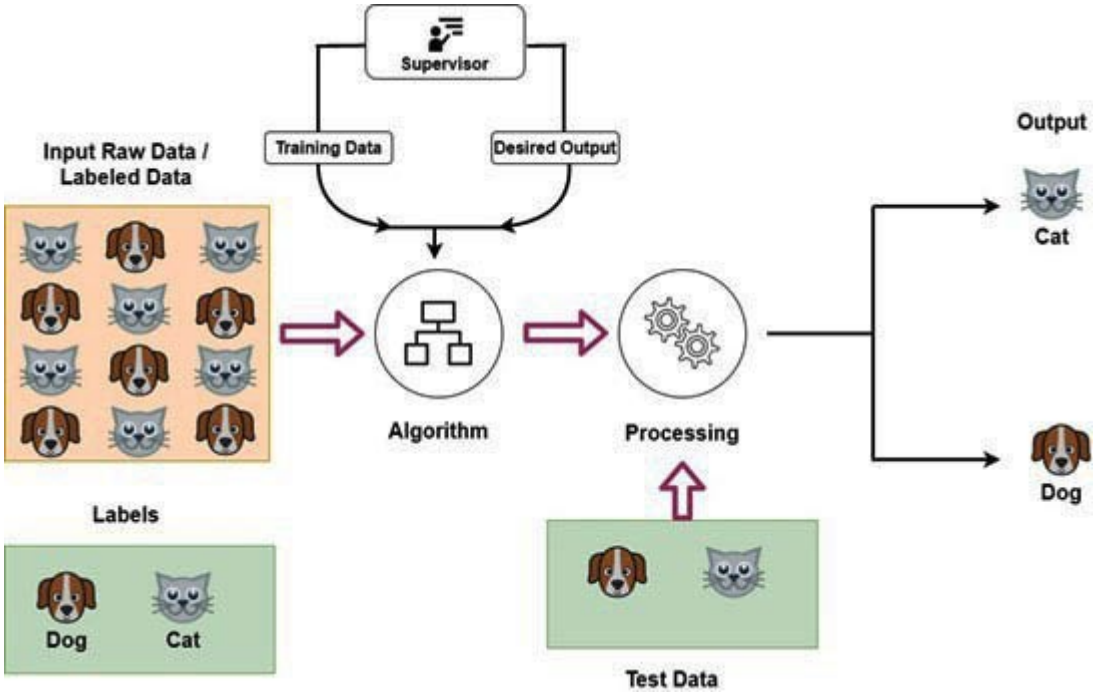

Figure 3. Supervised learning process (Doshi et al. 2022)

Regression is another type of supervised learning. Doshi et al. (2022) describe regression as a mathematical method that attempts to create relationships and determine their strengths between the output and input variables. In these cases, the output variable is a continuous or a real value, whether positive or a negative one. This type of ML is used to create predictions in cases such as weather forecasts, stock price predictions, and time-series forecasts, among others. Regression learns patterns and correlations from previous data and uses that to create predictions on what the desired output value should be. As these tend to approximations the evaluation is done by examining the error value instead of direct accuracy as is done in categorization models.

As a counterpart to supervised learning there is also unsupervised learning. In the book Doshi et al. (2022) explains how, unlike in supervised learning, unsupervised skips the supervision part and works on data that is not as clearly defined. In these cases, the data is unlabeled and is used to solve issues that have more complexity. These tasks involve issues such as finding patterns, grouping data into categories, or finding relations, associations, and dependencies in the data. Unsupervised machine learning is useful when the data can't be manually supervised but it still can be used to learn for example anomalies. An example of this would be machine malfunctions in a factory production line.

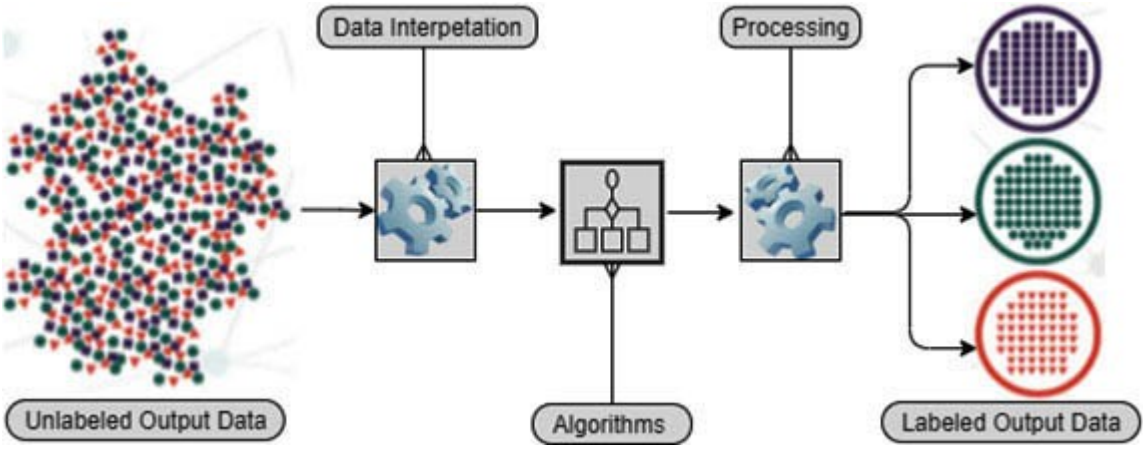

Figure 4. Unsupervised learning process (Doshi et al. 2022)

The third main category is reinforcement learning (RL). Doshi et al (2022) proceed to explain it as a trial-and-error approach to learning that usually uses a Markov decision Process to adapt to the environment it is given. This can be utilized when a machine must work within a limited space towards an objective that does not follow a clear path but instead focusing on finding a way to perform a task that can be done in multiple ways. One example of reinforcement learning is a car driving from point A to point B. While the goal is clearly set, the way to get there is not clearly defined. You can take the fastest route or the shortest route. You might encounter other drivers, pedestrians, big cars, small cars, animals, among other variables that make the task multidimensional. The way this is solved using RL is very similar to how humans operate. The operator is given the environment and a goal to work towards, in this example driving a car, then by doing the same task repeatedly and feeding the learned information back to the main algorithm, the algorithm gets better and better at it. By defining the task and giving options the algorithm chooses from on the way a plus or a minus score and telling it to maximize the score, the algorithm will eventually learn what to do and what not to do.

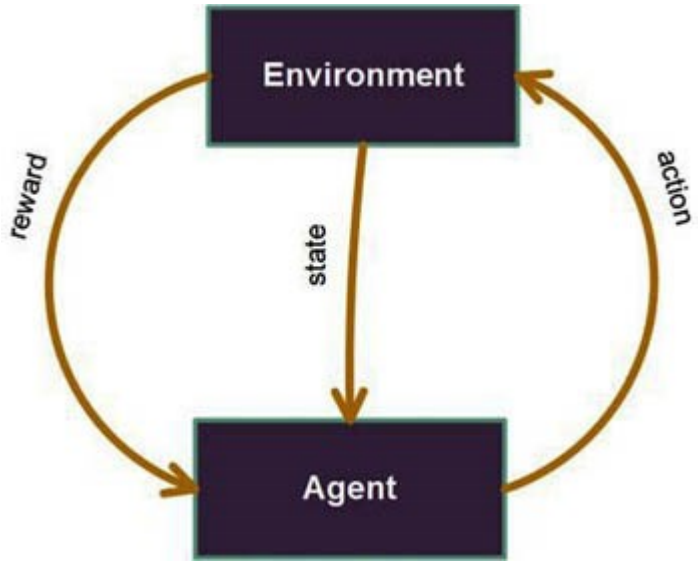

Figure 5. Markov Decision Process in RL (Doshi et al., 2022)

These are the basic categories of machine learning. While each category has multiple subcategories and categories can be combined to create middle grounds most of the ML algorithms fall under these main categories.

2.6 Algorithms and models

The core of ML consists of three components, an algorithm, the training process, and the resulting model. The algorithm is the approach for the problem, the training is what is being done with the data by the algorithm, for example prediction, and the model is the result of an algorithm that has been trained (Mattmann & Penberthy, 2020). Although model and algorithm terms are often being interchangeably used, the difference between them should be kept in mind.

As mentioned in the previous chapter, ML algorithms fall into three main categories. The main categories are supervised learning, unsupervised learning, and reinforcement learning (Mattmann & Penberthy, 2020). Although these are the three main categories, new categories emerge as the technology progresses. Mattmann & Penberthy (2020) present a fourth category of meta-learning and Doshi et al. (2022) introduces semi-supervised learning. It is likely that more and more categories are invented as time goes on. Since the use cases and underlying concepts differ different categories and algorithms solve different problems. It also might not prove feasible to try to mix and match algorithms from different categories together. For example, regression algorithm cannot be

used solve classification problems and classification algorithms cannot be used to solve reinforcement learning problems. Figure 6 illustrates the main categorization of ML algorithms.

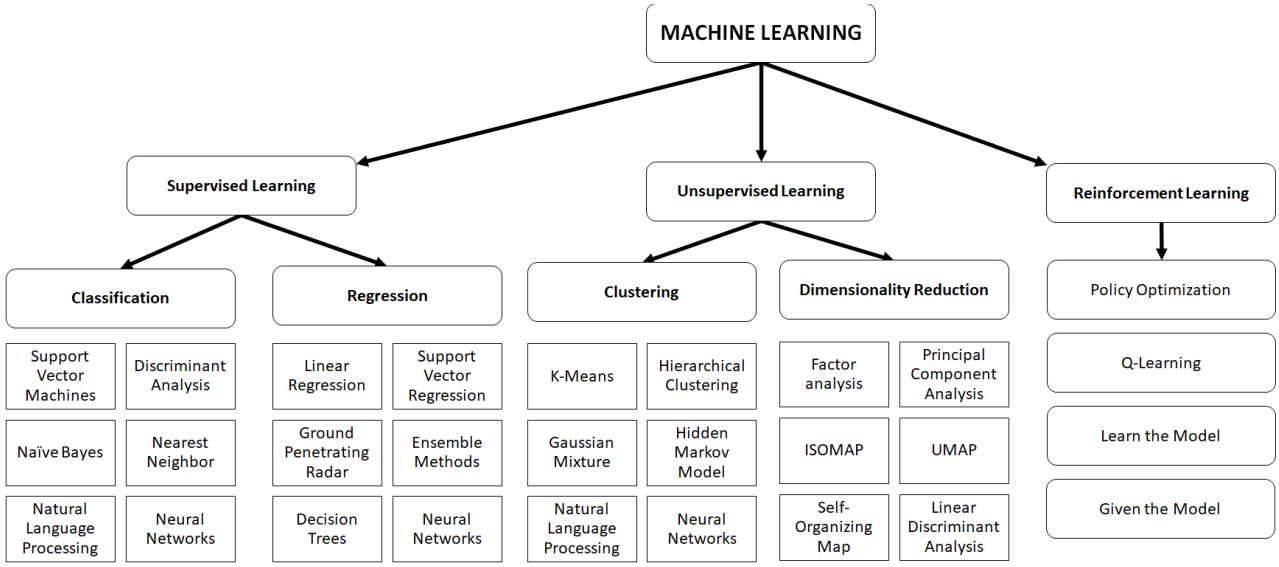

Figure 6. Machine learning algorithms by categories (adapted from Sivula, 2021)

Supervised learning includes but is not limited to models such as Linear regression, logistic regression, decision tree, and support vector machines. Doshi et al. (2022) and Hoang & Wiegratz (2023) describes supervised learning being used in cases where the training consists of a dataset that input variables and an output variable, and the desired output is clearly labeled. The output can be a category or a continuous value. The result of using a supervised learning model is an algorithm that can predict the correct label or value for data it has not previously seen. Thus, bringing value to the process.

Unsupervised category contains models such as K-means, hierarchical clustering, and principal component analysis. In contrast to supervised models, unsupervised ones are described by Doshi et al. (2022) as ML techniques that deal with unlabeled data. In these cases, the data does not have labels and it is being used to discover patterns. Clustering methods are the main way of finding patterns in these cases.

Doshi et al. (2022) describes the reinforcement learning category as consisting of models that are focused on learning through exploration and exploiting current knowledge. The aim for models

and algorithms in this category is to learn the environment they operate in and learn the best way to reach the given goal. For example, a Q-learning algorithm learns on a reward basis. Each action has a defined reward value, and by choosing and performing an action the model eventually learns which action sequence brings the best rewards. This eventually leads to the desired outcome by structuring the reward structure in such a way that it rewards wanted actions and punishes unwanted actions.

Artificial neural network (ANN) category is one of the exceptions of algorithms that can be used to solve issues ranging from classification to regression to clustering. Kattan et al. (2011) book describes an ANN as a series of artificial neurons that transfer data to each other. This kind of operation is akin how a brain operates. Hence making it flexible solution to multiple different problems. According to Mattmann & Penberthy (2020), Kattan et al. (2011) and Hagiwara (2022), the ANN category includes algorithms such as recurrent neural network, feed-forward neural network, LSTM and natural language processing. While this is not a comprehensive list, the applications for a brain like learning process' are vast. This also brings up the complexity of choosing a suitable model as each application of a neural network can have significantly different outcomes. This also highlights the importance of testing out multiple different models and algorithms to see which fit each situation the best. In figure 7 is a code example of an ANN model creation. This illustrates that while a basic model can be defined in a few lines of code, there is a seemingly infinite number of variations available as you can add and remove layers a seemingly indefinite amount. As each layer also contain multiple hyperparameters the number of variations increase.

```
# Define the model
model_dense = tf.keras.Sequential([
                      tf.keras.layers.Dense(20, activation=tf.nn.tanh),
                      tf.keras.layers.Dense(50, activation=tf.nn.relu),
                      tf.keras.layers.Dense(20, activation=tf.nn.relu),
                      tf.keras.layers.Dense(1)
                      ])
model_dense.compile(loss='mse',
              optimizer=tf.keras.optimizers.Adam(),
              metrics=['mae'])
```

Figure 7. Example of creating an three layer ANN model in tensorflow using keras

Due to the hype around AI and its applications there are multiple popular ways people are trying to take it into use. The services that have become popular include cases such as image creation (Midjourney), large language models (OpenAI ChatGPT, Google Bard, Microsoft Bing), AI generated content detection and AI generated content (Instagram filters, video, audio). As the technology develops and knowledge accumulates even automatic machine learning tool are starting to be available. In their book Sabharwal & Agrawal (2020) introduces the Google AutoML solution. This is a tool to automatically create every machine learning stage from data preparation to model selection to hyperparameter tuning to selecting evaluation metrics. While having tools such as these is very useful, it can also be dangerous as even unlawful applications of AI can readily be taken into use. It all depends how the technology will be used.

2.7 Over- and underfitting

In Xue's (2019) article ML algorithm overfitting and the problems caused by it are described. After getting good quality and quantity of data this problem occurs in the ML algorithm selection part. In cases where the chosen model for the algorithm cannot generalize accurately from the observed data it tends to overfit to the given dataset. This means that instead of learning the patterns in the data the ML algorithm learns the data itself. This in turn leads to poor performance on new data.

Jabbar & Khan (2015) explain how underfitting on the other hand causes the same result. Instead of learning the data like an overfitted model an underfitted model does not even learn the data but instead tries to generalize insufficiently.

In both cases the model requires additional tuning. As number of models to choose from and their respective hyperparameters increase, it can be a difficult challenge to find a well fitted model. Depending on the data used and the goal that is being achieved, the number of applicable models could be dozens. Trying to find one that fits the task in hand can be a long process and if the data that is being used consists of millions of rows the training and testing of an algorithm takes longer which in turn makes testing multiple different algorithms tedious and, in some cases, costly. This phase is commonly known as the hyperparameter tuning part, and it can be helped with methods such as Cartesian grid search (Ding et al. 2020).

In the article Ding et al. (2020) explains the process of where the machine learning engineers go through the case and selects and tests the algorithms that are best for the case and finetunes them to maximize the benefits. The hyperparameter tuning consists of examining key details of the machine learning model and its training phases. One of the most crucial parts of finetuning is how many times it is trained using the training data, these are called epochs. This is where overfitting and underfitting also need to be addressed. Figure 8 illustrates the possible outcomes of hyperparameter tuning and how it affects the model's behavior.

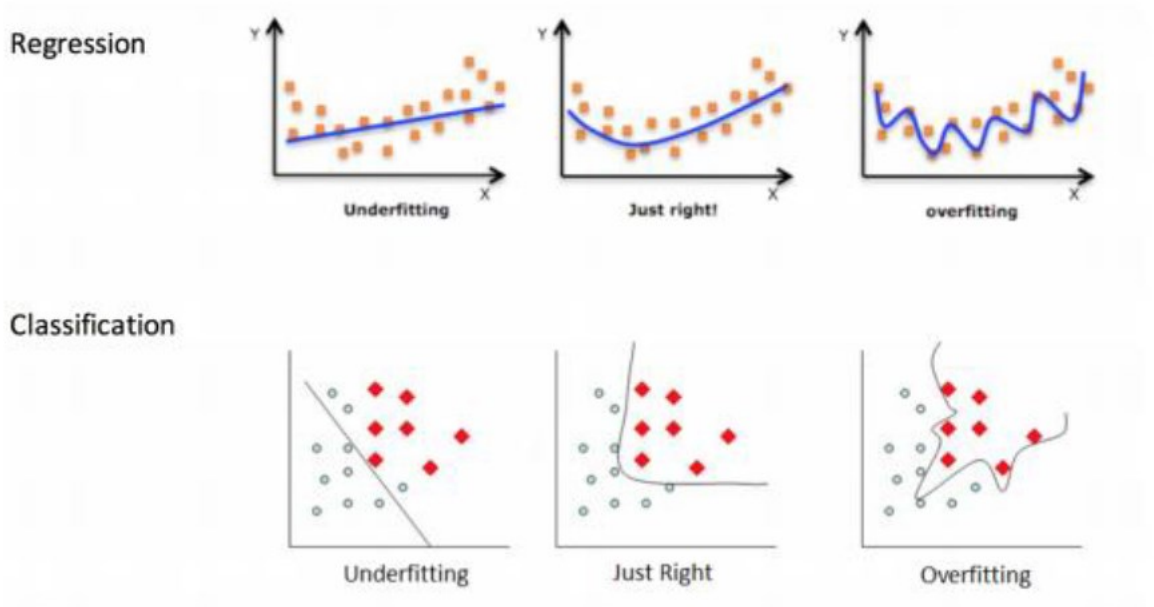

Figure 8. Under- and overfitting (Rajatheva, 2019)

2.7.1 Overfitting

In the article Xue (2019) explains how one of the possible outcomes of algorithm training is an overfitted model. Overfitting is the action of training an algorithm based on a training data set for multiple iterations until it memorizes the training data. This in turns creates an algorithm that cannot create accurate predictions or perform the wanted task with a reasonable accuracy on any other data. This is because unlike a well-trained model that has generalized and learned behaviors from the training data, an overfitted model is only good in identifying rows that are identical to those in the training data because it has been trained to memorize the data by training it on the same dataset too many times. This is especially true in the case of ANN due to how they operate in

a more human-like way. The way to identify an overfitted algorithm is to test it on both the training data and the test data. Should the algorithm be overfitted it will perform significantly better on the training data than on the test data. In classification overfitted models can for example reach a hundred percent accuracy in training data while performing at an eighty percent accuracy on test data. In some cases, it might be hard to spot an overfitted algorithm if it performs equally well on both training and test data. It will however become evident in new datasets as the performance tends to decrease.

2.7.2 Underfitting

Jabbar & Khan (2015) explain how underfitting is the other side of badly trained algorithms. Unlike an overfitted model that has seen training data too many times, an underfitted one has not been trained well enough. In these cases, the number of training epochs can be increased for the algorithm to better understand the data, but at the same time the number should not reach the territory of being overfitted. Like overfitting, underfitting also suffers from inaccurate performance. In this case however it can also be a case of a bad model as the results tend to be bad on both training and testing data or not giving the model enough epochs of training to reach a point where it can accurately create generalizations from the data. Even though increasing the count of epochs is relatively easy, depending on the model and data complexity it might be a time-consuming process to retrain or add epochs. It is usually better to give enough epochs to be closer to overfitting than underfitting as it tends to give better understanding of where the line is. In cases where each epoch is expensive in terms of computing power or time it takes it could also be taken into consideration to save the model after a reasonable number of epochs even during testing phases (Tensorflow, n.d.). This way you can continue training from the saved step and have a step to fall back onto in case it becomes overfitted. If even after the finetuning of epochs the models end up being inaccurate then the approach should be re-evaluated.

3 Research methods

The primary methods of research for the research question include both quantitative and qualitative methods. As most of the data that was required was already available and the research question was known, it led to approach of a case study as a research method. Priya (2021) introduces case study as a research strategy. The main objective of a case study approach is to investigate a

phenomenon in its context. As the data had already been gathered in an associated human resources and finances system, data gathering specifically for this question was not required. Rather extracting it from the system in a meaningful way and interviewing the subject matter and data experts was seen as a logical approach.

While the approach takes advantage of the Cross-industry standard process for data mining (CRISP-DM) model, it is not fully utilized. As the main objective was to find out whether the data is fit to the use case, the deployment phase was not utilized. However, the results could be used to proceed to the deployment phase. According to Martinez-Plumed (2021) the normal CRISP-DM process consists of the following steps: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. These phases are illustrated in figure 9.

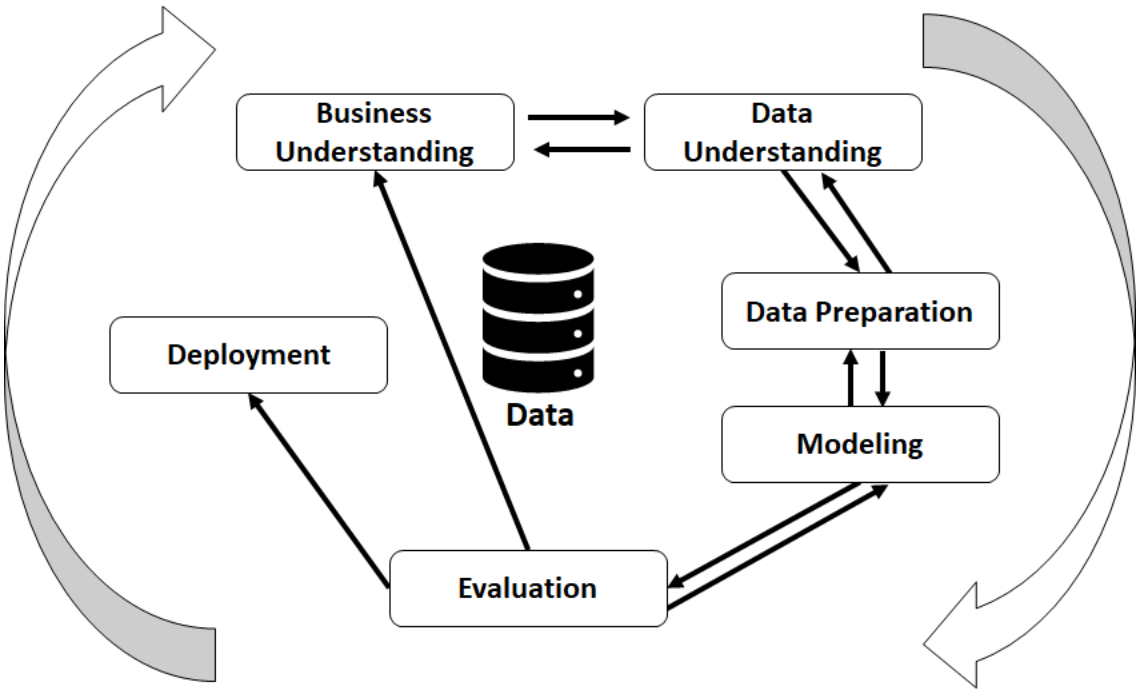

Figure 9. CRISP-DM data mining process (Sivula, 2021)

The data that was available had been collected in the associated system since 2016. This was used as the basis of the research. In the first sample that extracted, only the year 2022 data was present. This was later deemed inadequate, and it was further expanded to include last 4 years (2019 – 2022). The expanded dataset seemed to be an adequate sample size. The data contained infor-

mation about budgets, wages, and expenses, each in a separate file. This data was then be transformed into a single dataset that was be analyzed with Python and its extensions. The main tool used was a python extension called *Pandas* that is commonly used in machine learning applications (Leekha, 2021).

To gain business insight two subject matter experts were interviewed. This was done throughout the process as understanding of the data grew as the data was explored and as problems arose or further investigation was needed. As domain expertise helps to understand the data and its behavior and outliers it brings benefits to the EDA process (Yang et al., 2023). The business knowledge was then further transformed into data knowledge with an expert that was familiar with the data itself. This knowledge was also useful in working towards a technically compatible data.

The data was analyzed using EDA methods to understand the contents of the data. Martinez et al. (2017) describes the process as data examination without any pre-existing theories on how the data can be used to address the issue that is being examined. This was combined with a technical analysis to make sure that each of the data subsets were compatible with each other. These operations and processes made sure that a usable dataset was reached with as much of the data as possible. The net result for the four years was around 31 000 rows. This was the number of rows that were extracted, summarized, and curated from the full dataset as the result of EDA. While the combined row count for all datasets was roughly 842 000, the complexity of the budgeting process and the summarization reduced the actual dataset size to a fraction of the original.

After the data is understood well enough the CRISP-DM process proceeds into data preparation and in this phase the dataset is compiled into a singular one that has necessary attributes and can be used for a ML model (Sivula, 2021). As visualized in figure 9, data understanding and preparation can alternate as more data selection is done.

Sivula (2021) explains how these steps lead to the modeling and evaluation phases. In the modeling phase applicable models are chosen for the task and they are trained and assessed. From these models the best ones are chosen for the evaluation phase. In the evaluation phase the performance of each is assessed based on the criteria the business has determined.

Using these methods allowed for a streamlined process of getting from the data all the way to the resulting algorithm. In the book Leekha (2021) introduces multiple reasons for choosing Python as the language of choice and due to how current trend of doing machine learning applications heavily involve python as a coding language increases its value. Although other languages can be used as well and some of them may be better suited for larger scale analysis due to performance reasons, they were not an issue on this study. Since there are multiple machine learning libraries that can be used with Python and in this case even TensorFlow itself has them, there was no reason to deviate from it.

4 Background

The need for this ML implementation came from the organizations financial department as they had many different projects and funding subjects to keep track of and doing so manually would be a time-consuming task. In addition, doing calculations and predictions of whether the budget would be sufficient, would increase the workload by a considerable amount. To avoid the added manual labor, it was theorized that a ML model could be used to handle the task and provide monthly estimates on budget adequacy for the remainder of the year.

The results could be visualized on a report that would be easily accessible and quick to read through so that the appropriate actions could be taken as to oversee that the budget would be used accordingly. As spending tends to spread unevenly during the lifetime of the funding subject, using ML as a way of analyzing it could eventually make it learn the underlying human behavior as well. This would be harder to achieve with manual methods.

The result would be a ML model that takes data from the system monthly and creates predictions on the category where the tracked subject would be at the end of the year. These categories would be sufficient, needs attention and insufficient. The predictions would then be saved to a database and visualized with reporting tools. This would provide a tool for the financial department to increase efficiency on budget tracking.

5 Technology

Technology for this use case and environment were chosen to be open source and take advantage of cloud computing, specifically Microsoft Azure and their virtual machine services. This was because the organization was already using the cloud service, and it was the best way of tackling the project as the security issues were already taken into consideration by company-wide Azure policies. The server was placed in Azure Europe region as this was the preferred zone. Theoretically, the zone selection has little impact on the result but at the time of writing not all services were available in each zone. For example, a zone in Sweden was up and running but offered limited capabilities, and due to regulations (local government and EU GDPR for example) the region needs to be in Europe.

As for the coding language of choice, python was used due to its popularity on ML applications (Leekha, 2021). The code was handled in Jupyter Notebooks as they offered a lightweight platform for python coding. The base language was extended using *Pandas*, *Scikit-learn*, *Numpy*, *Matplotlib* and *Seaborn* libraries to better handle ML related issues. The main ML platform was chosen to be Tensorflow, as the library and documentation is readily available, and it can be converted into a production ready environment flexibly.

5.1 Cloud Computing

Cloud computing is not a new technology by itself, since has been around for years in one format or another (Surbiryala & Rong, 2019). However, as data centers grew and it became more accessible, it also started gaining popularity over the years. In the recent years, the Finnish government has been increasingly approving the usage of cloud services and computing. For example, the Ministry of Finance (2018) has published a document that takes a stance on the usage of cloud resources for the public sector. It was published in December 2018 and has been used as a baseline for cloud usage in many government offices since. Change, however, is not as fast as some government offices still consider cloud services as insecure or prefer to use self-hosted services. There is currently also ongoing debate about storing classified information in the cloud, and even some of the offices that do use cloud services do not bring data or services that handle security class IV data. While the current law (Lantto & Paatero, 2019) permits the storing and handling of data in the European security zone it does bring up security issues such as where to store encryption keys,

that need to be solved before the necessary precautions are taken into accord and the law is abided. Due to the law being worded as safe encryption method need to be used, it leaves room for interpretations and some government offices might not agree that cloud services can be safely encrypted or used. In this study it was not deemed as an issue due to the data not being sensitive.

5.2 Microsoft Azure

Microsoft Azure (n.d.-b) is a cloud computing service that offers a wide range of services for enterprise purposes. These vary from infrastructure as a service (IaaS), platform as a service (PaaS) to software as a service (SaaS) features. Examples include virtual machine services, cloud storage services, database services and networking services. New services are being added as the platform evolves.

The server that was chosen was a standard machine with 4 CPUs and 16 GiB ram with an Ubuntu Linux distribution. It was deemed as sufficient through performance testing during the data exploration and alteration and ML algorithm training processes. At the testing phase it also performed sufficiently so increasing the capacity would have just increased the costs while bringing minimal benefits. The server was isolated in its own Azure resource group and connections were limited to only allow a few select IP addresses from which the development operations were done.

Since the amount of data was quite small it was decided that a cloud SSD drive would be sufficient for the storing of the data. This way backups could be achieved by just taking a backup of the virtual machine itself and would not require any additional setup. As for production use however, it would be wise to separate the data to its own Azure storage account so that if the something happens to the server the data would remain on the storage account. It would also make it easier to move data to the service as you wouldn't have to move it directly to the machine itself but instead to the storage which could in turn be mounted on the virtual machine to enable more fluid usage. The results could also be stored on the storage account so that it would be easier to move for further development such as visualizations.

5.3 Python

For the base of the project Python was chosen as most machine learning tools are built upon it, which in turn made it a reliable baseline coding language (Leekha, 2021). Tools used consisted of software such as Jupyter notebooks and Tensorflow. In addition to these additional libraries that were used consisted of *Scikit-learn*, *Pandas*, *Numpy*, *Matplotlib* and *Seaborn*. These tools are quite commonly used so it is a good sign of stability and easily available material in case issues are experienced.

Pandas (n.d.) is an open-source data analysis tool for Python. It fulfils the needs to read data from files, to altering the data, to presenting and further developing the data contents. It creates dataframes that are like database tables. It is very similar to what you would get from using SQL language and a database but in a lighter format and without requiring a database itself. Lighter format in this case meaning that it has similar functions to SQL language but does not offer all the same tools. Pandas is generally used to handle and modify the data used in machine learning applications before the data is fed to a machine learning library or software.

Numpy (n.d.) is a scientific computing package. It provides tools to handle multiple different operations on arrays such as mathematical, logical, shape manipulation, sorting among others. It is a powerful library to compliment *Panda's* operations to create a dataframe that is optimized to each use case.

Scikit-learn (n.d.) is a machine learning toolkit for python. It consists of tools for predictive data analysis, such as regression, classification, and clustering models. One of the most common usages in ML model processes is creating test and train sets from a *Pandas* dataframe. These sets in turn are used to train a machine learning model and test its performance respectively. The toolbox offers a plethora of other tools as well and is a useful addition to any ML toolkit.

Matplotlib (n.d.) is a large visualization library that is good for multiple general use visualizations. It is a flexible tool to build graphs and plots using Python.

Seaborn (n.d.) is a library that visualizes data. It is based on Matplotlib and is works on Python. It is a multipurpose library and can make visualizations such as heatmaps and scatterplots. This library

extends the Matplotlib library by bringing in other visual tools that the base library does not necessarily have.

5.4 Jupyter notebook

Jupyter (n.d.) notebook is a notebook authoring application. It is a part of the Project Jupyter which in turn is a project that offers multiple different applications for different use cases. In this study only the Jupyter notebook was being used. In this context a notebook is a shareable document that can contain code in multiple languages, plain text, charts. It was used as a web-based UI to handle development on the Azure server as it was a lightweight and flexible way of handling machine learning application development. This also made it easier to split the operations into code blocks and test and develop each one of them separately. Notebooks can also be saved and exported out from the application in case they need to be backed up at a different location or in case the server needs to be changed.

5.5 Tensorflow

Tensorflow (n.d.) is an end-to-end platform for machine learning that is open source. This platform is developed and maintained by Google. It has extensive documentation and examples so even for a newer ML engineer there is enough material to get started. There are also a few tutorials that fetch a publicly available datasets and goes through the process of setting up a machine learning model from scratch. This makes it a solid platform for a case study such as this.

6 Security

To make sure that the information security of the implementation was secure and that the data that was being used was within ethical standards, General Data Protection Regulation (European Parliament & Council, 2016) and the upcoming EU standard for AI solutions (European Commission, 2021), multiple meetings with the information security officers and technology team were held. The boundaries that were set were that the data had to reside only on a secure server and not be handled with a personal computer other than moving purposes. The data itself was deemed to be usable for this instance due to it having no personal information nor any way to link it to identifiable natural persons. For the verification purpose, a sample dataset of a few rows from

each of the three data sets were delivered to the information security officers whom in turn deemed it to be safe and ethical to use.

As for the technology that was chosen for this project multiple different approaches were considered but as for the SaaS products that were available in the cloud (within Azure or third party provided cloud services), were deemed to be insecure. The issues arose from shared computing resources, which meant that the data being used could be compromised for example in cases where there are leakages in the memory of the shared computational devices. In the case of a self-created server, you can secure it better with your own encryption key and the resources are dedicated to your usage so there is no fear of data leaking from memory to other people using the shared computational power pool. Azure generated encryption keys were deemed to be safe and they were utilized.

As for other safety measures, the access to the server was limited to a select few IP addresses that were being used for the development and SSH and HTTPS with TLS 1.2 were being used to connect to the resources. TLS 1.3 was not available at the time. The server was encrypted using Azure provided security measures and the Azure resources used were only accessible to the organizations cloud service admin users. The service was also protected with general security policies that apply to resources in the cloud by the organization, but they are not disclosed due to them being out of the scope of this study.

7 Ethics

Ethics in services that utilize AI have been a heated topic for many years. Even in 1942 when Asimov wrote his science fiction short story Runaround, he mentioned the three laws of robotics (Haenlein & Kaplan, 2019). These laws were mainly aimed at preserving human life while also protecting robots' existence if it does not come at the cost of human life (Lee, 2020). This however is just the base level of ethical dilemmas. As the use cases of artificial intelligence increase so does the complexity of the issues. From simple prediction models to artificially generated voices and texts to self-driving cars. Each of these bring different set of dilemmas to consider. Who is responsible when a self-driving car gets into an accident? Who is responsible when an AI gives guides on how to create bombs? Should a large-scale natural language model, such as ChatGPT, be censored in some topics or should it be free to give responses to any question it is presented with? Should

artificially generated voices be allowed to replace voice actors in their field? Can you use AI to replicate a voice of a dead person or a celebrity? These are some of the ethical dilemmas that are being discussed increasingly. This is also where regulation becomes a necessity to avoid using the technology to create unwanted effects.

7.1 Regulation

Considering the many directions the AI technology can go, from self-driving cars (Tesla, n.d.) to large language models (OpenAI, n.d.), face recognition (Baynes, 2018), to eventually self-learning robots (Hao, 2020), it is evident that regulation becomes a necessity. Lassila (2021) goes through the five main principles of AI ethics that are mentioned in multiple different research papers. The principles that are presented are: transparency, justice and fairness, non-maleficence, responsibility/accountability, and privacy. When these categories are considered the baseline of AI usage should mostly be beneficial to the wellbeing of the people and the planet. However, since different cultures and countries have different views on what is acceptable in each principal, the usage of AI tools could vary a lot based on where it is used. The proposed regulation from the European Union, the United Kingdom, and the White house each take their own stance on these principles.

As regulation is being done on multiple fronts it also has variance in each approach. Even though the approach has variance, the necessity for it has been well recognized. The European Union (European Commission, 2021) is approaching the issue by dividing AI tools into categories that either permits, permits usage within a set of rules, or denies usage completely. This approach seems to be aimed to allow leniency in the law by giving categories with examples to help categorize the cases without having the need for constant law changes. The United Kingdom's approach (United Kingdom, 2023) is more aimed to target the principles as to what artificial intelligence should have. In the white paper they include the following: safety, security and robustness, appropriate transparency and explainability, fairness, accountability and governance, and contestability and redress. This approach is described as pro-innovation and, unlike the European Union regulation, seems to be aimed more to create trustworthy artificial intelligence instead of focusing on where it is being applied to. The White house (2022) approach highlights five key points: safe and effective systems, algorithmic discrimination protection, data privacy, notice and explanation and human alternatives, consideration, and fallback. This approach seems to aim at making AI solutions transparent and with opt out possibilities while making sure that discrimination does not happen,

and malicious systems are made illegal. It would seem to be aimed at protecting the general populus and protected classes from harmful applications.

7.2 Application

Although there are many different angles that need to be considered whenever using artificial intelligence services, in this specific study many of them do not apply. Although one could categorize a tool that monitors budget usage as a surveillance tool, in the same way it can be viewed as a data analytical point of view to an issue where no individuals are being analyzed. Since the initial use case did not contain data that could have been used to identify a single person, nor did it directly affect a natural person's life nor have direct interaction with users, it falls into a minimal or no risk category on the European Commission AI regulation (European Commission, 2023). However as mentioned in the European commission website if the algorithm and its use cases are developed further, it might fall into another category. This highlights the importance of keeping up with rules and regulations that the fast-evolving field of AI is encountering.

8 Data

In the AI field data is the most important part of the process as the quality and quantity play key parts in what the algorithms give as output. Mattman & Penberthy (2020) describe this issue in their book. A few pictures or rows of data is in most cases not enough to teach a machine to do a task like classification or prediction. Now while the amount of data varies depending on the task difficulty it is generally believed that you need enough of samples of different situations so usually the more data is available the better. The training and processing time for larger amounts of data should also be taken into consideration since that is usually the thing that drives infrastructure costs up. Either by requiring more hardware or more processing time. It is hard to give general guidelines on how much data is needed for an accurate ML algorithm so it should be done on a case-by-case basis. This is also a make-or-break point as insufficient or inaccurate data will lead to inaccurate ML model (Yang, 2023). You cannot turn a field of weeds into a field of roses just by applying ML. In figure 10 is an example of an early version of Midjourney where it is illustrated what happens when insufficient data is used to train a ML model.

Figure 10. A popular meme of a handshake generated by Midjourney (Immortal benelovence, 2022)

In the picture illustrated in figure 10 the goal was to get the model to create a realistic picture of people shaking hands. While in theory the goal was achieved, the details are far from reality. In the figure the hands are missing fingers, have extra fingers, bend in unnatural ways or are otherwise flawed. This in turn represents how a model that has not been properly trained can achieve a set goal, but the result might not be what was intended.

In the updated model, meaning the algorithm has received additional training, version of the same meme, figure 11, the impact on the difference the additional training has achieved on the model can be seen. This demonstrates the difference between a model that has not seen enough data or is not a good fit for the given task and a one that has been well adapted.

Figure 11. Midjourney version 5 handshake image (Lozmosis, 2023)

In figure 11, the hands are better defined, have all five fingers, and seem to be attached to bodies. It is closer to reality than the previous version. By ensuring that the algorithm is well trained, and the data used in the training is of high quality and it fits the purpose, the results should be of a similar quality model. This further solidifies the fact that you cannot turn bad data into a high-quality ML model.

8.1 Case data

The data used came from a Finnish national financial and human resources management system called Kieku. It is being broadly being used by Finnish government officials and it came into use in 2016 (Ronkainen, 2016). In case this study created a successful algorithm, it could potentially also

be used in many other government offices. This however is dependent on the ways that other government branches use the system. Also, since State Treasury is specializing in the field of government money flow, it could be theoretically useful on a grander scale. This however is speculation, but this study could pique the interest on their behalf.

The data that was being used consisted of wages, expenses, and budget. Some of the organizational data can be found in tutkihallintoa-website (Valtiokonttori, n.d.) and other data can be requested from the organization. They were compiled into a single dataset that was summarized to a monthly basis. This was because the period of which expenses were marked was retroactively after the month ended, wages were marked once per month and the financial department preferred once a month updates. The data contained all months of the calendar year but did not contain rows marked as internal budgets as they were removed from the data before the data was transferred to this usage. They were for internal usage and not necessarily reflecting the actual usage. These in turn did not help with the learning part as the actual outcome is more useful for learning purposes.

In the data were a few things that affected the accuracy of the predictions such as unpredictability on spending and retroactively updating budgets. The retroactive updating hindered the history data in a way that made it not reflect the events that happened in the past fully. This meant that if during the year the budget was increased, it overwrote the budget on the historical data. To get access to the changes in budget, a process would need to be created to capture the changes and store the data separately. The unpredictability in spending can be tackled with ML as through enough samples a proper algorithm should be able to learn the underlying behavior patterns. The complexity of the funding processes also created a problem of its own. The main issue was that the funding comes in multiple different forms. For example, the funding can be for a certain process, project, law, or office. In some cases, the budget is requested retroactively from the Ministry of Economic Affairs and Employment of Finland. These cases were removed from the final dataset due to not having a goal that can be tracked, which in turn made the data set smaller. In addition to the budgeted money issues, wages and expenses were, in some cases, tracked on a different level than what the funding was assigned to. For example, wages can be on an office level and the budget is on a project level. These in turn makes it harder to bring the machine learning process to an equal level on all sources of funding.

The data was split into three different files. These files each had an organization code, business accounting code, funding account code, two tracking codes, a project code, year, month, and the sum of the transaction. These three files were combined into a single dataframe using *Pandas* and summarized into a year and month basis. As for the hierarchy the levels were as follows: Organization code (*toimintayksikkö*) was the highest level followed by business accounting code (*LKPTili*), tracking code 1 (*seurantakohde_1*), tracking code 2 (*seurantakohde_2*), funding account code (*rahoitusrivi*) and project code (*projekti*). A sample and a row count are illustrated for each dataset in figure 12.

```
Jakopalkka

  Toimintayksikkö     LKPTili Rahoitusrivi Seurantakohde_1 Seurantakohde_2  \
0      3800191205  0041000100      3201024             NaN             NaN

        Projekti Kalenterivuosi/kuukausi    Summa
0  3800M-T002011                 201909  6171,84
Rows: 278298
-----
Tapahtuma

  Toimintayksikkö     LKPTili Rahoitusrivi Seurantakohde_1 Seurantakohde_2  \
0      3800261006  0093000001       320129             NaN             NaN

        Projekti  Tilikausi  Kirjauskausi   Summa
0  OH320S1410004       2019             4  198,68
Rows: 558641
-----
Budjetti

  Toimintayksikkö LKPTili Rahoitusrivi Seurantakohde_1 Seurantakohde_2  \
0      3800151401     NaN      3510631             NaN             NaN

  Projekti  Tilikausi  Kirjauskausi    Summa
0      NaN       2019             1  1199801
Rows: 5009
```

Figure 12. Base data example and row count

In the first iteration and testing phase a dataset which consisted only of year 2021 was used. This was to test if one year of data would be sufficient to use in the training part. During the data exploration part and data combination, the count of usable rows dropped down to around 6000 rows. This was deemed to be too small of a sample size considering that it was simple to add more data to the set. The second set of data consisted of data from 2019 to 2022 so four years in total.

The summary for each data set is illustrated in figure 12. Adding more years to the data set resulted in a net of 30 789 rows of usable data. This was a better training set for the algorithm as it had more examples to go by. The result is illustrated in figure 13.

```
   Toimintayksikkö  Rahoitusrivi  Tilikausi  Kirjauskausi         pvm  \
0           380011           600       2019             1  2019-01-01

   Budjetti  Palkat  Kulut  kk_jalj  kk_budjettia_kayt  v_budjettia_jalj  \
0   7841.23     0.0    0.0       11                0.0           7841.23

   v_budjettia_kaytetty  pros_kaytetty  pros_jalj_koko  riittaa
0                   0.0            0.0             1.0      2.0
Rows: 30789
```

Figure 13. Combined dataframe

8.2 Exploratory data analysis

In the book Myatt & Jonhson (2014) explore the issue of data exploration. While there might be different terms to describe the same process, the concepts have similarities. EDA is an important part of the machine learning process as it is used both in getting to know the data and the substance while finding out what works and what doesn't. This knowledge is then used to decipher whether the data is usable in the intended way and to give insights to the data handler to make better use of it. For example, if you are given data about ice cream eating habits, the weather and deaths by drowning, you might be able to technically link the data together and create a machine learning algorithm that is able to tell when people drown after eating ice cream, but the result might not prove to be useful in real life applications as correlation does not necessarily mean a causation. Or maybe the data is not technically joinable, so even though the data exists the data cannot be used in conjunction with one another. This is also true when the amount of data is insufficient to be able to be used in a broad enough way for the algorithm to make sufficient generalizations. An example of insufficient data would be giving someone one X-ray picture of a cancer patient and one picture of a healthy patient. An untrained eye would most likely not be able to make the distinction between them and it would essentially turn the decision process into a coin flip. The process of getting familiar with the data is crucial as the outcome is heavily dependent on the quality and quantity of the data that the ML algorithm was trained with.

8.2.1 Theory

In their journal Shoresh & Wong (2012) explain how data exploration is a process to understand the events or patterns in the data. While this process can be sped up by using visualization tools and techniques such as graphs, the base level of understanding the row level variance is important as well. Jambu (1991) describes how by examining the given data with statistical methods, the importance of each variable and its behavior becomes better understood. Once the knowledge from the analysis accumulates, it becomes easier to pick the data apart and point out flaws and outliers. This information can then be visualized to further understand the behavior or explain it to others. Both operations should work hand in hand in any given dataset analysis. As Shoresh & Wong (2012) explain that after gathering knowledge on row level operation, it should be used to create visualizations that bring added benefits to the user by making the outliers and anomalies easier to spot. This way the individual and general patterns are easier to detect. Visualization usually tends to bring more information to light than looking at rows or tables. However, as the authors write the visualizations should be kept concise as to not burden the viewer. This can be achieved by observing a small subset of data or splitting the visuals into smaller entities and group it by context.

By understanding the data and how it operates the dataset can then be further altered to fit the case in question and form a better ML model. This phase is called feature engineering, and it aims to optimize the behavior of the algorithm and the computational power requirements (Mattmann & Penberthy, 2020).

8.2.2 Application

To achieve a usable dataframe the base data required some additional alterations. The first attempts to combine the data into a singular dataframe were done at a level that was thought would be the most precise one by using as many columns as common denominators as possible. However, it was soon found out that although the data structure on each file was similar, the contents varied too much to be able to combine them directly.

For example, the funding can be on a main organization level while the expenses are marked on a sub organization or even on a project level. In most cases budget is not used in the business accounting code (*LKPTili*) level. This in turn means that the rows on each file might not necessarily

match before the modifications. In addition, project level following was not possible since the applications for funding for most projects were being done afterwards. There were also multiple rows that had no budgets assigned to them (marked as zero) and these rows were dropped due to them being out of the scope of useful data. Another set of dropping rows that did not contain useful data was done to rows that were assigned to a virtual budgeting month that was not part of the twelve months in a calendar year and they had no useful information to the dataset.

To combat the issue of streamlining the budget to a baseline where the necessary columns were kept and the budget able to be joined with the rest of the data was discussed multiple times with the domain experts. After getting input from the domain experts the methods of getting the data into a useful and a combinable state was reached. At first dropping some columns, namely the second tracking code, project codes and business accounting code, was tested and while the dataframe ended up improving it was still quite unusable as the organization codes did not match on each dataframe. To combat this even further it was decided that the organization codes would be trimmed down to match physical organizational structure and that seemed to correct the issue. After these alterations the data was deemed to be good enough for usage.

To further improve the combined dataframe and make it usable for ML algorithms there were multiple other alterations that needed to be done as well. This is quite a common occurrence in data handling as source data might not be enough on itself for the ML use case or it might have some technical issues that need to be addressed first (Yang, 2023). In this case the additional alterations consisted of operations such as adding a date column from the year and month columns, dropping unnecessary columns mentioned in the previous paragraph and business account code, convert sums to a float number from the Finnish way of using comma instead of period as a decimal separator, and replacing period, that was used as a zero in some cases, from business accounting code. Doing this made it easier to combine each dataframe into a singular one for further calculations. After going through these modifications in the original dataset only around 6000 rows remained, but after adding more years to the data a row count of around 31 000 was reached.

8.3 Data alterations

In addition to the baseline data changes made above, additional improvements and additions were made to help calculations and create categorization. These included the following actions, adding the available budget for the year to each row (*budjetti*), adding a remaining count of months left in the year (*kk_jalj*), adding a sum of used budget per month (*kk_budjettia_kayt*), calculate the used budget (*v_budjettia_kaytetty*), calculate the remaining budget (*v_budjettia_jalj*), calculate the percentage of used budget for the month from the yearly budget (*pros_kaytetty*) and calculate the percentage of remaining budget from the whole amount (*pros_jalj_koko*). These columns were used to give more of an overview to the data for the ML algorithm and to calculate the supervised ML category. After the addition of these calculated columns, it was possible to calculate the supervised learning category (*riittaa*) and mark each month with different categories such as sufficient, attention required, and insufficient based on how much of a budget is left for the rest of the year. Since December was the last month of the year it required some extra calculations. The dataset was sorted by organization (*toimintayksikkö*), funding account code (*rahoitusrivi*), and date (*pvm*) to form timeseries for LSTM model specifically. As the ordering did not make a noticeable difference in the other models this was done to the base dataset to keep the results comparable. In Table 1 is a summary of all the added columns and how they were calculated.

Table 1. Calculated columns added to the dataset

Name	**Description**	**Calculation**
Kk_jalj	Count of months left until December	12 – Kirjauskausi (accounting month)
Kk_budjettia_kayt	Sum of budget usage for the month	Palkat + Kulut (wages + expenses)
V_budjettia_kaytetty	Sum of total usage of the budget	For January: Palkat + kulut

		(wages + expenses) Other months: kk_budjettia_kayt cumulative sum
V_budjettia_jalj	Sum of yearly budget left	For the first month: Budjetti – palkat – kulut (budget – wages – expenses) Other months: Budjetti – v_budjettia_kaytetty (budget – v_budjettia_kaytetty)
Pros_kaytetty	Percentage of used budget for the month	Kk_budjettia_kayt / budjetti (used budget for the month divided by budget) If budget equals to 0 then 1
Pros_jalj_koko	Percentage of remaining budget for the year	V_budjettia_jalj / budget (remainder of the budget divided by the budget) If budget equals to zero, then zero

After the calculated columns were added, the dataset was further altered for supervised ML usage by creating a category that was used for the prediction. This category was called sufficiency (*riittaa*), and it was determined by a relation of the months remaining and percentage of budget used.

Since the category was added manually to the data it was directly translated to numbers to avoid issues with ML processing. The categories used are illustrated in Table 2.

Table 2. Definitions for budget usage categories

Category	**Months**	**Definition**
Insufficient (category 0)	January to October	Budget usage is greater or equal to 100%
	December	Budget usage is greater than 100%
Attention required (category 1)	January to December	Budget usage is more than 70% and less than 100%
Sufficient (category 2)	January to October	Budget usage is less than 70%
	December	Budget usage is less than or equal to 100%

After defining and applying the categories to the data, it was further explored to see how well each category would be represented. The categories were visualized to a stacked bar chart using matplotlib library. This illustrates the problem of attention required category. While sufficient and insufficient categories are well expressed in the data, the third category remains to be quite a small part of the overall dataset. The number of rows itself presents a problem for the ML algorithms as due to it being very minimal. This makes it harder for the algorithm to generalize. This also raises the question of whether the data is accurate or do the issues mentioned in the data exploration chapter skew the data significantly to one direction or another. In figure 14 the number of rows is illustrated per month. The categories do not consider the length of each tracked budget.

They could be budgets for a month or two or for the whole year. As such some of the categories could be over- or under-represented.

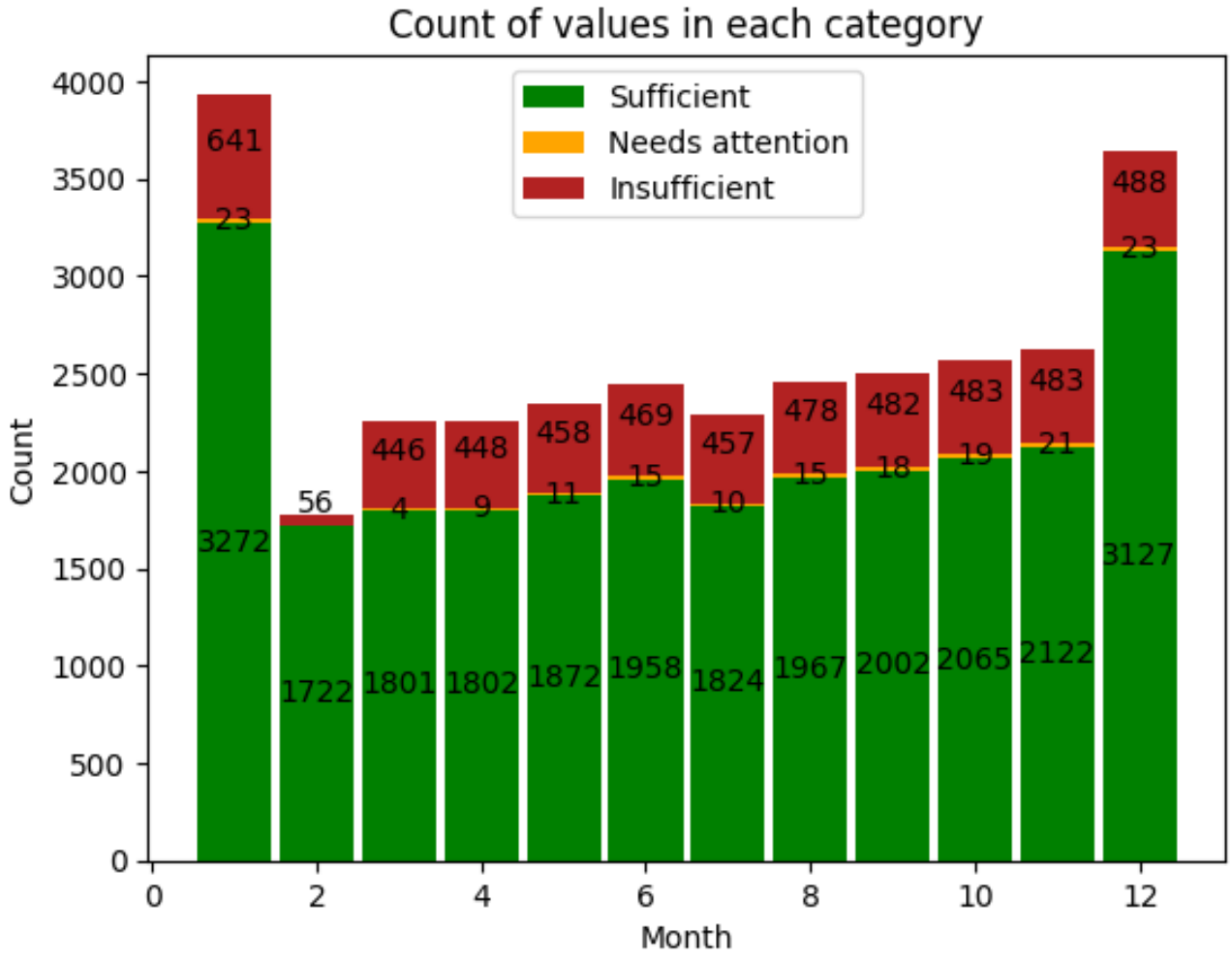

Figure 14. Count of values per category

To further gain insight to the data and how each column affects the outcome a heatmap was made. This was done by using the builtin *corr* method of *Pandas* in conjunction with *Seaborn*'s heatmap visualization. This produced an initial correlation matrix that was taken into analyzation. Figure 15 illustrates the correlation heatmap between each column.

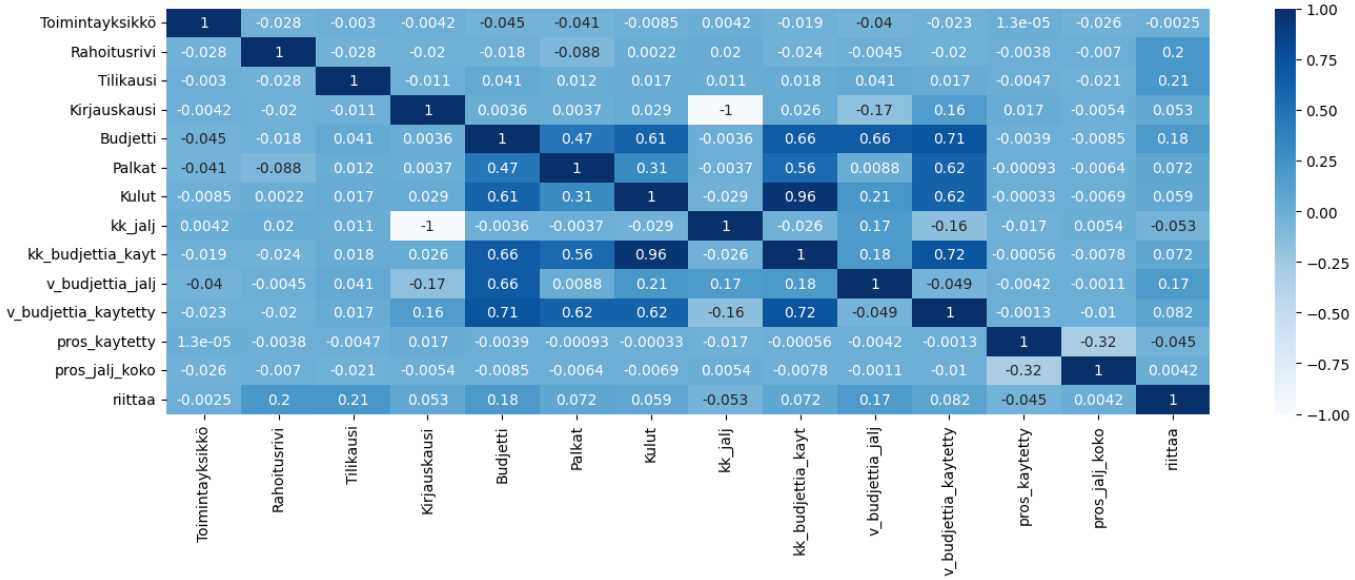

Figure 15. Heatmap of correlations between columns

The columns that correlated the most to the sufficiency column (*riittaa*) were the funding code (*rahoitusrivi*), year (*tilikausi*), and total budget remaining (*v_budjettia_jalj*). Other columns had a significantly smaller effect on the result.

Using the results from the first correlation matrix, dimensionality reduction was done to cut unnecessary columns and trim down the dataset to a more concise format. Figure 16 illustrates the effects of using the minimal number of columns that were discovered to have the most correlation to sufficiency.

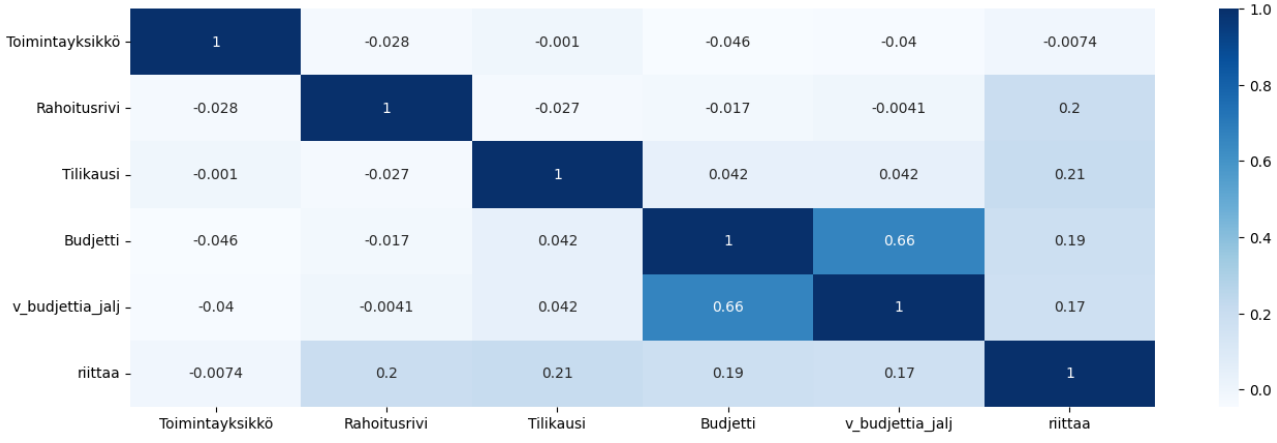

Figure 16. Heatmap after first iteration of dimensional reduction

At first glance the results seemed to be promising and the correlations from the chosen columns seemed to be more relevant. However, reducing the dimensionality this much created an issue of category 1 dropping from predictions as the class melded into others. This would mean that cases where interjection was needed, were no longer being predicted accurately by the models.

To combat the missing category, columns were added back to the mix but the categories that contained percentages were deemed to be overlapping information to the used budget columns. This prompted to drop the percentage categories from the dataset and form a third and final dataset that would be used in training of the ML algorithms. The correlation matrix from the third dataset is illustrated below.

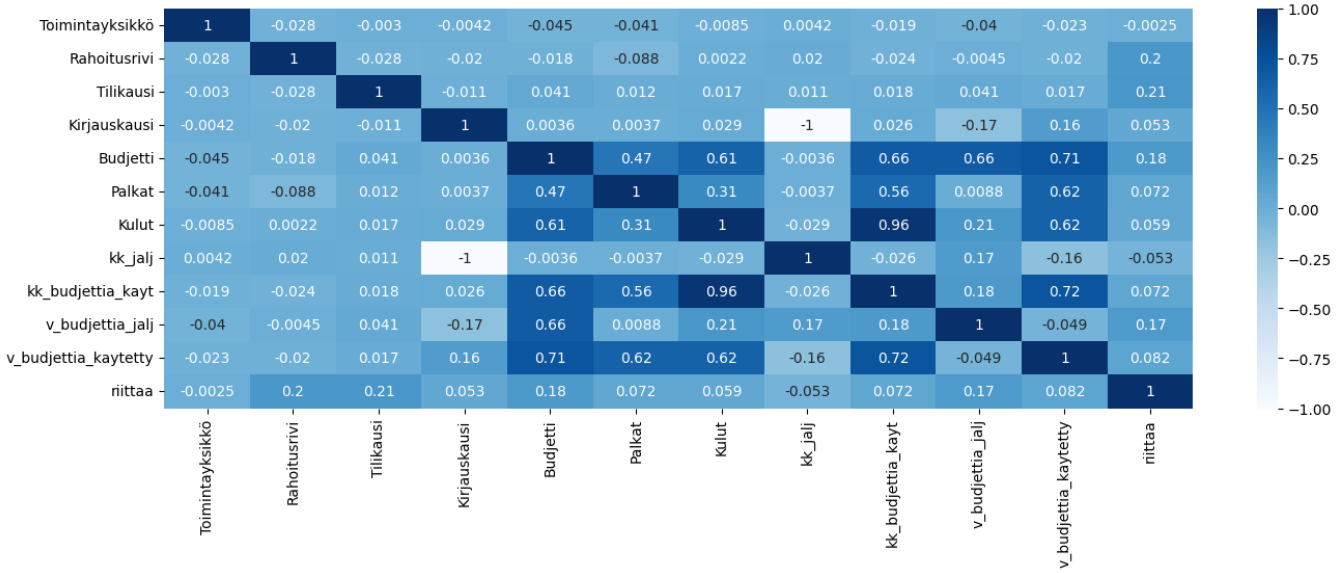

Figure 17. Heatmap after a second iteration of dimensionality reduction

By going through the dimensionality reduction, the final dataset was ready for ML algorithms. The dataframe that was derived from the operations is illustrated in figure 18. The key difference being the removal of categories that tracked percentages. This was done to reduce redundancy, but this also had a negative effect on models' accuracy. This is further explained in the next chapter.

```
   Toimintayksikkö  Rahoitusrivi  Tilikausi  Kirjauskausi         pvm  \
0           380011           600       2019             1  2019-01-01

   Budjetti  Palkat  Kulut  kk_jalj  kk_budjettia_kayt  v_budjettia_jalj  \
0   7841.23     0.0    0.0       11                0.0           7841.23

   v_budjettia_kaytetty  riittaa
0                   0.0      2.0
Rows: 30789
```

Figure 18. Dataframe after dimensional reduction

9 Applying machine learning

To find out which machine learning model would fit the best in this specific case, research was done to find out what kind of models were chosen in the financial field. In Rundo et al. (2019) research the writers compare multiple different models that have been applied to stock price prediction. Even though the primary objective is not the same, the operational field is similar and some of the models could be used in classification cases. The research also narrows down options models that might not offer good accuracy on the case. Using the same research as a baseline four models were tested on the case data. The models that were picked performed well in the research. These models were support vector classifier (SVC), LSTM, convolutional neural network (CNN) and random forest classifier (RFC).

The data used was split into training and testing datasets with the split of 75% into training and 25% into testing. It was also scaled by using *Scikit-learn*s standard scaler function to prevent bigger numbers from overwhelming smaller ones. Although the last chapter still shows the date (*pvm*) field in the dataset, it was dropped in this phase. The information it contained is also available in year *(tilikausi*) and month (*kirjauskausi*) fields.

The first model being tested was a support vector machine model (SVM). Since the aim was to classify the data into categories, the SVC model was specifically used. This model can be found in the *Scikit-learn* python package. The model reached an accuracy of 90% in the training data, and it reached the same accuracy in test data. This can vary a little depending on the train and test data split but even after multiple training rounds, the accuracy stays around 90%. It should also be

noted that category 1 data is scarce, so it seems to be quite a hard category to predict from the given dataset.

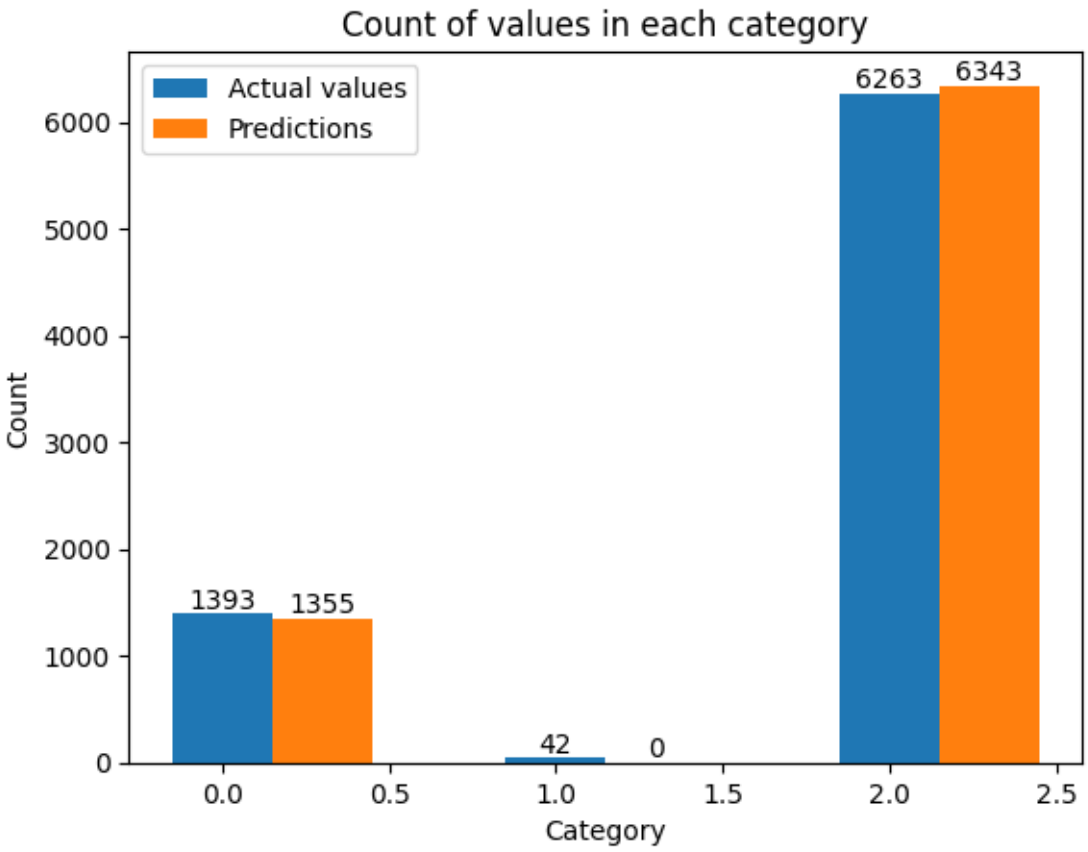

Figure 19. SVC model classifications

The second model that was tested was an LSTM model that predicts based on patterns in the past. The model was a simple 3-layer model with each layer having 100 neurons and an output layer with 3 different categories. While in general LSTM is a good model for time series prediction, in the dataset that was formed it seemed to underperform the other models. The reason for this behavior might come from the fact that although the dataset is technically a time series, it consists of multiple series that are not related to each other. To combat this the base dataset was sorted by organization, tracing number, and date. However even after sorting the data by time series that are related to each other and into a time series, an LSTM model might not be suitable for this kind of prediction. After 20 epochs the model reached an accuracy of 81% in train data and 80% in in test data. In figure 20 the inaccuracy becomes well illustrated. Category 0 is under predicted, category 1 is not being able to be predicted at all, and category 2 is over predicted. A model like this

would over emphasize the last category, making it seem that a significant count of budgets would last and thus not drawing attention to tracking codes that are about to exceed the given budget.

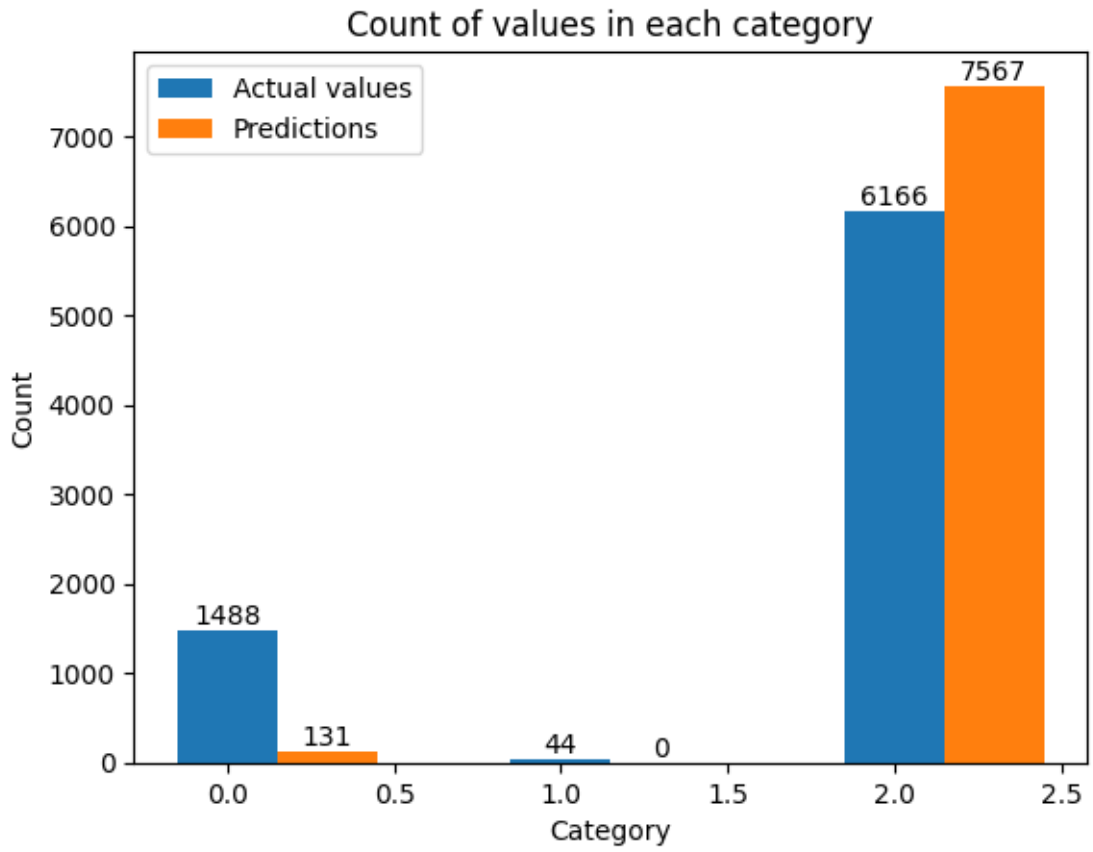

Figure 20. LSTM model classification predictions

The third model that was tested was a four-layer convolutional neural network model that used rectified linear unit as activation and had 100 neurons per layer. The last layer was a softmax layer with 3 outputs to match the specified categories. The loss function was sparse categorical cross entropy, optimizer was Adam and metric was accuracy. After training for 20 epochs the model reached 96% accuracy on train and test data. This model could be further optimized by additional hyperparameter tuning, but even without a significant amount of tuning the performance was already promising. Unlike LSTM and SVC, ANN could also predict category 1 rows with a reasonable accuracy.

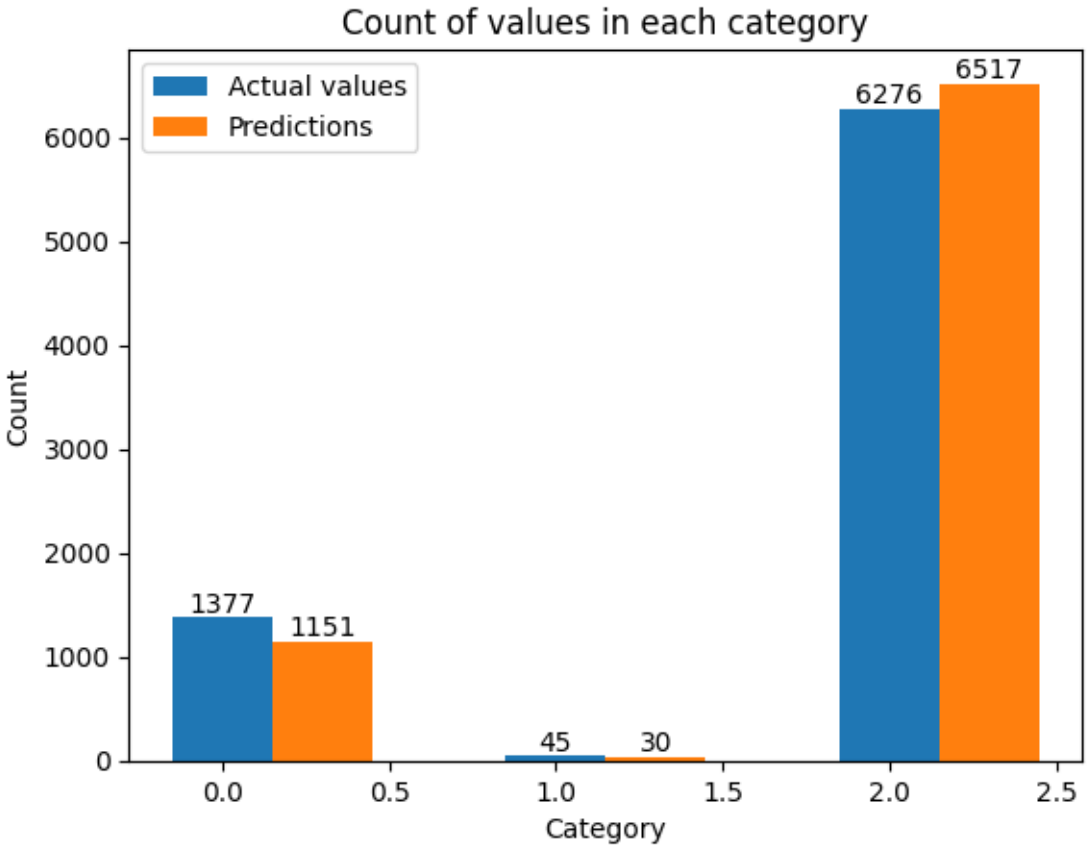

Figure 21. CNN model classification predictions

Fourth tested model was an RFC model. This model can also be found in *Scikit-learn* Python extension. The first iteration was done using default values and it produced an accuracy of 100% on training data and 99% on test data. From a surface look it seemed to be overfitted to the test data so further optimization was done. The optimization was done using *RandomizedSearchCV*-module. This module randomly searchers for optimal hyper parameters within the given model (Scikit-learn, n.d.). Even after the optimization the model resulted in similar results, giving 100% accuracy on training data and 99% on test data.

This could indicate that the data is somewhat insufficient to properly demonstrate each class, or the model is somewhat overfitted. However, the difference could also be minor enough not to make a noticeable difference in the predictions going forward. This would need to be verified in production usage to better understand if the difference is negligible or whether the model was overfitted. While the accuracy of categories 0 and 2 are quite close to 100% it seemed that category 1 is creating issues for RFC as well. This might be rectifiable by adapting the dataset to get

more accurate data. However, due to how the data is gathered it would require further development on the source of the data.

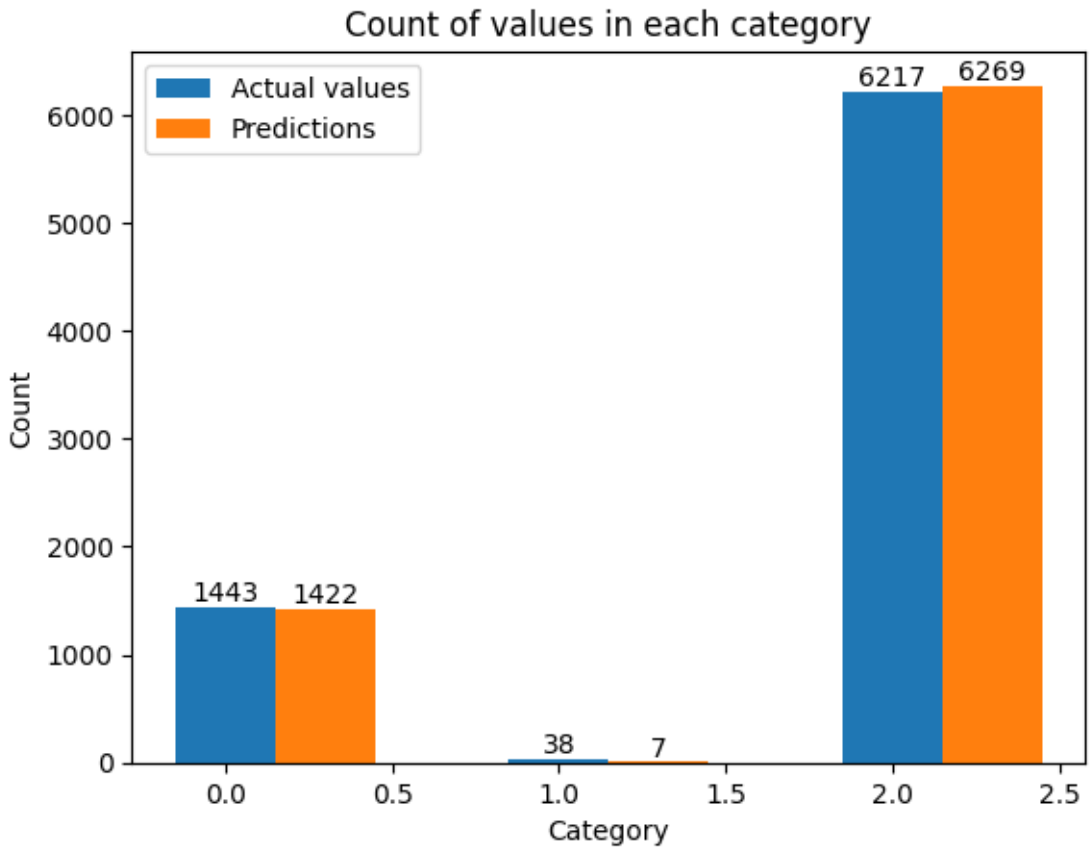

Figure 22. RFC model classification predictions

In Table 3 the predictions and actual counts are presented. In addition, the difference between the predicted and the actual counts are presented. A positive number represents that the category is under predicted while a negative value means that the category is over predicted. In most cases category 1 was the most difficult one to predict. This is most likely caused by low count of samples.

Table 3. Model test results

Model	**Category**	**Predicted count**	**Actual count**	**Difference**	**Overall test accuracy (%)**
SVC	0	1355	1393	38	90
	1	0	42	42	
	2	6343	6263	-80	
LSTM	0	131	1488	1357	80
	1	0	44	44	
	2	7567	6166	-1401	
CNN	0	1151	1377	226	96
	1	30	45	15	
	2	6517	6276	-241	
RFC	0	1422	1443	21	99
	1	7	38	31	
	2	6269	6217	-52	

Overall, RFC and CNN appear to be models that could be used to answer the research question. To verify that these models also work in production they should be run on live data and the predictions examined. In the final chapters the results and improvements are presented.

10 Results

From the models that were tested, RFC seemed to return the highest accuracy with the CNN being a close second. While RFC has superior accuracy to a CNN model it would be beneficial to run both models' side by side to test out if the accuracy stays on the long term. It would also be worthwhile to further tune the hyperparameters on the CNN model to test if the accuracy improves. As it is an arduous process it can take a significant amount of time so it might also require an evaluation of how much time should be spent on trying to optimize the model. Since the dataset used in this study was somewhat small, the training itself does not take a significant amount of time or computational power. However, model exploration and tuning the parameters per model is a time intensive process. This means that although the computational resources are not expensive, researching and testing different hyperparameters on a CNN model will eventually cumulate into hours, days, or weeks. At some point the improvements made will be miniscule. In Table 4 the accuracies of each model are displayed.

Table 4. Training and test accuracies on tested models

Model	**Training Accuracy**	**Test accuracy**
SVC	90%	90%
LSTM	81%	80%
CNN	96%	96%
RFC	100%	99%

As the day-to-day production data is different from the provided dataset, the accuracy and predictions should be monitored actively during the year to get a better understanding of whether the algorithm(s) performs as expected. Since the training data does not consider that budgets may have been altered during the year due to additional budgeting or other reasons, this could affect the prediction results of the chosen algorithm(s).

11 Additional development

Although the results look promising and seem to serve the business case, additional development should be done to further improve the situation. The core issues lie in the way the data is stored. Although budget changes are not stored separately in a database, they are available in a text format. For this study they were not being used but for future development and to improve the performance of the algorithm they should be used in conjunction to the history data. This way the training data could get a better generalization of the process. A better way however would be to track budget changes separately on the same database so that the training data that was used would be more accurate. As this data was not used in the training, taking it into usage could change the behavior of the algorithm itself. This also affects the categories that were added manually afterwards as the solution made in this study could categorize more rows into the sufficient category than there otherwise should be. Namely rows that had received additional budgets during the year. The effect from more accurate budget tracking should not massively change the outcome, but the chance it does, remains.

Another issue that was not addressed in the data is that while budgets could be thought of as a budget for the whole year, some tracked budgets could also be for only a few months. So, for example if a tracked budget would be marked for January and the project started in February and lasted to June, and it used all the funds assigned to it in June, then with the current category definitions this would fall into the insufficient category. While it is categorized correctly as insufficient for the rest of the year, it still is a misrepresentation of an actual correct case that could be categorized into the sufficient category as it would no longer be using the funds after June.

After implementing the algorithm and gathering the predicted rows and their predictions, a visualization tool such as Microsoft PowerBI would make the data easier to understand for the end users. In figure 23 is a simple example of report that shows the sufficiency (*riittaa*), year (*vuosi*),

month (*kuukausi*), organization (*toimintayksikkö*), and tracking number (*rahoitusrivi*) in a table format. Next to the table format is a stacked bar chart showing each sufficiency category per month. These visuals can then be filtered by the rightmost slicers of year, month, and sufficiency. While the visual elements, slicers and data organization can be further developed, even a simple illustration of how the data looks and the possibility of finding tracking numbers that are going over budget is a significant improvement.

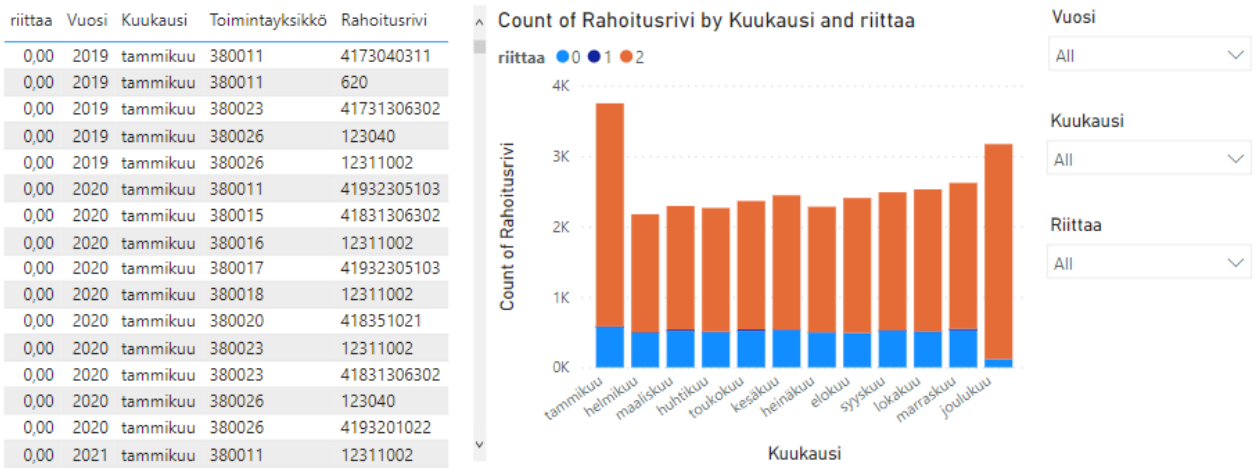

Figure 23. A simple example of an PowerBI visual based on the predictions

Other improvements should be discussed with the experts in the field and people involved in using the prediction data. Improvement ideas normally arise as processes reach maturity.

12 Conclusion

This study was conducted to answer the question "Can you use budget, wages and expenses data to create accurate predictions on whether a budget will last until the end of the year?". In the study the basics of what is AI and what is ML and how they are perceived in the current social environment were described. ML theory and its applications were introduced and how they could be applied in this use case. Technology, security, and ethics were introduced, and their applications were described. The study culminated in data exploration and alterations, testing of multiple ML models testing and describing improvements. After these steps the conclusions were reached, and they are described in this chapter.

While the data and ML models show promise of use there are still issues that need to be addressed or discussed before production use. Namely these are the data quality issues: verifying the definition of categories (sufficient, needs attention, insufficient), tracking budget changes and taking them into account to give better training data to the algorithm, and addressing budgets that are for smaller time frame than a year. After these steps are considered, it would be beneficial to run both CNN and RFC models' side by side for a year to see which one performs better on live data. Overall, the results from either model seem to offer a significant benefit to handling the budget control. If the budget change tracking issue is remedied, using ML would bring additional benefits as it can learn the process and underlying behavior on a more detailed level.

Overall, the application of ML seems to be beneficial in this case. However, the base data still requires additional development and the verification that the algorithm stays accurate and brings benefits to people handling the data takes at least a year. The faster the production usage can start the faster the results, benefits and the added value of ML can be evaluated.

References

AGI Workshop, Goertzel, B., & Wang, P. (2007). *Advances in artificial general intelligence: Concepts, architectures and algorithms: proceedings of the AGI Workshop 2006*. IOS Press.

Baynes C. (2018). Chinese police to use facial recognition technology to send jaywalkers instant fines by text. Independent. https://www.independent.co.uk/tech/china-police-facial-recognition-technology-ai-jaywalkers-fines-text-wechat-weibo-cctv-a8279531.html

Cacioppo, J. T. (2002). *Foundations in social neuroscience*. MIT Press.

Chhillar, D., & Aguilera, R. V. (2022). An Eye for Artificial Intelligence: Insights Into the Governance of Artificial Intelligence and Vision for Future Research. *Business & society, 61*(5), 1197-1241. https://doi.org/10.1177/00076503221080959

Ding, K., Lev, B., Peng, X., Sun, T., & Vasarhelyi, M. A. (2020). Machine learning improves accounting estimates: Evidence from insurance payments. *Review of accounting studies, 25*(3), 1098-1134. https://doi.org/10.1007/s11142-020-09546-9

Doshi, R., Hiran, K., Jain, R., Lakhwani, K. (2022). Machine learning: Master supervised and unsupervised learning algorithms with real examples. BPB Publications.

Du-Harpur, X., Watt, F., Luscombe, N., & Lynch, M. (2020). What is AI? Applications of artificial intelligence to dermatology. *British journal of dermatology (1951), 183*(3), 423-430. https://doi.org/10.1111/bjd.18880

European Commission (2021, April 21). *Proposal for a regulation of the European parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts*. EUR-Lex. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:52021PC0206

European Commission (2023, June 20). *Regulatory framework proposal on artificial intelligence.* European Commission digital strategy. https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai

European Parliament & Council (2016, April 27). *Regulation (EU) 2016/679 of the European Parliament and of the council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)*. EUR-Lex. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32016R0679

Google Market Summary (2023, October 5th). *NVIDIA Corp*. Google. https://www.google.com/search?q=google+nvda+stock

Haenlein, M., & Kaplan, A. (2019). A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence. *California management review, 61*(4), 5-14. https://doi.org/10.1177/0008125619864925

Hagiwara, M. (2022). *Real-world natural language processing*. Manning Publications.

Hao, K. (2020, March 2nd). *This robot taught itself to walk entirely on its own.* MIT Technology Review. https://www.technologyreview.com/2020/03/02/905593/ai-robot-learns-to-walk-autonomously-reinforcement-learning/

Hoang, D., & Wiegratz, K. (2023). Machine learning methods in finance: Recent applications and prospects. *European financial management: the journal of the European Financial Management Association, 29*(5), 1657-1701. https://doi.org/10.1111/eufm.12408

Immortal benelovence (2022). *Midjourney Reddit*. Reddit. https://www.reddit.com/r/midjourney/comments/yldngs/_/?rdt=51312

Jabbar, H., & Khan, R. Z. (2015). Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). *Computer Science, Communication and Instrumentation Devices*, *70*(10.3850), 978-981.

Jambu, M. (1991). *Exploratory and multivariate data analysis*. Academic Press.

Jupyter (n.d.). Jupyter. https://jupyter.org/

Kattan, A., Abdullah, R., & Geem, Z. W. (2011). *Artificial neural network training and software implementation techniques*. Nova Science Publishers.

Lantto, E., Paatero, S. (2019, November, 28). *Valtioneuvoston asetus asiakirjojen turvallisuusluokittelusta valtionhallinnossa 1101/2019*. Finlex. https://www.finlex.fi/fi/laki/alkup/2019/20191101

Lassila, S. (2021). *Designing the Trustworthy Principles of Artificial Intelligence - Case Finnish Police.* [Master's thesis, Laurea University of Applied Sciences]. Theseus. https://urn.fi/URN:NBN:fi:amk-2021110919576

Lavanchy, M. (2018, November). *Amazon's sexist hiring algorithm could still be better than a human*. IMD Business School for Management and Leadership Courses. https://www.imd.org/research-knowledge/digital/articles/amazons-sexist-hiring-algorithm-could-still-be-better-than-a-human/

Lee, R. S. T. a. (2020). *Artificial Intelligence in Daily Life* (1st ed. 2020.). Springer Singapore. https://doi.org/10.1007/978-981-15-7695-9

Leekha, G. (2021). *Learn AI with Python*. BPB Publications.

Lozmosis (2023). *Midjourney Reddit*. Reddit. https://www.reddit.com/r/midjourney/comments/11shla5/now_that_hands_are_better_heres_a_meme_update/

Mattmann, C. A., & Penberthy, S. (2020). *Machine Learning with TensorFlow* (Second edition.). Manning.

Matplotlib (n.d.). *Matplotlib: Vizualization with Python.* Matplotlib. https://matplotlib.org/

Martinez, W. L., Martinez, A. R., Martinez, W. L., & Solka, J. L. (2017). *Exploratory data analysis with MATLAB* (Third edition.). CRC Press. https://doi.org/10.1201/9781315366968

Martinez-Plumed, F., Contreras-Ochando, L., Ferri, C., Hernandez-Orallo, J., Kull, M., Lachiche, N., . . . Flach, P. (2021). CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories. *IEEE transactions on knowledge and data engineering, 33*(8), 3048-3061. https://doi.org/10.1109/TKDE.2019.2962680

Microsoft (2023), *Microsoft and OpenAI extend partnership*. The Official Microsoft Blog. https://blogs.microsoft.com/blog/2023/01/23/microsoftandopenaiextendpartnership/

Microsoft Azure (n.d.-a). *Pricing – Cloud Services*. Microsoft Azure. https://azure.microsoft.com/en-us/pricing/details/cloud-services/#pricing

Microsoft Azure (n.d.-b). *What is Azure*. Microsoft. https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-azure/

Ministry of Finance (2018). *Guidelines for Public Sector on Data Communications Services*. Institutional Repository for the Government. http://urn.fi/URN:ISBN:978-952-251-982-5

Myatt, G. J., & Johnson, W. P. (2014). *Making sense of data I: A practical guide to exploratory data analysis and data mining* (Second edition.). Wiley.

Numpy (n.d.). *Numpy version 1.26.0 software documentation*. Numpy. https://numpy.org/

OpenAI (n.d.). *Advice and answers from the OpenAI Team.* OpenAI. https://help.openai.com/en/

Pandas (n.d.). *Pandas 2.1.2 software documentation*. Pandas. https://pandas.pydata.org/docs/index.html

Pichai, S. (2023, May 10). *Google I/O 2023: Making AI more helpful for everyone*. Google. https://blog.google/technology/ai/google-io-2023-keynote-sundar-pichai/

Priya, A. (2021). Case Study Methodology of Qualitative Research: Key Attributes and Navigating the Conundrums in Its Application. Sociological Bulletin, 70(1), 94-110. https://doi.org/10.1177/0038022920970318

Rajatheva, R. (2019). *Application and Business Use Cases of Machine Learning.* [Oulu University of Applied Sciences, Bachelors Thesis]. Theseus. https://urn.fi/URN:NBN:fi:amk-2019052010663

Ronkainen H. (2016), *Mikä ihmeen kieku?*, Finnish State treasury, https://www.valtiokonttori.fi/blogi/mika-ihmeen-kieku/

Rundo, F., Trenta, F., Stallo, A. L. d., & Battiato, S. (2019). Machine Learning for Quantitative Finance Applications: A Survey. *Applied sciences, 9*(24), 5574. https://doi.org/10.3390/app9245574

Sabharwal, N., & Agrawal, A. (2020). *Up and Running Google AutoML and AI Platform*. BPB Publications.

Scikit-learn (n.d.). *Software documentation (version 1.3).* Scikit-learn. https://scikit-learn.org/stable/

Seaborn (n.d.). *Seaborn statistical data visualization*. Seaborn. https://seaborn.pydata.org/

Shoresh, N., & Wong, B. (2012). Data exploration. *Nature methods, 9*(1), 5. https://doi.org/10.1038/nmeth.1829

Sivula, A. (2021). *Exploring Machine Learning in Higher Education Industry: Student Performance Prediction* [Master's thesis, Jamk University of applied sciences]. Theseus. https://urn.fi/URN:NBN:fi:amk-202105189164

Solow, R. M., & Lo, A. W. (2012). *Rethinking the financial crisis*. Russell Sage Foundation.

Surbiryala, J., & Rong, C. (2019). *Cloud Computing: History and Overview*. https://doi.org/10.1109/CloudSummit47114.2019.00007

Tensorflow (n.d.). *Tensorflow version 2.14 software documentation*. Tensorflow. https://www.tensorflow.org/

Tesla (n.d.). *Autopilot and Full Self-Driving.* Tesla. https://www.tesla.com/en_EU/support/autopilot

The White House (2022, October). *Blueprint for an AI bill of rights making automated systems work for the American people*. White House. https://www.whitehouse.gov/wp-content/uploads/2022/10/Blueprint-for-an-AI-Bill-of-Rights.pdf

United Kingdom (2023, August 3). *A pro-innovation approach to AI regulation.* United Kingdom government. https://www.gov.uk/government/publications/ai-regulation-a-pro-innovation-approach/white-paper

Valtiokonttori. (n.d.). Tutkihallintoa.fi. Tutkihallintoa. https://www.tutkihallintoa.fi/

Wang, Q., & Bu, S. (2020). Deep learning enhanced situation awareness for high renewable-penetrated power systems with multiple data corruptions. *IET renewable power generation, 14*(7), 1134-1142. https://doi.org/10.1049/iet-rpg.2019.1015

Xue, Y. (2019). An Overview of Overfitting and its Solutions. *Journal of Physics*, *1168*, 022022. https://doi.org/10.1088/1742-6596/1168/2/022022

Yang, B., Nazari, R., Elmo, D., Stead, D., & Eberhardt, E. (2023). Data preparation for machine learning in rock engineering. *IOP conference series. Earth and environmental science, 1124*(1), 12072. https://doi.org/10.1088/1755-1315/1124/1/012072

Yu, L.L. (2004). Efficient feature Selection via Analysis of Relevance and Redundancy. *Journal of Machine Learning Research*, 5, pp. 1205-1224.

Appendices

Appendix 1. SVC code block

```
#SVC
#https://randomresearchai.medium.com/svc-model-in-python-2d7b6d9434b4
#create machine learning model
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.metrics import accuracy_score
from tensorflow.keras.layers import Layer
from sklearn.svm import SVC


class argmax_layer(Layer):
  def __init__(self):
    super(argmax_layer, self).__init__()

  def call(self, inputs):
    return tf.math.argmax(inputs, axis=1)

#load file
df_joined = pd.read_csv('joined_csv_reduced_d.csv')

y = df_joined['riittaa']
x = df_joined.drop(['riittaa','pvm'], axis=1)

# Available Columns
#Toimintayksikkö,Rahoitusrivi,Tilikausi,Kirjauskausi,pvm,Budjetti,Palkat,Kulut,
#kk_jalj,kk_budjettia_kayt,v_budjettia_jalj,v_budjettia_kaytetty,pros_kaytetty,p
ros_jalj_koko,riittaa

scaler = preprocessing.StandardScaler()
x_scaled = scaler.fit_transform(x)

x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.25)

model_svc = SVC()

model_svc.fit(x_train, y_train)

train_pred = model_svc.predict(x_train)
test_pred = model_svc.predict(x_test)

print('Accuracy in train data %.2f' % accuracy_score(y_train, train_pred))
print('Accuracy in test data %.2f' % accuracy_score(y_test, test_pred))
```

Appendix 2. RFC code block

```
#RandomForestClassifier
# https://www.datacamp.com/tutorial/random-forests-classifier-python
#create machine learning model
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn import preprocessing
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
recall_score, ConfusionMatrixDisplay
from tensorflow.keras.layers import Layer
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn.neighbors import KNeighborsClassifier as knn

from scipy.stats import randint


# Tree Visualisation
from sklearn.tree import export_graphviz
from IPython.display import Image
import graphviz

class argmax_layer(Layer):
  def __init__(self):
    super(argmax_layer, self).__init__()

  def call(self, inputs):
    return tf.math.argmax(inputs, axis=1)

#load file
df_joined = pd.read_csv('joined_csv_reduced_d.csv')

y = df_joined['riittaa']
x = df_joined.drop(['riittaa','pvm'], axis=1)

# Available Columns
#Toimintayksikkö,Rahoitusrivi,Tilikausi,Kirjauskausi,pvm,Budjetti,Palkat,Kulut,
#kk_jalj,kk_budjettia_kayt,v_budjettia_jalj,v_budjettia_kaytetty,pros_kaytetty,p
ros_jalj_koko,riittaa

scaler = preprocessing.StandardScaler()
x_scaled = scaler.fit_transform(x)

x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.25)

model_RFC = RandomForestClassifier()


model_RFC.fit(x_train, y_train)

train_pred = model_RFC.predict(x_train)
test_pred = model_RFC.predict(x_test)

print('Accuracy in train data %.2f' % accuracy_score(y_train, train_pred))
print('Accuracy in test data %.2f' % accuracy_score(y_test, test_pred))

param_dist = {'n_estimators': randint(20, 500), #[20,50,100,200,400],
              'max_depth': randint(2, 20)} #[2,5,10,20]}
```

```
rand_search = RandomizedSearchCV(model_RFC,
                                 param_distributions = param_dist,
                                 n_iter=5,
                                 cv=5)

# Fit the random search object to the data
rand_search.fit(x_train, y_train)

# Create a variable for the best model
best_model_RFC = rand_search.best_estimator_

# Print the best hyperparameters
print('Best hyperparameters:',  rand_search.best_params_)

# Generate predictions with the best model
y_pred_train_opt = best_model_RFC.predict(x_train)
y_pred_test_opt = best_model_RFC.predict(x_test)

# Create the confusion matrix
cm = confusion_matrix(y_test, y_pred_test_opt)

ConfusionMatrixDisplay(confusion_matrix=cm).plot();

print('Accuracy in train data %.2f' % accuracy_score(y_train, y_pred_train_opt))
print('Accuracy in test data %.2f' % accuracy_score(y_test, y_pred_test_opt))
```

Appendix 3. LSTM code block

```
#LSTM
# https://pythonalgos.com/2021/12/31/long-short-term-memory-lstm-in-keras/
#create machine learning model
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.metrics import accuracy_score
from tensorflow.keras.layers import Layer

class argmax_layer(Layer):
  def __init__(self):
    super(argmax_layer, self).__init__()

  def call(self, inputs):
    return tf.math.argmax(inputs, axis=1)

#load file
df_joined = pd.read_csv('joined_csv_reduced_d.csv')

y = df_joined['riittaa']
x = df_joined.drop(['riittaa','pvm'], axis=1)

# Available Columns
#Toimintayksikkö,Rahoitusrivi,Tilikausi,Kirjauskausi,pvm,Budjetti,Palkat,Kulut,
#kk_jalj,kk_budjettia_kayt,v_budjettia_jalj,v_budjettia_kaytetty,pros_kaytetty,p
ros_jalj_koko,riittaa

scaler = preprocessing.StandardScaler()
x_scaled = scaler.fit_transform(x)
x_scaled_LSTM = x_scaled.reshape(x.shape[0], x.shape[1], 1)

x_train, x_test, y_train, y_test = train_test_split(x_scaled_LSTM, y,
test_size=0.25)

model_LSTM = tf.keras.Sequential([
    tf.keras.layers.LSTM(100, return_sequences=True),
    tf.keras.layers.LSTM(100, return_sequences=True),
    tf.keras.layers.LSTM(100, return_sequences=False),
    tf.keras.layers.Dense(3),
])


model_LSTM.compile(
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    optimizer="sgd",
    metrics=["accuracy"],
)

model_LSTM.fit(x_train, y_train, validation_data=(x_test, y_test),
batch_size=64, epochs=20)

train_pred = model_LSTM.predict(x_train)
test_pred = model_LSTM.predict(x_test)

train_pred = np.argmax(train_pred, axis=1)
test_pred = np.argmax(test_pred, axis=1)

print('Accuracy in train data %.2f' % accuracy_score(y_train, train_pred))
print('Accuracy in test data %.2f' % accuracy_score(y_test, test_pred))
```

Appendix 4. ANN code block

```
# Neural network

#create machine learning model
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.metrics import accuracy_score
from tensorflow.keras.layers import Layer


class argmax_layer(Layer):
  def __init__(self):
    super(argmax_layer, self).__init__()

  def call(self, inputs):
    return tf.math.argmax(inputs, axis=1)

#load file
df_joined = pd.read_csv('joined_csv_reduced_d.csv')

y = df_joined['riittaa']
x = df_joined.drop(['riittaa','pvm'], axis=1)

# Available Columns
#Toimintayksikkö,Rahoitusrivi,Tilikausi,Kirjauskausi,pvm,Budjetti,Palkat,Kulut,
#kk_jalj,kk_budjettia_kayt,v_budjettia_jalj,v_budjettia_kaytetty,pros_kaytetty,p
ros_jalj_koko,riittaa

scaler = preprocessing.StandardScaler()
x_scaled = scaler.fit_transform(x)

x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.25)

model_dense = tf.keras.Sequential([
    tf.keras.layers.Dense(100, activation=tf.nn.relu,
                          input_shape=(x_train.shape[1],)),
    tf.keras.layers.Dense(100, activation=tf.nn.relu),
    tf.keras.layers.Dense(100, activation=tf.nn.relu),
    tf.keras.layers.Dense(3, activation=tf.nn.softmax),
])


model_dense.compile(loss='sparse_categorical_crossentropy',
                    optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
                    metrics=['accuracy'])


model_dense.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=20)

train_pred = model_dense.predict(x_train)
test_pred = model_dense.predict(x_test)

train_pred = np.argmax(train_pred, axis=1)
test_pred = np.argmax(test_pred, axis=1)

print('Accuracy in train data %.2f' % accuracy_score(y_train, train_pred))
print('Accuracy in test data %.2f' % accuracy_score(y_test, test_pred))
```