# Enhancing insights into spending through aggregation with automated document clustering of a large-scale multilingual corpus

## Bachelor Thesis

Part of the Examination for the

Bachelor of Science (B.Sc.)

of

International Business Administration and Information Technology

at the University of Business and Society Ludwigshafen

by

Lisa Rebecca Mirjam Schmidt

Sternstraße 93

67063 Ludwigshafen am Rhein

| | |
|---|---|
| Date of submission: | 07.06.2022 |
| Company Supervisor: | Dr. Karthik Muthuswamy |
| Academic Supervisor: | Prof. Dr. Joachim Melcher |

# Contents

# List of Abbreviations

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **AI BUS** | SAP AI Business Services |
| **BoW** | Bag of Words |
| **CBOW** | Continuous Bag of Words |
| **CRISP-DM** | Cross Industry Standard Process for Data Mining |
| **CoE** | Center of Excellence |
| **ERP** | Enterprise Resource Planning |
| **GIL** | Global Interpreter Lock |
| **JSON** | JavaScript Object Notation |
| **KDD** | Knowledge Discovery in Databases |
| **ML** | Machine Learning |
| **NLP** | Natural Language Processing |
| **NLTK** | Natural Language Toolkit |
| **SEMMA** | Sample Explore Modify Model Assess |
| **SQL** | Short Query Language |
| **TF-IDF** | Term Frequency - Inverse Document Frequency |

# List of Figures

# List of Tables

# Listings

# 1. Introduction

## 1.1. Motivation

An invoice is a document recording the main information on a sales transaction. Usually, an invoice contains the unit cost, a timestamp, and payment terms. Other information, such as shipping terms, shipping adress, or discounts may also be included.

Apart from their use for tax records, tracking the inventory and legal protection, invoices are essential for a company's financial reporting. Invoices are the main source of information for controlling [**investopediaInvoices**], as they record the complete history of cash flow [**invoicesPurpose**]. In general, internal financial reporting aims to provide information about the health of a company, and supply means for improvement. This in-depth analysis of spending is a time-consuming task.

More than 4 out of 5 financial departments are "overwhelmed by the high numbers of invoices they are expected to process" [**manualInvoiceProcessing**]. Already swamped departments of course struggle with providing information on savings potential. A solution for processing large amounts of invoice data with minimal human interference is desirable. With analysis results, financial advisors have a factual base for recommendations.

## 1.2. Current Situation

An essential part of economic counselling is the assessment of spending for different company segments. Spending of a firm usually is written down in invoice documents, which have to be grouped to analyze cost types. The global market is estimated to comprise 550 billion invoices annually, but 90% are exchanged paper-based [**kochEInvoicingJourney**]. With modern technology, these paper-based or digital documents can be transformed into a structured or semi-structured format. According to expert estimates, unstructured data makes up for more than 80% of enterprise data [**structuredAndUnstructuredData**].

Companies can not utilize unstructured data to its full potential, as this data is not leverageable with tradtitional data analysis tools.

## 1.3. Research Questions

Which methods exist for clustering large-scale multilingual corpora? Which combinations of feature-selection methods and clustering methods return the most valuable information?

## 1.4. Outline

The introductory chapter briefly explains the motivation behind the thesis. Also, the current situation is assessed and research questions are presented. Lastly, it presents the structure of the thesis.

The following chapter about objectives and criteria gives a detailes task description, and established criteria. In the section 2.2 the process model is explained, evaluated and adjusted.

Chapter 3 gives a glossary of terms. Chapter 3 sheds light on the corporate environment with respect to the aspects of history, organization an technological aspects.

The following chapters follow the structure of the research model presented in 2.2.

Chapter 4 explains the process of selecting a dataset. It presents fundamental types of projects and data sources. The chapter explains the sourcing of the used dataset.

Chapter 5 explains the business understanding.

Chapter 6 explains the data understanding.

In chapter 7, the preparation of the data is explained. The process of data cleaning is presented, as well as different means for feature extraction.

Chapter 8 compares algorithms and alternatives for measuring distances.

Chapter 9 presents the evaluation of the result. The chapter evaluated the output from the previous steps.

In chapter 10, a conclusion is drawn, and a further outlook is given.

# 2. Objectives and Criteria

## 2.1. Detailed Task Description

The goal of the thesis is to add value to real business documents by aggregating expenses into clusters of similar cost types. The supplied document dataset consists of 150.000 invoices. The invoices contain information about the vendor, billing amount and a description of the goods. Valuable information for companies would be insight into the different categories of expenses and the corresponding cost. With traditional data analysis methods, the company's controlling departments cannot identify which expenses are similar in nature, if they come from different cost centers.

The task is to perform a full data analysis on the supplied dataset. The dataset is to be prepared for processing with established methods. An evaluation for different means of feature extraction, machine learning, model evaluation and visualization should be performed. With the evaluation a complete flow for the data processing should be presented. The result should be an added value to the dataset in the form of aggregated expenses.

## 2.2. Criteria

The task includes different criteria from a corporate perspective which need to be considered. The analysis should be performed utilizing only available resources, which are the student's company laptop and already available instances for SAP internal services. The dataset which will later be presented is only available to SAP employees, and only after an access request for a specific use case is approved. Therefore, the tasks need to be completed with special attention to data protection.

Additional quality criteria were established before the start of the work. Firstly, the solution should be designed according to industry standards. This also includes the choice for a fitting research model. Secondly, the source code is to be documented in such a

way, that an expert third party can understand the workings of it in an appropriate time. Thirdly, the design of the source code should account for existing hardware limitations and should be optimized computationally.

## 2.3. Research Model

To solve the task described in chapter 1.2, this paper employs the Cross Industry Standard Process for Data Mining (CRISP-DM) [**CRISPDM2000**]. This model puts forward a structure for conducting data mining projects. CRISP-DM was developed in 1996 by three companies, which are now the partners of the CRISP-DM consortium: NCR, DaimlerChrysler AG and SPSS Inc.

A poll [**CRISPDMPopular2020**] conducted amongst visitors of a data science project management blog found that almost half of all respondents employ the CRISP-DM process model. Followed by Scrum and Kanban with a 18% and 12% of the user share, CRISP-DM is by far the most popular. Other methods such as Knowledge Discovery in Databases (KDD) and Sample Explore Modify Model Assess (SEMMA) are also noteworthy alternatives, but are less popular that CRISP-DM, Scrum and Kanban.

Being the most popular model, CRISP-DM is not necessarily always the best fit for all data science projects. In the case of this particular research effort, CRISP-DM proved the best suit for several reasons. Firstly, the process model follows the natural intuition of project design for data science tasks. Evaluation has to occur before the deployment, the modelling needs to occur before the evaluation, the preparation needs to occur before the modelling, and an adequate understanding of business and data aspects has to be developed in the beginning of the process. All those elementary dependencies are reflected in the model. Secondly, the CRISP-DM model adresses the iterative nature of data mining. Fundamentally, the model is of circular nature, reflecting the fact that data science projects underly the premise of continuous improvement. After the deployment of one solution, monitoring can give insights which allow for deeper business understanding, triggering the start of a new circuit of the model. Another model which puts forward an iterative approach is KDD [**KDD**]. Thirdly, the model allows to adapt the order of its phases. The free choice of path is more favorable compared to KDD, which allows for loops, but has a fixed order [**KDD**]. Fourth, CRISP-DM has no special requirements
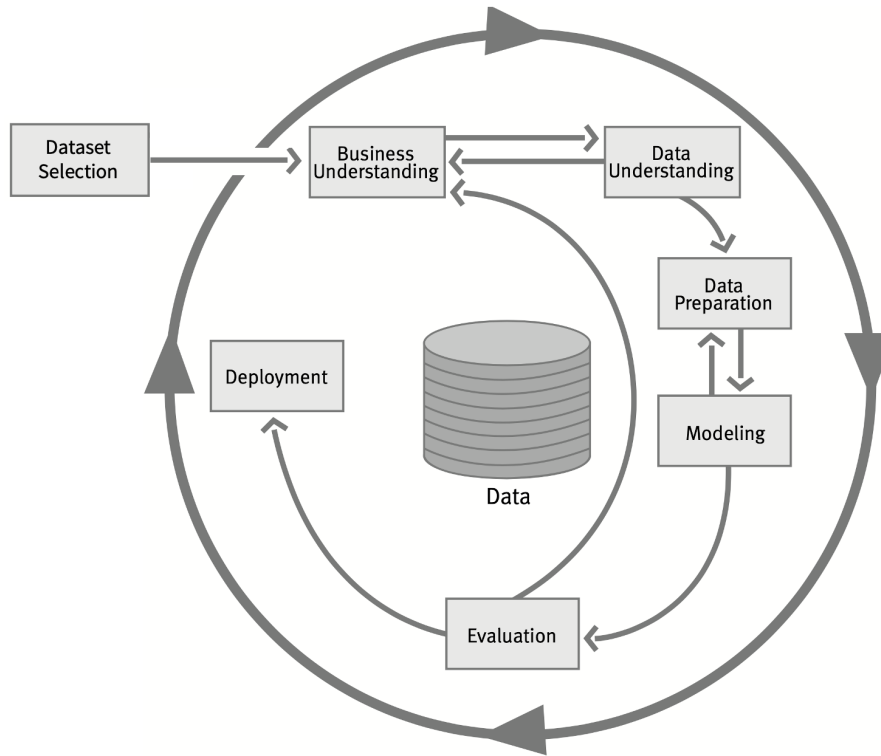
Figure 2.1.: Adjusted CRISP-DM Model

regarding team size or roles. Instead, a CRISP-DM project can be completed by only one person. This stands in contrast to SCRUM, which needs different roles represented by people in the team to work effectively [**SCRUMSolo**]. Fifth, the CRISP-DM model was publicized in the context of a 70-page guide with generic task descriptions and outputs for each phase. The detailed guide is especially valuable for teams with little experience.

Classically, the reference model consists of six phases. For this thesis the model was adapted to reflect all tasks encompassed. The phase "Dataset Selection" was added, resulting in a total of seven phases. The newly added phase includes the selection of a suitable dataset, the data retrieval, and the data provisioning. This results in a process model adapted to this specific project, and lays the groundwork for a successful undertaking.

# 3. Corporate Environment

## 3.1. Historical

SAP was founded in 1972 by five former IBM employees. The original company name was "Systemanalyse Programmentwicklung", which can be translated to "System analysis and program development". In 1976, a second company, the SAP GmbH was founded, where the acronym SAP denoted "Systems, Applications and Products for data processing" [**GeschichteSAP1972**]. The SAP GmbH is the company, which is today known as SAP SE.

Data processing being part of the company's name shows the importance of this field to SAP since the beginning of the company history. In 2017, SAP entered the Artificial Intelligence (AI) business with the SAP Leonardo Machine Learning Foundation [**rutschmannSAPLeonardo2021**]. SAP challenged the market for hyped products in the sectors machine learning, blockchain, big data, and design thinking. The name Leonardo refers to Leonardo Da Vinci, who is renowned for his interdisciplinary innovations [**schmitzLeonardo**]. SAP has the goal of driving the digital innovation strategies of customers with the help of SAP Leonardo.

With changes in the underlying hyperscalers for Leonardo, and evolving requirements of customers and partners, SAP adjusted their AI strategy. Two new products were introduced: SAP AI Core and SAP AI Launchpad. Both products are united under the collective name of AI Foundation [**rutschmannSAPLeonardo2021**]. With the general availability of AI Foundation in late 2021, SAP Leonardo is officially sunsetted.

## 3.2. Organizational

SAP SE has an executive board consisting of seven members, each attributed to one area. The AI division falls into the responsibility of Jürgen Müller, Chief Technology Officer and leader of the board area for technology and innovation [**JuergenMuellerBiography**].

Members of the organizational unit for AI are divided in different teams concerned with development, product success, operations and specific AI-services. Development Teams are organized into Centers of Excellence (CoEs), in which special expertise for designated areas is united. SAP has a team of researchers around the world concerned with state-of-the-art topics, including few-shot learning, sentiment analysis, privacy and fairness [**AIOverviewResearch**]. The research teams regularly publish articles on their advancements.

# 4. Dataset selection

## 4.1. Choosing a Dataset

The dataset is the fundament of a data-science project. The quality, size, and closeness to reality decide the helpfulness of findings made using the data. In this section, a classification for data-science projects is introduced as a guide for dataset selection.

### 4.1.1. Project Types

Table 4.1.: Fundamental Data Science Project Types

|  | Problem-First | Data-First |
|---|---|---|
| Systematics | Applied Data Science | Exploratory Data Science |
| Underlying Question | How can a problem be solved? | Which problems can be solved with the solution? |
| Role of the dataset | Different datasets can be considered for one problem statement. | The dataset is the core of the project, with a new dataset, a new project begins. |
| Requirements for the dataset | Require a ground truth or established methods for evaluating the goal achievement. | Compared to problem-first projects, larger datasets are required because there is no prior assumption of patterns. |
| Fixed component | Problem statement. | Dataset. |
| Innovative Aspect | Defined during formulation of the goal. | Found during the project. |

The problem-first project type is characterized by a predefined problem statement or research goal. The underlying question is how a specific problem can be solved

[**dataScienceProjectTypes**]. For the selection of the dataset, this requires that goal achievement can be measured with existing ground truth. In some cases, also other methods such as expert judgements can be sufficient. In a problem-first project different datasets can and should be considered. [**dataScienceProjectTypes**] give the metaphor of "mining for valuable minerals or metals at a given geographic location where the existence of the minerals or metals has been established". This means that it is already verified that the business goal can be reached with this specific dataset, hence the metaphor of already discovered mineral occurrence.

The complementary project type is the data-first project. This type is characterized by a more explorative approach, and the goal to find problems and patterns. Here, the dataset is at the core of the project and the fixed aspect of the project. In turn, this means swapping the dataset is the start of a new project. When turning to the metaphor of mining for minerals, this type of project would be the exploration of different test pits that promise mineral occurences [**dataScienceProjectTypes**]. It is not clear if a problem can be solved with the investigation, and problem to solve is not known.

## 4.1.2. Systematization of this Project

Usually, the project type is not decided on, but implicitly arises out of environmental parameters. This is also the case in this project. While there was no initial business goal defined, a dataset was chosen and agreed upon. The decision process is detailed in the next section, 4.2. The dataset being the anchor of the project, this data science project can easily be classified as a Data-First project.

## 4.1.3. Resulting Requirements for Selecting a Dataset

In a data-first project, there is no assumption of patterns. This for once means that a large dataset is required to cover enough ground for accurate derivation of insights. Second, a data-first project does not require structured data [**srivastavaDataMining**]. Instead, unstructured and semi-structured data is also applicable for data-first projects.

Table 4.2.: Comparison of Data Sources

|  | Internal | External |
| --- | --- | --- |
| Source | Internal or customer data, bought or generated | Publicly available, generated or supplied by companies |
| Relevance | Business-relevant | Anonymized, processed |
| Value | Relevant specific business | Of general relevance |
| Availability | Authorization processes in place, data protection measures | Ubiquitously available |
| Examples | Sales records, usage statistics, customer feedback | Historic weather information, consumer statistics, social media data |

## 4.2. Choosing a Data Source

With the project type as a decisive factor for the dataset explained, the other evaluation criteria for the dataset is described. In the corporate environment two fundamental sources for data exist.

### 4.2.1. Data Sources

Firstly, data can be sourced from inside the company. This can include customer data or data generated from observation and monitoring processes inside the company [**internalExternalData**]. Data is either directly or very closely related to the company's business. Because internally sourced data is of utter utility and a possible target for industrial espionage, internally sourced data is almost exclusively rated confidential, limiting even intra-company access to it. Authorization processes and more than often not existing registries for data may hinder project progress.

Secondly, data can be sourced outside the company. A vast number of online registries for data exist, both with paid and free of charge service offerings [**whyExternalData**]. Data sources include social media data, sensory data and weather data. Because of its publication, the data is sometimes stripped from all parts which could expose confidential

information such as corporate secrets. Additionally, data is anonymized for privacy reasons. External Data can give valuable insights into industries and markets.

## 4.2.2. Selected Data Source

It can be stated, that both sources are suited for different goals and different contexts. For data scientists with internal sources available, this type of source seems more appealing. The resulting data sets often are of higher quality and relevance. Of course, external data sets are important especially for independent data scientists without access to paid databases, or for small businesses without the option for an own data collection. An approach of connecting both internal and external data is also an option, but not within the scope of this project.

# 5. Business Understanding

In the guide complementing the CRISP-DM model, different tasks, and outputs for developing a business understanding are mentioned. The task and respective output will be dicussed in the following sections.

## 5.1. Business Objectives

Businesses without existing an Enterprise Resource Planning (ERP) solution in place can easily be overwhelmed by the number of invoices reaching them daily. Even more, the controlling department can easily lose the overview of spending. To quickly gain a perspective on the most important spending topics, spending should be sorted in categories of similar nature.

The primary goal is the development of a solution for automatic aggregation of documents, based on topics adressed in those documents. The focus is on shorter text segments, such as product descriptions. The business objective is an information gain, on how spending is distributed among cross-cutting topics in a company.

The created solution can be evaluated with the business success criterion: "Does the solution identify and give useful insights in the money pits?". The judgement of goal achievment is a subjective matter. Evaluating the success should therefore be distributed among several stakeholders, including but not limited to the author, the supervisor and the supplier of the data.

## 5.2. Assessment of the Situation

### 5.2.1. Inventory of Resources

An inventory of resources (A.1) was created for assessing the situation. Most notably is the availability of experts through excellent intercorporational cooperation. Also, a

large collection of datasets is available. Hardware platforms include personal machines as well as hosted environments with GPU capabilities. Available software are data science tools included in the Anaconda Navigator, such as Jupyter Notebook. All open-source libraries are of course also included.

## 5.2.2. Requirements, Assumptions, and Constraints

The project is to be completed the latest on June $7^{th}$ 2022.

Several assumptions underly the process of data mining. First, it is assumed that the descriptions of the invoices is speaking enough to identify the product referenced. Second, the analysis assumes that invoices can be logically grouped into clusters, in other words, several invoices refering to similar topics.

From a legal perspective, the project is constrained in the publication of data. While the use and processing of the supplied data is permitted within the context of the thesis, publication and further use is prohibited. The dataset is to be kept only on the local machine and SAP owned hyperscaler instances.

## 5.2.3. Risks and Contingencies

**Dataset too Large for Processing**   One specific risk that may arise in this project, is a dataset too large to be processed. Only a limited size of data is able to be loaded into memory. If the dataset turns out to be too big, four solutions are proposed.

1. The dataset can be compressed using either a lossy compression (sampling, truncating floating point values) or a lossless compression (choosing only specific columns, using a sparse-column representation or choosing efficient data types) [**largeDataSetMedium**].
2. Memory problems often arise out of computations that are not thought through. Most of the time, not the whole data needs to be in memory. Streaming the data or loading it progressively is a great option, if the algorithms permit this [**largeDataSetBrownlee**]. Programming languages often have built in lazy evaluation capabilities, such as Python's generators.

3. Another approach for storing and querying large datasets is the use of a relational database. The database can be queried using Short Query Language (SQL). Again, this option has the premise of Machine Learning (ML) algorithms permitting iterative learning [**largeDataSetBrownlee**].

4. Finally, using a platform for ML workloads, such as SAP AI Core helps with handling large amounts of data. This approach requires aligning of the source code to fit the specifications and endpoints of the platform.

**Messy Data**    Of course, data collected from operational sources is not perfect and ready to feed into an algorithm. Data cleaning is one of the main activities of data scientist's everyday workload. But what if the dataset is untidy to such a severe extent, that it is not able to be cleaned in a reasonable amount of time? Particular problems can be column shifts in tables, different units for values without naming the unit, or feature encodings without keys for decoding. Depending on the extent of the case, a different dataset should be evaluated. Also, the remark has to be made that this case is highly unlikely as the datasets have been used for other applications in the past.

**Inefficient Calculation**    Calculations can quickly grow into inefficient and obfuscated code, taking hours or even days to complete. To mitigate this risk the following contingencies are proposed:

1. Different methods for calculating the same operation should be identified, evaluated and implemented.

2. Regular benchmarking of smaller chunks of the data helps to extrapolate processing times and decide for one solution.

3. Implementations in libraries should be, in general, favored over self-made implementation. Literature research in industry-specific blogs helps to find even more efficient implementations than those contained in popular libraries.

4. A sophisticated design of data structures is crucial in utilizing optimized code. Even the most elaborate way of calculating operations can turn into a resource-intensive task if the input data is structured poorly. Considering different data structures and selecting the most appropriate one for each specific task is crucial.

**Results are of no Value**   The business objectives were determined in a prior section. But what is the procedure if the business objectives are not reached? Especially the criterion of giving useful insights is the crux. A simple laissez-faire attitude towards the definition of "useful" is unsatisfactory, although creating the illusion of a positive outcome of the project. With not reaching the original goal of a research project, the work does not automatically become useless. Identifying problems and causes for the failure can facilitate other research.

## 5.2.4. Terminology

To summarize both relevant data mining and business terminology, a glossary was compiled. This is not yet completed, as the glossary of terms will be expanded while writing the thesis.

**Clustering Algorithm**   A clustering algorithm is a sequence of instructions, which arranges a set of instances into groups, which contain items of high similarity to each other.

**Document**   A document is a unit of data most typically containing natural language text.

**Hyperscaler**   A hyperscaler is a company offering achitecture to adapt to changing workloads in cloud computing, networking and internet services.

**Natural Language Processing (NLP)**   NLP is often attributed to computer science, but after closer examination, NLP is a discipline comprised of linguistics, computer science, artificial intelligence an mathematics [**chowdhury2003**].

**Script**   A script is a small piece of code contained in a single file, most commonly written in a dynamic high-level programming language.

**SAP AI Foundation**   The SAP AI Foundation is a term describing SAP's offerings of the AI Launchpad and AI Core.

**SAP AI Launchpad**   The SAP AI Launchpad is a tool for the operation of AI content inside an SAP system.

**SAP AI Core**   SAP AI Core allows for training and serving AI scenarios [**schmitzLeonardo**].

## 5.3.  Data Mining Goals

The following data mining goals were identified during the phase of business understanding:

1. Identifying and applying appropriate methods for feature extracting tailored to this type of dataset.
2. Identifying and applying appropriate methods for clustering documents in this type of dataset.
3. Identifying and applying appropriate methods for topic modelling with this type of dataset.
4. Aggregating expenses by their clusters and visualizing the output.

The successful outcome is defined by reaching all named criteria. The achievment will be evaluated by the author and the supervisor, also people referenced in the inventory of resources will be considered to evaluate the outcome.

## 5.4.  Project Plan

### 5.4.1.  Project Plan

I don't think I will need this...

## 5.4.2. Initial assessment of tools and techniques

**Programming Language**   A multitude of programming languages for statistical analyses and machine learning applications exist. The most popular and best supported are Python and R. Both languages are open source and targeted at data science tasks. Following criteria will be used to evaluate the best fit for this project:

1. Understandability, as the availability of learning resources and the ease of reading and writing source code.
2. Availability of resources such as libraries, packages and modules.
3. Compatibility with existing solutions, availability of scaling and deployment options.

Python is a general purpose programming language with a straightforward syntax. Python is regularly taught at schools and suited for beginners. R is a language built by statisticians. Therefore, R is suited primarily for data-science tasks. While a data analysis can be created easily, more complex functionality requires experience and is considered challenging [**pythonVsR**]. For understandability, Python is rated as the more favorable choice.

For the availability of resources, R clearly scores. In general, Python has a large number of libraries available. Since python is not exclusively for data science, only a fraction of them are suited for statistical analyses. For R on the other hand, over 13.000 packages are available in the R archives [**pythonVsR**].

Deploying a python solution is easily possible via APIs, webapps and containerization technologies. R is also displayable on the web using specific packages. Both R and Python can be deployed in docker containers [**rDocker**] [**pythonDocker**] paving the way to be deployed on all major hyperscalers. It can be stated that python has superior ability to be scaled and deployed as it is a general purpose programming language.

**Programming Paradigm**   With the programming language decided, the basic programming paradigm has to be chosen. Python is an high-level object-oriented language, containing also functional aspects [**corePython**]. Python allows for programming using only scripts, but also supports and encourages the use of modules to separate files. For data scientist, the decision between modular code and code contained in a notebook presents itself in the beginning of every project. Python code in modules can be or-

ganized into several files calling each other when needed. This option requires careful documentation, as there is no graphical interface showing dependencies between the modules.

Python notebooks are structured to represent code, descriptions and the output. The most popular notebook format are Jupyter Notebooks, representing code and additional information as JavaScript Object Notation (JSON) data [**jupyterteamArchitecture**]. While notebooks allow to tell a story using visualizations and descriptions, they lack in ability to be easily deployed using common methods. Of course, the deplyoment of a Jupyter Notebook is not impossible, but disproportionately more difficult than deploying python modules.

Since not every part of a data-science project has to be deployed, a hybrid approach to Juypter Notebooks and modular coding makes sense. The data understanding part of the project will be completed using notebooks. The data preparation and the modelling will be completed in python modules to ensure deployability if desired.

# 6. Data Understanding

## 6.1. Initial Data Collection Report

**Data Requirements Planning**   The business objective is the development of a solution for the automatic aggregation of documents based on their topics. To achieve this goal, a sufficient number of documents are required. Also, the documents must contain enough text to be assigned a topic. After clustering the value of each expense in a cluster have to be summed, to gain the desired insight of spendings in one category. So the value an expense amounts to has to be included in the data.

**Selection Criteria**   An initial look a the data has shown that each invoice item can easily contain more than 100 pieces of information, which can be considered as attributed in the dataset. The part of the data that is of interest for the research is the description and payment data. Other data needs to be retained for a coherent and informative presentation of the results, such as location information and information on seller and buyer.

**Insertion of Data**   The insertion of data is concerned with the theoretical utilization of the data and problems arising during this process. The encoding and grouping of free text items is referenced as one concern in the CRISP-DM user guide [**CRISPDM2000**]. In this research the focus is on free text items, therefore this step will be discussed in detail in the following chapter. Other concerns are missing attributes. The dataset has already undergone a surface-level evaluation, which concludes that all relevant attributes are contained.

## 6.2. Data Description Report

The data is available as a local folder of size 5.12 GB, it contains 152,591 JSON files. Each file represents one invoice document. The files are between 11 and 540 KB in size.

Each invoice contains specific header data, the detailed list of the 172 header fields is contained in the appendix A.2.

The items listed in one invoice are line items. The information about line items is contained in 32 features, also detailed in the appendix A.3.

## 6.3. Data Exploration Report

The target for aggregation being the descriptions of line items, this feature is explored first.

**Descriptions** The descriptions of invoice items vary greatly in their length. While the largest portion of descriptions is under 600 characters, there are outliers with over 1000 characters.
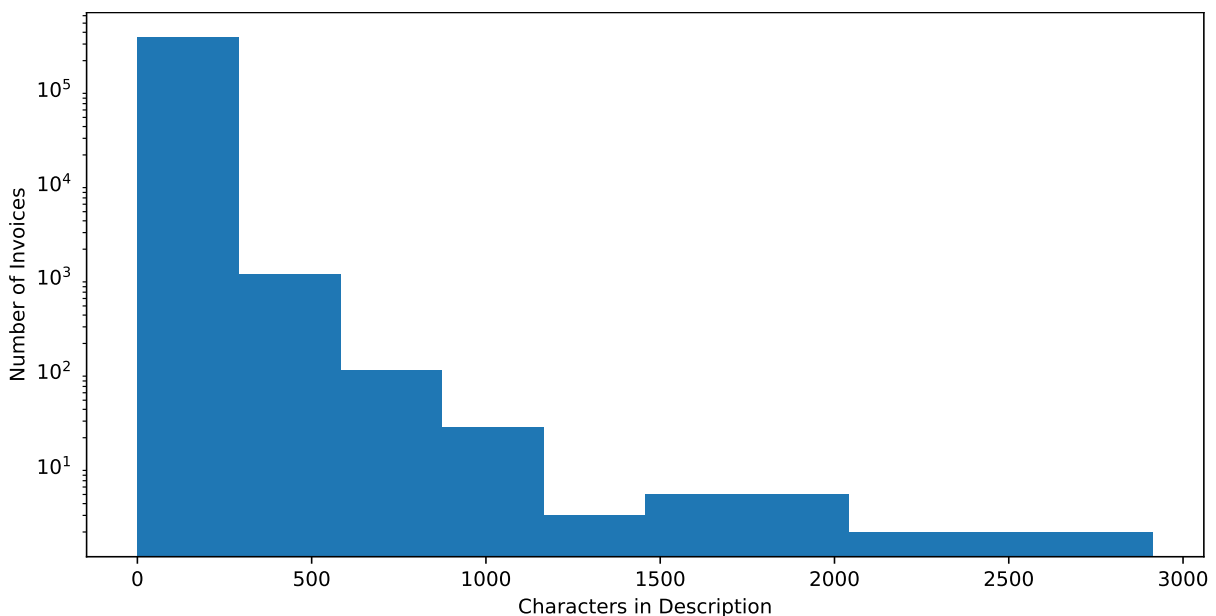


Figure 6.1.: Histogram of Description Length

**Language Distribution**   The description of the invoices is related to the feature 'language', which describes the language of the invoice and its text fields. The most popular languages are English, Spanish, and German. It is noticeable, that there are over 700 languages or combinations of more than one language. 58878 invoices were not assigned a language, which is roughly one third of the total number.
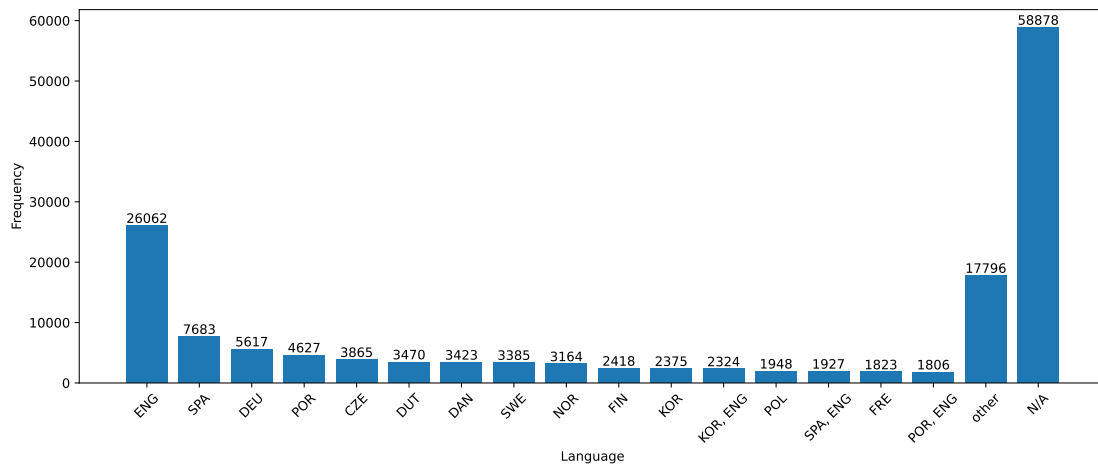


Figure 6.2.: Number of Invoices per Language

For the cumulative distribution of the languages, N/A values were omitted. From the chart it can be inferred, that 80% of invoices are in the top 16 languages. The slow rise after the top 80 languages indicates, that there are a lot of languages with a small number of invoices. This is also explained with a huge number of possible combinations of more than one language.

## 6.4. Data Quality Report

DOes the data contain errors? Are there missing values? are all values plausible? plot some stuff here check number of fields in each record (?) maybe optional
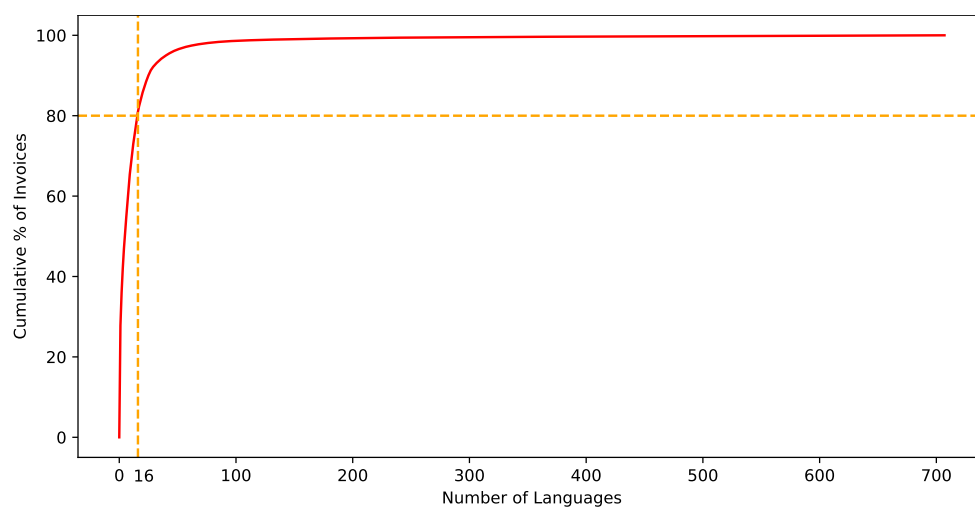
Figure 6.3.: Percentage of Invoices by cumulative Number of Languages

# 7. Data Preparation

## 7.1. Data processing and data wrangling

Data preparation is the process of transforming the acquired dataset into a dataset which can be fed into learning algorithms and is cleaned of impurities in such a way, that the learning results are likely satisfactory. Impurities may be missing values, shifted columns, encoding errors or different data formats. Data preparation includes data wrangling, data cleaning, and feature extraction. What is left should be a uniform dataset, which is in a format that is understandable by the desired algorithms.

### 7.1.1. Alternatives

The chosen dataset consists of files in JSON. Several alternatives for storage and transforming the data exist.

**Storage Location**   Within the constraints of the thesis, three options are available for data storage. Firstly, data can be stored locally on the available machine. The data is available without internet access, and the local machine has high read/write speeds. This option requires no additional learning, and is free of charge for the department. A multitude of libraries exist for accessing files on a local filesystem. A downside is the limited scalability in terms of storage capacity and read/write operations. Additionally, local storage makes the data only available to this specific machine.

Storing data on a cloud filesharing system is easy to operate and free of usage-bound charges. The storage space is virtually unlimited. Accessing files on sharing platforms is feasible but tedious. The access can be shared within the company context, but everyone is subject to the limited accessibility. Using a files storage service could be of use for transferring data without a physical connection. Still, this is the only recommended use case in this context.

The third option is storing the data using a specialized storage service, such as AWS S3. While the learning overhead is higher in the beginning, the scalability is a convincing argument. Billing is according to usage, but adequate because of the high connectivity with other cloud-based ML service offerings inside and outside of SAP.

Table 7.1.: Comparison of available Storage Options

|  | **Local** | **Cloud Filesharing** | **Specialized Storage Services** |
|---|---|---|---|
| Example | 2.6GHz 512GB SSD | Microsoft OneDrive | AWS S3 |
| Ease of use | high | high | higher learning overhead |
| Cost | free of charge | free of charge | billing according to used storage space |
| Scalability | limited | unlimited | unlimited |
| Data access | local | limited remote access capacity | local, remote, high connectivity to ML service offerings |

**Data Format**   The data is still available only in the from JSON documents.

- transforming json documents into dataframe rows

## 7.1.2. Theoretical Implementation

For the data wrangling the invoices have to be read into memory and then processed into a reusable structure. Each document will be read into memory, then the necessary information will be extracted. All invoices will be combined into one data structure, which then is persisted for later use.

The data for an invoice and its contained items is stored in an array of objects. This is also shown in code example (7.2) of a shortened JSON Schema for one invoice. The array "annotations" contains objects. One object represents information such as the invoice
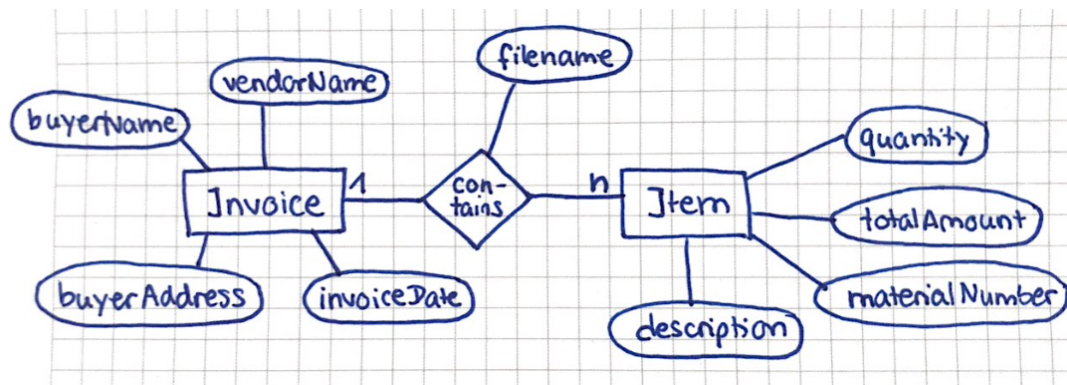
Figure 7.1.: Entity Relationship Diagram for Invoce Documents

date or the unit price. Information about invoice items have labels containing the prefix "lineItem". One invoice containing several items is represented by duplicate values of one label (in the example 7.1 the label "lineItem.description.value"). Here, the order of the labels has to be retained during data wrangling to ensure not mixing up information about different invoice items.

```
 1  {
 2      "annotations": [
 3          {
 4              "label": "vendorName.value",
 5              "text": "Example IT Vendor"
 6          },
 7          {
 8              "label": "invoiceDate.value",
 9              "text": "2020-02-32"
10          },
11          {
12              "label": "lineItem.description.value",
13              "text": "Mouse"
14          },
15          {
16              "label": "lineItem.description.value",
17              "text": "HDMI Adapter"
18          },
19          {
```

```
20                "label": "lineItem.description.value",
21                "text": "Shipping Cost"
22            }
23        ]
```

Listing 7.1: JSON of one invoice

### 7.1.3. Practical Implementation

The files are processed in python, using the library "json". Each file is opened, and decoded with python's build in json decoder. The hierarchy is traversed until the array "annotations" is reached. Now, the tuples of label and text are saved. This process is split up into invoice and item labels.

The information on invoices is stored in a tabular data structure, a python pandas DataFrame. Worth mentioning is that some invoices contain more information than others. In this case, invoices are still appended into one table, but fields for non-existent values are left empty. The filename is the unique identificator for one invoice.

| filename | vendorName.value | invoiceDate.value | totalAmount.value |
|---|---|---|---|
| 73d8feb2-b01d-11ec-b909-0242ac120002.json | Example Telephone Company | 2020-03-12 | |
| 942d7318-b01e-11ec-b909-0242ac120002.json | Example IT Vendor | 2020-02-31 | 24.48 |

Figure 7.2.: Exemplary depiction for processed invoices in a DataFrame

Similarly, information on the invoice items is retrieved from the documents and stored in a pandas DataFrame. Every invoice item has an unique id and can be linked to the respective invoice through the filename.

The resulting two DataFrames are serialized with the python standard library "pickle". The data can be efficiently loaded into memory and reserialized again with this library.

This processing step allowed for storing the initial 5.12GB of data in a more usable format and takes up only a total of 393MB. Even further improvements to storage will be dicussed later on.

| | filename | lineItem.totalAmount.value | lineItem.description.value |
|---|---|---|---|
| **0** | 73d8feb2-b01d-11ec-b909-0242ac120002.json | 250.00 | Cell phone bill may |
| **1** | 73d8feb2-b01d-11ec-b909-0242ac120002.json | 256.00 | Cell phone bill june |
| **2** | 73d8feb2-b01d-11ec-b909-0242ac120002.json | 249.00 | Cell phone bill july |
| **3** | 73d8feb2-b01d-11ec-b909-0242ac120002.json | 280.00 | Cell phone bill august |
| **4** | 73d8feb2-b01d-11ec-b909-0242ac120002.json | 300.00 | Cell phone bill september |
| **5** | 942d7318-b01e-11ec-b909-0242ac120002.json | 12.99 | Mouse |
| **6** | 942d7318-b01e-11ec-b909-0242ac120002.json | 4.99 | HDMI Adapter |
| **7** | 942d7318-b01e-11ec-b909-0242ac120002.json | 6.50 | Shipping Fee |

Figure 7.3.: Exemplary depiction for processed invoice items in a DataFrame

The process of reading files from the disk is not inherently an expensive one, but in the realms of many thousand documents, processing times soon reach several days. Observing the execution of the python code showed a peculiarity: While the utilization of the used processor was consequently at the maximum, only one process is executed, leaving the total CPU usage at only 20%.



Figure 7.4.: Exemplary depiction for processed invoice items in a DataFrame

One question that may now arise is: Why doesn't python split up the workload and employ the full capacity of the machine? This can be explained with the design of the Python interpreter. The Global Interpreter Lock (GIL) controls access to the Python Virtual Machine executing the code. This lock only allows exactly one thread to run at a time [**corePython**]. This of course is not favorable, as valuable CPU capacity is unused. Fortunately, the GIL behaves in a special way regarding C code, and I/O operations in Python utilize C code: the lock is released before executing a C routine [**corePython**]. This allows to bypass the (in this case) inconvenient locking mechanism.

Different python libraries exploit this speciality and allow to spawn a pool of different processes, which then execute calls asynchronously. One example is the ProcessPoolExecutor. A notable restriction is that only picklable objects can be submitted for multiprocessing. This concerns both fuction and its parameters. While this is not a problem in this task, this restriction will become important later on in the section about feature extraction.

```json
1  {
2    "$id": "https://example.com/arrays.schema.json",
3    "$schema": "https://json-schema.org/draft/2020-12/schema",
4    "description": "A representation of information extracted ↙
         ↳ from invoices.",
5    "type": "object",
6    "properties": {
7      "annotations": {
8        "type": "array",
9        "items": { "$ref": "#/$defs/annot" }
10      }
11    },
12    "$defs": {
13      "annot": {
14        "type": "object",
15        "required": [ "label", "text" ],
16        "properties": {
17          "label": {
18            "type": "string",
19            "description": "The identifier of the extracted ↙
                 ↳ information."
20          },
21          "text": {
22            "type": "string",
23            "description": "The value of the extracted ↙
                 ↳ information."
```

Listing 7.2: Shortened JSON Schema of one invoice document representation

## 7.2. Data cleaning

- removing stopwords from several languages

- removing numbers and interpunction

- tokenization The task of data cleaning was also completed using python scripts. The library Natural Language Toolkit (NLTK)

```
1 $ python3 -c 'import cleaner; print(cleaner.clean("500grams of ↙
    ↳ special baking flour type 504"))'
2 >>> ['of', 'special', 'baking', 'flour', 'type']
```

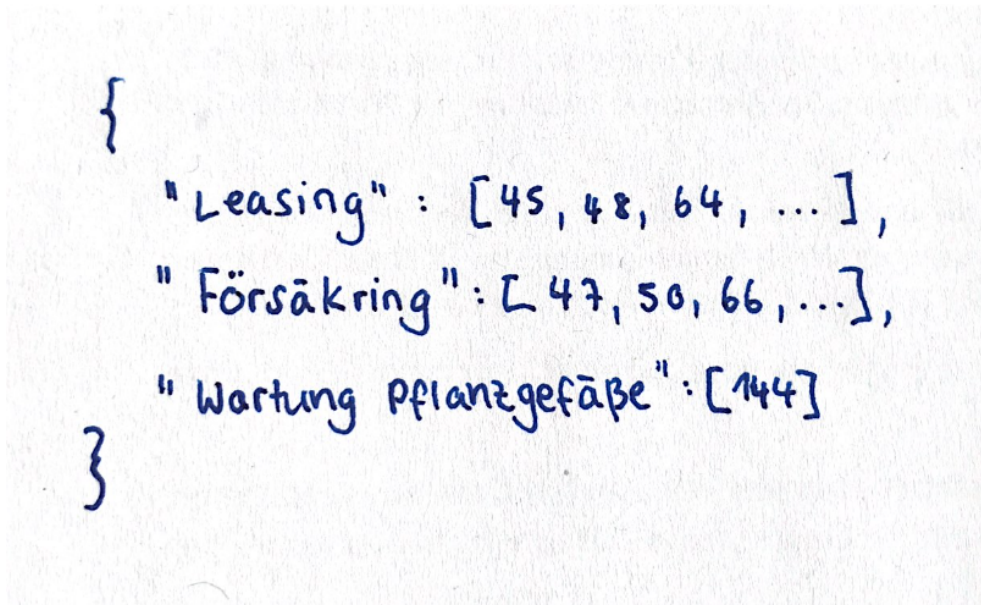## 7.3. Considerations of Space and Time Complexity

The dataset consists of over 150.000 invoices, and in those invoices, over 350.000 items are listed. With hardware-limitation in place, an optimized approach for storing and processing the data is required. Several considerations for speeding up processing time and reducing storage space can be made. In the following, the observations are explained and approaches for improvement are given.

**Duplicates and space complexity**

Investigation shows, there are only 79.741 unique descriptions for the listed items. By saving only the unique values, the required space is reduced to less than one fourth compared to before. Additionally, this step is required by most machine learning models, as duplicate input values can skew the outcome. The model is chosen later, so this processing step leaves the model selection more open to different kinds of learning algorithms.

**Reconstructing Relationships and time complexity of searching**

After the deletion of duplicate descriptions (documents in the corpus), the

```
{
    "Leasing" : [45, 48, 64, ...],
    "Försäkring": [47, 50, 66, ...],
    "Wartung Pflanzgefäße": [144]
}
```

Explain how I want to store the data. Explain the successive artifacts.

## 7.4. Feature Extraction and Feature Engineering

The majority of pupular ML algorithms require the input of scalar, vector or matrix data. A form of ML models, which work with textual input will be discussed later in this section. But since many algorithms were not designed to work with textual data, a transformation is required before already existing algorithms can be applied. Several methods for representing text as mathematical object will be discussed in the following.

### 7.4.1. Alternatives

**One-hot encoding**

**Bag of Words Model**

Following three documents will be considered to explain the workings of the Bag of Words (BoW) model:

| | content |
|---|---|
| **document** | |
| **1** | car repair |
| **2** | flight ticket processing and ticket reservation |
| **3** | parking ticket |

A corpus is transformed into the BoW representation in two steps.

Firstly, the vocabulary is determined. The vocabulary is a collection of all words occuring in the corpus. Every word is contained exactly once, regardless of the actual number of occurrences. The vector represenation of one document is of the same length as the vocabulary. One document being represented by one vector, a corpus of several documents can be represented as a matrix. The resulting matrix has the size $|D| * |V|$, with $|D|$ being the number of documents in the corpus, and $|V|$ being the size of the vocabulary.

Secondly, for each combination of one document $d_i$ and one word $v_j$ in the vocabulary, the occurrences are counted. The times, how often the specific word is contained in the document is noted in the matrix at location $ij$.

| | and | car | flight | parking | processing | repair | reservation | ticket |
|---|---|---|---|---|---|---|---|---|
| **car repair** | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| **flight ticket processing and ticket reservation** | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 2 |
| **parking ticket** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

The result of vectorization are three vectors:

$$d_1 = [0, 1, 0, 0, 0, 1, 0, 0]$$

$$d_2 = [1, 0, 1, 0, 1, 0, 1, 2]$$

$$d_3 = [0, 0, 0, 1, 0, 0, 0, 1]$$

This vectorization method can be implemented with ease and in a computationally efficient manner. The BoW model allows for a very intuitive understanding of documents, since texts consisting of the same words are considered topically related.

One of the drawbacks of this method is that no consideration is paid to words being repeated in one document. Additionally, no semantic relationship between words or

documents can be inferred. Further, it can be stated, that the BoW representation fails to capture the meaning of synonyms. This becomes obvious with an example: document $d_1$ and $d_3$ would be considered to belong into the topic of transportation or automotives. The BoW representation suggests topical proximity between document $d_2$ and $d_3$ , through the shared word "ticket". "Ticket" here is used in both the meaning of an entrace pass ($d_2$) and in the meaning of a note for a traffic offence ($d_3$).

To conclude, the BoW model is a straightforward text representation method. Still, it fails to capture several aspects of the natural language.

**Tf-Idf**

The Term Frequency - Inverse Document Frequency (TF-IDF) model aims to capture more meaning from the corpus by considering the composition of the whole corpus for the calculation of individual document vectors.

TF-IDF makes two assumptions about natural language:

**1 Term Frequency**  A word $t_i$ which occurs very frequently in one document is considered to describe a text very well. The occurences of one word in one document is denoted as $\#(t_i)$. One additional consideration needs to be made regarding the document length. In a document of length $|d_2| = 6$ and a document of length $|d_3| = 2$, the word $t_i$ occuring once would be considered equally important to each document. Of course, the word should be considered more important to $d_3$, since it accounts for a larger share of the text. The measure resulting from both ideas is the term frequency:

$$TF(t_i, d_j) = \frac{\#(t_i)}{|d_j|} = \frac{occurences\ of\ word\ t_i\ in\ document\ \ d_j}{legth\ of\ d_j}$$

**2 Inverse Document Frequency**  A word $t_i$ which occurs in a large number of documents does not describe one document well. Words occuring in many documents often are articles or pronouns (stopwords) which do not provide value when inspecting the content of a text. The inverse document frequence is a measure accounting for this fact. The inverse document frequency of a word is the proportion between the number of documents in the corpus and the number of documents containing the word. The

logarithm is applied, as the importance of a word does not increase proportionally to the number of occurrences.

$$TF(t_i, d_j) = \log \frac{|D|}{\#(d_{t_i})} = \log \frac{number\ of\ documents\ in\ corpus\ D}{number\ of\ documents\ containing\ word\ t_i}$$

Combining both assumptions, the TF-IDF measure is created:

$$TFIDF(t_i, d_j) = TF(t_i, d_j) * TF(t_i, d_j)$$

For the corpus displayed in the previous section the document vectors calculated with the TF-IDF measure are:

$$d_1 = \begin{bmatrix} 0\ 0.707107\ 0\ 0\ 0\ 0.707107\ 0\ 0 \end{bmatrix} \tag{7.1}$$

$$d_2 = \begin{bmatrix} 0.39798\ 0\ 0.39798\ 0\ 0.397980\ 0.397980.605349 \end{bmatrix} \tag{7.2}$$

$$d_1 = \begin{bmatrix} 0\ 0.707107\ 0\ 0\ 0\ 0.707107\ 0\ 0 \end{bmatrix} d_2 = \begin{bmatrix} 0.39798\ 0\ 0.39798\ 0\ 0.397980\ 0.397980.605349 \end{bmatrix} d_3 = \begin{bmatrix} 0\ 0\ 0\ 0.7$$

The TF-IDF measure corrects some of the pitfalls of the BoW model. It certainly is less vulnerable to skewing by stopwords as words are ranked by importance to each document and the all over corpus.

Just as the BoW representation, TF-IDF suffers from high-dimensionality. The vectors contain one element for each word in the vocabulary, resulting in vectors which are inefficient to handle. Also, TF-IDF fails to represent the topical relationship between $d_1$ and $d_3$.

**Word Embeddings**

Again, the most desirable vector representation consists of dense low-dimensional vectors which are close to each other if the words are considered similar. The lack of "under-

standing" of related words, and the problem of high-dimensionality is corrected with the third presented option: Word Embeddings.

An embedding is a translation of a word into a high-quality vector. The model does so, as it has been trained on a large set of natural language data. Popular embeddings include word2vec, fasttext and doc2vec. Word2Vec is the original model and will be discussed in the following.

The word2vec model assumes that words appearing in a similar context are also similar to each other. There are two options for the input and output: the word $w_i$ and its context $c_i$.

**document**

| | |
|---|---|
| **1** | The car repair was expensive. |
| **2** | My secretary did the ticket reservation. |
| **3** | I got a parking ticket yesterday. |

**Continuous Bag of Words**    With the CBOW architecture the word2vec model is trained to predict a word using the context as input. A sliding window of a predefined size is moved along the text.
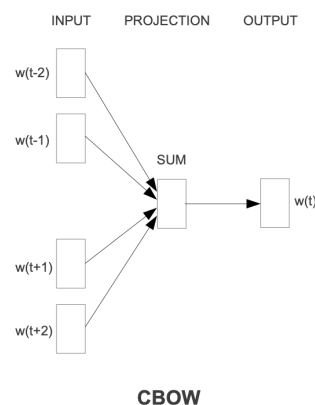


Figure 7.5.: Continuous Bag of Words architecture
with sliding window of size $C = 5$

**Skipgram**   With the skipgram architecture the word2vec model is trained to predict the context of the input word.



INPUT   PROJECTION   OUTPUT

w(t-2)

w(t-1)

w(t)
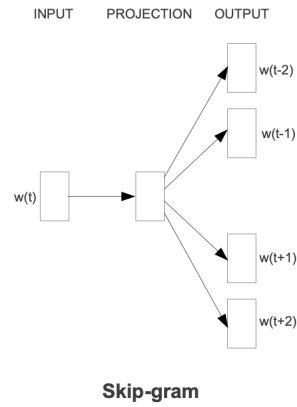
w(t+1)

w(t+2)

**Skip-gram**

Figure 7.6.: Skipgram architecture
with sliding window of size $C = 5$

**Negative Sampling**   Word2Vec uses either SKIPGram or CBOW.

Word2vec can utilize two models for selecting observations: either skipgram or CBOW.

## 7.4.2. Theoretical Implementation

## 7.4.3. Practical Implementation

Tf-Idf weighed USEM Word vectors

| input | target |
|---|---|
| secretary | my |
| secretary | did |
| did | secreatary |
| did | the |
| the | did |
| the | ticket |
| ticket | the |
| ticket | reservation |

Figure 7.7.: Observations for training with skipgram architecture

# 8. Modeling

The objective of this thesis is the grouping of expenses with the methods of NLP. Grouping, or clustering, is the practise of sorting data points into groups in such a way that the similarity inside a group (intra-cluster similarity) is high while the similarity between clusters (inter-cluster similarity) is low. The defintion of similarity as well as different clustering algorithms are presented and contrasted in the following sections.

## 8.1. Similarity and distance measures

A distance measure is a quantification of how near objects in space are. Distance measures can be defined for spaces of arbitrary numbers of dimensions.

**Euclidean Distance**   The most intuitive and popular distance measure is the euclidean distance. This measure is applicable for vector spaces $F^n$ with $n \in \mathbb{N}_0$. For low-dimensional applications, this distance works well and shows great results. Euclidean distance can be calculated in a highly efficient manner, even for n dimensions. This suggests, that euclidean distance is particularly suitable for high-dimensional data. Unfortunately, euclidean distance falls victim to the curse of dimensionality. In [**beyerNearestNeighbor**] it is proven that with increasing dimensions, the distance between data points approaches an uniform value for all datapoints. This effect could be demonstrated for spaces with as little as ten dimensions. Therefore, it can be said that euclidean distance is not suitable for high-dimensional data. With vectors in NLP ranging from 100 to 800 dimensions, this distance measure is not suitable.

**Dot Product**

**Cosine Distance**

## 8.2. Clustering Algorithms

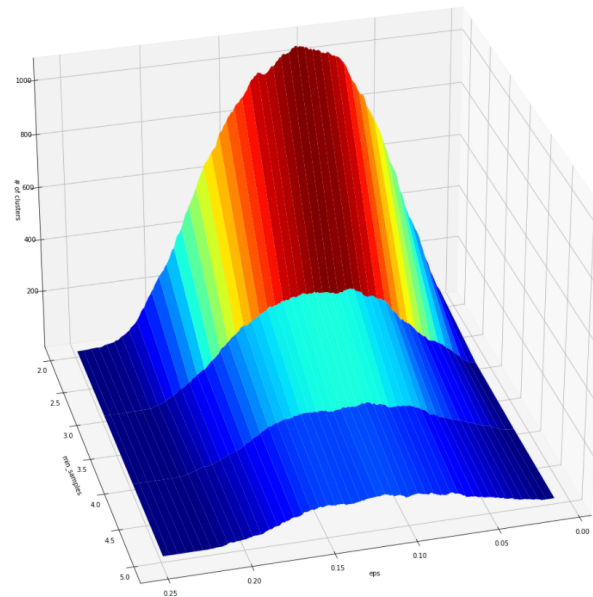### 8.2.1. K-Means with Euclidean Distance

### 8.2.2. DBSCAN



Figure 8.1.: DBSCAN hyperparameters and resulting number of clusters

### 8.2.3. Machine Learning

Already Alan Turing understood that for laymen a learning machine can be perceived as a paradox. How can a machine learn, if a human has to define its behavior beforehand? There are three major subfields in the discipline of artificial intelligence that fundamentally explain how a computer can learn how to behave despite predefined behavior.

### 8.2.4. Supervised Learning

A supervised learning algorithm learns its decision with the help of a data set (input) that also contains the correct decision (output) as information. It is trained with only a

part of the entire data set, so that the model can be tested in a later step with the help of unknown data. This way, a statement can be made about the accuracy of the model.

## 8.2.5. Unupervised Learning

Unsupervised learning is complementary to supervised learning. All algorithms that fall into the category of unsupervised learning are trained with data that does not contain the correct output (label) as information. Here, the categorization is not constrained by the given data, but decided on by the algorithm.

## 8.2.6. Reinforcement Learning

The third way in which an algorithm can make better decisions as it gains experience is called reinforcement learning. Reinforcement learning is about letting algorithms solve very complex tasks. The special feature is that there is no defined solution path, but the algorithm is rewarded for goal-oriented behavior and punished for wrong decisions. The definition of goal-oriented behavior has to be put into place by the engineers setting up the training of the model. Real-world tasks are extremely complex, so not all possible solution paths can be calculated and compared to find the optimal path. Parking a car is a routine task for a human after a few hours of driving, but a computer sees only an infinite set of possibilities for turning angles. This problem can be solved by reinforcement learning. The algorithm is rewarded for each parking attempt where the car ends up seeing in the parking space. For the remaining attempts, the algorithm is penalized. Over many thousands of attempts, the reinforcement learning model is trained in this way.

The three major ways of learning even with previously defined behavior can now be implemented by specific models. For example, there are several ways to create and train a model using Unsupervised Learning.

# 9. Deployment

# 10. Evaluation of the result

## 10.1. Visualization

## 10.2. Measures

# 11. Conclusion and Outlook

## 11.1. Conclusion

## 11.2. Outlook

Other process model could have been used.

# A. Appendix

## A.1. Inventory of Resources

## A.2. Invoice Header Data

accountNumber.key

accountNumber.value

barCode.value

buyerAddress.cityTownVillage.value

buyerAddress.country.value

buyerAddress.district.value

buyerAddress.extraName.value

buyerAddress.full.key

buyerAddress.full.value

buyerAddress.houseNumber.value

buyerAddress.stateProvince.value

buyerAddress.street.value

buyerAddress.zip.value

buyerName.key

buyerName.value

comments.key

comments.value

country.value

currency.key

currency.value

deliveryDate.key

deliveryDate.value

deliveryNoteNo.key

deliveryNoteNo.value

discount.key

discount.value

dueDate.key

dueDate.value

employeeName.key

employeeName.value

exchRate.key

exchRate.value

exchRateSrcCurr.key

exchRateSrcCurr.value

exchRateTarCurr.key

exchRateTarCurr.value

filename

index

invoiceAmount.key

invoiceAmount.value

invoiceDate.key

invoiceDate.value

invoiceNo.key

invoiceNo.value

invoiceType.value

language

paymentTerms.key

paymentTerms.value

pii.address.key

pii.address.value

pii.email.key

pii.email.value

pii.name.key

pii.name.value

pii.other.key

pii.other.value

pii.phone.key

pii.phone.value

purchaseOrderNo.key

purchaseOrderNo.value

shipToAddress.cityTownVillage.value

shipToAddress.country.value

shipToAddress.district.value

shipToAddress.extraName.value

shipToAddress.full.key

shipToAddress.full.value

shipToAddress.houseNumber.value

shipToAddress.stateProvince.value

shipToAddress.street.value

shipToAddress.zip.value

shippingAmount.key

shippingAmount.value

Table A.1.: Inventory of Resources

| Type of resources | Kind of resources | Quantification |
|---|---|---|
| Personnel | business experts | 1 person |
| | data experts | 1 person |
| | technical support | 1 person |
| | data mining experts | 1 person |
| Data | fixed extracts | - |
| | live data | - |
| | warehoused data | - |
| | operational data | 1 dataset |
| Computing resources | hardware platforms | 1 machine<br>access to SAP AI Core |
| Software | data mining tools | anaconda, jupyter |
| | other relevant software | Excel, Visual Studio Code, Git |

*) An asterisk indicates theoretical availability of more resources, if needed.

subtotalAmount.key
subtotalAmount.value
tableHeader.batchNumber.value
tableHeader.description.value
tableHeader.discount.value
tableHeader.materialNumber.value
tableHeader.purchaseOrderNumber.value
tableHeader.quantity.value
tableHeader.serialNumber.value
tableHeader.tableHeaderBox
tableHeader.tax10Amount.value
tableHeader.tax10Rate.value
tableHeader.tax1Amount.value
tableHeader.tax1Rate.value
tableHeader.tax2Amount.value
tableHeader.tax2Rate.value
tableHeader.tax3Amount.value
tableHeader.tax3Rate.value
tableHeader.tax4Amount.value
tableHeader.tax4Rate.value

tableHeader.tax5Amount.value
tableHeader.tax5Rate.value
tableHeader.tax6Amount.value
tableHeader.tax6Rate.value
tableHeader.tax7Rate.value
tableHeader.totalAmount.value
tableHeader.unitOfMeasure.value
tableHeader.unitPrice.value
tax10Amount.key
tax10Amount.value
tax10Description.key
tax10Description.value
tax10Rate.key
tax10Rate.value
tax1Amount.key
tax1Amount.value
tax1Description.key
tax1Description.value
tax1Rate.key
tax1Rate.value

tax2Amount.key
tax2Amount.value
tax2Description.key
tax2Description.value
tax2Rate.key
tax2Rate.value
tax3Amount.key
tax3Amount.value
tax3Description.key
tax3Description.value
tax3Rate.key
tax3Rate.value
tax4Amount.key
tax4Amount.value
tax4Description.key
tax4Description.value
tax4Rate.key
tax4Rate.value
tax5Amount.key
tax5Amount.value

# A.3. Invoice Line Item Data

lineItembaxIteNumbeRateare7Amount.value

lineItemdescriptionExAmountRatervalue

lineItemdiscountvaBRatetax8Amount.value

lineItemlineItemBoxBAmountax8Ratervalue

lineItemmaterialNurBRatetaxx10Amount.value

lineItempurchasxOrHAmtumbeRatelue

lineItemquantityaxItRatetotalAmount.value

lineItemserItNumbFAmountOfMeasure.value

lineItemtaxIt0AmountRateunitPrice.value

lineItemtaxIt0Ratex6Amount.value

lineItemtaxItAmountx6Ratervalue