Πρόγραμμα Μεταπτυχιακών Σπουδών

στην Αναλυτική των Επιχειρήσεων και Επιστήμη των Δεδομένων

Τμήμα Οργάνωσης και Διοίκησης Επιχειρήσεων

<u>Διπλωματική Εργασία</u>

«Επεξεργασία φυσικής γλώσσας (NLP) στο ψηφιακό μάρκετινγκ – Ανάλυση συναισθήματος στις κριτικές του Amazon χρησιμοποιώντας το OpenAI»

«Natural Language Processing (NLP) in Digital Marketing – Sentiment Analysis in Amazon Reviews using OpenAI»

Κωνσταντίνα Δρίζη του Κωνσταντίνου

Υποβλήθηκε ως απαιτούμενο για την απόκτηση του μεταπτυχιακού διπλώματος στην Αναλυτική των Επιχειρήσεων και Επιστήμη των Δεδομένων

Θεσσαλονίκη, Αύγουστος 2023

Acknowledgements

I would like to express my gratitude to my thesis advisor, Mr, Tarampanis and Mrs. Karamanou for their guidance and support.

I would also like to acknowledge the support and encouragement of my parents and my brother, without whom this journey would not have been possible. Their belief in me and their constant encouragement has been a source of inspiration throughout the entire process.

Table of Contents

Abstract

In the evolution of technology, data is highly valued. Artificial Intelligence (AI) has become increasingly integrated into our lives. With the rise of digital channels and customer-generated text data, NLP techniques offer the ability to extract valuable insights into customer preferences and sentiments. This thesis explores the potential of NLP in digital marketing, through exploring the literature for NLP and Digital Marketing and then through making a sentiment analysis of Amazon customer reviews by fine-tuning Open AI.

Περίληψη (Abstract in Greek)

Στις μέρες μας τα δεδομένα αξίζουν όσο ο χρυσός. Η Τεχνητή Νοημοσύνη (AI) έχει ενσωματωθεί όλο και περισσότερο στη ζωή μας. Με την άνοδο των κοινωνικών δικτύων και των δεδομένων που δημιουργούνται από τους πελάτες, η επεξεργασία φυσικής γλώσσας (NLP) προσφέρει τη δυνατότητα εξαγωγής πολύτιμων πληροφοριών για τις προτιμήσεις των πελατών. Αυτή η εργασία διερευνά τις δυνατότητες του NLP στο ψηφιακό μάρκετινγκ, ιδιαίτερα μέσω της ανάλυσης του κειμένου των κριτικών πελατών χρησιμοποιώντας το Open AI.

Chapter 1. Introduction

The changes in technology over the last years have been tremendous. We live in the era of information were data worth as gold.

Artificial intelligence (AI) has become increasingly integrated into our lives in recent years through the increasing production of AI-powered products and services, such as virtual assistants, chatbots and recommendation systems. AI has also transformed various industries, including marketing, finance and transportation, by enabling new capabilities and efficiencies. Additionally, AI has influenced a user's online experiences, from personalized content recommendations to targeted advertising.

Natural Language Processing (NLP) is a rapidly growing field that has the potential to change the way businesses interact with customers. With the increasing use of digital channels for marketing, NLP can help companies to better understand and engage with their audience.

Digital marketing that is also in our lives lately defines (and has changed) the way we buy, eat, and even the way we live. The growth of digital channels, such as social media and chatbots, has led to an increase of customer-generated data in the form of text. This data can be analyzed using NLP techniques to extract valuable insights about customer preferences, needs, and sentiments. By leveraging these insights, businesses can create more targeted and personalized marketing campaigns, resulting in increased customer engagement and conversion rates.

In this thesis, we are going to perform a sentiment analysis in Amazon reviews (Chapter 4). For that purpose we will firstly, review the literature on NLP in digital marketing and identify key challenges and opportunities, we will then explore the current state of NLP in digital marketing and investigate its potential to improve customer engagement and conversion rates. Moving forward, we will present a sentiment analysis of customer reviews on Amazon, though fine-tunning GPT. Our research aims to provide a deeper understanding of the potential of using GPT in digital marketing and to demonstrate its value to businesses looking to improve their customer engagement and conversion rates.

Chapter 2. Literature Review

This chapter will elaborate on Natural Language Processing and Digital Marketing concepts. This is essential so the reader can get familiar with the basic concepts of the topic. The literature can be distinguished into two categories. The first one contains bibliography with an approach to Artificial Intelligence (AI), NLP and Digital Marketing concepts and terminologies. In the second one, the bibliography is mainly for Sentiment Analysis, Text Classification (TC) in Marketing (but within the framework of NLP) and ChatGPT.

2.1 Artificial Intelligence

2.1.1 Definition

According to Brittanica Artificial Intelligence (AI) refers to the capability of a machine to carry out tasks that were once thought to be only doable by humans. Meaning it can perform tasks that would normally require human intelligence, such as reasoning, learning, and decision-making. AI systems can process and analyze enormous amounts of data, identify patterns and insights, and make predictions and recommendations based on that information. This can lead to significant efficiencies and improvements in various industries, such as marketing, finance, transportation, national security and healthcare (Darrell M. West and John R. Allen).

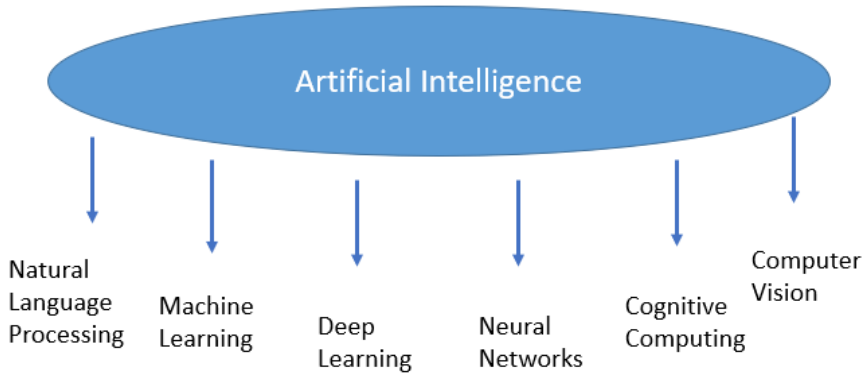


Figure 1: Fields of Artificial Intelligence

Artificial intelligence (AI) is a wide field that involves many subfields and applications, such as (Jurafsky, D., & Martin, J. H. (2019)):

- **Machine learning:** a field of AI that focuses on the development of algorithms and statistical models that allow machines to learn from data and improve their performance over time.
- **Natural language processing (NLP):** the field of AI that focuses on developing algorithms and techniques to enable machines to understand, interpret, and generate human language.
- **Deep learning:** an area of machine learning that uses neural networks with multiple layers to analyze and learn from large amounts of data.
- **Neural Networks:** a type of machine learning algorithm replicating the functions of the human brain.
- **Cognitive Computing:** is an interdisciplinary field of AI that combines machine learning, natural language processing, and other techniques to enable machines to simulate human thought processes.
- **Computer Vision:** focuses on enabling machines to understand visual information from the world around them.

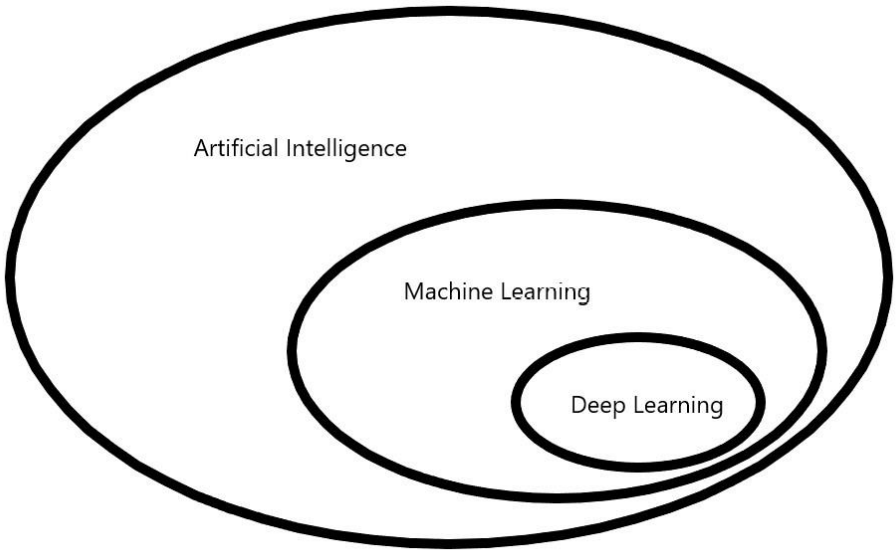


Figure 2: Overview of AI, ML and DL

2.2 Natural Language Processing (NLP)

2.2.1 Definition

Natural Language Processing (NLP) is a field of Artificial Intelligence (AI) that focuses on the relations between computers and human languages. It involves the development of algorithms and models that can understand, explain and create human language. NLP is a multidisciplinary field that draws on linguistics, computer science, and cognitive psychology.

NLP is also accustomed to perform a wide range of tasks such as language translation, sentiment analysis, text summarization, and question answering (Manning and Schutze, 1999). It is widely used in various applications like speech recognition, language-based search engines, machine translation, and more. It is also used in emerging technologies like virtual assistants and chatbots, which can understand and respond to users' queries in natural language (GPT).

It has become increasingly important for businesses and marketers that are looking to engage with customers and extract insights from customer-generated data to use NPL processes (Gershman et al., 2015).

2.2.2 Use of Natural Language Processing

Below are a few examples of how Natural Language Processing (NLP) can be used:

- **Text Classification:** it can be used to classify text into defined categories, such as spam detection in emails or sentiment analysis in customer reviews. One popular approach is using machine learning algorithms to train a model on labeled text data (Manning et al., 2008).
- **Named Entity Recognition:** it can be used to identify and extract specific information from text, such as people, organizations, or locations. One common approach is using conditional random fields (CRF) or recurrent neural networks (RNN) (Lample et al., 2016)

- **Machine Translation:** translate text from different languages. Some common approaches include using statistical machine translation or neural machine translation.
- **Text Summarization:** automatically summarize text, such as extracting the main points from a news article or a research paper (Nallapati et al., 2016)
- **Speech Recognition:** transcribe spoken language into text and can be used in applications like voice-controlled virtual assistants and speech-to-text dictation.
- **Language Generation:** generate text that mimics human writing, such as chatbot responses or automated content creation. One approach is using neural networks, such as the GPT (Generative Pre-trained Transformer)

2.3 Text Classification

2.3.1 Definition

As stated previously text classification, also known as text categorization, can be used to classify text into predefined categories, such as spam detection in emails or sentiment analysis in customer reviews. It applies in a lot of applications like sentiment analysis, spam filtering, topic detection, and news classification.

Some of the most popular approaches in text classification include rule-based methods where rules or patterns are manually created, supervised learning which uses labeled examples to train a model to predict the class of new texts, and unsupervised learning such as clustering and topic modeling which identifies patterns or topics in the text without labeled data. Deep learning methods like convolutional neural networks and recurrent neural networks have gained popularity in recent years and have shown promising results in text classification, as reported by Zhang and Wallace (2015).

2.3.2 Detailed view of Text Classification

2.3.2.1 Rule-based methods

Rule-based methods for text classification involve creating specific rules based on certain words or phrases present in the text. These rules help decide which category the text belongs to. A benefit of using rule-based methods is that it's easy to understand how the classification decision tree was made.

Rule-based methods can also be customized to fit a specific area like the medical field. For example, medical texts like clinical notes or radiology reports have specific language, so rule-based methods can be made to match the words and terms of the medical field. This makes rule-based methods more effective than general classification methods.

However, rule-based methods may not be able to handle complicated or detailed language, and creating rules can be time-consuming. Even with these limitations, rule-based methods are still useful in certain areas where language is structured and predictable.

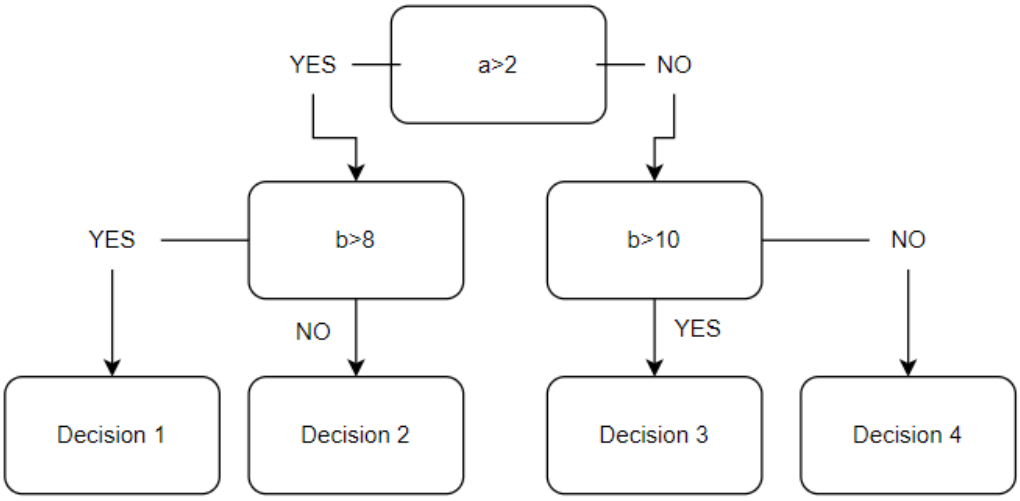


Figure 3: Example of a Decision Tree

2.3.2.2 Supervised Learning

Supervised learning is a field of machine learning in which an algorithm "learns" to classify new data based on patterns it has learned from a labeled dataset. In text classification, supervised learning algorithms are commonly used to assign one or more predefined labels to a given piece of text. These labels can be anything from sentiment categories (positive, negative, or neutral) to topic categories (sports, politics, technology, etc.)

One of the advantages of supervised learning in text classification is that it allows for a high degree of accuracy. By training on a labeled dataset, a model can learn to recognize patterns and relationships between words and phrases, and use them to accurately classify new, unlabeled text samples. This makes supervised learning particularly useful for applications where accuracy is important, such as spam filtering or legal document classification.

Another advantage of supervised learning is that it can be tailored to specific domains or applications. Like the previous example, in the medical field, a supervised learning model can be trained on a dataset of medical records to accurately identify and classify patient diagnoses. Similarly, in the field of finance, a model can be trained on a dataset of financial reports to identify fraud or make predictions about stock prices. This flexibility makes supervised learning a powerful tool for a wide range of applications.

However, there are also some disadvantages to supervised learning in text classification. One challenge is the need for large amounts of labeled data. In order to train a model that can accurately classify text, it is typically necessary to have a large dataset of labeled text samples. This can be also a time-consuming job, particularly for specialized domains where labeled data may be limited. Additionally, supervised learning models may struggle with handling complex language, such as sarcasm or irony.

In conclusion, supervised learning is a widely used approach in text classification that involves training a model on a labeled dataset to accurately classify new, unlabeled text samples. This approach offers high accuracy and flexibility, making it a valuable tool for a wide range of applications. However, it also has limitations, such as the need for large amounts of labeled data and the potential for difficulty in handling complex language

2.3.2.3 Unsupervised Learning

Unsupervised learning is a machine learning technique where the algorithm is trained on a dataset without labeled data. In text classification, unsupervised learning is particularly useful for discovering patterns and structures within large text datasets. One common approach to unsupervised text classification is clustering, where the algorithm groups together similar documents based on their content. This can be useful for tasks such as topic modeling or sentiment analysis, where the goal is to identify common themes or opinions within a large corpus of text. For example, an unsupervised clustering algorithm could be used to group together customer reviews based on the topics they mention, such as product features, customer service, or pricing.

Another application of unsupervised learning in text classification is dimensionality reduction, which involves reducing the number of features in a dataset while preserving as much of the original information as possible. This can be particularly useful for text classification tasks where the number of features (such as individual words or phrases) is very large, making it difficult to train a model using traditional supervised learning techniques. Principal Component Analysis (PCA) is one commonly used technique for dimensionality reduction in text classification. By reducing the number of features, PCA can make it easier to train models that accurately classify text documents.

One important consideration when using unsupervised learning for text classification is that it can be difficult to evaluate the performance of the algorithm, since there is no ground truth against which to compare the results. However, there are several metrics that can be used to evaluate clustering algorithms, such as silhouette scores or normalized mutual information. These metrics can help to determine the quality of the clusters generated by the algorithm and provide insights into the underlying structure of the text data. In addition, it is important to carefully consider the choice of algorithm and hyperparameters when using unsupervised learning for text classification. (Zhou, X., & Li, C. (2010))

2.3.2.4 Deep Learning

Deep Learning (DL) is a subset of Machine Learning (ML) that has proven to be an effective technique for text classification by using deep neural networks to learn complex patterns and features from raw text data.

Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) networks, are popular approaches for text classification using DL. CNNs extract local features such as phrases and word sequences, while RNNs model the context and dependencies between words in a document.

However, DL models require significant computational resources and large amounts of labeled training data, and their black-box nature makes it difficult to interpret their predictions. Advances in explainable DL and transfer learning techniques have shown promise in addressing these challenges. To fully understand DL, one should be familiar with terms such as loss score, loss function, optimizer, batch size, and epochs. The learning process in DL involves predicting a value, comparing it to the actual value, and using the loss score to adjust the layers' weights through the optimizer. The process is repeated n times, with each repetition considered an epoch, and the ideal number of epochs should be determined to avoid overfitting or underfitting. The batch size parameter determines the number of instances trained in an epoch, with the epoch referring to the training procedure on the entire dataset and the batch size referring to the chunks of data trained in an epoch.

2.3.3 Types of Predictions

Machine Learning and Deep Learning models are capable of performing regression tasks, where they predict a specific value, or classification tasks, where they determine the class to which an instance of data belongs. Regression tasks are straightforward and commonly used for estimating house prices, stock prices, weather forecasting, and similar applications. In contrast, classification problems exhibit more complexity and can be further categorized as binary, multi-class, multi-label, or multi-output, as described by Geron (2019).

Binary classification models are employed in scenarios where the data is divided into only two categories. Certain machine learning models, including Logistic

Regression and Support Vector Machine classifiers, are specifically designed for binary classification tasks.

On the other hand, multi-class classification models are trained to differentiate data across multiple categories. These classifiers are useful for tasks like Optical Character Recognition (OCR), where the objective is to determine if a character falls within the range of characters from "0" to "9". In such cases, each character can only belong to one category, and the model must make a decision regarding its appropriate class.

Multi-label classification models are utilized when the dataset can have multiple labels assigned to each instance. For instance, news articles can be assigned multiple labels simultaneously, and these classifiers have the capability to assign more than one class to a given instance.

2.3.4 Pre-Processing for Text Classification

Text classification is a commonly used technique in Natural Language Processing (NLP) that involves categorizing text into specific classes. This technique is known by various names such as “text categorization”, “document categorization”, “topic classification”. (Vajjala, Majumder, Gupta, & Surana, 2020). The applications of text classification are vast and include areas such as content classification, customer support, e-commerce, and language identification. To create a text classification system, one needs to follow a few steps such as collecting and preprocessing the dataset, splitting the dataset into training and test sets, transforming text into vectors, applying and training a classifier, evaluating the results, and finally deploying the model into new data and assessing its performance.

Next, we will elaborate more on the details related to Text Classification.

2.3.5 Text Pre-processing

In Text Classification the dataset consists of alphanumerical values. The first step is to pre-process the text, so it is easier to analyze it further and to perform feature engineering, in the most efficient way. There is a handful of pre-processing practices such as Word Tokenization, Lowercasing – Uppercasing, Stemming – Lemmatization, Removing stop-words (Vajjala, Majumder, Gupta, & Surana, 2020).

2.4 Digital Marketing

2.4.1 Definition

Digital marketing is the practice of promoting services, products, or brands using digital channels such as the social media, Internet, search engines, mobile devices, and other digital platforms. It includes a wide range of activities such asb search engine optimization (SEO), pay-per-click (PPC) advertising, social media marketing, email marketing, content marketing, and more. Digital marketing aims reaching and engaging with customers through digital channels, in order to drive sales and increase brand awareness (Kotler et al., 2016).

Digital marketing has become popular as more and more customers use digital channels to research and purchase products and services such as Instagram, Facebook etc. It allows businesses to reach audiences globally and target specific segments of the population with precision through personalization. Additionally, digital marketing allows easy tracking and measurement of the performance of campaigns, which is important for quick adjustments to improve the results.

Digital marketing differs from traditional marketing. It is highly data-driven and results-oriented. It allows for real-time monitoring and optimization of campaigns, which can lead to more effective and efficient marketing efforts. The term digital marketing also includes all the activities that are related to digital channels and digital assets, such as web analytics, search engine optimization, email marketing, mobile marketing, and many more (Kotler et al., 2016).

2.4.2 Use of Digital Marketing

Below are some examples of how digital marketing can be used to drive more traffic to the website and increase conversions, according to Afzal Beg, Dr. Nandita Tiwari (Digital Marketing with Natural Language Processing, ISSN NO:2347-6648):

- **Search Engine Optimization (SEO):** improve a website's visibility in search engine results, by using techniques such as keyword research, content optimization, and link building.
- **Pay-Per-Click (PPC) Advertising:** create and run online advertising campaigns, such as Google AdWords or Facebook Ads, that target exact

audiences based on demographics, interests, and behaviors in order to drive more traffic to a website by targeting specific groups of users.

- **Social Media Marketing:** manage social media accounts and campaigns, such as Facebook, Instagram, in order to engage with customers and promote products or services.
- **Email Marketing**: send targeted email campaigns to specific groups of customers or prospects.
- **Content Marketing:** distribute valuable, relevant and consistent content in order to attract and engage a specific target audience.
- **Influencer Marketing:** Depending on the product making partnerships with people that tend to influence others, could be a successful tactic to drive brand awareness and sales.

2.5 NLP in Digital Marketing

2.5.1 Definition

Natural Language Processing (NLP) is a primary aspect of digital marketing as it could understand human language, which is increasingly used in customer interactions and feedback through digital channels such as social media, email, and customer reviews websites (Pang and Lee, 2008).

2.5.2 Use of NLP in Digital Marketing

NLP is used in many digital marketing tasks like sentiment analysis, content generation, personalization, search engine optimization, and data analysis. Sentiment analysis, foe example, means figuring out the overall emotion conveyed in a piece of writing or speech, such as a customer review. Algorithms using NLP can be trained to identify positive, negative or neutral sentiment, which can help businesses to understand customer sentiment to their products or services to adjust their marketing strategies accordingly and gain profit (Pang and Lee, 2008).

Content generation uses NLP algorithms to automatically generate content, such as product descriptions, social media posts, and email campaigns. This could

save time and money, while also helping to improve the effectiveness of marketing campaigns (Zhang et al., 2018).

Personalization can also be used in NLP in digital marketing, it can be used to understand customer preferences, needs and behavior by analyzing their past interactions with a company and use this information to personalize the content and offer to each customer. A/B testing can also be applied to this field, allowing businesses to track customers behavior and preferences. Lastly, in search engine optimization (SEO), NLP algorithms can be used to identify the keywords and phrases that are most relevant to a business's products or services, which can then be used to improve the content of a website for better visibility on search engines (Li and Liu, 2011)

2.6 ChatGPT

2.6.1 Definition

As organizations increasingly rely on data-driven insights to make informed decisions, the need for strong natural language processing (NLP) models has never been greater. Among the most advanced and popular NLP models in use today is OpenAI's Generative Pre-trained Transformer (GPT). With its ability to classify text into various categories, GPT can be used to help people save time, solving complex problems faster and more efficiently than using any other method. Although GPT models are highly accurate and adaptable by default, fine-tuning can further improve their performance.

Beginning this thesis, GPT-3 was the latest model that could be fine-tuned with custom data. On August 2023 Open AI released GPT-3.5 Turbo fine-tunning. In the literature below, all mentions to GPT is for GPT-3 fine-tunning.

2.6.2 Benefits of Fine-Tuning GPT

The GPT model has the ability to create text that resembles that of a human, answer inquiries, finish sentences, and perform other tasks. However, fine-tuning GPT model can offer more advantages for analytical purposes, such as:

- Enhanced accuracy and relevance of results

Enhancing the accuracy and relevance of a GPT model to meet specific needs can be achieved through fine-tuning. This involves training the model on proprietary data to teach it the language and terminology unique to a particular industry or domain. By doing so, the model's ability to provide tailored insights that are significant to a business has improved.

- Customizing a model for specific tasks

Fine-tuning a model for a specific task can improve its performance, making it more efficient in processing and analyzing large amounts of data compared to a generalized model. A generalized model may only provide general root causes when analyzing trends in digital analytics, requiring additional manual research and intervention to identify the actual root cause. Moreover, in some cases, these suggestions may confuse the analyst rather than help them identify the root cause. On the other hand, a fine-tuned model can produce more accurate results, reducing the need for manual intervention and the time required to complete the task. This is because a fine-tuned model includes past experiences from resolved cases, enabling it to provide more relevant and accurate suggestions.

For example, if a business has a fine-tuned model for detecting fraudulent transactions in its financial data, the model can accurately identify fraudulent activity and help reduce the need for manual intervention. This can save the business time and resources by identifying fraudulent transactions early on, preventing them from causing further damage.

Furthermore, fine-tuning a model can help businesses adapt to changes in their industry or domain. For instance, a fine-tuned model can help a medical organization keep up with the latest research and terminologies by analyzing scientific papers and providing insights relevant to the organization's needs. This ensures that the organization stays updated with the latest advancements and can make more informed decisions.

- Real-world use of fine-tuned GPT

In the analytics industry, it's essential to keep in mind that numbers can have different meanings depending on the business's specific operations. While industry benchmarks and trends can be helpful, it's crucial to recognize that these general insights may not apply to every business or use case. This is where fine-tuning a GPT

model can be particularly valuable. By incorporating a business's unique knowledge and experience into the analytics process, a fine-tuned model can generate insights tailored to the specific needs of the business. This can provide a more accurate and nuanced understanding of a business's operations, enabling more informed data-driven decisions.

Customizing a GPT model for a specific task, known as fine-tuning, can greatly enhance its performance and save time and effort in processing and analyzing large volumes of data compared to using a generic model. For instance, if a general model is used to analyze a sudden increase in digital analytics trends, it may provide general causes that require additional manual research. In contrast, a fine-tuned model produces more accurate results, reducing the need for manual intervention and the time it takes to complete the task. This is because the fine-tuned model incorporates past experiences from resolved cases, providing more precise and relevant insights.

Fine-tuning a GPT model is particularly beneficial for incorporating a business's language and terminology specific to its industry or domain. By training the model on its own data, the business can teach the model to understand its specific language and improve its ability to provide relevant insights. This can improve the model's accuracy and relevance to the business's specific needs, leading to better data-driven decisions.

In conclusion, fine-tuning a GPT model can provide many benefits for analytics. By training the model on a business's own data, it can generate insights tailored to the specific needs of the business. This can lead to a more accurate understanding of a business's operations, reduce the need for manual intervention, and improve the model's accuracy and relevance to the business's specific needs. As the demand for data-driven insights continues to grow, fine-tuning GPT models will become increasingly valuable for businesses looking to make more informed data-driven decisions.

Chapter 3. Methodology

3.1 Steps for Sentiment Analysis

In this study, we followed systematic steps to conduct sentiment analysis using OpenAI's language model. The following steps, outline the approach we hired to achieve our objectives:

Step 1: Data Selection and Acquisition

The first step involved identifying a suitable dataset for our sentiment analysis. After comprehensive exploration, we chose Amazon reviews dataset available on Kaggle. This dataset presented a diverse range of product reviews, offering multiple textual content for our analysis. The dataset's wide availability and relevance to real-world consumer opinions made it an ideal candidate for our study.

Step 2: Data Preprocessing and Refinement

Upon procuring the Amazon reviews dataset, we recognized the need for data refinement to ensure optimal analysis outcomes. As part of this step, we undertook data cleaning procedures. We reorganized the dataset's categories to enhance clarity and straightforwardness. This preprocessing step streamlined our subsequent analysis, enabling us to focus on sentiment trends without unnecessary complexity.

Step 3: OpenAI API Integration

To use the advanced capabilities of OpenAI's language model, we established an account and acquired an API key. This allowed us to access the language model's sentiment analysis functionality programmatically. The integration of the OpenAI API was necessary in order to execute our sentiment analysis task.

Step 4: Code Implementation for Sentiment Analysis

The main body of our methodology was the development of code to perform sentiment analysis using the OpenAI language model. We wrote code that harnessed the API's capabilities to analyze sentiment within the Amazon reviews dataset. This step marked the bridge between raw data and insightful sentiment predictions, demonstrating the practical application of cutting-edge AI technology in real-world datasets.

Step 5: Creating Examples and Parameter Fine-Tuning

In order to fine-tune our model and understand which parameters worked better for our data set we designed a set of test examples, meticulously chosen from

the dataset, ensuring a representative distribution across various categories. The scale of these test examples was deliberately limited to a maximum of 50 reviews per category. This constrained selection allowed us to gain insights into the interplay between parameter configurations and sentiment predictions within a controlled environment. Having identified parameter settings that demonstrated promise in the limited test examples, we transitioned towards applying these settings to the entire dataset. This final application phase represented the culmination of our efforts to ensure that the model's predictions resonated with sentiments expressed across the broad scale of reviews.

Step 6: Result Extraction and Interpretation

The final step of our methodology involved extracting sentiment analysis results from the OpenAI-powered code. We obtained sentiment predictions for each review text in the dataset. This comprehensive collection of results served as the foundation for our subsequent analyses and insights. By examining the predictions across the dataset, we discerned patterns, discrepancies, and trends in sentiment expressions, providing valuable context for our findings.

Chapter 4. Applying Sentiment Analysis in Amazon Reviews

Implementing Machine Learning (ML) from model into practice can be challenging at times. There are a lot of recourses that one can use to implement ML in the web, or "hand-crafted" algorithms can be used for every specific data set.

In this chapter, we will analyze the methodology that is followed, like:

- the dataset that is used
- the resources (hardware and software)
- the processing procedure

4.1 Data Collection

The data set used is from Kaggle. It contains product reviews and metadata from Amazon. Specifically, every row is a review. The data set includes 10,972 reviews (ratings, helpfulness votes, text), product metadata (descriptions, brand, category information, price, and image features), and links. We chose this dataset due to its diverse collection of product reviews, providing rich textual content for our analysis. Its relevance to real-world consumer opinions made it suitable for our thesis's objectives.

4.2 Data Set

The data set contains 28 distinct columns, each holding vital information about the products and corresponding reviews. Specifically, the dataset's columns are outlined in Table 1 with explanations:

Value	**Explanation**
id	The ID of the product
asins	ASIN stands for Amazon Standard Identification Number. It's a unique identifier of 10 letters and/or numbers for a product that's assigned by Amazon.com. It's primarily used for product-identification within their product catalog of billions of items. *Source: https://www.nchannel.com/blog/amazon-asin-what-*

	is-an-asin-number/
brand	The Brand of the product
categories	The category of the Product
colors	What color the product is (not contains accurate data)
dateAdded	The date that the product was added in Amazon
dateUpdated	The date that the product was last updated
dimension	The size of the product
ean	is a standardized barcode image that represents the 13-digit GTIN on most products customers shop for in the UK and globally, except North America (USA & Canada). (not contains data)
keys	A unique key combining the Name of the product and the ID
manufacturer	Who is the manufacturer of the product
manufacturerNumber	The number of the manufacturer
name	The name of the product
prices	The price of the product, displayed as a json combined with other parts values
reviews.date	The date of the review
reviews.doRecommend	Binary variable containing "false", "truth"
reviews.numHelpful	unknown value
reviews.rating	The rating of the review in numerical value (1-5)
reviews.sourceURLs	The URL of the product
reviews.text	The text of the review
reviews.title	The title of the review
reviews.userCity	The Geolocation of the user (City)
reviews.userProvince	unknown value (Blank)
reviews.username	The username of the user writing the review
sizes	The size of the product
upc	The upc of the product

weight	The weight of the product

Table 1: Overview of the data set's values

4.3 Data Set Alterations

After reviewing the data set we came up with the need to make some alterations to the raw data, such as reduce the data set and rename the categories:

1. Reduced the Data Set

Conscious of the Open AI charge (see 3.1.4) we have reduced the data set to 1.269 rows. This was done randomly, keeping 10% of the reviews for every category and only for reviews that contain rating from 1-5 stars (not blank values). The new reduced data set contains the same columns as the initial Data set.

2. Renamed the categories.

Due to the confusing naming conventions of the initial categories, we have changed the categories with the help of excel (find & replace) to → Accessories, Amazon Echo, Amazon kindle store, Headphones, Home, Mobiles, Power Adapters & Cables, Streaming Media Players, as shown also in table 2.
More specifically the initial categories were 21 and were reduced to 8. That will help for better classification on the categories and for better understanding of the data set.

Previous Value	**New Value**
Amazon Devices,Kindle Store,Kindle Accessories	Accessories
Amazon Devices,Kindle Store,buy a kindle	Accessories
Amazon Devices & Accessories,Amazon Device Accessories,Controllers & Remote Controls,Kindle Store,Amazon Echo Accessories,Remote Controls	Accessories
Amazon Devices & Accessories,Amazon Device Accessories,Kindle Store,Kindle E-Reader Accessories,Kindle Oasis Accessories	Accessories
Amazon Devices,Corded Headsets,Electronics Features,Electronics,Audio,Headphones,Kindle Store,Kindle	Accessories

Accessories	
Amazon Devices & Accessories,Amazon Device Accessories,Controllers & Remote Controls,Kindle Store,Fire TV Accessories,Controllers & Remotes,Controllers	Accessories
Amazon Devices,Electronics,Kindle Store,Amazon Echo	Amazon Echo
Amazon Devices,mazon.co.uk	Amazon kindle store
Kindle Store,Amazon Devices,Electronics	Amazon kindle store
Amazon Devices	Amazon kindle store
Amazon Devices,Kindle Store	Amazon kindle store
Electronics,Amazon Devices	Amazon kindle store
Amazon Devices,Kindle Accessories	Amazon kindle store
Amazon Devices,Electronics,Kindle Store	Amazon kindle store
Flipkart Headphone, Devices & Accessories, Bluetooth Headphone	Headphones
Amazon Devices,Home,Smart Home & Connected Living,Smart Hubs & Wireless Routers,Smart Hubs,Home Improvement,Home Safety & Security,Alarms & Sensors,Home Security,Amazon Echo,Home, Garage & Office,Smart Home,Voice Assistants,Amazon Tap,Electronics Features,TVs & Electronics,Portable Audio & Electronics,MP3 Player Accessories,Home Theater & Audio,Speakers,Featured Brands,Electronics,Kindle Store,Frys,Electronic Components,Home Automation,Electronics, Tech Toys, Movies, Music,Audio,Bluetooth Speakers	Home
Cell Phones & Accessories,Accessories,Screen Protectors,Cell,Amazon Devices,Electronics	Mobiles
Amazon Devices & Accessories,Amazon Device Accessories,Power Adapters & Cables,Kindle Store,Kindle E-Reader Accessories,Kindle Paperwhite Accessories	Power Adapters & Cables

Categories,Amazon Devices,Streaming Media Players	Streaming Media Players
Categories,Amazon Devices,Electronics Features,Streaming Media Players,Consumer Electronics,See more Amazon Fire TV Digital HD Media Streamer (Late...	Streaming Media Players

Table 2: Overview of the product categories' alterations

4.4 Data Set Exploration

The reduced data set contains 1.269 values distributed as in Figure 4 to the respective categories. The mean is 158 values per category. 'Home' and 'Amazon Kindle Store' categories seem to have most of the reviews 542 and 378 respectively, while on the other side 'Amazon Echo' and 'Mobiles' have the lower number of reviews.

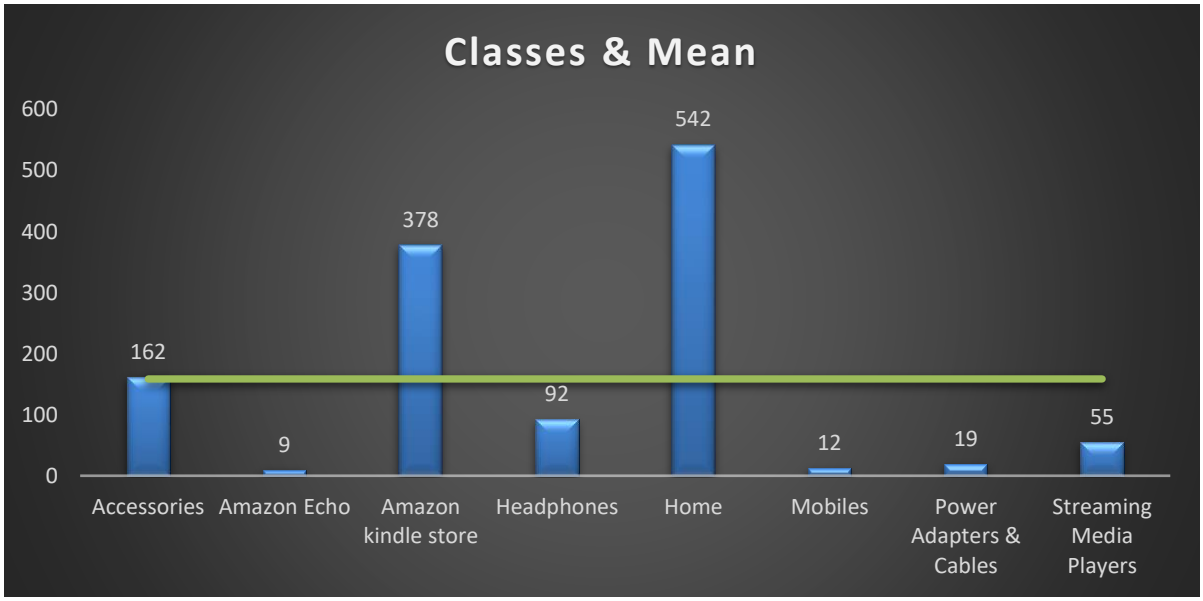


Figure 4: Distribution of classes

Below there are some more information regarding the raw data file that will help us understand the data set.

Figure 5 is showing ratings per month. In January and July the users rated the products more positevly, but this can be attributed to the fact that there are more ratings recorded for these months.

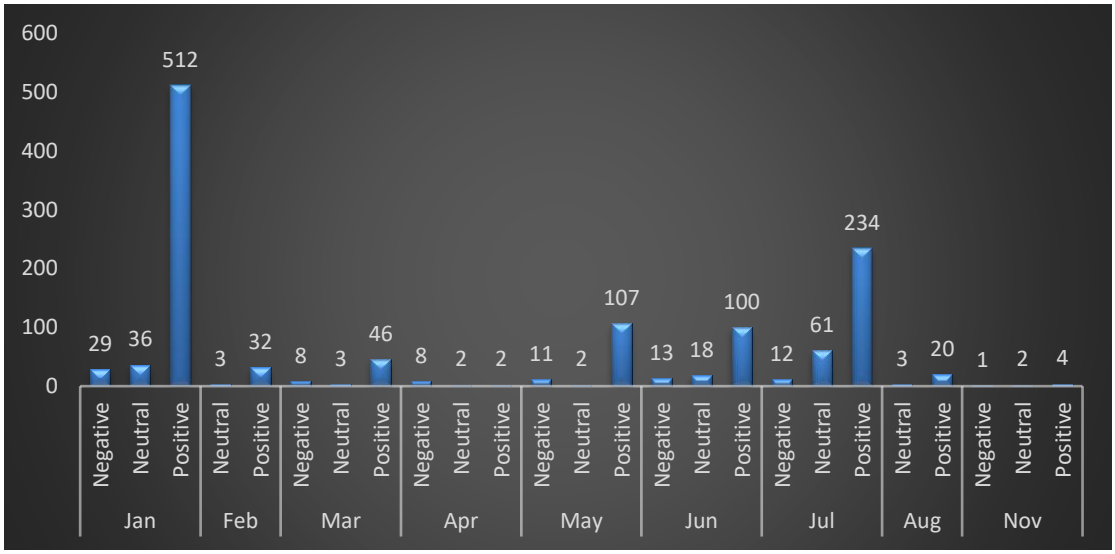


Figure 5: Count of Ratings per month

In the below Figures 6,7,8 one can see how the Positive, Neutral and Negative reviews can be attributed through time.

The common trend across all sentiment types is the higher review ratings observed in January and lower values in April and November. This pattern could be attributed to various factors, such as holiday seasons, product launches, or market trends. These figures collectively offer a dynamic visualization of how consumer sentiments, whether Positive, Neutral, or Negative, manifest across different months, providing valuable insights into the subside and flow of product reviews over time.

Figure 6: Trendline of Positive Ratings per month

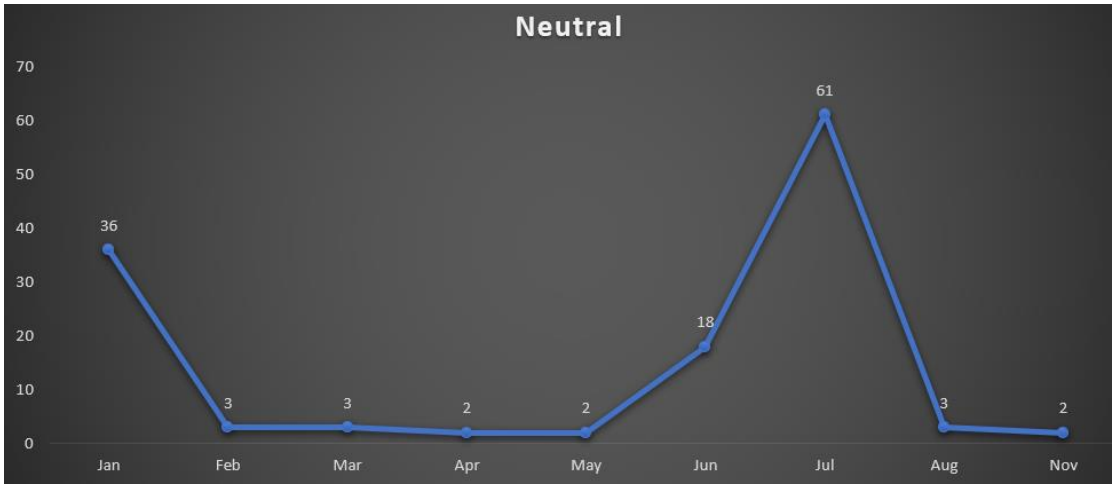


Figure 7: Trendline of Neutral Ratings per month

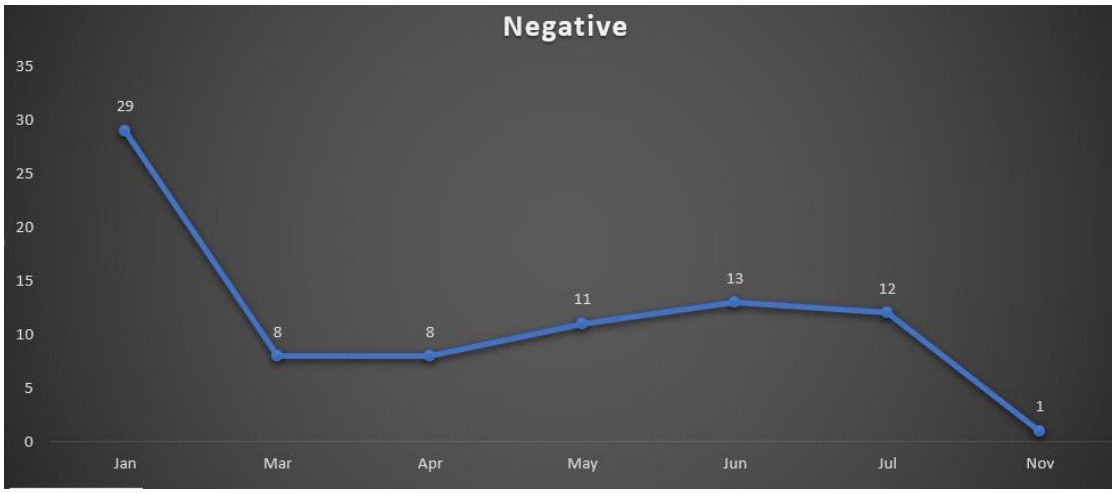


Figure 8: Trendline of Negative Ratings per month

4.5 Software Resources

To perform all the tasks for this thesis, the Python programming language was used.

4.6 Hardware Resources

The program run on a Dell laptop with the below characteristics:

- CPU: 11th Gen Intel(R) Core (TM) i7-1165G7 @ 2.80GHz
- RAM: 16 GB

- Windows 10 PRO

4.7 Cloud Services

- Open AI, the model used "text-davinci-003"
- The Python script was executed on Google's Colab cloud service, which allows users to upload and run Python scripts online. It also provides the capability to save files, including the results, directly to the user's Google Drive or computer.

4.8 Open AI charge

Open AI charges 24.8$ per month to subscribe to GPT-plus that is necessary to obtain an API key. It also charges 0.02$/1K tokens when using the “davinci” model. (see appendix C)

4.7 Sentiment Analysis

Below there is an overview of the procedure step by step, displaying also the outcome of the code

4.7.1 Steps for preparation

Steps on using Chat GPT on text classification:

1. Subscribe to Chat-GPT Plus.
2. Generate an API secret key. (see appendix - C)
3. Create an account to Google Collab

4.7.2 First steps to the analysis

- **Import libraries and the raw data**

There are multiple ways to upload the data.

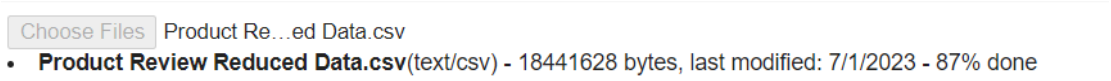
- Directly from the PC
- From Google Drive using code
- From Google Drive using Google Colab

In this example the data set was loaded directly from the PC.

```
import os
import pandas as pd
import io
import matplotlib.pyplot as plt

from google.colab import files

uploaded = files.upload()
```



➢ Load the data set

In the below step the data set was loaded and also the rows were reduced to maximum 50 per category, this is done in order to do some test when fine-tuning the model with a reduce data set. Then **when choosing the most appropriate values for the parameters the sentient analysis will run for all 1.269 reviews**.

```
df = pd.read_csv('Product Review Reduced Data.csv',
encoding='latin-1')

df = df.groupby('main category').head(50)
```

➢ Calculate the tokens

In order to calculate the average number of tokens we took a specimen of 100 different reviews. This helped us understand what value to insert to the max_token parameter when making call to API. More specifically, a token refers to a distinct unit of text. It can be a single word, a punctuation mark, or even a subword in some cases. When processing and analyzing text, it's common to break down the text into tokens to understand and manipulate it better.

For example, consider the sentence: "Today the sun was shining" When tokenized, this sentence becomes a series of tokens: ["Today", "the", "sun", "was", "shining", "."]. Each word and the period at the end is considered a token. Please also refer to appendix – B in order to see a visual calculation of the tokens in OpenAI

```
subset_df = df.head(100)
average_tokens = subset_df['reviews.text'].apply(lambda text:
len(text.split())).mean()
print("The average number of tokens in the 'reviews.text'
column:", average_tokens)
```

Result:

```
The average number of tokens in the 'reviews.text' column: 270.74
```

For this reason we will set the max_tokens in the sentiments analysis code near this value, 200, 300, 400.

- **Sentiment Analysis**

In order to make sentiment analysis using Open AI, the below code with the parameters needs to be used. The below code is a template. Later, we will review all the possible combinations to the parameters and the results each combination gave.

```
def get_sentiment(text):
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt,
          max_tokens=--,
          temperature=--,
          top_p=--,
          frequency_penalty=--,
          presence_penalty=--
      )
```

The parameters used (temperature, max_token, top_p, frequency, presence) have specific roles in controlling the behavior of the OpenAI language model during text completion. Below is an explanation of each parameter and its purpose:

- **Temperature:** The temperature parameter controls the randomness of the generated output. A higher temperature (e.g., 1.0) produces more diverse and creative responses, while a lower temperature (e.g., 0.2) produces more focused and deterministic responses. Setting the temperature to 0 means that the output will be deterministic, with the model always generating the same response given the same input.

- **Max Tokens:** The max_tokens parameter limits the length of the generated text to a specific number of tokens. Tokens can be thought of as pieces of text, such as words or characters. By setting a maximum token limit, one can control the length of the response generated by the model.
- **Top-p (Nucleus) Sampling:** The top_p parameter, also known as nucleus or probabilistic sampling, controls the diversity of the generated output by limiting the cumulative probability distribution of the predicted tokens. Setting it to 1.0 means the model considers all possible tokens, while setting it to a lower value like 0.8 restricts the model to consider only the most likely tokens that add up to 80% cumulative probability. This helps ensure that the generated responses are more focused and clear.
- **Frequency Penalty:** The frequency_penalty parameter controls the penalty applied to tokens based on their frequency in the training data. A higher value (e.g., 1.0) penalizes frequently occurring tokens, discouraging the model from repeating common phrases or responses. By setting it to 0.0,the frequency penalty is disabled, allowing the model to generate responses freely without considering token frequency.
- **Presence Penalty:** The presence_penalty parameter controls the penalty applied to tokens based on their presence in the input prompt. A higher value (e.g., 1.0) penalizes tokens that are already present in the prompt, encouraging the model to generate more diverse and novel responses. Setting it to 0.0 means no presence penalty is applied, allowing the model to use words from the prompt as part of the generated output.

4.8 Fine-tuning the sentiment analysis

In this thesis the sentiment analysis was performed using 2 methods:

- Without specifying the prompt
- With specifying the prompt

A prompt is a specific input provided to a language model to instruct it on what task to perform or what information to generate. It's a text-based instruction or question that sets the context for the model's response. Prompts are useful tools that

allow users to harness the power of language models for a wide range of tasks, from generating text to answering questions, summarizing content, and more.

In order to count how accurate the results are we took the 'reviews.rating' column, from the raw data, that contains numerical values (1-5) and we assigned the values from <=2 to 'Negative', equal to 3 to 'Neutral' and >=4 to 'Positive'. Counting these results there are:

- Positive: 1057 (83%)
- Negative: 130 (10%)
- Neutral: 82 (7%)

That way we can count the number of matches and mismatches with the generated sentiment from the OpenAI.

Firstly we will run and fine-tune the model in the reduced data set that it is maximum 50 rows per category, as stated in 3.2.2 step 'Load the data set'.

- Without specifying the prompt

At first we will run the sentiment analysis without specifying the prompt. In the below code the prompt was assigned as 'text' which is the text of the review as displayed in the raw data.

```
def get_sentiment(text):
    try:
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=text,
          max_tokens=200,
          temperature=1.0,
          top_p=1.0,
          frequency_penalty=0.1,
          presence_penalty=0.1
      )
    except InvalidRequestError:
      return "Error"
    sentiment = response['choices'][0]['text'].strip().lower()
```

When the prompt wasn't specified, the results were shown in table 3, using different parameters to find the ones that generated the best-suited results.

Parameters used	Results
prompt=text, max_tokens=200, temperature=1.0, top_p=1.0, frequency_penalty=1.0, presence_penalty=1.0	Sentiment based on Open AI: Neutral 276 Positive 11 Negative 2 Error 1 Name: sentiment, dtype: int64 Number of matches: 114 Number of mismatches: 176
prompt=text, max_tokens=400, temperature=0.5, top_p=0.5, frequency_penalty=0.5, presence_penalty=0.5	Sentiment based on Open AI: Neutral 276 Positive 11 Negative 2 Error 1 Name: sentiment, dtype: int64 Number of matches: 114 Number of mismatches: 176
prompt=text, max_tokens=200, temperature=1.0, top_p=1.0, frequency_penalty=0.1, presence_penalty=0.1	Sentiment based on Open AI: Neutral 282 Positive 6 Negative 1 Error 1 Name: sentiment, dtype: int64 Number of matches: 114 Number of mismatches: 176

Table 3: Parameters used and results when the prompt is not specified

In the above, we can understand how adjusting the parameters, can affect the results of sentiment analysis using OpenAI's language model. Most of the time, changing these parameters didn't drastically alter the sentiment predictions. However, in one specific case, we noticed something different.

When we changed the parameters in the last example, the OpenAI model produced fewer positive sentiment predictions and more neutral ones. This highlights that adjusting the parameters can sometimes lead to variations in how the model interprets sentiment in text. The time to execute the request without specifying the prompt was approximately 6-7 minutes.

This finding tells us that, while most examples showed consistent sentiment predictions, tweaking the parameters can have an impact. It's a reminder that picking the right parameter values is important to get the sentiment analysis results we want. It also raises questions about how the model's behavior and parameter choices are connected, which is worth exploring further when specifying the prompt.

- With specifying the prompt

In the 2nd case we will run some examples with specifying the prompt, the results were the below using different parameters and prompts to find the ones that generated the best-suited results. In every example, in the highlighted part one can see the changes made comparing to the previous example.

Example 1

```
prompt1 = f"This is a review about a product taken from Amazon.
The review text is: {text}. Which is the sentiment of the
review?"
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt1,
          max_tokens=200,
          temperature=1.0,
          top_p=1.0,
          frequency_penalty=0.1,
          presence_penalty=0.1
```

Code 1: Code of Example 1

This prompt and parameters used in Code 1 generated **215 matches** with the rating of the raw data. At first sight we can understand that the results are much more accurate than the previous case when the prompt wasn't specified. Also, the code run in 3 minutes.

```
Sentiment based on Open AI:
Positive    203
Neutral      61
Negative     23
Error         3
Name: sentiment, dtype: int64

Number of matches: 215
Number of mismatches: 75
```

Figure 9: Results of Example 1

Example 2

```
prompt1 = f"This is a review about a product taken from Amazon.
The review text is: {text}. Can you tell if the sentiment of the
review is positive negative or neutral?"
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt1,
          max_tokens=300,
          temperature=1.0,
          top_p=1.0,
          frequency_penalty=0.5,
          presence_penalty=0.5
```

Code 2: Code of Example 2

This prompt and parameters used in Code 2 along with changing the input of the prompt, generated **236 matches**, hence increasing the 'frequency_penalty' and the 'presence_penalty' increased the success of the results by +8%. This part of the code also run in 3 minutes.

```
Sentiment based on Open AI:
Positive    222
Neutral      43
Negative     22
Error         3
Name: sentiment, dtype: int64

Number of matches: 236
Number of mismatches: 54
```

Figure 10: Results of Example 2

Example 3

```
prompt1 = f"This is a review about a product taken from Amazon.
The review text is: {text}. Can you specify if the sentiment of
the review is positive negative or neutral?"
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt1,
          max_tokens=200,
          temperature=1.0,
          top_p=1.0,
          frequency_penalty=1.0,
          presence_penalty=1.0
```

Code 3: Code of Example 3

In this example (Code 3) we have increased the “frequency_penalty” and the “presence_penalty” to 1.0 and also we have changed one word of the prompt (tell→ specify). These small changes resulted to **227 matches** which are less than the previous example.

```
Sentiment based on Open AI:
Positive    210
Neutral      54
Negative     23
Error         3
Name: sentiment, dtype: int64

Number of matches: 227
Number of mismatches: 63
```

Figure 11: Results of Example 3

Example 4

```
prompt1 = f"This is a review about a product taken from Amazon.
The review text is: {text}. Can you tell if the sentiment of the
review is positive negative or neutral?"
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt1,
          max_tokens=400,
          temperature=1.0,
          top_p=0.8,
          frequency_penalty=0.8,
          presence_penalty=0.8
```

Code 4: Code of Example 4

In this example (Code 4) we have increased the “max_tokens” to 400 and lowered the ‘frequency_penalty’, ‘presence_penalty’ and the “top_p” to 0.8, allowing the model to generate responses freely without considering token frequency. Also, we reverted the ‘specify’ word to ‘tell’. Using this approach the model generated **241 matches** which is the best performance out of all the examples made.

```
Sentiment based on Open AI:
Positive    224
Neutral      39
Negative     24
Error         3
Name: sentiment, dtype: int64

Number of matches: 241
Number of mismatches: 49
```

Figure 12: Results of Example 4

Example 5

```
prompt1 = f"This is a review about a product taken from Amazon.
The review text is: {text}. Can you tell if the sentiment of the
review is positive negative or neutral?"
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt1,
          max_tokens=400,
          temperature=1.0,
          top_p=0.8,
          frequency_penalty=0.5,
          presence_penalty=0.5
```

Code 5: Code of Example 5

Lastly, in this example (Code 5) we have lowered the 'frequency_penalty' and the 'presence_penalty' to 0.5. Using this approach the model generated **237 matches** which is the 2nd best performance out of all the examples made.

```
Sentiment based on Open AI:
Positive    221
Neutral      43
Negative     23
Error         3
Name: sentiment, dtype: int64

Number of matches: 237
Number of mismatches: 53
```

Figure 13: Results of Example 5

The 5 above examples were the ones worth mentioning as we did more tests on fine-tuning the model. In most of the cases, the results had small variations between them.

4.9 Final sentiment analysis

After fine-tuning the model using 2 different cases (with/without prompt) and multiple examples, we have run the model to the whole data set using the parameters and the prompt of the 'Example 4' that generated the most accurate results, to the whole data set.

We have included in the prompt the following question "*This is a review about a product taken from Amazon. The review text is: {text}. Can you tell if the sentiment of the review is positive negative or neutral*?". This question aimed to provide context and guide the model's understanding of the sentiment expression within the review text.

The parameters were set as:

- max_tokens=400,
- temperature=1.0,
- top_p=0.8,
- frequency_penalty=0.8,
- presence_penalty=0.8

Based on Figure 14 Open AI generated 993 Positive reviews (78%), 195 Neutral (15%) and 78 Negative (7%). The errors refer to reviews that are more than 4000 tokens and Open AI cannot process. **The number of matches is 81%.**

```
Sentiment based on Open AI:
Positive    993
Neutral     195
Negative     78
Error         3
Name: sentiment, dtype: int64

Number of matches: 1028
Number of mismatches: 241
```

Figure 14: Results of Final Sentiment Analysis

Figure 15 provides a visual representation that illustrates the distribution between the two outcomes of our sentiment analysis comparison: "match" and "mismatch." This representation offers an insightful visualization of the correspondence or discrepancy between the sentiments generated by OpenAI and the textual ratings we established based on the reviews' numerical ratings.

Each bar in the graph represents either a "match" or a "mismatch" between the sentiments. A "match" indicates instances where OpenAI's sentiment prediction aligned with the manually derived textual rating. On the other hand, a "mismatch" signifies cases where there was a disparity between OpenAI's prediction and our assigned textual rating.

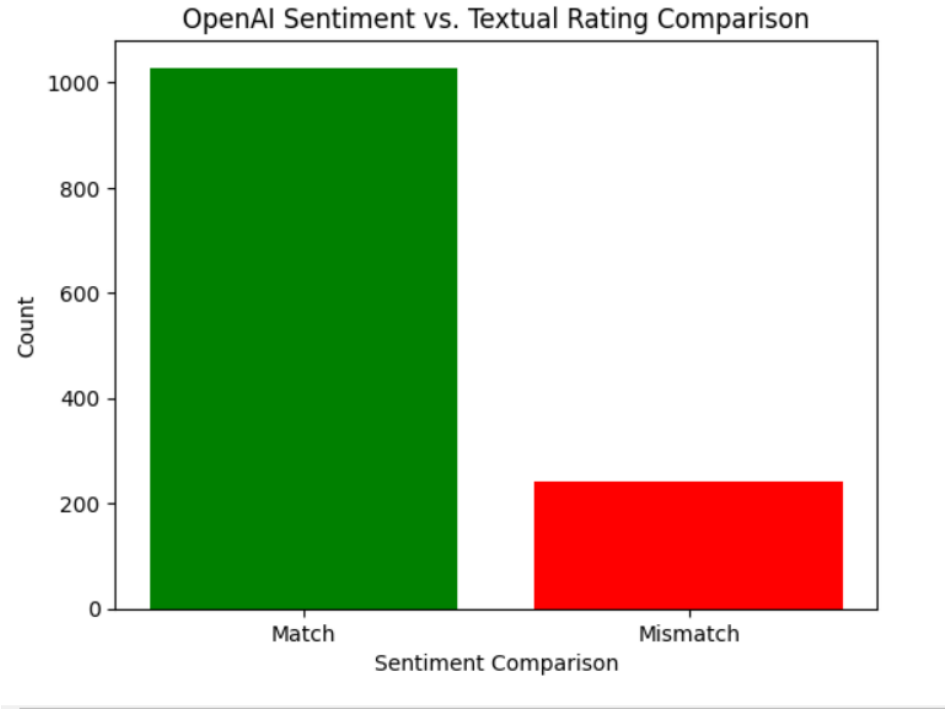


Figure 15: Sentiment Comparison

4.10 Comparison of the raw data rating and OpenAI

Using specific prompts in our sentiment analysis process brought out really impressive results. These prompts helped the model better understand the sentiments in the reviews, and this improvement is evident when we look at the table below.

When we look at how sentiments matched up (Table 4), we can see that positive sentiments were very close in both the raw data and OpenAI's predictions. About 83% of the time, the raw data sentiments matched the 78% positive predicted

by OpenAI. This small difference of 5% shows that OpenAI did a good job understanding the positive feelings expressed in the reviews.

Similarly, negative sentiments also had a good match. Around 10% of the raw data sentiments aligned with the 15% predicted by OpenAI. This 5% difference indicates that OpenAI was effective in recognizing and reflecting the negative opinions shared by reviewers.

Additionally, the neutral sentiments were pretty much the same in both raw data (7%) and OpenAI's predictions (7%). This consistency highlights OpenAI's ability to understand neutral expressions accurately. Lastly, there were 3 errors when running the model.

Table 4 clearly shows that OpenAI's sentiment predictions greatly improved with the help of specific prompts. The fact that the percentages of sentiments matched so closely indicates that OpenAI can accurately understand the sentiments in reviews when guided by these prompts. **The number of matches were 1.028**, so Open AI could successfully understand the sentiment of the analysis 81%.

Sentiment	Raw data sentiment	OpenAI sentiment	% Raw data sentiment	% OpenAI sentiment	Difference
Positive	1,057	993	83%	78%	5%
Negative	130	195	10%	15%	5%
Neutral	82	78	7%	7%	0%
Errors	-	3	-	-	-
	1,269	1,266			

Table 4: Comparison of the results

Chapter 5. Findings & Discussion

This chapter presents the results from the research and analysis conducted in this study. The analysis revealed patterns, trends, relationships, and important discoveries. However, as previously said this thesis is more about finding how marketers can take advantage of this new AI feature, than finding results from the data set.

5.1 Findings of the analysis

The below table shows the text of the review with the rating of the raw data (Textual Rating), the generated sentiment of the OpenAI (OpenAI Sentiment) and the comparison of the 2 sentiments which assigns ‘Match’ or ‘Mismatch’ (Sentiment Comparison’.

Main Category	Review Text	Textual Rating	OpenAI Sentiment	Sentiment Comparison
Amazon kindle store	I initially had trouble deciding between the paper white and the voyage because reviews more or less said the same thing: the paperwhite is great, but if you have spending money, go for the voyage.Fortunately, I had friends who owned each, so I ended up buying the paperwhite on this basis: both models now have 300 ppi, so the 80 dollar jump turns out pricey the voyage's page press isn't always sensitive, and if you are fine with a specific setting, you don't need auto light adjustment).It's been a week and I am loving my paperwhite, no regrets! The touch screen is receptive and easy to use, and I keep the light at a specific setting regardless of the time of day. (In any case, it's not hard to change the setting either, as you'll only be changing the light level at a certain time of day, not every now and then while reading).Also glad that I went for the international shipping option with Amazon. Extra expense, but delivery was on time, with tracking, and I didnt need to worry about customs, which I may have if I used a third party shipping service.	Positive	Positive	Match
Amazon kindle store	I liked the case but unfortunately I purchased the wrong one for my Kindle, my fault. Amazon, as usual, there was no problem getting a refund.	Positive	Neutral	Mismatch
Amazon kindle store	I'm loving my new Fire 7. I was looking for another Fire 6, and since it was not available and this 7 is only a little larger, comes in great colors, AND has Alexa - I was hooked and couldn't wait for it to arrive.	Positive	Positive	Match
Amazon kindle store	While this case fits a 7th Gen Fire perfectly, there is no comfortable way to hold it and it doesn't stand up on it's own for viewing. Returning to try another style	Negative	Neutral	Mismatch
Amazon kindle store	Good looking and works well. If you're having trouble standing it up vertically or horizontally check out Amazon's video (link below). My only slight quibble is that I find the Fire 7 slippery (it's fallen out of my hand) and I was hoping this case might have more of a grippy material.	Positive	Positive	Match
Amazon kindle store	Fits beautifully - snug and sweet.	Positive	Positive	Match
Amazon kindle store	Love the color of the cas and seems to protect well. Loving this case.	Positive	Positive	Match

Figure 16: Sentiment per category and review

Taking as an example the above reviews, can see that OpenAI can distinguish the text's sentiment when the sentiment is stated very clearly like review "Fits beautifully – snug and sweet" where the result of OpenAI matches the rating of the raw data.

However, texts like the in Figure 17 have mixed sentiment. Although the user liked the product and the rating was Positive, he uses the words "but unfortunately" that can confuse the model that indicated this review as neutral.

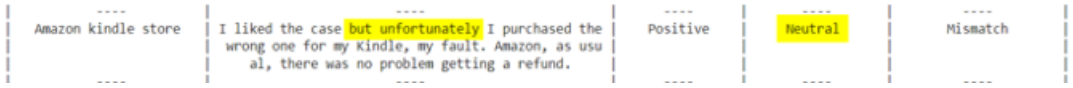


Figure 17: Example of a review

Likewise, in Figure 18, while the user rate the product with a negative review, he used some words with clearly positive sentiment. OpenAI here has also confused and rated this review as neutral.

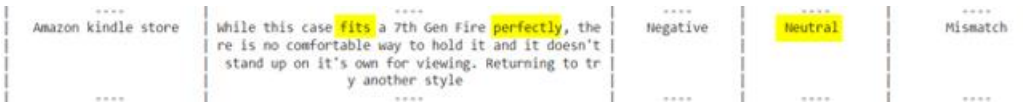


Figure 18: Example of a review

All the above cases make us dig deeper. All words used and language used, highlight the complex mix of language, context, and model behavior that users use and can influence the sentiment analysis.

Moving on with the analysis of the results, below there is a bar chart that shows the distribution of the generated sentiment by negative, positive, neutral.

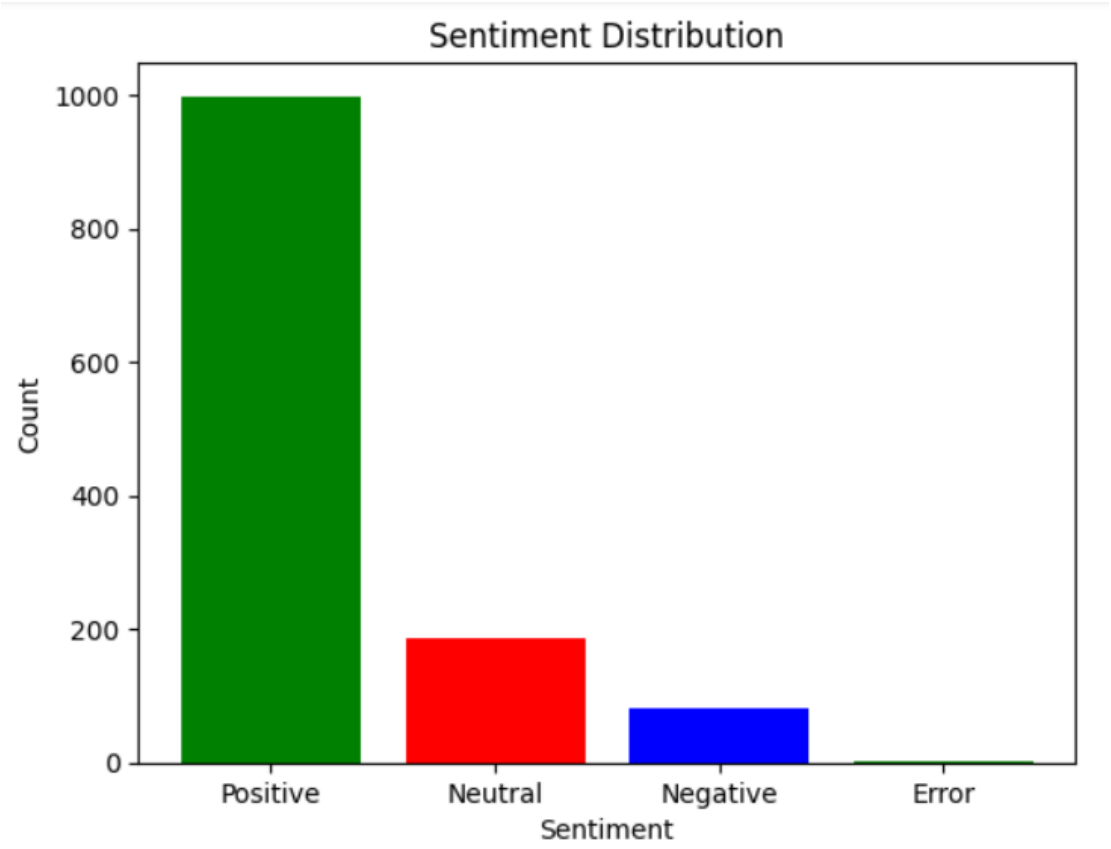


Figure 19: Sentiment Contribution

5.1.1 Results

Our research reveals a strong consistency in the sentiment predictions produced by OpenAI's language model when the prompt is specified. This suggests that the model understands sentiment tones within text naturally more when specifying the prompt.

By specifying a prompt, we essentially guide the model to focus on a specific task or direction, creating a contextual framework for its analysis. This structured approach appears to result in a more refined and accurate understanding of sentiment tones embedded within the text. The prompt, acting as a cue, seems to prompt the model to pay closer attention to the underlying emotional expressions conveyed by the reviewers.

This observation underscores the model's capacity to adapt to contextual signals, leveraging the additional guidance provided by the prompt to enhance its sentiment analysis. The strengthened alignment between predicted sentiments and the

actual emotions expressed in the text highlights the model's remarkable ability to decipher and reflect the sentiment tones with greater precision. Ultimately, the role of the prompt becomes pivotal in fine-tuning the model's comprehension of sentiments, leading to predictions that closely resonate with the human-generated textual ratings.

Lastly it is strongly agreed that there are two important ways to make the model's predictions better.

Firstly, we can adjust the model's settings (fine-tuning), like how much it sticks to the rules (parameters). Changing these settings can make the model's predictions match what we expect.

Secondly, having more examples in our dataset can also help. Some positive and negative reviews didn't get predicted correctly, and that shows we need more variety in the examples the model learns from. It's like learning from more types of music to understand songs better.

In conclusion, **for better sentiment predictions, we should fine-tune the model's settings, specify the prompt and give it a wider range of examples to learn from.**

5.2 Comparing the limitations of Open AI with Python's classic sentiment analysis method

5.2.1 Sentiment Analysis with Python

Python's classic sentiment analysis methods have been widely used in various domains to analyze and understand textual data. These methods involve traditional machine learning techniques and rule-based approaches to classify text into positive, negative, or neutral sentiments. While these methods have been effective in certain scenarios, they have limitations that need to be considered when comparing them to more advanced techniques.

One limitation of classic sentiment analysis methods is the reliance on handcrafted features. These methods typically require manual feature engineering, where specific linguistic or contextual features are extracted from text, such as word frequencies, n-grams, or syntactic patterns. This process, which was also described in the literature review, can be time-consuming and may not capture all the aspects of

sentiment. In contrast, advanced sentiment analysis techniques, such as deep learning-based models, can automatically learn relevant features from raw text, eliminating the need for manual feature engineering.

Another limitation is the need for labeled training data. Classic sentiment analysis methods often rely on pre-labeled datasets for training their models. These datasets require human annotation, which can be subjective and time-intensive. These models may not generalize well to new domains or languages without additional labeled data. In contrast, advanced sentiment analysis techniques, such as transfer learning or pre-trained language models, can leverage large-scale labeled datasets from various sources, enabling them to capture a broader range of sentiments and generalize to different domains and languages.

Classic sentiment analysis methods also struggle with handling complex linguistic incidents, such as sarcasm, irony, or context-dependent sentiment. These methods may rely on simple lexical or syntactic patterns, which can lead to misclassification or inaccurate sentiment analysis results. Advanced techniques, such as deep learning models with attention mechanisms or contextual embeddings, can capture more subtle linguistic cues and contextual dependencies, enabling them to better handle complex sentiments.

Furthermore, classic sentiment analysis methods may not effectively handle sentiment polarity shift, where sentiment changes within a sentence or document. These methods often assign a single sentiment label to the entire text, disregarding the fine-grained sentiment variations. In contrast, advanced techniques, such as aspect-based sentiment analysis or fine-grained sentiment classification, can identify sentiment shifts at a more granular level, providing more detailed and accurate sentiment analysis results.

In conclusion, while Python's classic sentiment analysis methods have served as a fundamental approach to sentiment analysis, they have limitations in terms of manual feature engineering, reliance on labeled training data, handling complex linguistic phenomena, and capturing fine-grained sentiment variations. Advanced sentiment analysis techniques, such as deep learning models and contextual embeddings, offer more powerful and flexible solutions, allowing for automated feature learning, better generalization, and improved handling of complex sentiments. By leveraging these advanced techniques, researchers and practitioners can achieve more accurate and nuanced sentiment analysis results in diverse applications.

5.2.2 Sentiment Analysis with OpenAI Comparing to classic sentiment analysis

While OpenAI provides powerful natural language processing capabilities, it also has certain limitations when compared to traditional Python sentiment analysis methods and it is important to consider these limitations when evaluating the suitability of OpenAI for sentiment analysis tasks.

One limitation of OpenAI is the reliance on cloud-based API services. OpenAI's language models, such as **GPT**, **require an internet connection** and API access to generate responses. This dependence on external services can have limitations in terms of usage quotas or pricing plans. In contrast, traditional **Python** sentiment analysis methods **can be implemented locally** without relying on external services, offering more control and flexibility.

Another limitation is the complexity and potential "black-box" nature of OpenAI's models. While **OpenAI** provides high-level APIs for accessing their models, **the underlying architecture and training processes are complex and not directly transparent** to users. This lack of transparency can make it challenging to understand how the model generates its responses or to debug and troubleshoot any issues that may arise. In contrast, traditional **Python** sentiment analysis methods are often based on well-established algorithms and **can be more transparent** and interpretable.

OpenAI's models are primarily trained on large-scale general-purpose datasets, which may not always align perfectly with specific sentiment analysis tasks or domain-specific nuances. **Fine-tuning** or customizing these models for sentiment analysis tasks **can be a complex and resource-intensive process**, requiring significant amounts of labeled training data and computational resources. In contrast, traditional **Python** sentiment analysis methods **can be tailored and fine-tuned specifically for the task** at hand, allowing for more targeted analysis and customization.

Furthermore, **OpenAI's** language models, **may generate responses that are not necessarily accurate** or aligned with specific sentiment analysis requirements. These models are trained on vast amounts of diverse data and may exhibit biases or generate outputs that do not fully align with human judgment or domain-specific knowledge. In contrast, traditional **Python** sentiment analysis methods **can be**

designed and fine-tuned to prioritize specific sentiment analysis criteria and align with expert knowledge or specific requirements.

5.2.3 Best-suited Approach for Digital Marketing

OpenAI offers several advantages for marketers when it comes to sentiment analysis compared to traditional Python-based approaches. Here are five main reasons why OpenAI can be more useful for marketers:

1. OpenAI's language model, such as GPT, is trained on large amounts of data, enabling them to understand and generate human-like text. This advanced natural language processing capability allows marketers to analyze sentiments in a more context-aware manner. The model can comprehend complex language structures, idiomatic expressions, and even detect sensitive sentiment cues like sarcasm, enabling marketers to gain deeper insights from customer feedback, social media posts, or product reviews.
2. It can handle large volumes of text data efficiently. This scalability is particularly beneficial for marketers dealing with vast amounts of customer feedback, social media conversations, or online reviews. By utilizing OpenAI, marketers can process and analyze extensive datasets quickly, gaining a comprehensive understanding of overall sentiment trends and patterns across various channels and platforms.
3. It also allows marketers to perform sentiment analysis in real-time, providing near-instantaneous results. This capability is crucial in fast-paced marketing environments where immediate feedback and insights are necessary for making timely decisions. Marketers can leverage OpenAI to monitor brand sentiment, track campaign performance, and respond swiftly to emerging trends or customer sentiments, enabling proactive engagement and reputation management.
4. It shines at understanding the contextual meaning of text, considering the broader context in which sentiments are expressed. This contextual understanding is invaluable for marketers, as sentiments can vary significantly based on factors such as product features, brand reputation, or industry-specific terminology. OpenAI can help marketers identify nuanced sentiments, detect sarcasm or irony, and distinguish between positive sentiments towards

specific aspects and overall negative sentiments, providing a more comprehensive and accurate sentiment analysis.

5. OpenAI's model is not limited to sentiment analysis alone. It has the potential to generate creative content and personalized responses based on user input. Marketers can leverage this capability to enhance customer experiences, personalize marketing messages, and create engaging content that resonates with their target audience.

In conclusion, OpenAI's advanced natural language processing capabilities, scalability, real-time analysis, contextual understanding, and potential for creativity and personalization make it a valuable tool for marketers in sentiment analysis tasks. By harnessing OpenAI's capabilities, marketers can gain deeper insights from large-scale data, respond promptly to customer sentiments, understand the contextual nuances of sentiment expressions, and enhance their marketing strategies with AI-generated content. Ultimately, OpenAI empowers marketers to make data-driven decisions, engage effectively with customers, and drive impactful marketing campaigns.

Chapter 6. Conclusion – Further discussion

Summing up, in this thesis we reviewed the literature on NLP in digital marketing and identified key challenges and opportunities, then explored the current state of NLP in digital marketing and investigated its potential to improve customer engagement and conversion rates. To back up the belief that OpenAI can be a game changer in the Marketer's job, a sentiment analysis of customer reviews on Amazon, though fine-tunning GPT was performed. Our research aimed to provide a deeper understanding of the potential of using GPT in digital marketing and to demonstrate its value to businesses looking to improve their customer engagement and conversion rates. The results were impressive as OpenAI could successfully identify 81% of the sentiment of the review correctly (1.028 out of 1.269 reviews).

In conclusion, leveraging OpenAI for sentiment analysis offers significant advantages for marketers compared to traditional Python-based methods. OpenAI's advanced natural language processing capabilities enable a more understanding of sentiments, allowing marketers to gain deeper insights from customer feedback, social media posts, and product reviews. The scalability of OpenAI's models also enables efficient processing of large volumes of text data, providing a comprehensive view of sentiment trends across different channels and platforms. Real-time analysis empowers marketers to make timely decisions, monitor brand sentiment, and respond quickly to emerging trends.

OpenAI's contextual understanding is a game-changer for marketers. It can detect subtle sentiments, sarcasm, irony, and differentiate between positive sentiments towards specific aspects and overall negative sentiments. This contextual understanding enhances the accuracy of sentiment analysis, giving marketers a more comprehensive understanding of customer perceptions in different contexts.

Moreover, OpenAI's potential for creativity and personalization opens up new opportunities for marketers. AI-generated responses can be used to create engaging content and deliver personalized marketing messages. This capability allows marketers to enhance customer experiences and run tailored marketing campaigns that resonate with their target audience.

Looking ahead, it is essential to stay updated with OpenAI's advancements in sentiment analysis. Marketers can explore integrating

OpenAI's sentiment analysis with other marketing analytics tools and platforms to gain a holistic view of customer sentiment.

Ethical considerations should also be taken into account, such as ensuring diverse and unbiased training datasets to mitigate potential biases in sentiment analysis results.

Bibliography

1. Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
2. Bock, E. (2017). Chatbots in customer service: How to create an effective virtual assistant. Harvard Business Review Digital Articles.
3. Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., & Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. Computational linguistics, 19(2), 263-311.
4. Cambria, E., Schuller, B., Xia, Y., & Havasi, C. (2013). New avenues in opinion mining and sentiment analysis. IEEE Intelligent Systems, 28(2), 15-21.
5. Gershman, S., Blei, D., & Tenenbaum, J. (2015). Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. Science, 349(6245), 273-278.
6. Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine, 29(6), 82-97.
7. Jurafsky, D., & Martin, J. H. (2019). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Cambridge University Press.
8. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 260-270).
9. Li, N., Chen, C., & Liu, Y. (2019). Personalized recommendation with deep neural networks. IEEE Transactions on Cybernetics, 49(7), 2153-2165.
10. Li, X., & Liu, Y. (2011). Opinion mining and sentiment analysis: A survey. ACM Computing Surveys (CSUR), 43(2), 1-19.
11. Li, Y., Li, X., & Liu, Y. (2015). Sentiment analysis and opinion mining: A survey. Synthesis Lectures on Human Language Technologies, 8(1), 1-167.
12. Manning, C. D., & Schutze, H. (1999). Foundations of statistical natural language processing. MIT press.
13. Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge university press.
14. Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B., & Xiang, L. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. arXiv preprint arXiv:1602.06023.
15. Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1-2), 1-135.
16. Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1-2), 1-135.

17. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2018). Language models are unsupervised multitask learners.
18. Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. AAAI/IAAI, 98, 55-62.
19. Wang, C., Li, X., & Liu, W. (2020). A survey of spam filtering techniques. Journal of Network and Computer Applications, 166, 102736.
20. Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. IEEE Computational Intelligence Magazine, 13(3), 55-75.
21. Zhang, L., Liu, B., & Li, Y. (2018). Deep learning for sentiment analysis: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), e1253.
22. Zhang, X., & Lee, W. S. (2006). A machine learning approach to anti-spam filtering. ACM Transactions on Information Systems, 24(4),
23. Zhang, Y., Zhang, D., & Li, H. (2018). Generative models for content generation in marketing. Proceedings of the ACM Conference on Recommender Systems.
24. Zhou, X., & Li, C. (2010). Dimensionality reduction in text classification via principal component analysis. Expert Systems with Applications, 37(6), 4216-4223.
25. Tsesmetzis, S. (2023) A comprehensive guide for fine-tuning a GPT-3 model, https://baresquare.com/blog/guide-for-fine-tuning-a-gpt-3-model-for-analytics, accessed on 20/06/2023
26. Hufford, J. (2023) Amazon ASIN: What is an ASIN number?, accessed on 01/07/2023
27. Dieckmann, J. (2023). How To Use ChatGPT API for Direct Interaction From Colab or Databricks, https://pub.towardsai.net/how-to-use-chatgpt-api-for-direct-interaction-from-colab-or-databricks-39969a0ead5f, accessed on 22/6/23
28. OpenAI website, https://platform.openai.com/examples/default-adv-tweet-classifier, accessed on 02/07/2023
29. Text Manipulation using OpenAI, https://platform.openai.com/examples/default-adv-tweet-classifier, accessed on 03/07/2023
30. Webnadu Digital Marketing Company, https://www.webnadu.in/2023/01/webnadu-digital-marketing-company.html, accessed on 03/07/2023

Appendix

Appendix - A

OpenAI offers different language models for various tasks. These models include GPT-3, which is versatile for things like writing and translation. The "davinci" version is good for creative writing, while "curie" can be tailored for specific jobs. "Babbage" is more budget-friendly, and "Ada" is efficient for text tasks. These models are used by developers to make chatbots, websites, and other programs understand and generate language better. The model you choose depends on what you want to do and how much computer power you have.

In the table below you can find the different models of the OpenAI.

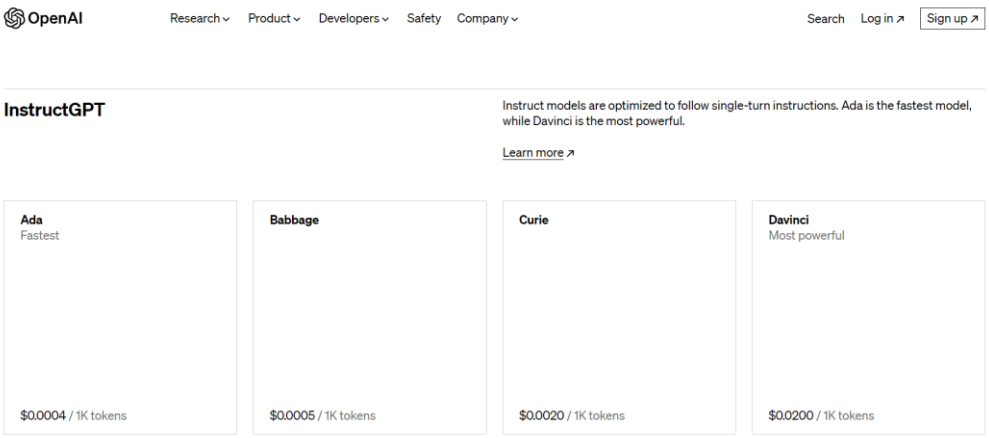


Figure 20: OpenAI models

Appendix – B

A token refers to a distinct unit of text. It can be a single word, a punctuation mark, or even a subword in some cases. When processing and analyzing text, it's common to break down the text into tokens to understand and manipulate it better.

Below is an example of how many tokens just one review has.

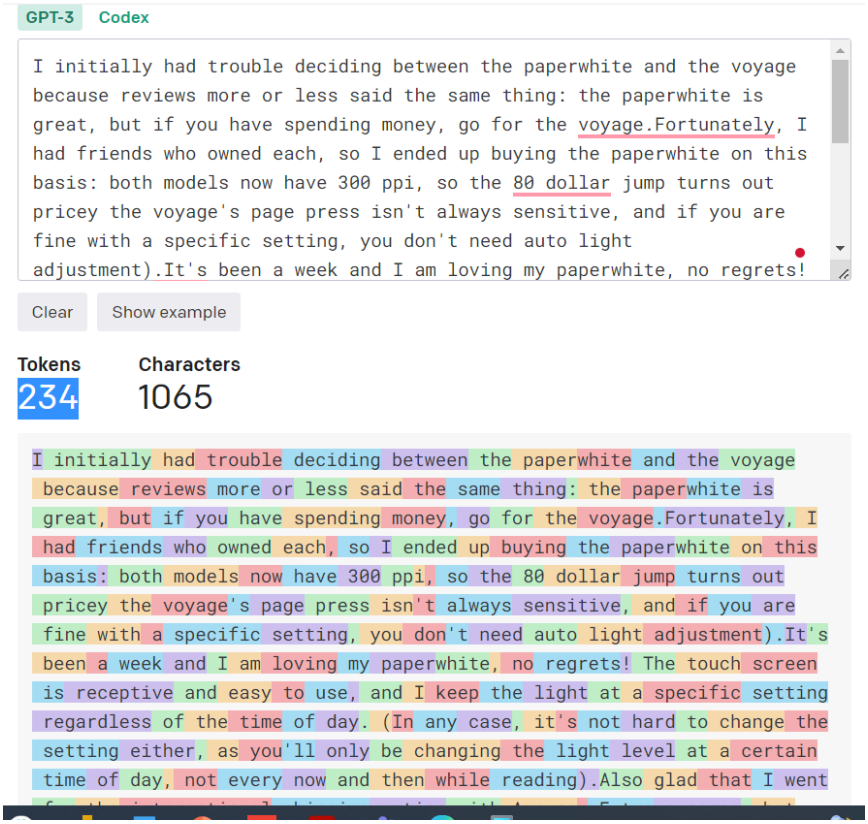
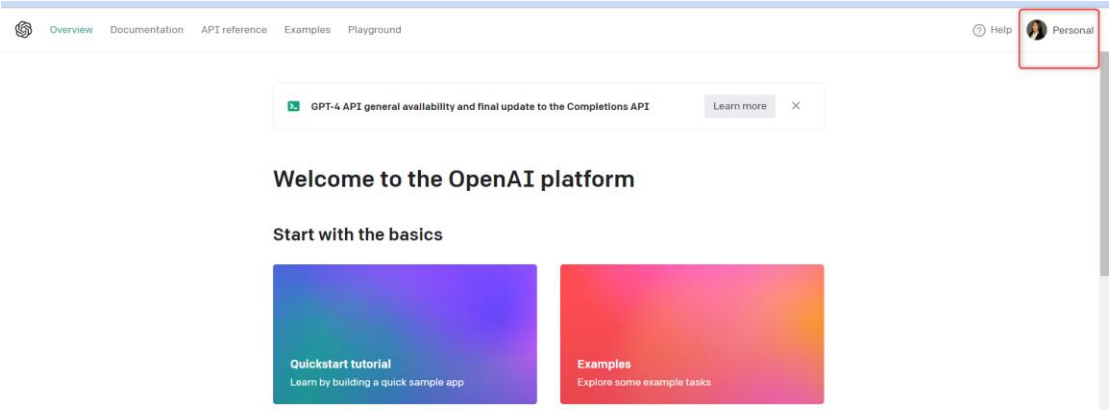


Figure 21: Example of tokenization

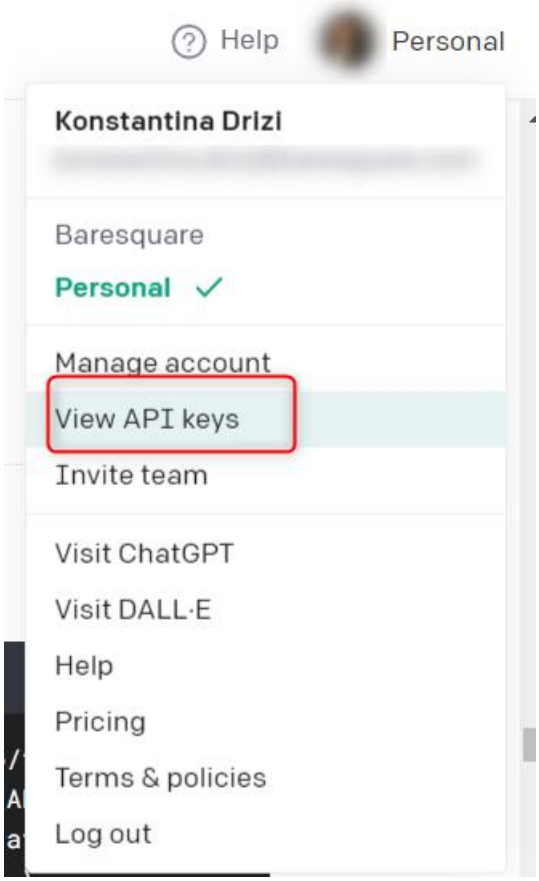
Appendix – C

Steps to generate the key to OpenAI API after subscribing to ChatGPT plus.

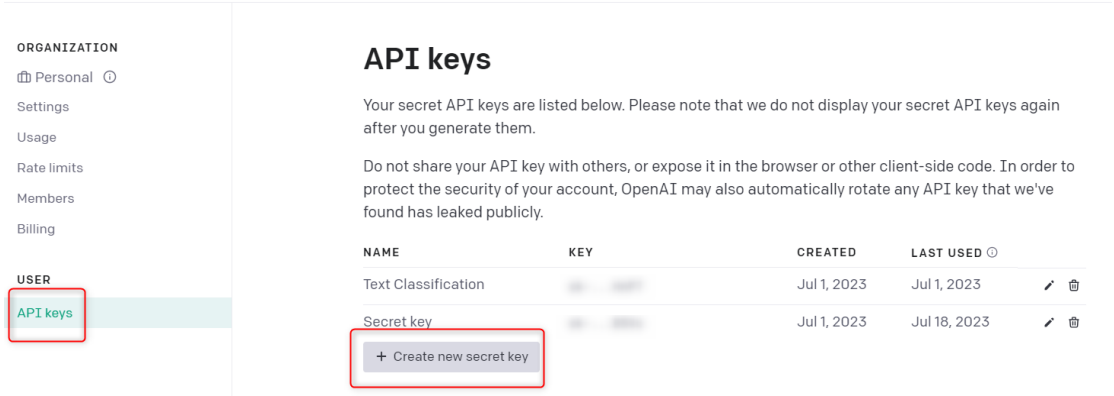
1. Enter OpenAI



2. Go to ‘View API keys’



3. Create new secret key



Appendix – D

Charge of OpenAI to conduct this thesis. OpenAI also charges + 24.8$ the subscription per month in order to be able to generate an API key.

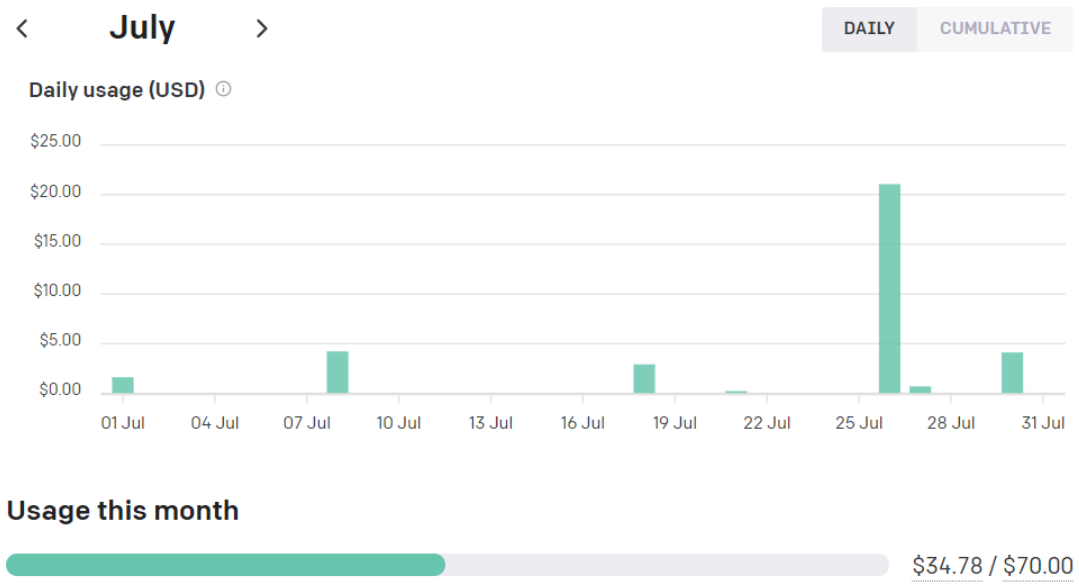


Figure 22: Usage for July

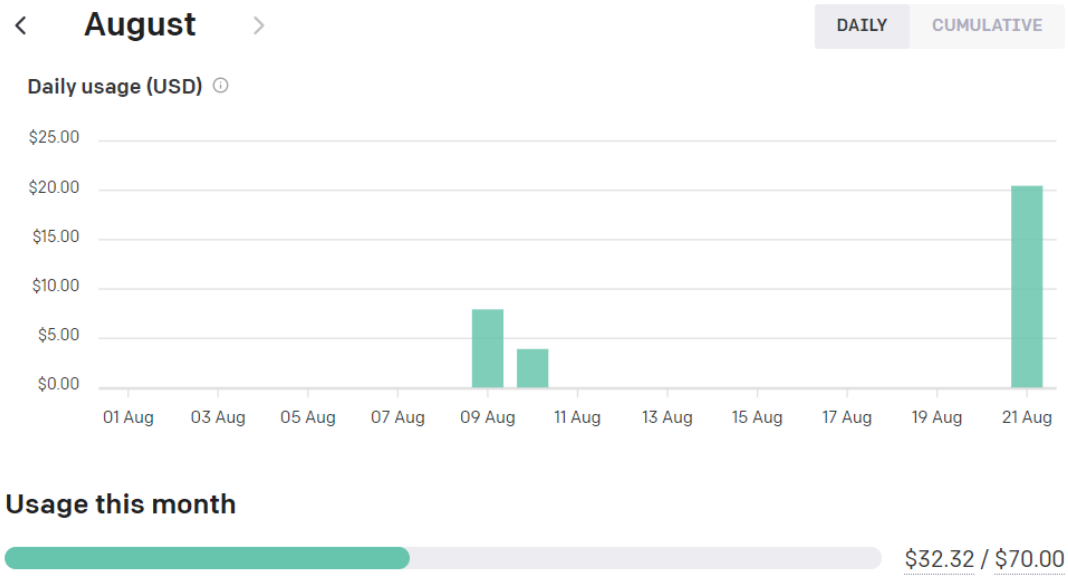


Figure 23: Usage for August

Appendix – E

Below the whole used code is displayed with comments.

```
#Import libraries and the raw data
import os
import pandas as pd
import io
```

```
import matplotlib.pyplot as plt

from google.colab import files

uploaded = files.upload()

#Install OpenAi and also imprort InvalidRequestError, which is
essential in order to exclude this error from the sentiment loop

!pip install openai

import openai

from openai import InvalidRequestError

#Install prettytable library for easily displaying tabular data
in a visually appealing table format.

!pip install prettytable

from prettytable import PrettyTable

# Load the dataset
df = pd.read_csv('Product Review Reduced Data.csv',
encoding='latin-1')

# Calculate the average number of tokens in the "reviews.text"
column

subset_df = df.head(100)  # 100 is the speciment
average_tokens = subset_df['reviews.text'].apply(lambda text:
len(text.split())).mean()
print("The average number of tokens in the 'reviews.text'
column:", average_tokens)

openai.api_key = "sk-……"


# Perform sentiment analysis on "reviews.text" using OpenAI API
def get_sentiment(text):
    try:
      prompt1 = f"This is a review about a product taken from
Amazon. The review text is: {text}. Can you tell if the sentiment
of the review is positive negative or neutral?"
      response = openai.Completion.create(
          engine="text-davinci-003",
          prompt=prompt1,
          max_tokens=400,
```

```
            temperature=1.0,
            top_p=0.8,
            frequency_penalty=0.8,
            presence_penalty=0.8
        )
    except InvalidRequestError:
        return "Error"
    sentiment = response['choices'][0]['text'].strip().lower()

# Convert sentiment into "Positive," "Negative," or "Neutral"
    if 'positive' in sentiment:
        return 'Positive'
    elif 'negative' in sentiment:
        return 'Negative'
    else:
        return 'Neutral'

# Apply sentiment analysis on the 'reviews.text' column
df['sentiment'] = df['reviews.text'].apply(get_sentiment)

#Convert numerical ratings from "reviews.rating" into textual
ratings
def map_rating_to_text(rating):
    if rating >= 4.0:
        return 'Positive'
    elif rating <= 2.0:
        return 'Negative'
    else:
        return 'Neutral'

df['textual_rating'] =
df['reviews.rating'].apply(map_rating_to_text)

raw_data_positive = (df['textual_rating'] == 'Positive').sum()
raw_data_negative = (df['textual_rating'] == 'Negative').sum()
raw_data_neutral = (df['textual_rating'] == 'Neutral').sum()



# Function to compare OpenAI sentiment analysis with textual
ratings
def compare_sentiment(textual_rating, openai_sentiment):
    if 'positive' in openai_sentiment.lower() and textual_rating
== 'Positive':
        return 'Match'
    elif 'negative' in openai_sentiment.lower() and
textual_rating == 'Negative':
        return 'Match'
```

```
    elif 'neutral' in openai_sentiment.lower() and textual_rating
== 'Neutral':
        return 'Match'
    else:
        return 'Mismatch'

# Apply comparison between OpenAI sentiment and textual ratings
df['sentiment_comparison'] = df.apply(lambda row:
compare_sentiment(row['textual_rating'], row['sentiment']),
axis=1)

# Count the number of matches and mismatches
match_count = (df['sentiment_comparison'] == 'Match').sum()
mismatch_count = (df['sentiment_comparison'] == 'Mismatch').sum()

# Count the occurrences of each sentiment generated by OpenAI
sentiment_counts = df['sentiment'].value_counts()


# Set the maximum width for each column in the table
max_text_width = 50

# Prepare a list to store the wrapped text
wrapped_texts = []
for text in df['reviews.text']:
    # Wrap the text if it exceeds the maximum width
    wrapped_text = [text[i:i+max_text_width] for i in range(0,
len(text), max_text_width)]
    wrapped_text = '\n'.join(wrapped_text)
    wrapped_texts.append(wrapped_text)

# Add the wrapped text, main category, and sentiment comparison
to the DataFrame
df['wrapped_text'] = wrapped_texts

# Create a new DataFrame with the desired columns
result_df = df[['main category', 'wrapped_text',
'textual_rating', 'sentiment', 'sentiment_comparison']]

# Save the DataFrame to an Excel file
result_df.to_excel('sentiment_analysis_results.xlsx',
index=False)

# Display the results in a table with wrapped text, main
category, and line separator
table = PrettyTable()
table.field_names = ['Main Category', 'Review Text', 'Textual
Rating', 'OpenAI Sentiment', 'Sentiment Comparison']
```

```
for _, row in result_df.iterrows():
    table.add_row([row['main category'], row['wrapped_text'],
row['textual_rating'], row['sentiment'],
row['sentiment_comparison']])
    table.add_row(["----", "----", "----", "----", "----"])

print(table)

# Display the counts of turning the 'review.rating' to Negative,
Neutral, Positive
print("Sentiment based on the numerical ratings on the raw
data:")
print(f"Positive: {raw_data_positive}")
print(f"Neutral: {raw_data_neutral}")
print(f"Negative: {raw_data_negative}")



# Display the counts of the analysis
print("\nSentiment based on Open AI:")
print(sentiment_counts)

# Display the counts of the matches & mismatches
print(f"\nNumber of matches: {match_count}")
print(f"Number of mismatches: {mismatch_count}")

# Display the match and mismatch counts in a bar chart
labels = ['Match', 'Mismatch']
counts = [match_count, mismatch_count]

plt.bar(labels, counts, color=['green', 'red'])
plt.xlabel('Sentiment Comparison')
plt.ylabel('Count')
plt.title('OpenAI Sentiment vs. Textual Rating Comparison')
plt.show()

labels = sentiment_counts.index
counts = sentiment_counts.values

plt.bar(labels, counts, color=['green', 'red', 'blue'])
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.title('Sentiment Distribution')
plt.show()

# Save the DataFrame to an Excel file
df.to_excel('sentiment_analysis_results.xlsx', index=False)
```