AMAZON PRODUCT FEATURE EXTRACTION, SCORING & SUMMARIZATION

Text Analytics – Fall, 2022
Professor Anol Bhattacherjee

Amitoj Singh Lotey
Durga Prasad Somarouthu
Sahil Shah

Table of Contents

Figures and Tables

EXECUTIVE SUMMARY

Online retail platforms provide access to millions of products across all conceivable categories. A vast variety of products combined with the convenience of location agnostic "window shopping" access helped fuel the recent surge in online shopping. During the COVID-19 pandemic alone, online shopping volume increased 43% from $571.2 billion in 2019 to $815.4 billion in 2020 (www.census.gov). Purchasing decisions on these platforms are largely driven by product reviews. Making sense of hundreds, or thousands of reviews is a challenge. Currently platforms help customers make informed decisions by utilizing a combination of user input based product ratings (1-5 stars) and access to user textual reviews.

Prior research has validated our own anecdotal experience that textual reviews and product ratings often provide divergent perspectives. Minimizing the gap between what customers care about and what companies highlight about their products is critical for online marketplaces like Amazon. The purpose of this analysis is to harness online textual reviews to extract important product features that customers care about, score them, and provide a comprehensive summary of all reviews of a product. The goal is to simplify the review analysis phase for customers, while also providing retailers insight into what product features they should highlight.

Our dataset consists of 400+ reviews of the Apple 2020 M1 MacBook Air. In this study, we focus on two primary workstreams in the text analysis phase – feature extraction and scoring, and review summarization. We use multiple feature extraction and sentiment analysis techniques to compare extracted features with those highlighted by the retailer, and also provide textual review-based feature scoring. We then utilize Open AI's cutting edge GPT3 model to produce a summary of the entire body of reviews. We use analysis output to demonstrate how previously mentioned discrepancies can be resolved using our methodology.

This analysis walks the reader through all major steps in the process from data corpus creation, text analysis process, results, and insights discussion including limitations, and future scope.

PROBLEM DEFINITION & SIGNIFICANCE

Online shopping surged 43% during the COVID-19 Pandemic from $571.2 billion in 2019 to $815.4 billion in 2020 (www.census.gov). 82% of users confirmed that reviews directly influence their buying decision, and 69% shared the reviews with others including: family, friends, and co-workers. However, online reviews and ratings are often biased and tend to over-represent the most extreme views (Harvard Business Review, 2018). Furthermore, there is a gap between the numeric product ratings and review text sentiments. (Almansour, A. et al, 2022).

Amazon has implemented a product feature rating system that requires manual input from the customer, but does not necessarily capture what customers care about most. The features available to rate are the same for all laptops. This creates an issue when feature rating options do not correlate to features actually available in the product – i.e. a feature that product does not even have is scored and made available for user review. Also, the average consumer is not able

to validate whether the numeric (star) rating of a product accurately reflects the sentiments of reviewers. Amazon does provide a "most useful" positive and negative review for a product. However, these reviews do not provide a holistic summary of all reviews submitted.

Application of text analytics methods could help resolve these discrepancies. Bridging these gaps would provide consumers clarity and help to expedite the decision-making process. It would also provide retailers insight into what product features to highlight, based on extraction of product features customers care about most.

PRIOR LITERATURE

We examined numerous existing publications with three primary intentions; First, to better understand the context, breadth and significance of the problem. Second, to understand the approach of other teams that have built and interpreted textual analysis models in a similar context. Third, to explore novel methodologies that could be applied to our use case. Of the many publications examined, key takeaways from seven that were most impactful are included here. A table of prior research papers, their relevance to our project and key takeaways is included in the appendix.

Amazon Review classification and sentiment analysis (Bhatt, A., et al., 2015) had a strong focus on data preprocessing and multiple possible methods of sentiment analysis. It provided insight regarding rule based feature extraction using 11 POS combinations.

Feature-level rating system using customer reviews and review votes (Jerripothula, K. R, et al. 2020) focused on mobile phone ratings via feature extraction. Their method of scoring reviews by extracting only sentences in which the feature appears is the approach that we modeled our feature extraction and scoring upon.

Analysis of text feature extractors using deep learning on fake news (Ahmed, B., Ali, G., Hussain, A., Baseer, 2021) had a strong focus on evaluating different methods that we can use for feature extraction. It also provides a succinct example of how to use sklearn model metrics to summarize the model performance.

Automatic product feature extraction from online product reviews using maximum entropy with lexical and syntactic features (Gamgarn S., Pattarachai L.) takes into account lexical and syntactic features. It approaches mining and summarizing reviews by extracting opinion sentences regarding product features. It also provides method to extract only features that customers are interested in.

Retrieving Product Features and Opinions from Customer Reviews (L. García-Moya, H. Anaya-Sánchez and R. Berlanga-Llavori) focused on use of grammatical dependency relations between words in modeling the language of features. It combines a probabilistic model and a stochastic mapping model. It also proposes a new methodology based on language models to retrieve product features and opinions from a collection of free-text customer reviews about a product or service.

Understanding what concerns consumers: a semantic approach to product feature extraction from consumer reviews (Chih-Ping, W., Yen-Ming, C., Chin-Sheng, Y., Yang, C.) adds extra step of co-occurrence based "feature pruning" in process flow. It also provides context for the problem significance, discussing the increasing difficulty for the individual consumer and retailer to obtain a comprehensive view of consumer opinions pertaining to the products of interest as number of product reviews grow. It also provides novel approach to feature extraction that incorporates opinion words and review semantics.

Text-Rating Review Discrepancy (TRRD): An Integrative Review and Implications for Research (Almansour, Amal, et al.) builds models based on either rating scores or review texts, assuming that they were correlated implicitly with each other. It provides a novel approach proven by using scientific methods that shows the importance of correlation with user reviews and ratings. It proves discrepancy between user reviews and user ratings and shows both need to be used to get a better understanding about the product. In this way, this paper also provides context for problem significance. It directs our attention to the causes of discrepancies and inconsistencies between text reviews and ratings to mitigate and reduce their negative effects.

DATA SOURCE AND PREPARATION

In this report we consider a body of 464 reviews of one product, the Apple 2020 M1 MacBook Air. Extracted fields include review id, title, date, rating, helpfulness and the text review itself. A screenshot of the resultant data frame is shown below.

id	title	date	rating	helpful	review
R3BX8WHB2MIR4A	The Best Laptop I have ever owned coming from ...	Reviewed in the United States on October 21, 2022	5.0 out of 5 stars	16 people found this helpful	This thing is amazing. I have owned several wi...
REDYTHJ9K9DX	Love this notebook!	Reviewed in the United States on September 24,...	5.0 out of 5 stars	118 people found this helpful	I recently migrated to Apple after many years ...
R3OM2CA0JOK4JU	Good laptop and amazing delivery	Reviewed in the United States on October 16, 2022	4.0 out of 5 stars	25 people found this helpful	Got this as a gift during prime day sale. It c...
R2HNC31O1P2O53	Brand New Easy set up	Reviewed in the United States on October 25, 2022	5.0 out of 5 stars	One person found this helpful	This is my very first MacBook finally, got it ...
R1ZW71ZTQ5Z2IR	Amazing Student Laptop	Reviewed in the United States on October 19, 2022	5.0 out of 5 stars	22 people found this helpful	I've only been using my MacBook Air for a few ...

Figure 1 - Data Frame

The process for dataset preprocessing is shown in the following flowchart

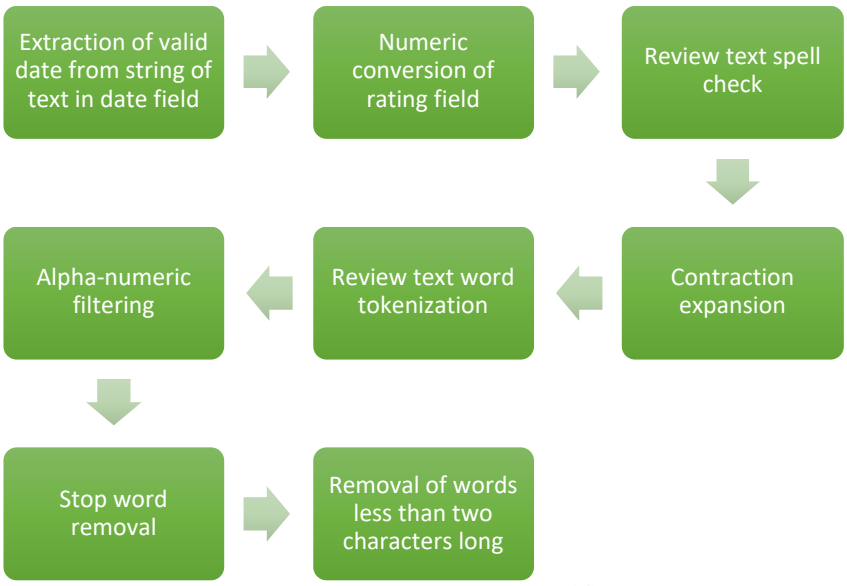

Figure 2 - Data Preprocessing Workflow

Upon scraping, all fields were of string type. In the date column we first removed the text specifying review origin, and then converted the remaining date string into a date type object. We parsed out the numeric portion of the rating and converted it to float. We then ran spell check and contraction expansion on the review text column, tokenized into words and filtered to only keep alpha numeric tokens. Finally, we removed stop words and words less than two characters long. The resultant dataset was used for exploratory analysis that is described in the following section. Further processing steps required for text analysis are detailed in the Text Analysis and Results section.

EXPLORATORY ANALYSIS

We conducted exploratory analysis to better understand the data before doing text analysis. We wanted to understand whether there were any important trends in the data that should be accounted for before doing the text analysis. We explored two primary relationships

1. Count and average rating of reviews over time
2. Relationship between length of review and ratings/helpfulness

Count and average rating of reviews over time

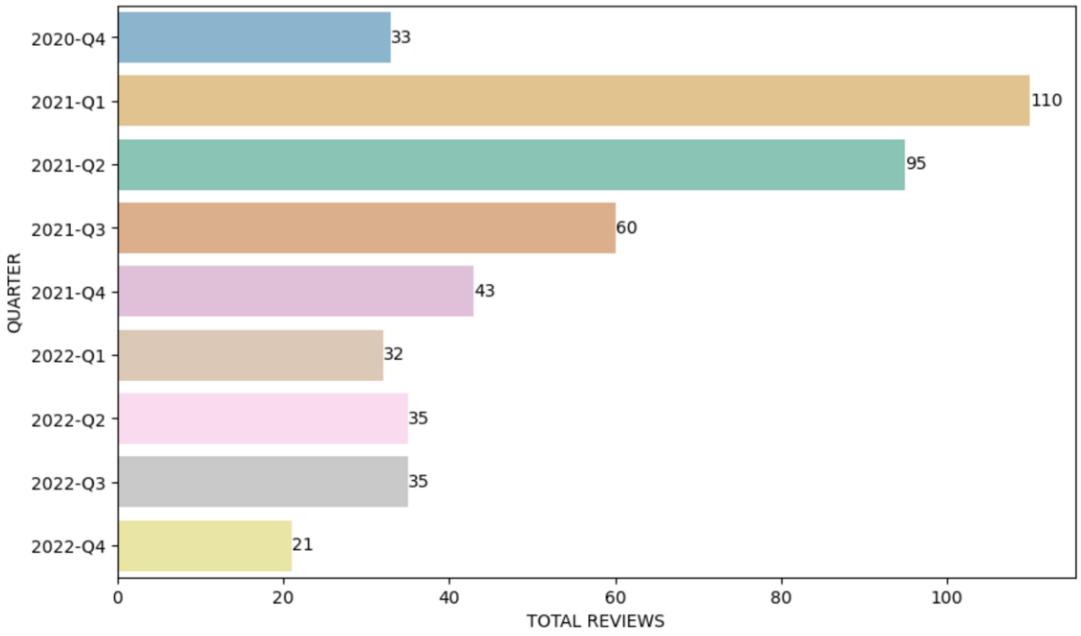

Figure 3 - Number of Reviews per Quarter

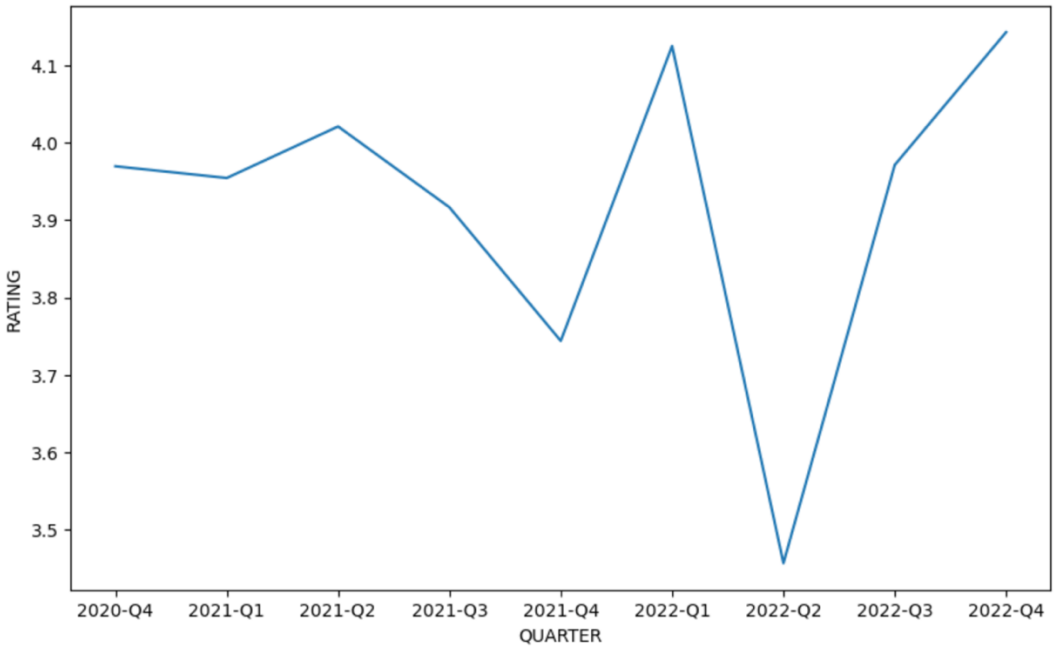

Figure 4 - Average Rating Per Quarter

This plot shows us that highest number of reviews are in quarters 1 & 2 of 2021. Average ratings fluctuate between ~3.4 – 4.1. Interestingly, ratings dropped from their near peak in 2022 Q1 to the minimum in the following quarter, 2022 Q2.

Relationship between length of review and ratings/helpfulness

We were curious to understand how length of review might correlate with ratings and review helpfulness. Review helpfulness is measured by the number of people who found the review helpful, and is a metric that is captured during the scraping process.

The histogram below shows the average number of words for each rating level (1-5). Based on this histogram it doesn't seem like there is a consistent pattern between review length and rating.

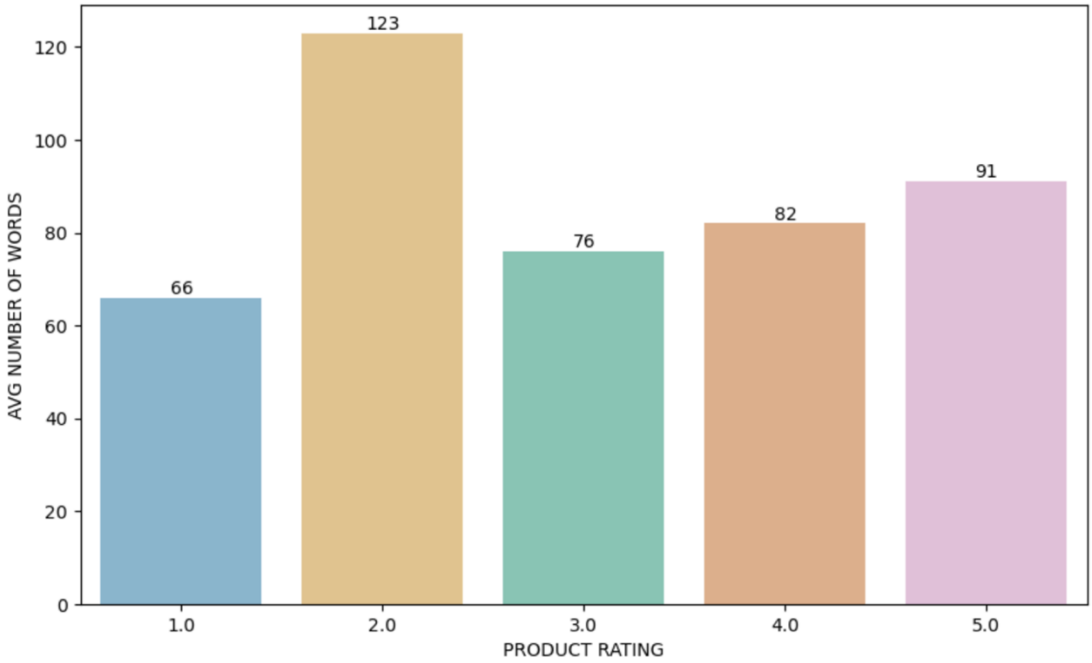

Figure 5 - Rating vs Avg Number of Words

Computed correlation coefficients confirm the lack of consistent patterns between rating, review length and helpfulness.

Table 1 - Correlation: Rating Review Length, Helpfulness

Variables	Correlation
Rating & Review Length	0.07
Helpfulness & Review Length	0.22
Rating & Helpfulness	-0.03

TEXT ANALYTICS AND RESULTS

Our analysis is a combination of two primary workstreams - feature extraction and scoring, and summarization. This section details the methodology and results for both workstreams.

1. FEATURE EXTRACTION AND SCORING

Methodology

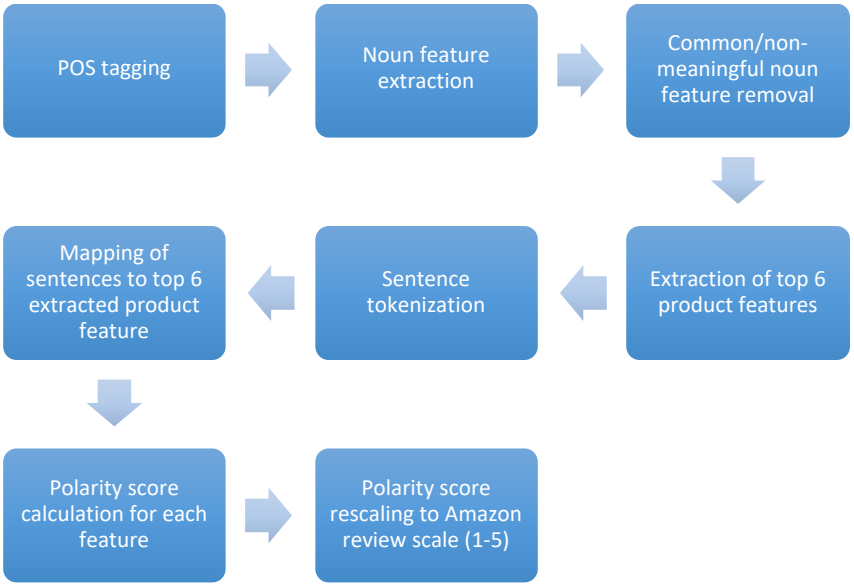

Figure 6 - Feature Extraction and Scoring Workflow

On previously cleaned reviews we applied POS tagging using the nltk library. We extracted nouns and then removed those that were common/non-meaningful like 'work', 'life', 'apple' etc that do not reflect meaningful product features. From the list of meaningful noun features, top six features by frequency were extracted. We chose to extract six features to align with the number of "by feature" features that Amazon presents for laptops.

We sentence tokenized the review column and then created a dictionary in which the key was the feature and value was a list of all the sentence tokens containing that feature. Using SentimentIntensityAnalyzer, we calculated polarity score for each key in the dictionary and a list of dictionaries was made containing each feature as key and its sentiment score as its value. The resultant sentiment score is on a decimal scale between the range of -1 to 1. To compare with Amazon 'by feature' review scores, we rescaled the output of these polarity scores to a scale from 1 to 5

Results

The top six feature extracted from the review corpus are battery, screen, camera, quality, chip and keyboard. Frequency of extracted feature appearance in the corpus is shown in the figure 7. The output indicates that battery is the feature customers care about most. The corresponding scaled scores based on sentiment analysis are shown in figure 8. 'By Feature' scores shown on Amazon are compared with features and scores generated by our methodology in figure 9.

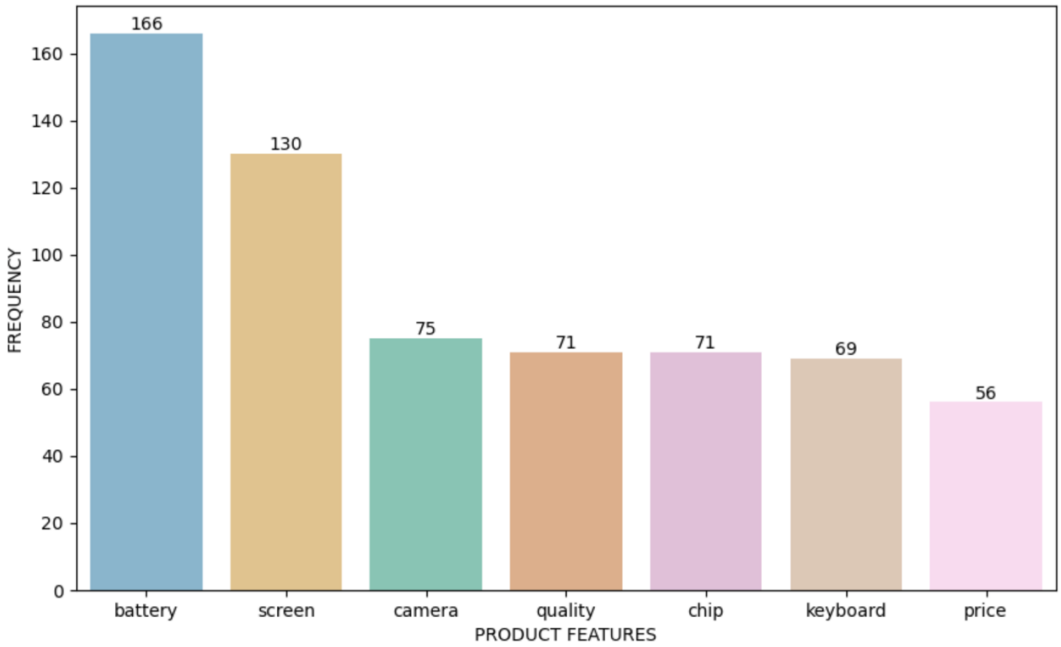

Figure 7 - Top Feature Frequency

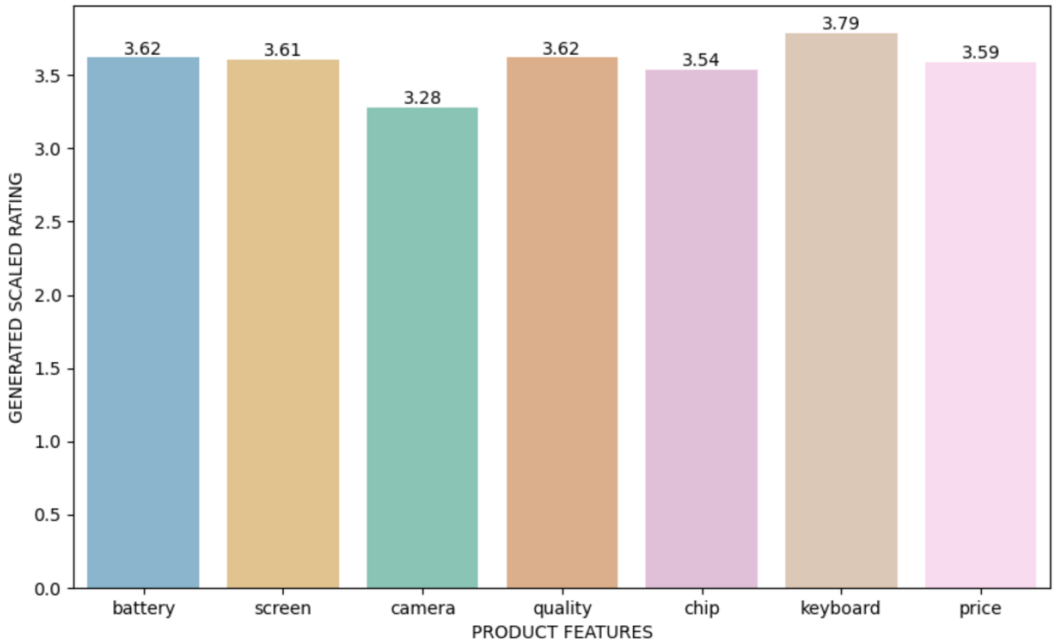

Figure 8 - Extracted Feature Rating

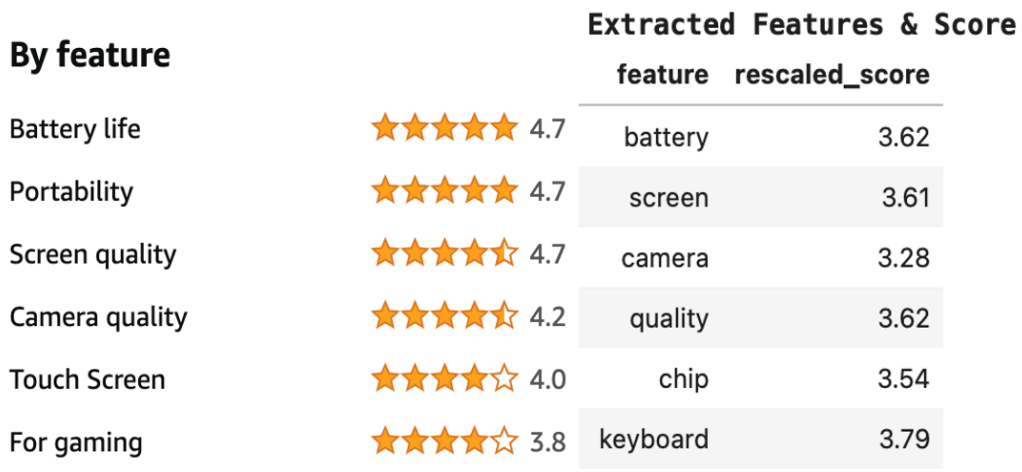

Figure 9 - Comparison with Amazon 'By Feature'

Overall, we see that our extracted features align closely to those presented by Amazon. While sentiment based ratings of matching features like battery, camera and screen are in the same (positive) direction as the Amazon 'By Feature' rating, there are clear discrepancies between the two sets. All extracted feature scores are equal to or lower than their "By Feature' counterpart by an average of 0.85.

'Touch Screen' and 'For Gaming' are two particularly interesting Amazon 'By Feature' scores. Touch screens do not exist on the product in question. Also, this laptop is not typically utilized for gaming, and unsurprisingly gaming did not appear on our list of top features mentioned by reviewers.

2. REVIEW SUMMARIZATION

Methodology

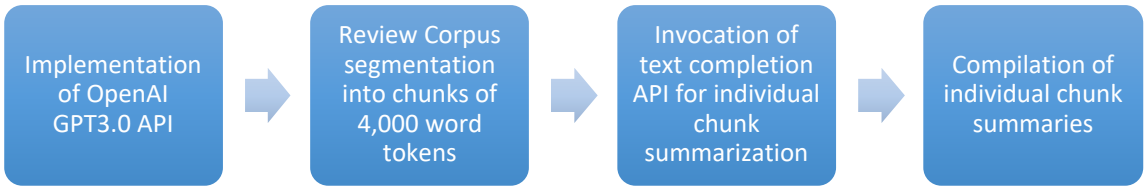

Figure 10 - Review Summarization Workflow

We used OpenAI's powerful GPT-3 API to generate a summary of all reviews in the corpus. Since Davinci is the most capable GPT-3 model for this purpose, we selected the recent model *text-davinci-002* which is pretrained on an immense diverse corpus of text. It can provide a summary for a body of text that is up to 4000 tokens in length.

Interaction of API through HTTP requests requires an authentication key which is retrieved by signing up on the website. As the API has a limitation on length of input, we segmented the combined list of reviews of ~40,000 words into 4,000 token chunks. We invoked the text completion API to generate the summary.

Since this model uses an abstractive text summarization methodology, the output of text completion/summarization differs for every invocation. Below are two significant parameters that we used to tune the output. Remaining parameters were set to default values.

- **Temperature** (0.7): Text completion is based on probability prediction of next word and is controlled by temperature parameter. A higher temperature value gives the model freedom to take more risks in generating text.
- **max_tokens** (300): It is the maximum number to tokens to generate the text

This model performs various tasks as driven by the "prompt" parameter from the input. The generated summary is further analyzed by verifying the sentiment score with OpenAI's GPT-3 model. Prompts used in our analysis are included in the appendix.

Results

Using the above methodology, we produced a ~2,500 word summary from an original body of ~40,000 words. An examination of the resultant summary shows us that it is grammatically sound, but often repetitive. The first 150 words of the summary are shown below.

```
The 2020 MacBook Air with the M1 processor is an excellent laptop with great
build quality, a long-lasting battery, and a beautiful Retina display. It is a
little short on USB ports and does not have a built-in card slot or touchscreen,
but it is a fast and smooth computer that is easy to learn and use. It also
integrates well with Google and Microsoft services. The 2020 MacBook Air is a
great computer with a great screen, keyboard, and battery life. The only
downsides are the webcam and the lack of ports. The Apple MacBook Air M1 is a
great laptop with a beautiful display, accurate and smooth keyboard, and great
battery life. It integrates well with other Apple devices, but there is some
concern about the security of sharing the iPhone\'s lock screen password over the
air with the laptop. The M1 processor is very impressive, delivering outstanding
```

Figure 11 – Review Summary

Sentiment analysis of the generated text summary is positive, and aligns with our feature scoring results.

INSIGHTS AND RECOMMENDATIONS

Discussion

The vast field of text analytics has barely been explored. Our objective was to apply text analytics methodologies to a real-world business problem. We identified discrepancies in online 'by feature' product ratings and reviews and used a combination of text analytics methods to resolve them. Our analysis consisted of three parts; review feature extraction, scoring and summarization.

Our feature extraction & scoring methodology can be used to provide a customized "By Feature" scoring section for products. We found that the extracted features were relevant and meaningful. Since users primarily discuss pros and cons of available features, the likelihood of extracting features that do not exist in the product is low using our extraction process.

Our feature scoring methodology based on sentiment analysis of sentences in which the feature appears is robust, and aligns closely with expected results based on comparison with Amazon's 'By Feature' rating section. That said, there are clear discrepancies between the two sets. Further analysis may be useful to determine which set is the more accurate reflection of user sentiment. Finally, the feature extraction and scoring pipeline can be automated, eliminating the need for user input beyond text reviews, and would provide scoring that is more reflective of true user sentiment.

Based on our analysis, price is an important feature that customers discuss in text reviews, but is not included as a feature to rate on Amazon. This may be a seventh feature to consider adding on the platform.

The summarization approach is a strong step in the right direction, but is not production ready. We were able to successfully summarize a body of reviews consisting of ~40,000 words into a single ~2,500 word summary. However, since the approach required us to segment the total corpus into chunks of 4,000 tokens, and then consolidate the results, the summary produced is sometimes repetitive. In addition, while ~2,500 words is a major reduction of the total corpus, it is still prohibitively long for a user to read through. Ideally, we need to find a way to resolve the repetition issue, and condense the summary into 500 words or less.

The methodology and results presented in this analysis can be utilized to produce business value. This can be integrated into existing online retail platforms or as a feature for third party tools that analyze online reviews.

Recommendations

We provide the following recommendations to online retailers, specifically Amazon:

1. Include price among 'By Feature' features to rate
2. Utilize text reviews for
    a. determining what features customers care about most
    b. avoiding the inclusion of features that don't exist in the product
3. Automate the manual user input based feature scoring system by implementing appropriate sentiment analysis of text reviews

FUTURE SCOPE

The potential business value of this analysis is largely unexplored. Some preliminary next steps to extend the current analysis include:

1. Extracting and scoring secondary level product features
2. Resolving repetition in review summarization
3. Exploring other pretrained text models for various tasks including
    a. Summarization
    b. Query Response
    c. Translation
4. Interactive dashboard creation

REFERENCES

JOURNALS

Ahmed, B., Ali, G., Hussain, A., Baseer, A., & Ahmed, J. (2021). Analysis of text feature extractors using deep learning on fake news. *Engineering, Technology & Applied Science Research*, *11*(2), 7001–7005. https://doi.org/10.48084/etasr.4069

Almansour, A., Alotaibi, R., & Alharbi, H. (2022). Text-rating review discrepancy (TRRD): An integrative review and implications for research. *Future Business Journal*, *8*(1). https://doi.org/10.1186/s43093-022-00114-y

Bhatt, A., Patel, A., Chheda, H., Gawande, K., Amazon Review classification and sentiment analysis - *IJCSIT. (n.d.)*. Retrieved October 7, 2022, from https://www.ijcsit.com/docs/Volume%206/vol6issue06/ijcsit2015060652.pdf

Garcia-Moya, L., Anaya-Sanchez, H., & Berlanga-Llavori, R. (2013). Retrieving product features and opinions from customer reviews. *IEEE Intelligent Systems*, *28*(3), 19–27. https://doi.org/10.1109/mis.2013.37

Jerripothula, K. R., Rai, A., Garg, K., & Rautela, Y. S. (2020). Feature-level rating system using customer reviews and review votes. *IEEE Transactions on Computational Social Systems*, *7*(5), 1210–1219. https://doi.org/10.1109/tcss.2020.3010807

Somprasertsri, G., & Lalitrojwong, P. (2008). Automatic product feature extraction from online product reviews using maximum entropy with lexical and syntactic features. *2008 IEEE International Conference on Information Reuse and Integration*. https://doi.org/10.1109/iri.2008.4583038

Wei, C.-P., Chen, Y.-M., Yang, C.-S., & Yang, C. C. (2009). Understanding what concerns consumers: A semantic approach to product feature extraction from Consumer Reviews. *Information Systems and e-Business Management*, *8*(2), 149–167. https://doi.org/10.1007/s10257-009-0113-9

WEBSITES

Amazon.com: 2020 Apple MacBook Air Laptop: Apple M1 chip, 13" retina ... (n.d.). Retrieved November 6, 2022, from https://www.amazon.com/Apple-MacBook-13-inch-256GBStorage/dp/B08N5M7S6K

Beautiful Soup documentation. Beautiful Soup Documentation - Beautiful Soup 4.9.0 documentation. (n.d.). Retrieved November 5, 2022, from https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Online reviews are biased. here's how to fix them. Harvard Business Review. (2018, March 14). Retrieved November 3, 2022, from https://hbr.org/2018/03/online-reviews-are-biased heres-how-to-fix-them

OpenAI API. (n.d.). Retrieved November 5, 2022, from https://beta.openai.com/docs/guides/completion

OpenAI API. (n.d.). Retrieved November 5, 2022, from https://beta.openai.com/playground

APPENDIX

Prior Research

PRIOR WORK AND ITS RELEVANCE TO OUR PROJECT			
Research Paper	**Relevance**	**Methodology**	**Key Findings/Take aways**
Bhatt, A., Patel, A., Chheda, H., Gawande, K., Amazon Review classification and sentiment analysis - IJCSIT. (n.d.). Retrieved October 7, 2022, from https://www.ijcsit.com/docs/Volume%206/vol6issue06/ijcsit2015060652.pdf	Methodology	1. Classification done via pipeline of POS tagging, Opinion Word Extraction, OW polarity Identification, Sentence Polarity Identification, Summary generation 2. Discusses possibility of using of Naïve Bayes, Probabilistic ML, etc.	1. Strong focus on the multiple steps used in data preprocessing 2. Gives an overview of the methods we can use for the sentiment analysis portion of our project – similar to what we have already learned to do 3. Key highlight is explanation on using rule based extraction (11 POS combinations used to extract feature and its sentiment)
Jerripothula, K. R., Rai, A., Garg, K., & Rautela, Y. S. (2020). Feature-level rating system using customer reviews and review votes. IEEE Transactions on Computational Social Systems, 7(5), 1210–1219. https://doi.org/10.1109/tcss.2020.3010807	Methodology	1. Feature Selection: features in which people are generally interested and neglect the words having their frequency less than 0.02% of the total number of reviews in the review data 2. spelling correcting function (using autocorrect package of python 3.group of continuous tokens as a sentence if last token and the token just before the first one are periods . 4.Uses compound score as the required sentiment analysis score 5.report the number of phones close to different integer ratings	1) use of spelling correction 2) Relevant sentence extraction: only sentences containing feature keywords are retained. 3)Table representing number of phones with different integer ratings for each feature
Ahmed, B., Ali, G., Hussain, A., Baseer, A., & Ahmed, J. (2021). Analysis of text feature extractors using deep learning on fake news. Engineering, Technology & Applied Science Research, 11(2), 7001–7005. https://doi.org/10.48084/etasr.4069	Methodology	1.Split Datasets into train and test splits 2.Feature extraction via TF-IDF, Glove, BERT 3.Running ANN model 4. Evaluation	1. Provides evaluation of possible methods we can use for feature extraction 2. Based on this paper, we may consider using NN for classification tasks in our project 3. Provides succinct example of how to use sklearn model metrics to summarize the model performance

Gamgarn Somprasertsri, & Pattarachai Lalitrojwong. (2008). Automatic product feature extraction from online product reviews using maximum entropy with lexical and syntactic features. 2008 IEEE International Conference on Information Reuse and Integration. https://doi.org/10.1109/iri.2008.4583038	Methodology	1.extract learning features from the annotated corpus 2.train the maximum entropy model, uses trained model to extract product features, 3. apply a natural language processing technique in postprocessing steps to discover the remaining product features. 4.re-formulated as a classification problem, in which the task is to observe some syntactic dependency and context information and predict the class. classes such as product feature and non-product feature 5.product feature extraction problem as a classification task: given a sequence of words in a sentence, we generate a sequence of labels indicating whether the word is a product feature or non-product feature.	1. Highlights how to train a model to extract features from text 2. Takes into account lexical and syntactic features 3. Indicates that product feature extraction can be viewed as classification task 4. Concentrated on mining and summarizing reviews by extracting opinion sentences regarding product features 5. Provides method to extract only features that customers are interested in
L. García-Moya, H. Anaya-Sánchez and R. Berlanga-Llavori, "Retrieving Product Features and Opinions from Customer Reviews," in IEEE Intelligent Systems, vol. 28, no. 3, pp. 19-27, May-June 2013, doi: 10.1109/MIS.2013.37.	Methodology	1. Focuses on aspect-based summarization - aspect identification, sentiment classification, and aspect rating (aspect is intended to represent the opinion or sentiment targets, which are also referred to as product features) 2. Combines probabilistic model of opinion words and a stochastic mapping model between words to approximate a lan- guage model of products. 3. Generates set of candidate product features via identification of word sequences of arbitrary length 4. prunes the candidates that fall below a threshold according to functions defined in the paper 5. ranks candidates by combining the prob- ability of their head under the refined language model of product features to- gether with a normalized probability value of the sequence of modifiers con- ditioned on the head 6. Statistical mapping model T based on word co-occur- rencesfromlocalcontextsinD,wecon- sider defining p(wi /wj) (i, j ∈ {1, ..., n}) to be proportional to the number of times word wi occurs in a local con- text of words from D containing an occurrence of wj. 7. kernel-based density es- timation approach to learn the model of opinion words Q from a (minimal) knowledge source of sentiments or opinions	1. A new methodology based on language models retrieves product features and opinions from a collection of free-text customer reviews about a product or service. The proposal relies on a language-modeling framework that can be applied to reviews in any domain and language provided with a minimal knowledge source of sentiments or opinions 2. Combines probabilistic model and a stochastic mapping model 3. Allows for the use of grammatical de- pendency relations between words in modeling the language of features 4. it can effectively retrieve product features without relying on natural language processing (NLP) techniques. 5. Generally, NLP-based approaches present good precision but low recall figures because they depend on the definition of extraction patterns, which are depen- dent on both the particular language and the reviews' application domain.

Understanding what concerns consumers: a semantic approach to product feature extraction from consumer reviews Chih-Ping Wei Æ Yen-Ming Chen Æ Chin-Sheng Yang Æ Christopher C. Yang https://link.springer.com/content/pdf/10.1007/s10257-009-0113-9.pdf	Context + Methodology	1. Used a modified version of the unsupervised PFE technique in (Hu and Liu 2004a, b). 2. Since original Hu paper technique did not take into account semantic perspective, developed Semantic Product Feature Extraction Technique 3. adopt a list of subjective positive and negative adjectives defined in the General Inquirer as OPINION WORDS 4. Allows us to exclude non-product features and opinion-irrelevant product features, effectively identify product features with rare occurrences in the input consumer reviews 5. Using the known semantic orientation of these opinion words, the opinion orientation identification task can be incorporated easily into the proposed SPE technique 6. The proposed SPE technique begins with the preprocessing of the input consumer reviews, where the preprocessing task includes part-of-speech (POS) tagging, noun phrase chunking, and stemming.	1. Provides novel approach to feature extraction that incorporates opinion words and review semantics 2. Adds extra step of "feature pruning" in process flow 3. We may be able to use co-occurrence based feature pruning to ensure that extracted review features actually relate to product features 4. Used manually tagged features to evaluate model 5. Discusses increase in difficulty for the individual consumer and retailer to obtain a comprehensive view of consumer opinions pertaining to the products of interest as number of product reviews grow 6. Sentiment analysis essentially consists of two main tasks: product feature extraction for extracting product features stated in consumer reviews and opinion orientation identification for determining the sentiment
Almansour, Amal, et al. "Text-Rating Review Discrepancy (TRRD): An Integrative Review and Implications for Research." *Future Business Journal*, vol. 8, no. 1, 2022, https://doi.org/10.1186/s43093-022-00114-y.	Context + Methodology	1. Research studies that built their models based on either rating scores or review texts, assuming that they were correlated implicitly with each other 2. Four machine learning methods: Naïve bayes (NB), decision tree (DT), neural network (NN), and support vector machine (SVM) were demonstrated with the linguistic features, and their performances were compared by accuracy and the harmonic mean between precision and recall (F1 score). 3. Neural network (NN) and support vector machine (SVM) classification showed acceptable performance under every condition 4. Using a corpus contains 5531 restaurants, with associated a set of 52,264 reviews. Reviews contain structured metadata (star rating, date) along with text. The experiment showed there was a positive correlation between positive reviews and star ratings and a negative correlation between negative reviews and star ratings. 5. These results motivated the authors to include text reviews in the context of recommender systems.	1. Provides a novel approach proven by using scientific methods that shows the importance of correlation with user reviews and ratings 2. Could use proposed OCR model research methodology for our project 3. Proves discrepancy between user reviews and user ratings and shows both need to be used to get a better understanding about the project 4.Makes us pay more attention to the causes of discrepancies and inconsistencies between text reviews and ratings in order to mitigate and reduce its negative effects on developed OCRs solution models

Prompts

```
Prompt1: To generate a summary of the review
Write a concise summary of the following:
<< Review >>
CONCISE SUMMARY:'

Prompt2: To check sentiment of the generated summary
```

```
'Decide whether review sentiment is positive, neutral, or
negative.
<<Review>>
Sentiment:'
```

Python Code

```
# -*- coding: utf-8 -*-
"""amazon-review-analysis.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1XmPF0OUrJORunLwEsGGKOWZCSeSlh1ZK
"""

## common libraries
import pandas as pd
import nltk
from collections import Counter
import numpy as np
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('omw-1.4')
nltk.download('averaged_perceptron_tagger')
nltk.download('universal_tagset')

##loading the dataset
df=pd.read_csv("reviews_final_1k.csv")
df.head()

df.dtypes

len(df)

df.dropna()

"""## Data cleaning"""

df['review'] = df['review'].astype(str)
df['helpful'] = df['helpful'].apply(lambda x: str(x).replace(" people found this
helpful",""))
df['helpful'] = df['helpful'].apply(lambda x: str(x).replace("One person found this
helpful",'1'))
df['helpful'] = df['helpful'].apply(lambda x: str(x).replace(",",''))
df['helpful'] = df['helpful'].astype(int)
```

```
df['date'] = df['date'].apply(lambda x: x.replace("Reviewed in the United States on
",""))
df['rating'] = df['rating'].apply(lambda x: x.replace(" out of 5 stars","").strip())
df['date']= pd.to_datetime(df['date'])
df['rating'] = df['rating'].astype(float)

df['review_length'] = df['review'].apply(lambda x: len(str(x).split(" ")))

df.describe()

"""### Correlation check"""

df['rating'].corr(df['review_length']).round(decimals = 2)

df['helpful'].corr(df['review_length']).round(decimals = 2)

df['rating'].corr(df['helpful']).round(decimals = 2)

df[(df['helpful'] > 500)]

df[(df['review_length'] > 500)]

avg_review_length_df=df.groupby('rating')['review_length'].mean().astype(int).reset_in
dex()
avg_review_length_df

#!pip install contractions
import contractions
def expand_words(data):
    expanded_words=[]
    for word in data.split():
        expanded_words.append(contractions.fix(word))
    return ' '.join(expanded_words)

df['clean_review'] = df['review'].apply(lambda x: expand_words(x))

df.head(3)

df['sent_tokenized'] = df['clean_review'].apply(lambda x: nltk.sent_tokenize(x))

df.head(3)

!pip install textblob

from textblob import TextBlob,Word

def sentenceCorrection(sentences):
    sentence_list=[]
```

```
    for sentence in sentences:
        word_list=[]
        for word in sentence.split(" "):
            #sentence_correct.append(str(TextBlob(sent).correct()))
            word_list.append(str(Word(word)))
        sentence_list.append(' '.join(word_list))
    return ' '.join(sentence_list)

df['clean_review'] = df['sent_tokenized'].apply(lambda x: sentenceCorrection(x))

df.head()

nltkStopwords = nltk.corpus.stopwords.words('english')
lemmatizer = nltk.stem.WordNetLemmatizer()
def tokenizeCleanLemmatizeWords(data):
    nltkWords = nltk.tokenize.word_tokenize(data)
    nltkCleanWords = [w for w in nltkWords if w.isalnum() and len(w)>1]
    nltkCleanWords = [w.lower() for w in nltkCleanWords if w not in nltkStopwords]
    return [lemmatizer.lemmatize(w) for w in nltkCleanWords]

df['clean_review'] = df['clean_review'].apply(lambda x:
tokenizeCleanLemmatizeWords(x))

df.head()

"""## Feature Extraction"""

def getNounPhraseFeatures(data):
    nltkPos = nltk.pos_tag(data, tagset='universal')
    pattern = 'NP: {<DT>?<JJ>*<NN>}'
    parser = nltk.RegexpParser(pattern)
    nltkParsed = parser.parse(nltkPos)
  #print(nltkParsed)
    word, pos = zip(*nltkParsed)
    nltkNouns = []
    for i, w in enumerate(word):
        if pos[i] == 'NOUN' and len(w)>2:
            nltkNouns.append(word[i])
    return nltkNouns

def getTop(data):
    total_freq=[i  for i, w in Counter(data).most_common()]
    unique_pos=getNounPhraseFeatures(list(set(data)))
    return [w for w in total_freq if w in unique_pos][0:6]

words=[]
common_words=['buy','well','laptop','apple', 'product','mac','look','get','macbook',
'computer',
```

```
'air','use','love','thing','work','life','window','need','time','mode','drive','issue'
,'year','day','port','run']
for i,row in df.iterrows():
        for w in row['clean_review']:
            if w not in common_words:
                words.append(w)
len(words)

getTop(words)

df.head(2)

sentences=[]
features=['battery','screen','camera','quality','chip','keyboard','price']
#features=['screen','ram', 'price', 'keyboard', 'ssd', 'quality']
for i,row in df.iterrows():
    for sent in row['sent_tokenized']:
        sentences.append(sent)
len(sentences)

from nltk.sentiment.vader import SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()

feature_sentences={}
for word in features:
    sentences_present=[]
    for sent in sentences:
        if word in sent:
            sentences_present.append(sent)
    feature_sentences[word]=sentences_present
len(feature_sentences)

reviews_list=[]
for key in feature_sentences:
    sentiment=0.0
    for sent in feature_sentences.get(key):
        sentiment+=analyzer.polarity_scores(sent)['compound']
    x=sentiment/len(feature_sentences.get(key))
    new_row = {'feature':key, 'score':x}
    reviews_list.append(new_row)
reviews_list

result_df = pd.DataFrame(reviews_list,columns=['feature','score'])

result_df.head(10)

def reScale(OldValue,NewRange,OldRange,OldMin,NewMin):
    return (((OldValue - OldMin) * NewRange) / OldRange) + NewMin
```

```
result_df['rescaled_score']=result_df['score'].apply(lambda x: reScale(x,4,2,-1,1))

result_df['rescaled_score']=result_df['rescaled_score'].round(decimals = 2)

print ('\033[1m' + '  Extracted Features & Score')
result_df[['feature','rescaled_score']].head(6)

print(len(words))
print(features)
freq= Counter(words)
result_df['frequency']=''
for i,row in result_df.iterrows():
    result_df['frequency'][i]=freq.get(row['feature'])
result_df

from datetime import datetime
df['quarter']=''
for i,row in df.iterrows():
    df['quarter'][i]=str(row['date'].year)+'-Q'+str(row['date'].quarter)

import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
def generatePlot(plot_type,data_var,x_var,y_var,x_label,y_label):
    plt.clf()
    matplotlib.rc_file_defaults()
    ax = sns.set_style(style=None, rc=None )
    # figsize=(12,6)
    plt.figure(figsize=(10,6))
    my_cmap=sns.color_palette("colorblind")
    if plot_type=='bar':
        ax=sns.barplot(data = data_var, x=x_var,y=y_var, alpha=0.5,palette=my_cmap)
    if plot_type=='line':
        ax=sns.lineplot(data = data_var, x=x_var,y=y_var,palette=my_cmap)
    ax.set(xlabel=x_label, ylabel=y_label)
    for i in ax.containers:
        ax.bar_label(i,)
#sns.lineplot(data = result_df, x='feature',y='rescaled_score',marker='o', sort =
False, ax=ax2)

generatePlot('bar',result_df,'feature','frequency','PRODUCT FEATURES','FREQUENCY')

generatePlot('bar',result_df,'feature','rescaled_score','PRODUCT FEATURES','GENERATED
SCALED RATING')

df['rating'] = df['rating'].astype(float)
trend_df=df.groupby('quarter')['id'].count().reset_index()
```

```
trend_df.columns=['quarter', 'id']
trend_df1=df.groupby('quarter')['rating'].mean().reset_index()
trend_df1.columns=['quarter', 'rating']

trend_df1=df.groupby('quarter')['rating'].mean().reset_index()
trend_df1.columns=['quarter', 'rating']

trend_df

generatePlot('bar',trend_df,'id','quarter','TOTAL REVIEWS','QUARTER')

generatePlot('line',trend_df1,'quarter','rating','QUARTER','RATING')

avg_review_length_df

generatePlot('bar',avg_review_length_df,'rating','review_length','PRODUCT RATING','AVG
NUMBER OF WORDS')

avg_review_length_df

generatePlot('bar',avg_review_length_df,'rating','review_length','Product
rating','Average No. of words')

"""## OPENAI GPT-3 Integration of Text Completion API"""

!pip install openai

import openai
import os
from time import time,sleep
import textwrap
import re

openai.api_key ='**************************************'
def getSummary(review_var,modelname,length):
    return openai.Completion.create(
        model=modelname,
      prompt=review_var,
  temperature=0.7,
  max_tokens=length,
  top_p=1.0,
  frequency_penalty=0.0,
  presence_penalty=0.0
  )

import textwrap
chunks = textwrap.wrap(' '.join(df['review']), 4000)
```

```
# len(reviews_text)

prompt_header='Write a concise summary of the following: \
<<SUMMARY>> \
CONCISE SUMMARY:'

count = 0
import time

result=[]
for chunk in chunks:
    count = count + 1
    prompt = prompt_header.replace('<<SUMMARY>>', chunk)
    prompt = prompt.encode(encoding='ASCII',errors='ignore').decode()
    summary = getSummary(prompt,'text-davinci-002',300)['choices'][0]['text'].strip()
    time.sleep(1)
    result.append(summary)

len(chunks)

print(len(' '.join(df['review']).split(" ")))

print(len(' '.join(result).split(" ")))

result



result=" ".join(result)

result

prompt = prompt_header.replace('<<SUMMARY>>', result)
prompt = prompt.encode(encoding='ASCII',errors='ignore').decode()
getSummary(prompt,'text-davinci-002',100)['choices'][0]['text'].strip()

sentement_header='Decide whether review sentiment is positive, neutral, or negative. \
<<SUMMARY>> \
Sentiment:'

def getSentiment(data):
    prompt = sentement_header.replace('<<SUMMARY>>',data )
    prompt = prompt.encode(encoding='ASCII',errors='ignore').decode()
    time.sleep(3)
    return getSummary(prompt,'text-davinci-002',100)['choices'][0]['text'].strip()

getSentiment(result)
```