Intel Products Sentiment Analysis from Online Reviews

A PROJECT REPORT

Submitted by

Raaghashree M (RA2211030010040)

Under the Guidance of

Dr. M. Safa

(Assistant Professor, Networking and Communications)

Mr. Debdyut Hazra

(Project Mentor, Intel)

in partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING
with specialization in (Cyber Security)

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR- 603 203

JULY 2024

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr C. Muthamizhchelvan,** Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology**, Dr T.V.Gopal,** for his invaluable support.

We express our gratitude to **Debdyut Hazra**, Project Mentor, Intel for supporting us during the critical stages of the project work.

We wish to thank **Dr Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We extend my gratitude to our **HoD Dr. Annapurani. K Head of the Department, Networking and Communications** and my departmental colleagues for their support.

We register our immeasurable thanks to our Faculty Advisor, **Mr. V. Rajaram**, Assistant Professor, Department of Networking and Communications, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, **Dr. M. Safa**, Assistant Professor, Department of Networking and Communications, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under her mentorship.

We sincerely thank the Department of Networking and Communications, staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

-RAAGHASHREE M [RA2211030010040]

ABSTRACT

User sentiment in online reviews plays a crucial role in assessing and evaluating a product's performance and understanding consumers interests. The project implements a robust sentiment analysis approach to evaluate online user reviews of Intel processors, utilizing two approaches: the rule-based VADER model and the machine learning-based DistilBERT model. The reviews sourced from Amazon were collected through API and then pre-processed to clean and standardize the text. Further, feature extraction using TF-IDF was done to identify common and significant terms from user sentiments. VADER was chosen for its efficiency in handling short texts, while DistilBERT showcased its advanced natural language understanding capabilities. By evaluating the model based on accuracy, precision, recall, and F1 score, the analysis provides valuable insights about the consumer interests and opinions of various Intel processor generations, highlighting the strengths and areas for improvement. This comparative study helps to view the importance of extensive sentiment analysis to understand consumer feedback and thus improving the products to be offered in future.

TABLE OF CONTENTS

ABBREVIATIONS

NLP	**Natural Language Processing**
API	**Application Programming Interface**
VADERS	**Valence Aware Dictionary and sEntiment Reasoner**
BERT	**Bidirectional Encoder Representations from Transformers**
DistilBERT	**Distilled Bidirectional Encoder Representations from Transformers**
TF-IDF	**Term Frequency–Inverse Document Frequency**
ML	**Machine Learning**
NLTK	**Natural Language ToolKit**

1. INTRODUCTION

User reviews give detailed and insightful feedback on the performance, reliability and overall user satisfaction of a product and any issues that are not always mentioned in professional reviews. This project aims to analyze user sentiments on Intel processors, specifically focusing on both mobile and desktop variants across the 12th, 13th, and 14th generations, by through online reviews sourced from Amazon. By examining these reviews, the overall user sentiment about the processors and the improvement and be analysed. Further, identification of common features and keywords in the positive and negative feedback and also tracking the sentiment trends over time by comparing the performance of different processors is done.

To achieve a comprehensive analysis, two different approaches were implemented: a rule-based model and a machine learning-based model. The rule-based approach utilizes the VADER (Valence Aware Dictionary and sEntiment Reasoner) sentiment analysis tool, which is suitable for handling social media text. Meanwhile, the machine learning approach uses a pre-trained DistilBERT model, implemented through a pipeline, to analyze the sentiment of the reviews. This dual-method strategy not only increases the accuracy of the analysis but also helps do a comparative study of the performance of rule-based and machine learning models in sentiment classification. By incorporating multilingual reviews and focusing on different processor generations, the project provides valuable insights into the global consumer sentiment and the ever-changing performance trends of Intel processors.

2. LITERATURE REVIEW

The sentiment analysis of customer reviews is a widely researched area, with various studies and research papers exploring different methodologies and datasets to understand user sentiments.

"Sentiment Analysis of Customer Reviews Using Machine Learning Techniques" by M. Zhang, X. Chen, and Y. Li uses Amazon customer reviews to build and implement different machine learning techniques, including Support Vector Machines (SVM) and Naive Bayes. This approach highlights the effectiveness of traditional machine learning models in sentiment analysis. However, the study is limited to only certain product categories and hence lacks a comparative analysis across different generations and models, which is crucial for a more detailed understanding of product evolution and user preferences.

"A Comparative Study of Sentiment Analysis Techniques on Social Media Texts" by A. Tripathy, A. Agrawal, and S. Rath focuses on Twitter and Facebook posts to compare multiple sentiment analysis techniques, including deep learning models. The study's focus is on social media texts rather than specific product reviews.

"Sentiment Analysis on Product Reviews Using Lexicon-Based Approach and Machine Learning" by S. Arora and S. Kaur analyses both Flipkart and Amazon product reviews, combining lexicon-based and machine learning approaches for sentiment analysis. Although this method is effective, the study is not comprehensive because of the limited dataset size and scope, and it does not include multilingual or non-English reviews, which are important in a global consumer market.

"Aspect-Based Sentiment Analysis on Amazon Reviews" by S. Nasukawa and J. Yi analyses Amazon product reviews by focusing on aspect-based sentiment analysis. This method provides detailed information about specific product features but may result in high computational costs and also focuses less on overall sentiment trends across product generations, which should offer a broader perspective on user satisfaction and product performance over time.

"Using Deep Learning for Sentiment Analysis on Product Reviews" by P. J. Liu, X. Qiu, and X. Huang uses advanced deep learning models such as LSTM and BERT on Yelp and Amazon reviews to achieve a higher accuracy in sentiment classification. Despite its accuracy, the approach needs a lot of computational resources and longer training times, and it is limited to a few product types, restricting its inclusivity.

Each study and paper have its own strengths (comparing machine learning models or detailed analysis) but also have certain limitations (less focus on specific product categories, smaller dataset sizes, higher computational requirements). In contrast, this project uses a rule-based approach (VADER) and a machine learning pipeline (DistilBERT) on a large dataset of Intel processor reviews from Amazon. This dual approach increases the accuracy and allows for a broader comparison across different processor generations. Additionally, the inclusion of multilingual review translation ensures a more inclusive analysis of world-wide global user sentiments.

3. DATA COLLECTION

The sentiment analysis on Intel products was focused on the 12th, 13th and 14th generations of mobile and desktop processors including the i3,i5,i7 and i9 cores. To perform a comprehensive analysis, the data was collected and categorised by the three generations and then analysed to find the overall user sentiment as well as the comparison of performance and trends across different generations of Intel processors.

3.1. Data Source:

The data for the sentiment analysis on Intel products was sourced for Amazon, a global e-commerce platform. Amazon's extensive user base provides a diverse and inclusive set of reviews, thus making it an extremely ideal source for obtaining user sentiments on the Intel processors.

3.2. Data Scraping:

The data was extracted, using an API provided by Apify.com, a tool for web scraping. This API helped with efficient extraction of user reviews from Amazon. Further, with the help of the tool, specific data in regards to review id and rating scores were extracted. Reviews extracted were categorized based on the generations of Intel processors, covering both mobile and desktop variants.

3.3. Dataset:

The dataset is classified by the processor generations and houses a total of 2000 reviews.

The dataset includes key features like:

- Review ID - A unique identifier for each review
- Product ASIN - Amazon Standard Identification Number used to identify specific Intel processor models

- Country - The country from which the review was posted
- Rating score – Rating given by user (1-5)
- Review title
- Review description

All the reviews were sourced from recent user entries, ensuring that the dataset represents the present user sentiments. This comprehensive and up-to-date dataset serves as a reliable foundation for the sentiment analysis.

4. DATA PREPROCESSING

Data preprocessing is an important step in sentiment analysis, impacting the effectiveness of both rule-based and machine learning approaches. The collected data underwent several preprocessing steps to ensure the overall quality, and consistency of the data, which would increase the accuracy and also the reliability of the sentiment analysis models.

4.1. Steps in Data Preprocessing

Fig 4.1. Data preprocessing

Figure 4.1 shows the data preprocessing mechanism that the raw data undergoes before the sentiment analysis is done. It consists of several steps to ensure the data is cleaning, tokenized and normalised for accurate analysis.

4.1.1. Removing Noise:

- **Duplicates**: Removing duplicates ensures each review contributes equally to the sentiment analysis.
- **Punctuation**: Punctuation marks can be removed to simplify the user data.
- **Non-Alphabetic Characters**: Characters like numbers and symbols are do not serve any purpose to sentiment analysis and thus removing them would help in bringing the focus on to the meaningful words.

4.1.2. Handling Missing Values:

- Some user reviews may have missing text fields like title or review description, so they are filled to ensure the dataset is complete so as to prevent any errors during both rule-based and machine learning model training.

4.1.3. Text Standardization:

- **Lowercasing**: Converting all text to lowercase will ensure the uniformity of data.
- **Stop Words Removal**: Some common words that may not contribute to sentiment and feature extraction section of the project, such as "the", "and", "is", are removed to reduce noise.
- **Lemmatization**: This is the process of converting words to their base or root form, done to ensure that the different forms of a word are treated as a single entity. This helps in reducing the overall complexity, running time and further improves the model's understanding.

4.1.4. Translation of Non-English Reviews:

- This diverse dataset contains reviews from global users, thus the non-English reviews are translated to English to maintain consistency in the dataset, helping the sentiment analysis models to process all reviews uniformly without any issues.

4.2.Importance of preprocessing in analysis:

4.2.1. Rule-Based VADER:

- In NLP, VADER is a tool designed to perform sentiment analysis by evaluating the emotional tone of a particular text.
- This tool relies on a set of predefined lexicons and rules to analyze sentiment.
- Thus preprocessing ensures that the text to be analysed is clean and standardized, which allows the tool to accurately match words and phrases to its sentiment

lexicon.

- This help the model to perform more efficiently without being confused by irrelevant characters in the text.

4.2.2. **Machine Learning Pipeline**:

- Machine learning models need structured and clean data to learn patterns and make accurate predictions.
- Further pre-processed data also improves the feature extraction process, ensuring the model captures the most relevant information.
- Techniques like tokenization, stop word removal, and lemmatization help in creating an accurate sentiment analysis, leading to better model performance in terms of accuracy, precision, recall, and F1 score.

Hence, data preprocessing is an essential and important step for not only improving the performance of both rule-based and machine learning sentiment analysis models but also to help produce reliable and accurate sentiment analysis.

5. SENTIMENT ANALYSIS METHODOLOGY

Two different approaches were made use to perform sentiment analysis on the online reviews: a rule-based method and a machine learning-based method. The objective was to compare the performance of both these models to get a more accurate, comprehensive and inclusive report. By using both methodologies, the project aims to identify the strengths and weaknesses in each approach, and ensures a strong sentiment analysis framework.

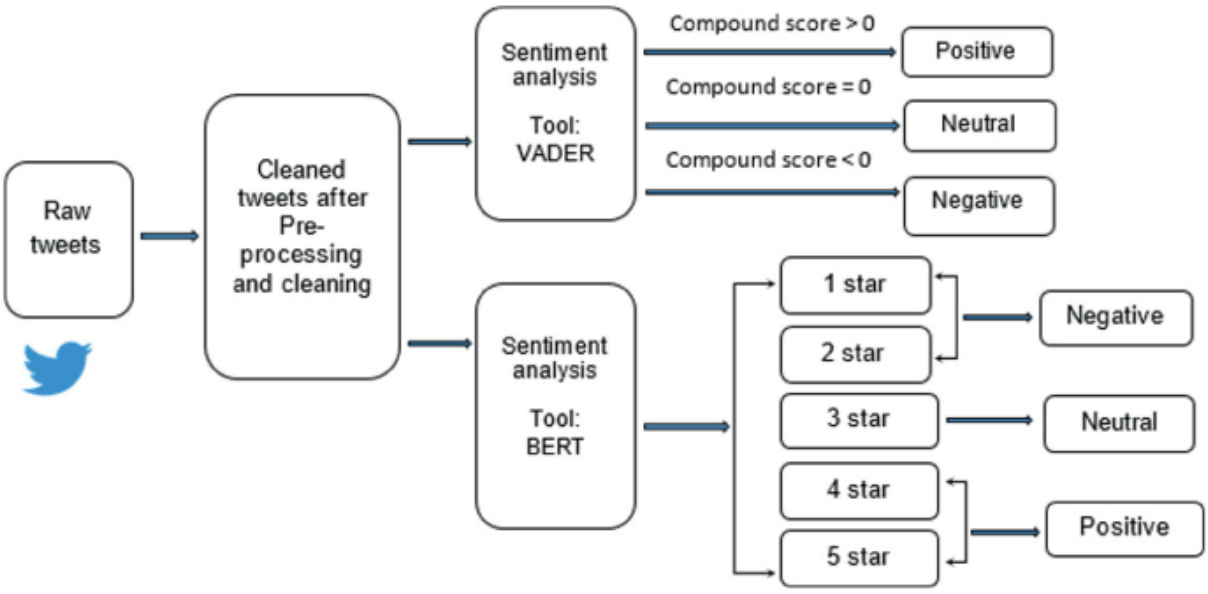

Fig 5.1. Sentiment analysis

Figure 5.1 shows a sample sentiment analysis methodology and procedure for a twitter analysis model and it is evident how the VADER and the BERT model differs in the analysis methodology

5.1.VADER Approach:

The Valence Aware Dictionary and sEntiment Reasoner (VADER) is a rule-based sentiment analysis tool designed for efficient handling of social media text and short reviews. Therefore this tool is particularly effective for analyzing the user sentiments expressed in informal contexts, such as online product reviews.

The sentiment analysis begins with tokenization, where the input text is broken down into individual words and punctuation marks, referred to as tokens. The VADER uses a predetermined lexicon of sentiment-related words, of which each are assigned a particular score, called valence score which indicates the intensity of the sentiment. Therefore, according to this, the positive words have positive scores, the negative words have negative scores, and neutral words have scores closer to zero.

VADER not only identifies the sentiment of each and every word but also checks the overall context of the sentience in which the words are used. This is done by several procedures like:

- **Negation Handling:** Adjusting sentiment scores if a negation word precedes a positive or negative word.
- **Intensity Modifiers:** Recognizing adverbs and adjectives that affect sentiment intensity.
- **Punctuation and Capitalization:** Using exclamation marks and capitalized words to adjust sentiment intensity.
- **Special Text Handling:** Recognizing emoticons, emojis, slang, and abbreviations to accurately gauge sentiment.

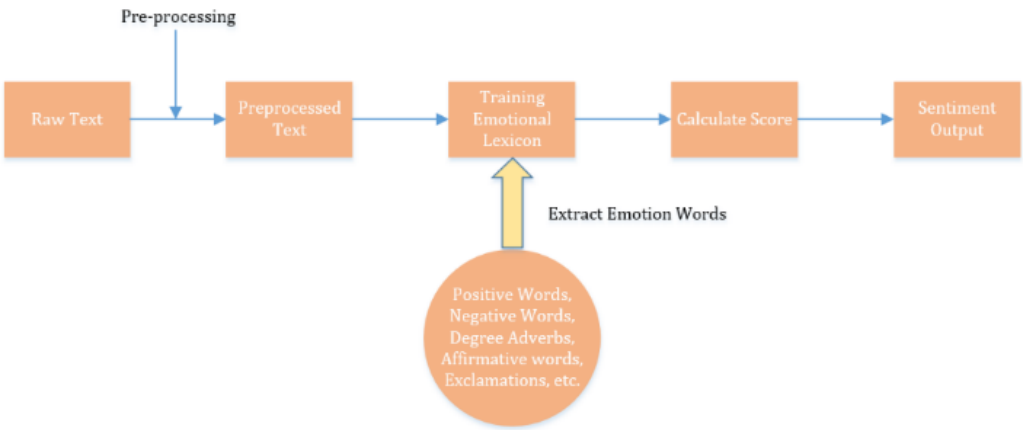

Fig 5.2. Working of VADER Model

Figure 5.2 showcases the VADER model's sentiment analyses procedure and mechanism

The sentiment scores are classified into four categories:

- Positive
- Negative
- Neutral
- Compound - a sum of the valence scores of each word ranging from -1 (most extreme negative) to +1 (most extreme positive)

This scoring system allows VADER to provide a accurate sentiment analysis, making it effective for texts written in informal language, for text with slang, abbreviations, emoticons, punctuation, and special characters. This makes VADER a suitable choice for analyzing product reviews where users often express sentiments casually.

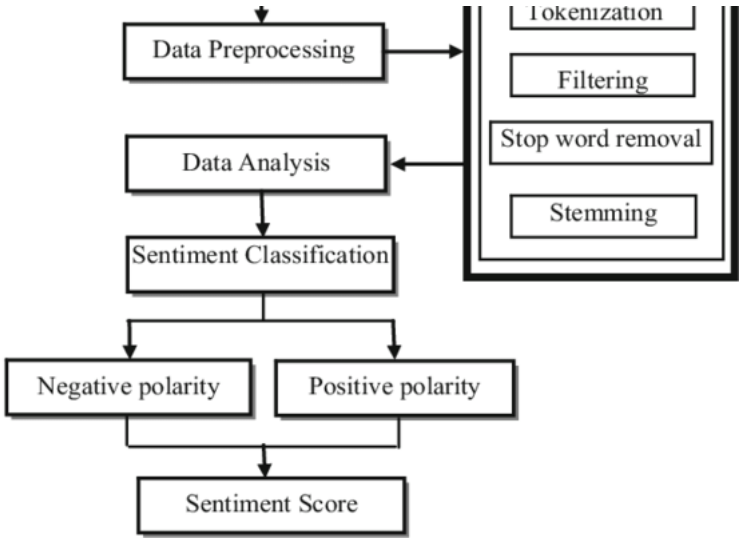

Fig 5.3. Steps in Sentiment Analysis

5.2.Machine Learning Approach:

DistilBERT, short for Distilled Bidirectional Encoder Representations from Transformers, is a machine learning model based on neural network architecture. It is a smaller, faster, and lighter version of the BERT model. DistilBERT retains 97% of BERT's performance while being 60% faster and 40% smaller, making it an efficient and powerful tool for sentiment analysis.

For this project, the pre-trained DistilBERT model from HuggingFace was utilized, specifically the multilingual base case model. This model was chosen for its proven effectiveness in understanding context and nuances in text across different languages. The multilingual capability is particularly beneficial given the diversity of languages in user reviews. The model was pre-trained on a large corpus of text data, allowing it to capture a wide range of linguistic patterns and semantics, which is crucial for accurate sentiment analysis.

Working of the model:

- **Tokenization**: The input text is tokenized into subwords, which allows the model to handle out-of-vocabulary words by breaking them into smaller, meaningful components.
- **Embedding**: Each token is converted into a high-dimensional vector (embedding) that captures its semantic meaning.
- **Transformer Layers:** The embeddings are passed through multiple transformer layers that apply self-attention mechanisms to understand the relationships and dependencies between tokens in the context of the entire sentence.
- **Contextual Understanding:** DistilBERT processes the entire text bidirectionally, meaning it takes into account both the left and right context of each token. This bidirectional approach enables the model to capture complex semantic and syntactic patterns in the text.
- **Output:** The final output is a set of embeddings for each token, which are then used

to predict the sentiment of the entire review.

5.2.1. Sentiment Classification

The embeddings generated by DistilBERT are fed into a classification layer to predict the sentiment of the text. This involves training a logistic regression classifier on top of the embeddings to categorize the sentiment as positive, negative, or neutral. The model's predictions are based on the nuanced understanding of context and word relationships learned during pre-training.

5.2.2. Advantages of this approach:

- **Contextual Understanding:** DistilBERT's ability to understand context and relationships between words leads to highly accurate sentiment predictions, even in complex sentences.
- **Multilingual Capability:** The model's ability to handle multiple languages ensures that reviews in different languages are accurately analyzed.
- **Efficiency:** DistilBERT's optimized architecture allows for faster processing without significant loss of accuracy, making it suitable for large-scale sentiment analysis tasks.

5.3. Feature Extraction

Feature extraction is a crucial step in the sentiment analysis pipeline, transforming raw text into numerical representations that models can process. For both the rule-based and machine learning approaches, the TF-IDF (Term Frequency-Inverse Document Frequency) method is used.

$$TF = \frac{Number\ of\ times\ a\ word\ "X"\ appears\ in\ a\ Document}{Number\ of\ words\ present\ in\ a\ Document}$$

$$IDF = log\left(\frac{Number\ of\ Documents\ present\ in\ a\ Corpus}{Number\ of\ Documents\ where\ word\ "X"\ has\ appeared}\right)$$

$$TF\ IDF\ =\ TF * IDF$$

Fig 5.4. TF-IDF Features

5.3.1. TF-IDF Method:

- TF-IDF is applied to convert the text reviews into numerical vectors.
- TF-IDF measures the importance of a word in a document relative to a collection of documents, helping to identify the most significant words that contribute to the sentiment.
- This method was applied to both the rule-based VADER approach and the machine learning-based DistilBERT approach to ensure a consistent and comprehensive analysis.
- Each review is tokenized, and stop words are removed to reduce noise.
- The TF-IDF vectorizer then generates a matrix where each column represents a word from the corpus, and each row represents a document.
- TF-IDF helps in identifying the most significant words in the reviews, aiding both VADER and DistilBERT in understanding the sentiment conveyed by the text.

By employing these two methodologies, the aim is to provide a comprehensive analysis of user sentiments towards Intel products. The combination of rule-based and machine learning approaches ensures that the sentiment analysis is both accurate and insightful, leveraging the strengths of each method to deliver a nuanced understanding of the reviews.

6. IMPLEMENTATION

The implementation of the sentiment analysis models was carried out using different software tools and libraries, programmed using Python for data analysis and machine learning. The coding environment chosen for this project was Google Colab, for clean and efficient execution of the code and further evaluation of performance. The online reviews data was imported for analysis through Google File Upload for further processing.

Data Handling:

The raw data was to be converted to a Python pandas data frame to edit and manipulate the data and perform analysis. The primary modules used for data handling and manipulation included:

- **Pandas:** Data manipulation and analysis, for a flexible data structure for loading, cleaning, and preprocessing the dataset.
- **Matplotlib and Seaborn:** For data visualization and plotting results and graphs to understand data distributions and trends.
- **Tqdm:** Module used for progress tracking, for a smoother user operation by acting as a progress indicator for possibly longer running operations.

Data Preprocessing:

To prepare the dataframe containing raw data for sentiment analysis, several preprocessing steps were carried out. The following libraries and tools were used for these tasks:

- **NLTK (Natural Language Toolkit):** Library for natural language processing tasks, like tokenization, stop word removal, and lemmatization.

- **Langdetect**: Used for language detection to ensure the multi lingual reviews were also processed in the correct language.
- **Googletrans**: For translating non-English reviews into English.

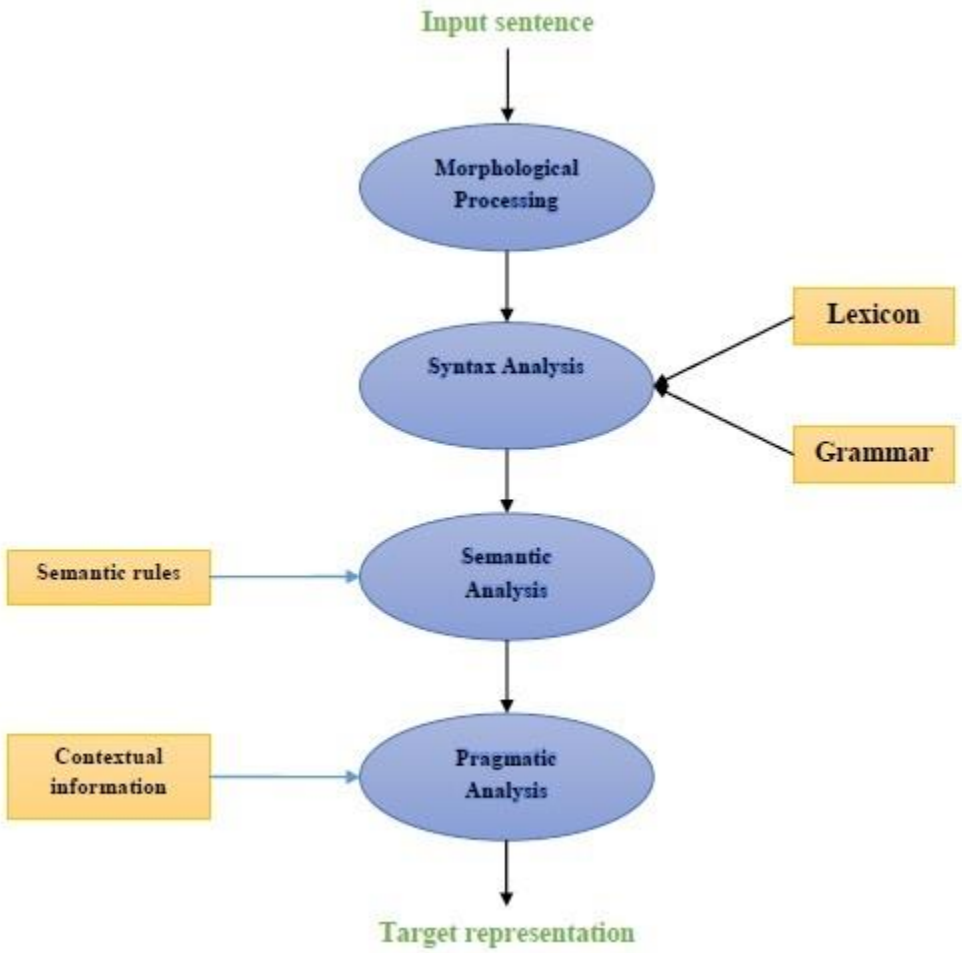

Fig 6.1. NLP Analysis

This phase included cleaning of data by removing duplicates, handling missing values, translating non-English reviews, and processing the text by tokenization, stop word removal, and lemmatization.

Sentiment Analysis:

The sentiment analysis of reviews was implemented using two approaches: a rule-based method with VADER and a machine learning approach with a pre-trained DistilBERT model.

Rule-Based VADER Approach:

- NLTK Sentiment Intensity Analyzer: Library used for performing sentiment analysis using the VADER model, which is specifically used in handling social media text and short reviews by analyzing the sentiment intensity of words in sentences and assigning a score based on that.

Machine Learning Approach:

- Transformers: The Transformers library was used to implement a pipeline the ML model by Hugging Face, that is a pre-trained DistilBERT base case multilingual model. This model was trained and implemented using logistic regression for performing analysis.

Feature Extraction:

Feature extraction was performed using the TF-IDF (Term Frequency-Inverse Document Frequency) method, which converts the textual data into particular numerical features and tokens making sure they are ideal for analysis by machine learning algorithms.

- Scikit-learn: Library used for feature extraction using the TfidfVectorizer, that converts the text from review into a matrix TF-IDF features and scores.

Model Training:

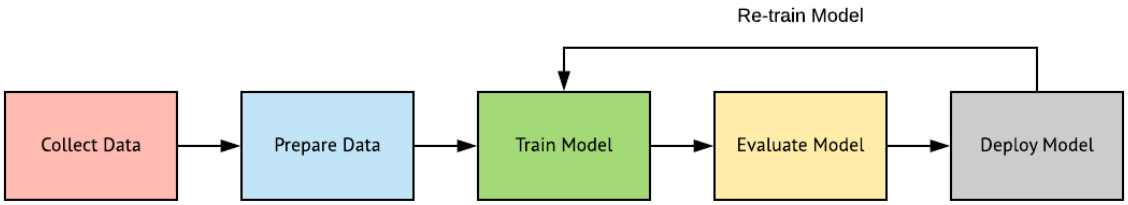

Fig 6.2. Machine Learning Approach

The distilBERT model was trained by splitting the dataset into training and testing sets to evaluate the model's performance effectively.

- **Data Split**: The data was divided into training and testing sets based on the standard 80-20 rule, with 80% of the data used for training and 20% for testing.
- **Model Training:** The DistilBERT base case multilingual model was fine-tuned on the training data. Logistic regression was used for classification within the pipeline, to improve and optimize the model's performance.

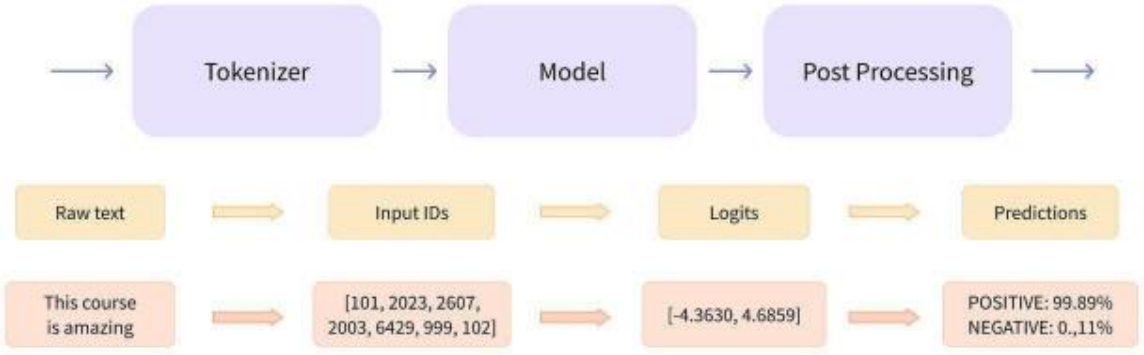

Fig 6.3. ML Model Mechanism

6.1.Evaluation Metrics

To evaluate the performance of the ML model, several metrics were used:

- **Accuracy**: Measures the overall correctness of the model by calculating the ratio of correctly predicted instances to the total instances.
- **Precision**: Indicates the proportion of true positive predictions among all positive predictions, reflecting the model's ability to identify relevant instances.
- **Recall**: Measures the proportion of true positive predictions among all actual positive instances, indicating the model's ability to capture all relevant instances.
- **F1 Score**: The harmonic mean of precision and recall, providing a single metric that balances both precision and recall.

The performance of the distilBERT model was evaluated using these metrics.

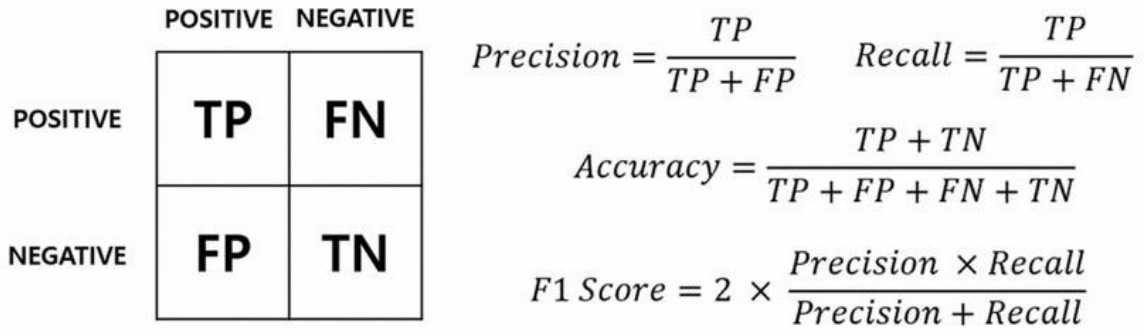

Fig 6.4. Model evaluation metrics

6.2.Plotting results:

To visually represent the results of the sentiment analysis, different plotting techniques has been used. This is done to help in understanding the distribution of sentiments and identifying the key features in particular user sentiments.

- **WordCloud Library:** This library was used to generate word clouds for both positive and negative sentiments. Word clouds were used to provide a visual representation of the most common words in positive and negative reviews. This technique highlights the frequency of words, with more frequent words appearing larger in the word cloud.
- **Matplotlib for Sentiment Distribution:** Matplotlib was used to create bar plots and other types of visualizations to showcase the overall distribution of positive and negative sentiments in the dataset.

By comparing the results, the processors strengths and weaknesses were identified, leading to a comprehensive understanding of their performance in sentiment analysis of Intel product reviews.

7. RESULTS AND DISCUSSION

VADER'S APPROACH:

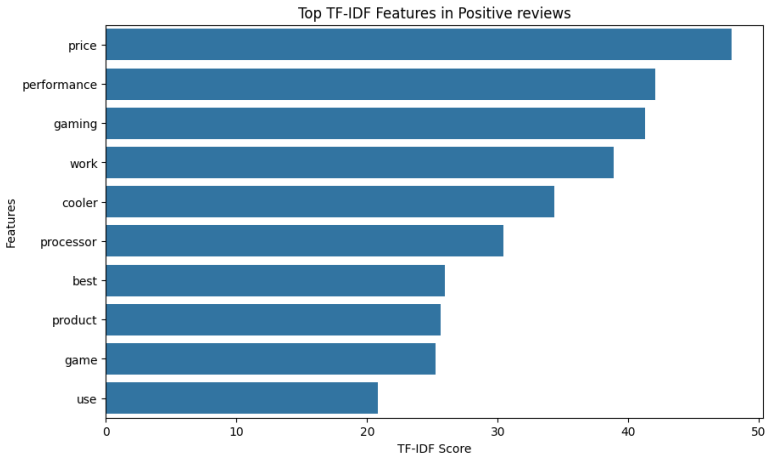

Fig 7.1. Positive TF-IDF of 12th generation

Figure 7.1 and 7.2 showcases the top positive and negative TF-IDF features for the 12th generation Intel processors respectively, obtained through VADER'S model highlighting the common sentiment in user reviews

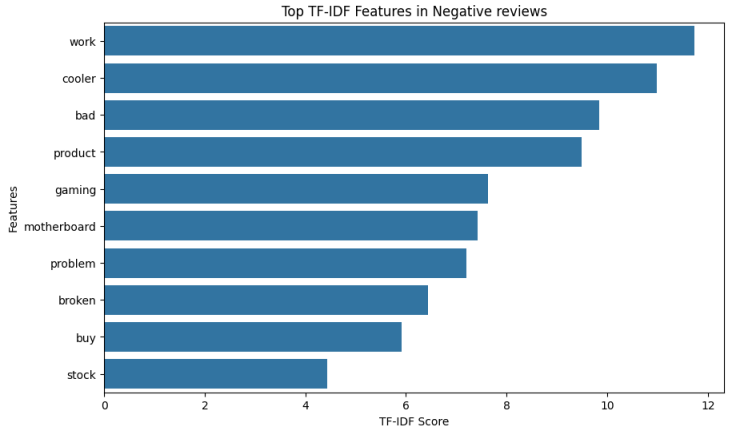

Fig 7.2. Negative TF-IDF of 12th generation

Fig 7.3. Word Clouds of 12th generation

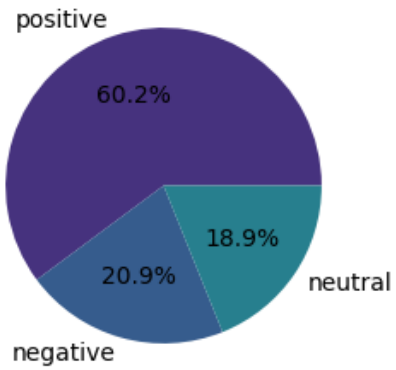

Fig 7.4. Sentiment distribution of 12th generation processors

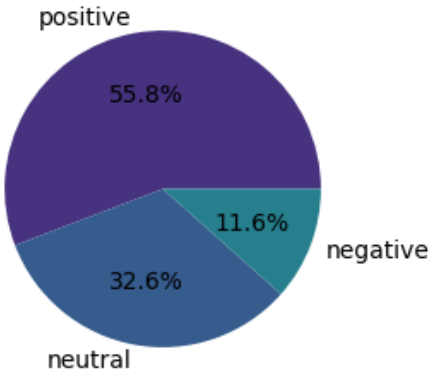

Fig 7.5. Sentiment distribution of 13th generation processors

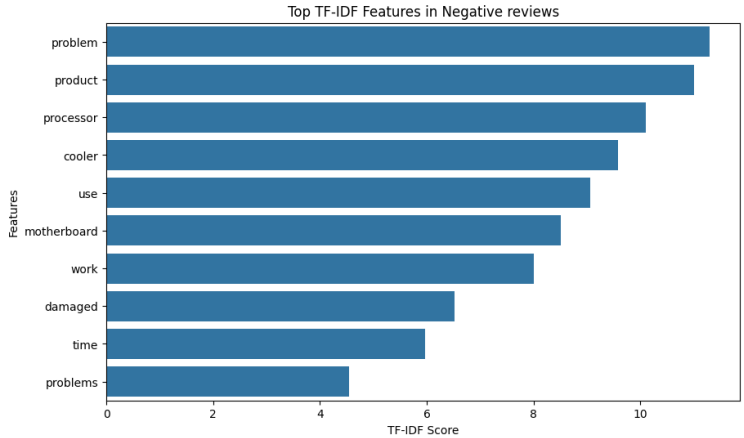

Fig 7.6. Negative TF-IDF of 13th generation

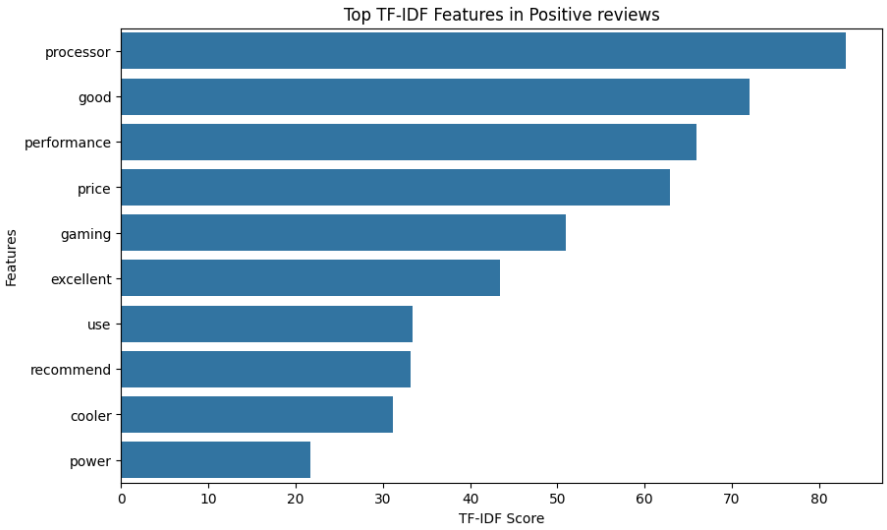

Fig 7.7. Positive TF-IDF of 13th generation

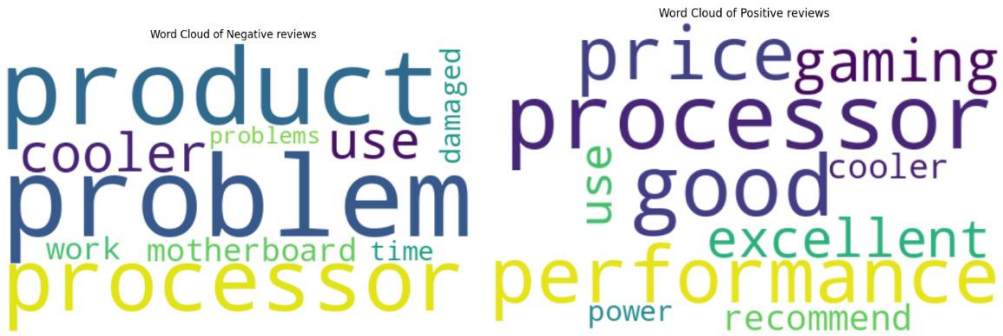

Fig 7.8. Word Clouds of 13th generation

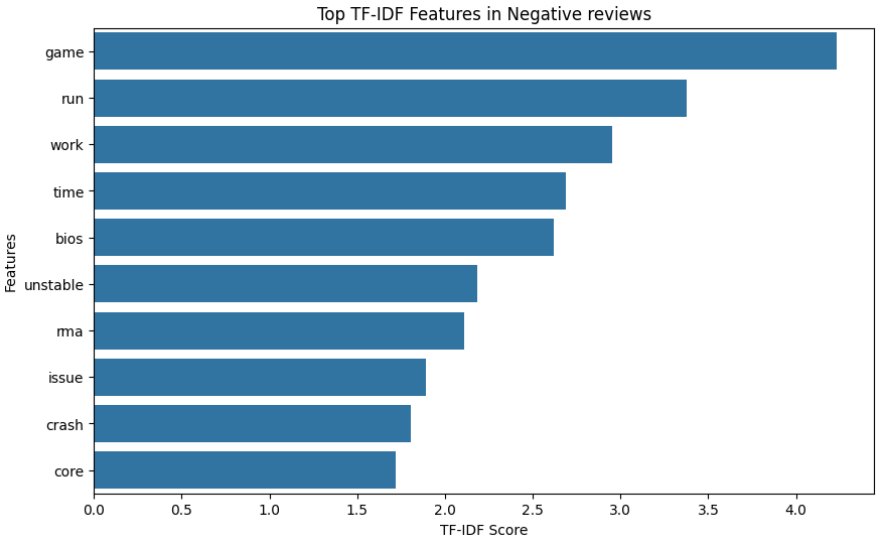

Fig 7.9. Negative TF-IDF of 14th generation

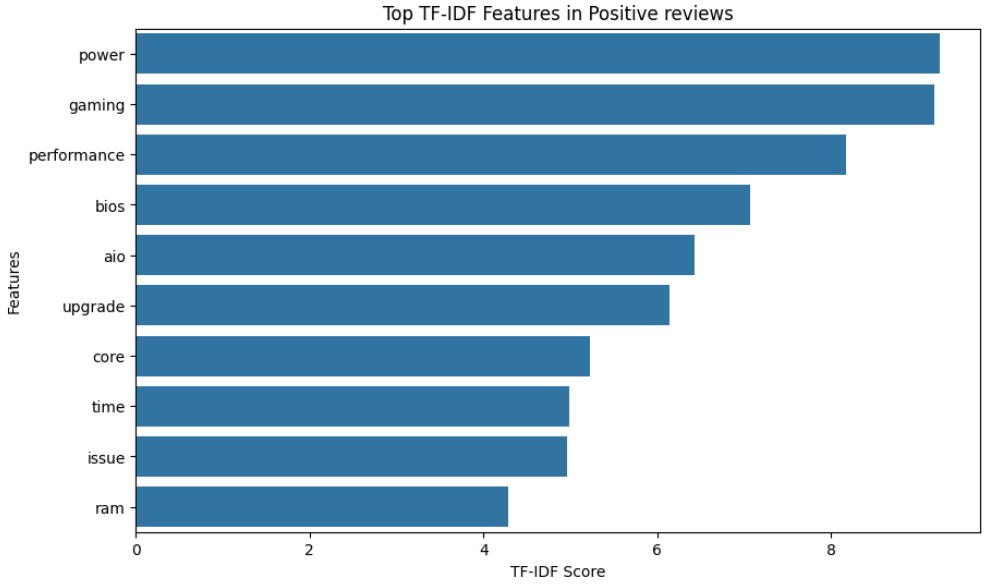

Fig 7.10. Positive TF-IDF of 14th generation

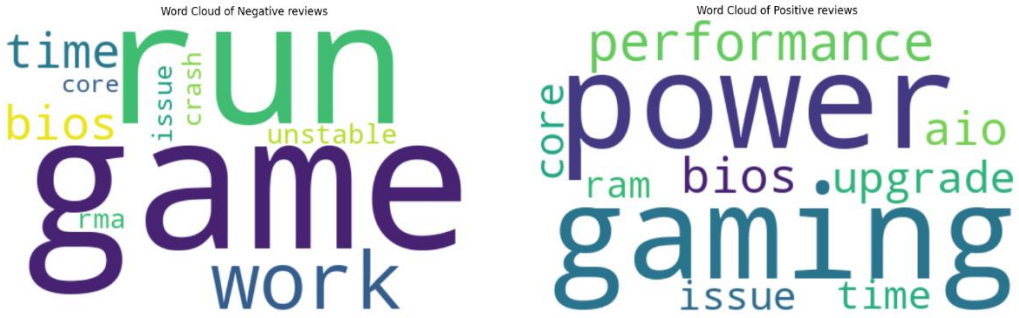

Fig 7.11. Word Clouds of 14th generation

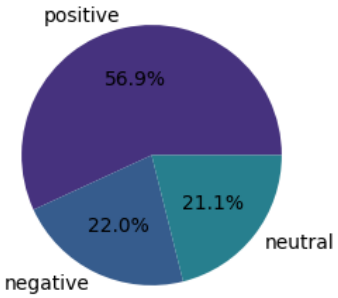

Fig 7.12. Sentiment distribution of 14th generation processors

DistilBERT APPROACH:

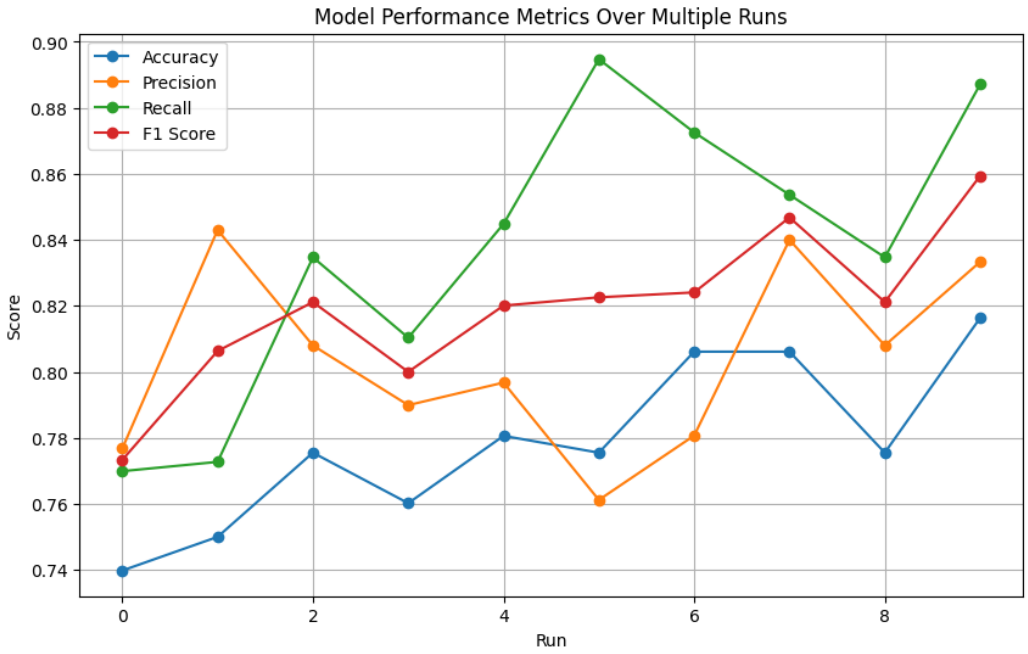

Fig 7.13. DistilBERT Model Performance Metrics

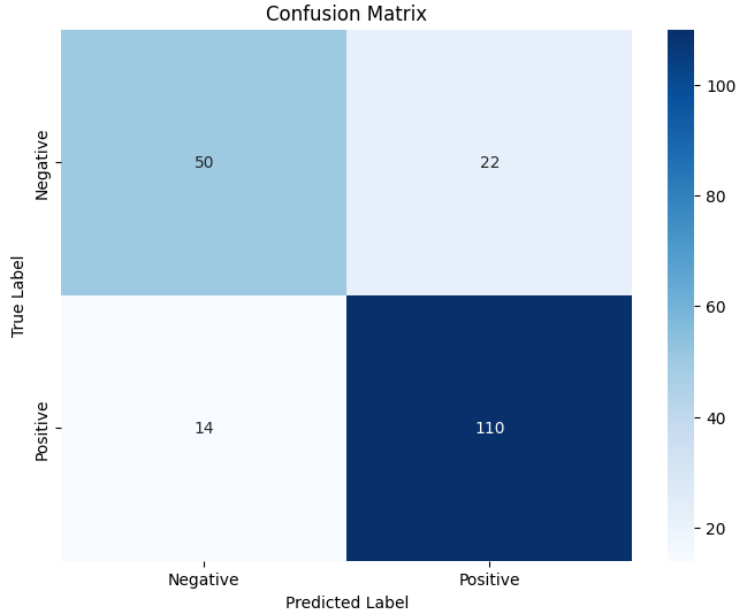

Fig 7.14. Confusion Matrix of DistilBERT Model

Generation wise Analysis:

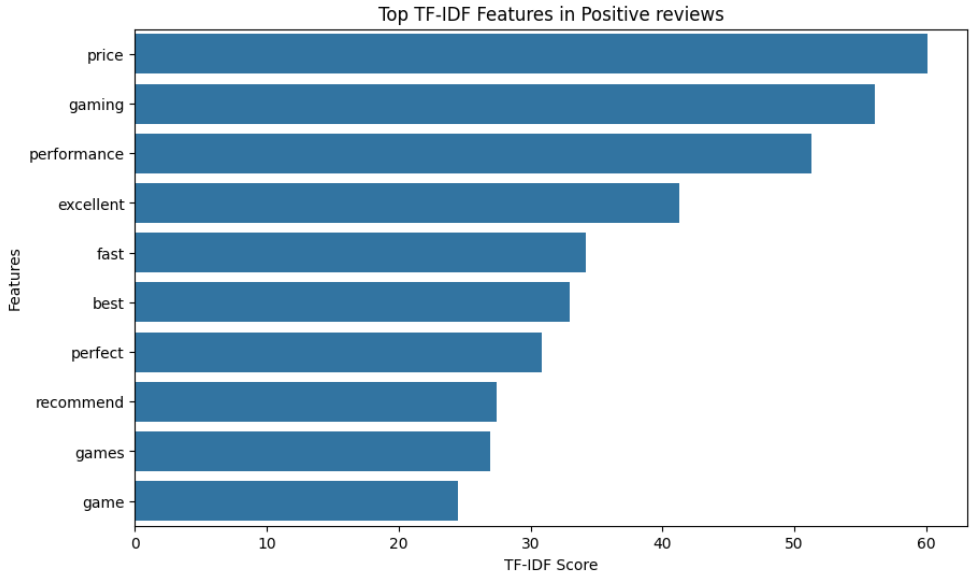

Fig 7.15. Positive TF-IDF of 12th generation

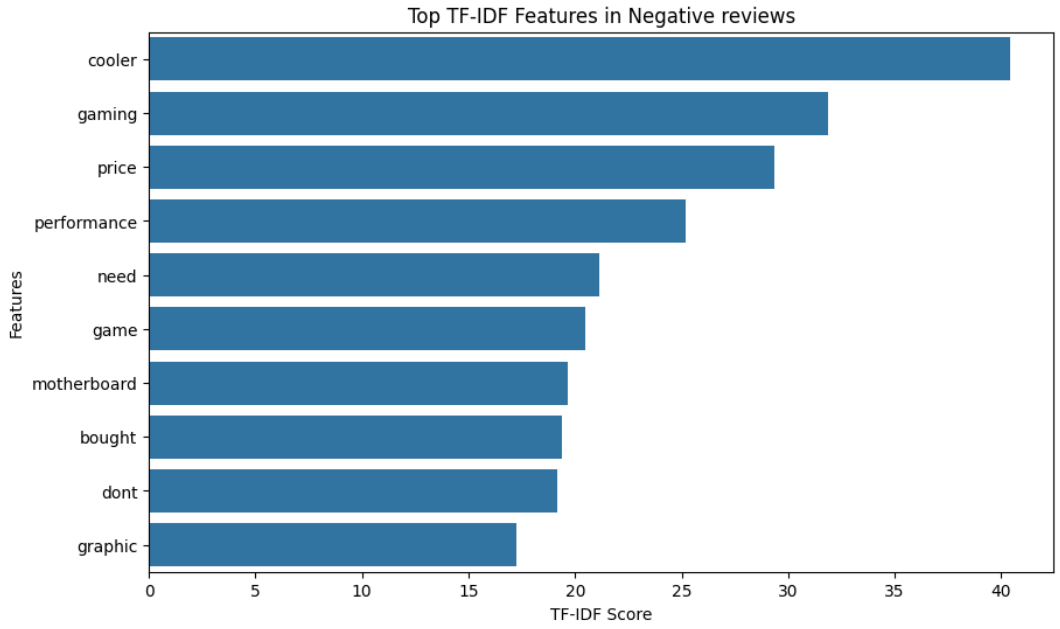

Fig 7.16. Negative TF-IDF of 12th generation

Fig 7.17. Word Clouds of 12th generation

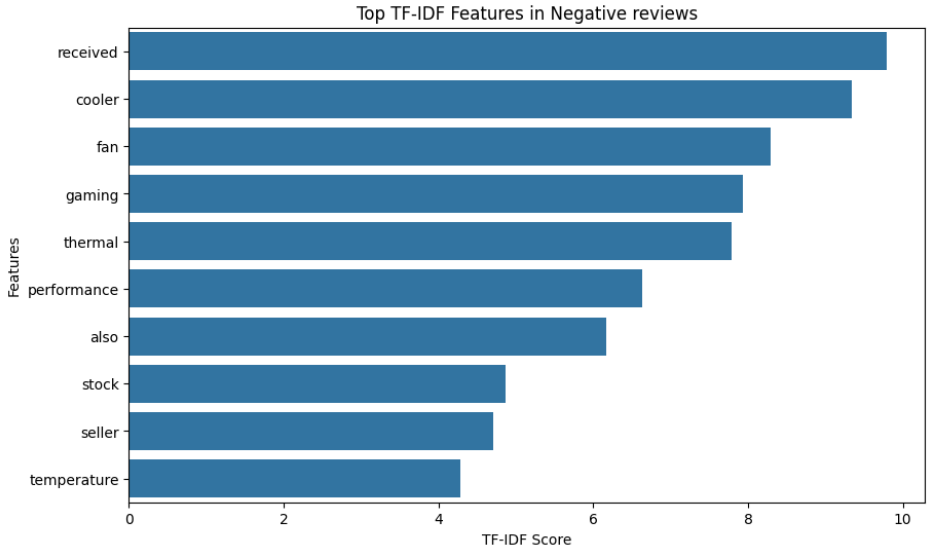

Fig 7.18. Negative TF-IDF of 13th generation

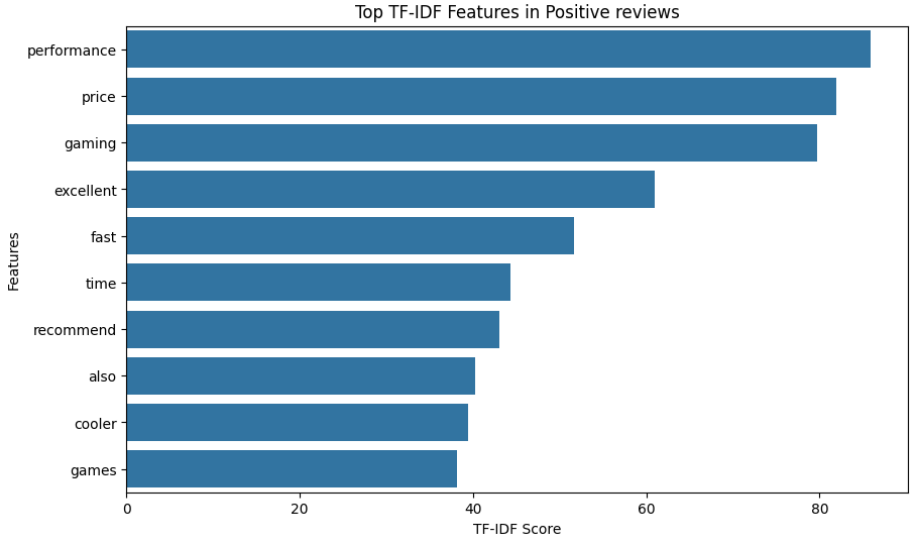

Fig 7.19. Positive TF-IDF of 13th generation

Fig 7.20. Word Clouds of 13th generation

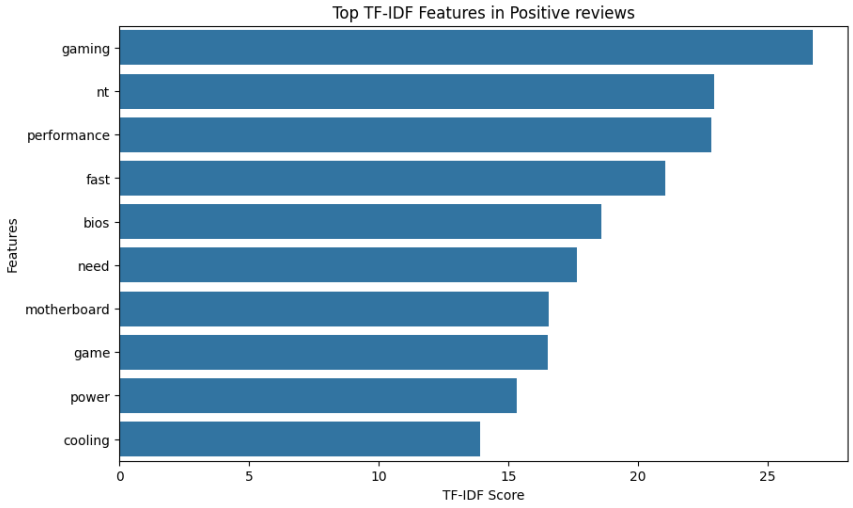

Fig 7.21. Positive TF-IDF of 14th generation

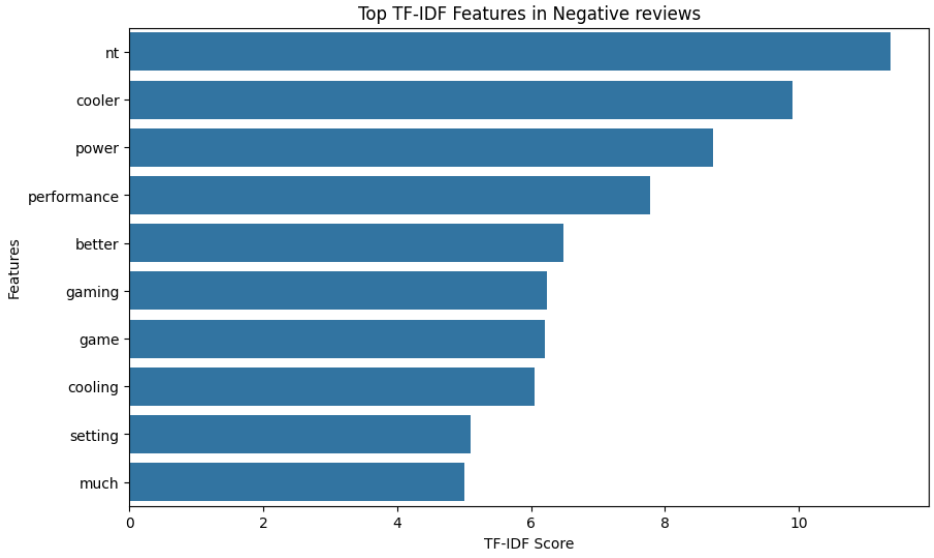

Fig 7.22. Negative TF-IDF of 14th generation

Fig 7.23. Word Clouds of 14th generation processors

Generation	Positive Features	Negative Features
12th Gen	Price, Performance, Gaming, Good, Fast, Excellent	Cooler, Working, Broken, Motherboard
13th Gen	Price, Excellent, Performance, Recommend, Fast	Problem, Processor, Cooler, Damaged, Thermal, Temp
14th Gen	Power, Performance, BIOS Upgrade, RAM, AIO, Motherboard	Game Run, Work, Unstable, BIOS, Cooler Not

Fig 7.24. Generation wise Performance Comparison

Figure 7.24 highlights the overall comparison of the performance of the different Intel processors and the difference in the negative and positive sentiments. Through this it is apparent that the 12th generation processors exhibit a relatively higher positive sentiment, with keywords such as "performance," "gaming," "good," "fast," and "excellent" being frequently mentioned. These terms underscore strong user satisfaction related to performance and gaming capabilities. In contrast, the 13th generation reviews also highlight positive aspects like "price," "performance," and "fast," with a notable inclusion of "recommend," indicating robust consumer endorsement. However, this generation is marked by an increase in negative comments focusing on issues like "thermal temp" and "damaged," which reflect some concerns about reliability. The 14th generation introduces a broader range of positive

keywords, including "power," "bios," "upgrade," and "ram," suggesting advancements and additional features. Despite these improvements, negative sentiment also intensifies with mentions of "unstable," "bios," and "cooler," indicating ongoing issues with stability and hardware.

The 14th generation shows significant improvements with the introduction of new positive keywords such as "power," "bios upgrade," and "ram," which were less emphasized in earlier generations. This suggests enhancements in hardware capabilities and feature sets. Despite these advancements, the 14th generation continues to face issues with stability and reliability, as indicated by the increased negative comments. This suggests that while new features are introduced, some fundamental concerns about system stability persist.

8. CONCLUSION

The sentiment analysis was performed to find patterns in user reviews across several Intel processor generations by using this methodology. This gave an overview of each product's advantages and shortcomings as well as a look at how customer satisfaction has changed over time. An in-depth evaluation of Intel's advancement in meeting customer wants and preferences was made possible by the comparison of user opinions across generations. The VADER and DistilBERT methodologies both demonstrated exceptional efficacy, with their respective performances being comparable. It was clear from the data that customer opinions differed between Intel CPU generations. Through the identification of frequently occurring positive and negative terms, it became feasible to identify the particular aspects that users found deficient or appreciated.

The analysis reveals a trend of increasing feature richness and performance in newer processor generations, with the 14th generation reflecting notable improvements in hardware capabilities and features. However, it also highlights that some issues, particularly related to system stability and cooling, continue to impact user satisfaction across generations. The 12th generation stands out for its relatively higher positive sentiment, while the 13th and 14th generations show both improvements and emerging concerns.

This information is valuable for Intel to continue improving their products and addressing any recurring issues. To sum up, sentiment analysis of user evaluations is a crucial technique for assessing the effectiveness of products and comprehending client input. The integration of rule-based and machine learning methodologies yielded a thorough and precise evaluation of user feelings, delivering valuable insights for the advancement and enhancement of products. This study provided a comprehensive picture of the changing patterns in consumer satisfaction with Intel processors, highlighting the need of ongoing innovation and adaptability to market demands.

REFERENCES

[1] Abraham, M. P. (2020). Feature based sentiment analysis of mobile product reviews using machine learning techniques. *International Journal of Advanced Trends in Computer Science and Engineering*, *9*(2), 2289-2296. https://doi.org/10.30534/ijatcse/2020/210922020

[2] Ahmed, H., & Aljuboori, F. (2023). undefined. https://doi.org/10.21203/rs.3.rs-2841831/v1

[3] Analysis of product reviews based on sentiment analysis by using machine learning and deep learning. (2023). *neuroquantology*, *20*(03). https://doi.org/10.48047/nq.2022.20.3.nq22956

[4] Chockalingam, N. (2018). Simple and effective feature based sentiment analysis on product reviews using domain specific sentiment scores. *Polibits*. https://doi.org/10.17562/pb-of-6k39

[5] Datt, J. S. (2023). Sentiment analysis using customer feedback. *International Journal of Trendy Research in Engineering and Technology*, *07*(04), 09-13. https://doi.org/10.54473/ijtret.2023.7402

[6] Devasia, N., & Sheik, R. (2016). Feature extracted sentiment analysis of customer product reviews. *2016 International Conference on Emerging Technological Trends (ICETT)*. https://doi.org/10.1109/icett.2016.7873646

[7] A, G., Baranwal, S. K., Wali Haider Zaidi, S. M., & Srivastava, D. (2023). Perceptual based sentiment analysis of consumer reviews using rational machine learning techniques for e-Commerce applications. *2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT)*. https://doi.org/10.1109/easct59475.2023.10393149

[8] Kazemian, S., Zhao, S., & Penn, G. (2014). Evaluating sentiment analysis evaluation: A case study in securities trading. *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. https://doi.org/10.3115/v1/w14-2620

[9] Mboutayeb, S., Majda, A., & Nikolov, N. S. (2021). Multilingual sentiment analysis: A deep learning approach. *Proceedings of the 2nd International Conference on Big Data, Modelling and Machine Learning*. https://doi.org/10.5220/0010727700003101

[10] Nedjah, N., Santos, I., & De Macedo Mourelle, L. (2019). Sentiment analysis using convolutional neural network via word embeddings. *Evolutionary Intelligence*, *15*(4), 2295-2319. https://doi.org/10.1007/s12065-019-00227-4

[11] Onan, A. (2020). Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks. *Concurrency and Computation: Practice and Experience*, *33*(23). https://doi.org/10.1002/cpe.5909

[12] Prakash, Y., & Sharma, D. K. (2023). Aspect based sentiment analysis for Amazon data products using PAM. *2023 6th International Conference on Information Systems and Computer Networks (ISCON)*. https://doi.org/10.1109/iscon57294.2023.10112193

[13] Purohit, A. (2021). Sentiment analysis of customer product reviews using deep learning and compare with other machine learning techniques. *International Journal for Research in Applied Science and Engineering Technology*, *9*(VII), 233-239. https://doi.org/10.22214/ijraset.2021.36202

[14] Sahu, T. P., & Khandekar, S. (2022). A machine learning-based lexicon approach for sentiment analysis. *Research Anthology on Implementing Sentiment Analysis Across Multiple Disciplines*, 836-851. https://doi.org/10.4018/978-1-6684-6303-1.ch044

[15] Shahid, A. (2023). Natural language processing techniques for sentiment analysis in social media. https://doi.org/10.31219/osf.io/qgdbz

[16] Soni, J., Xavier, S., Saxena, R., & Waghela, P. (2024). Leveraging web scraping for aspect-based sentiment analysis: A case study on Flipkart customer reviews of mobile phones. *International Journal of Research Publication and Reviews*, *5*(6), 6747-6752. https://doi.org/10.55248/gengpi.5.0624.1638

[17] Taparia, A., & Bagla, T. (2020). Sentiment analysis: Predicting product reviews' ratings using online customer reviews. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3655308

[18] Victor Rajan, K. (2024). Sentiment analysis of social media using artificial intelligence. *Artificial Intelligence*. https://doi.org/10.5772/intechopen.113092

[19] Wang, C., Zhu, X., & Yan, L. (2022). Sentiment analysis for e-Commerce reviews based on deep learning hybrid model. *2022 5th International Conference on Signal Processing and Machine Learning*. https://doi.org/10.1145/3556384.3556391

[20] Yadav, A., Yadav, D., & Jain, A. (2018). An improvised feature-based method for sentiment analysis of product reviews. *ICST Transactions on Scalable Information Systems*, 165670. https://doi.org/10.4108/eai.13-7-2018.165670

APPENDICES

VADER'S APPROACH:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from tqdm.notebook import tqdm
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
from langdetect import detect
from googletrans import Translator
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import string
from sklearn.feature_extraction.text import TfidfVectorizer
from wordcloud import WordCloud
nltk.download('vader_lexicon')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
df = pd.read_csv("Intel 14th gen data.csv")
sia = SentimentIntensityAnalyzer()
translator = Translator()
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))
stop_words.update(['cpu','good','processor','intel','chip','core','i9',
'gen','run','pc','used','return','arrived','box','new','amazon','issue'
,'day',"don't",'open','great','open','core','got','im','fine','like','d
idnt','tried','despite','month','case','high'])
print("Size of data:")
print("Before preprocessing:",df.shape)

def preprocessing(df):

    df.drop_duplicates(subset='reviewId', keep='first', inplace=True)
    df['reviewDescription'].fillna('', inplace=True)
    df['reviewDescription'] =
df['reviewDescription'].astype(str).str.lower()

    def clean_text(text):
        text = text.translate(str.maketrans('', '',
string.punctuation))  # Remove punctuation
        words = nltk.word_tokenize(text)
```

```
        words = [word for word in words if word.isalpha()]  # Remove
non-alphabetic tokens
        words = [word for word in words if word not in stop_words]  #
Remove stop words
        lemmatized_words = [lemmatizer.lemmatize(word) for word in
words]
        return ' '.join(lemmatized_words)

    df['reviewDescription'] = df['reviewDescription'].apply(clean_text)
    return df

df = preprocessing(df)
print("After preprocessing:",df.shape)

def Translate(text):
    try:
        lang = detect(text)
    except:
        lang = 'unknown'

    if lang != 'en':
        try:
            text = translator.translate(text, src=lang, dest='en').text
        except:
            text = ''

    return text

def analyze_sentiment(df):
    res = {}
    positive_comments = []
    negative_comments = []

    for i, row in tqdm(df.iterrows(), total=len(df)):
        text = str(row['reviewDescription'])
        text = Translate(text)

        scores = sia.polarity_scores(text)
        myid = row['Id']
        res[myid] = scores

        if scores['compound'] >= 0.05:
            positive_comments.append(text)
        elif scores['compound'] <= -0.05:
            negative_comments.append(text)

    return res, positive_comments, negative_comments
```

```
res, positive_comments, negative_comments = analyze_sentiment(df)

for myid, scores in res.items():
    print(f"Review ID: {myid}, Sentiment Scores: {scores}")

scores_df = pd.DataFrame.from_dict(res, orient='index')
scores_df.index.name = 'Review ID'

vaders = df.merge(scores_df, left_on='Id', right_index=True)
vaders.head()
def features(comments, n_features=10):

    vectorizer = TfidfVectorizer(max_features=n_features,
stop_words='english')
    tfidf_matrix = vectorizer.fit_transform(comments)
    feature_names = vectorizer.get_feature_names_out()
    sums = tfidf_matrix.sum(axis=0)
    data = [(feature_names[col], sums[0, col]) for col in
range(sums.shape[1])]
    ranking = pd.DataFrame(data, columns=['feature',
'tfidf']).sort_values(by='tfidf', ascending=False)

    return ranking

positive_features = features(positive_comments)
negative_features = features(negative_comments)
#TF-IDF features positive reviews
plt.figure(figsize=(10, 6))
sns.barplot(x='tfidf', y='feature', data=positive_features)
plt.title('Top TF-IDF Features in Positive reviews')
plt.xlabel('TF-IDF Score')
plt.ylabel('Features')
plt.show()

print("\n" * 3)

#TF-IDF features negative reviews
plt.figure(figsize=(10, 6))
sns.barplot(x='tfidf', y='feature', data=negative_features)
plt.title('Top TF-IDF Features in Negative reviews')
plt.xlabel('TF-IDF Score')
plt.ylabel('Features')
plt.show()

print("\n" * 3)

# Word Cloud Positive reviews
plt.figure(figsize=(10, 6))
```

```
wordcloud_pos = WordCloud(width=500, height=300,
background_color='white').generate_from_frequencies(dict(positive_featu
res.values))
plt.imshow(wordcloud_pos, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Positive reviews')
plt.show()

print("\n" * 3)

# Word Cloud Negative reviews
plt.figure(figsize=(10, 6))
wordcloud_neg = WordCloud(width=500, height=300,
background_color='white').generate_from_frequencies(dict(negative_featu
res.values))
plt.imshow(wordcloud_neg, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Negative reviews')
plt.show()

sentiment_count = vaders['compound'].apply(lambda x: 'positive' if x >=
0.05 else 'negative' if x <= -0.05 else 'neutral')
sentiment_distribution = sentiment_count.value_counts()
plt.figure(figsize=(4, 3))
plt.pie(sentiment_distribution.values,
labels=sentiment_distribution.index, autopct='%1.1f%%',
colors=sns.color_palette('viridis'))
plt.title('Sentiment Distribution')
plt.show()

print("\n\n")

fig, axs = plt.subplots(1, 3, figsize=(12, 3))
sns.barplot(data=vaders, x='ratingScore', y='pos', ax=axs[0])
sns.barplot(data=vaders, x='ratingScore', y='neu', ax=axs[1])
sns.barplot(data=vaders, x='ratingScore', y='neg', ax=axs[2])
axs[0].set_title('Positive')
axs[1].set_title('Neutral')
axs[2].set_title('Negative')
plt.tight_layout()

plt.show()
```

DistilBERT Approach:

```
#Replace with different file name if required
```

```
df = pd.read_csv("Intel 14th gen data.csv")
tqdm.pandas()

def analyze_sentiment(review):
    try:
        result = sentiment_model(review[:512])[0]
        return result['label'], result['score']
    except Exception as e:
        print(f"Error processing review: {review[:50]}... | Error:
{e}")
        return None, None

df[['sentiment_label', 'score']] =
df['reviewDescription'].progress_apply(lambda x:
pd.Series(analyze_sentiment(x)))

df = df.dropna(subset=['sentiment_label', 'score'])

df['sentiment'] = df['sentiment_label'].apply(lambda x: 'positive' if x
== 'LABEL_1' else 'negative')

def store_comments(df):
    positive_comments = df[(df['sentiment'] ==
'positive')]['reviewDescription'].tolist()
    negative_comments = df[(df['sentiment'] ==
'negative')]['reviewDescription'].tolist()
    return positive_comments, negative_comments

positive_comments, negative_comments = store_comments(df)

# Print the count of positive and negative comments
print(f"Number of positive comments: {len(positive_comments)}")
print(f"Number of negative comments: {len(negative_comments)}")

positive_count = len(positive_comments)
negative_count = len(negative_comments)

sentiment_distribution = pd.Series([positive_count, negative_count],
index=['positive', 'negative'])
plt.figure(figsize=(4, 3))
plt.pie(sentiment_distribution.values,
labels=sentiment_distribution.index, autopct='%1.1f%%',
colors=sns.color_palette('viridis'))
plt.title('Sentiment Distribution')
plt.show()

print("\n\n")
fig, axs = plt.subplots(1, 2, figsize=(12, 3))
```

```
positive_df = df[df['sentiment'] == 'positive']
sns.barplot(data=positive_df, x='ratingScore', y='score', ax=axs[0])
axs[0].set_title('Positive')
axs[0].set_ylabel('Score')

negative_df = df[df['sentiment'] == 'negative']
sns.barplot(data=negative_df, x='ratingScore', y='score', ax=axs[1])
axs[1].set_title('Negative')
plt.show()

tqdm.pandas()
translator = Translator()

def detect_and_translate(review):
    try:
        lang = detect(review)
        if lang != 'en':
            translation = translator.translate(review, src=lang,
dest='en')
            return translation.text
        else:
            return review
    except Exception:
        return None

def extract_features(comments, n_features=10):
    stop_words_list = list(stop_words)
    vectorizer = TfidfVectorizer(max_features=n_features,
stop_words=stop_words_list)

    translated_comments = []
    for comment in tqdm(comments, desc="Translating comments"):
        translated_comment = detect_and_translate(comment)
        if translated_comment:
            translated_comments.append(translated_comment)

    tfidf_matrix = vectorizer.fit_transform(translated_comments)
    feature_names = vectorizer.get_feature_names_out()
    sums = tfidf_matrix.sum(axis=0)
    data = [(feature_names[col], sums[0, col]) for col in
range(sums.shape[1])]
    ranking = pd.DataFrame(data, columns=['feature',
'tfidf']).sort_values(by='tfidf', ascending=False)

    return ranking

def plot_features_and_wordclouds(positive_features, negative_features):
```

```
    plt.figure(figsize=(10, 6))
    sns.barplot(x='tfidf', y='feature', data=positive_features)
    plt.title('Top TF-IDF Features in Positive reviews')
    plt.xlabel('TF-IDF Score')
    plt.ylabel('Features')
    plt.show()

    print("\n" * 3)

    plt.figure(figsize=(10, 6))
    sns.barplot(x='tfidf', y='feature', data=negative_features)
    plt.title('Top TF-IDF Features in Negative reviews')
    plt.xlabel('TF-IDF Score')
    plt.ylabel('Features')
    plt.show()

    print("\n" * 3)

    plt.figure(figsize=(10, 6))
    wordcloud_pos = WordCloud(width=500, height=300,
background_color='white').generate_from_frequencies(dict(positive_featu
res.values))
    plt.imshow(wordcloud_pos, interpolation='bilinear')
    plt.axis('off')
    plt.title('Word Cloud of Positive reviews')
    plt.show()

    print("\n" * 3)

    plt.figure(figsize=(10, 6))
    wordcloud_neg = WordCloud(width=500, height=300,
background_color='white').generate_from_frequencies(dict(negative_featu
res.values))
    plt.imshow(wordcloud_neg, interpolation='bilinear')
    plt.axis('off')
    plt.title('Word Cloud of Negative reviews')
    plt.show()

positive_features = extract_features(positive_comments)
negative_features = extract_features(negative_comments)

plot_features_and_wordclouds(positive_features, negative_features)

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
from sklearn.linear_model import LogisticRegression
import numpy as np
```

```
import pandas as pd
X = df['reviewDescription']
y = df['sentiment']

num_runs = 10
accuracies = np.zeros(num_runs)
precisions = np.zeros(num_runs)
recalls = np.zeros(num_runs)
f1_scores = np.zeros(num_runs)

for i in range(num_runs):
    # Splitting data into sections
    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=i)

    vect = TfidfVectorizer()
    X_train_tfidf = vect.fit_transform(X_train)
    X_test_tfidf = vect.transform(X_test)

    model = LogisticRegression()
    model.fit(X_train_tfidf, y_train)

    y_pred = model.predict(X_test_tfidf)

    #Storing metrics
    accuracies[i] = accuracy_score(y_test, y_pred)
    precisions[i] = precision_score(y_test, y_pred,
pos_label='positive')
    recalls[i] = recall_score(y_test, y_pred, pos_label='positive')
    f1_scores[i] = f1_score(y_test, y_pred, pos_label='positive')

plt.figure(figsize=(10, 6))
plt.plot(range(num_runs), accuracies, label='Accuracy', marker='o')
plt.plot(range(num_runs), precisions, label='Precision', marker='o')
plt.plot(range(num_runs), recalls, label='Recall', marker='o')
plt.plot(range(num_runs), f1_scores, label='F1 Score', marker='o')

plt.xlabel('Run')
plt.ylabel('Score')
plt.title('Model Performance Metrics Over Multiple Runs')
plt.legend()
plt.grid(True)
plt.show()

# Print average metrics
print(f"\nAverage Accuracy: {np.mean(accuracies)}")
print(f"Average Precision: {np.mean(precisions)}")
```

```
#Recall may be affected due to false positives
print(f"Average Recall: {np.mean(recalls)}")
print(f"Average F1 Score: {np.mean(f1_scores)}")
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred, labels=['negative', 'positive'])

print("\nConfusion Matrix:\n")

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=['Negative', 'Positive'], yticklabels=['Negative',
'Positive'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()

tn, fp, fn, tp = cm.ravel()

false_positives = fp

accuracy = (tp + tn) / (tp + tn + fp + fn)
precision = tp / (tp + fp)
recall = tp / (tp + fn)
f1_score = 2 * (precision * recall) / (precision + recall)

print("\nResults derived from Confusion Matrix")
print(f"\nAccuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1_score}\n")
```