

INTEL – UNNATI INDUSTRIAL TRAINING PROGRAM
2024-2025

PROJECT NAME

PS-11 Intel Products Sentiment Analysis from Online Reviews

TEAM NAME : UNITY CIRCLE

TEAM MEMBER 1 : KIRTHANA MOHAN R

TEAM MEMBER 2 : SWETHA S

TEAM MEMBER 3:_KRITHIKA S

COLLEGE NAME : SRI SAI RAM INSTITUTE OF TECHNOLOGY

ACKNOWLEDGEMENT

We would like to express Our heartfelt gratitude to Intel Unnati Industrial Training Program 2024-2025 for providing us with the opportunity to undertake this industrial training project. The training and resources provided were instrumental in the successful completion of our sentiment analysis project.

Our sincere thanks go to DEBDYUT HAZRA and other mentors at Intel Unnati Industrial Program for their invaluable guidance, insightful feedback, and continuous support throughout the course of this project. I am also grateful to the entire Intel Unnati team for their encouragement and for creating an environment conducive to learning and development.

We express our gratitude and sincere thanks to our Guide Dr.SU.SUGANTHI for helping us complete the project.

We would like to acknowledge the open-source community for developing the tools and libraries, including NLTK, Gensim, and Hugging Face Transformers, which were crucial for building and implementing the sentiment analysis models used in this project.

Special thanks to our peers and colleagues who provided constructive feedback and support during this journey. Lastly, We are deeply thankful to our family and friends for their unwavering support and patience, which kept us motivated throughout the project.

TABLE OF CONTENTS

SNO	TITLE
1	INTRODUCTION
	1.1 Problem statement
	1.2 Abstract
	1.3 Objective
	1.4 Scope
2	LITERATURE SURVEY
	2.1 Related Work
	2.2 Sentimental Analysis Technique
3	DATA COLLECTION
	3.1 Data source
	3.2 Data Collection Methods
	3.3 Data Description
	3.4 Data Pre-processing
4	SENTIMENTAL ANALYSIS METHODOLOGY
	4.1 Sentimental Analysis Approach
	4.2 Model Selection
	4.3 Feature Extraction
5	IMPLEMENTATION
	5.1 Tools and Libraries
	5.2 Model training
	5.2.1 LSTM model
	5.2.2 Random Forest model
	5.2.3 Roberta model
	5.2.4 Chatbot – recommends based on queries & data
6	RESULTS AND DISCUSSION
7	TEAM CONTRIBUTION
8	CONCLUSION
	8.1 Summary
	8.2 Challenges Faced
	8.3 Future Enhancement

1.INTRODUCTION

1.1 PROBLEM STATEMENT

Customer feedback and Reviews plays a crucial role in the continuous improvement of products and services. Companies often receive vast amounts of textual reviews from end users and tech reviewers on various platforms, making it challenging to manually analyse and derive actionable insights from this data.

Our project aims to address this issue by developing an automated sentiment analysis system that can process customer reviews, detect trends over time, and provide recommendations for INTEL Products (MOBILE/DESKTOP PROCESSOR) using a chatbot

1.2 ABSTRACT

The raise of online reviews has provided a wealth of information for business seeking to understand the customer feedback and tech reviews. This project aims to develop an sentiment analysis system using natural language processing (NLP) techniques , machine learning (ML) models and deep learning (DL)models to classify sentiments in customer reviews efficiently. It involves data preprocessing, sentiment analysis with ML , VADER and LSTM models, and exploratory data analysis (EDA) to identify trends and correlations By visualizing sentiment trends, analyzing product feature correlations, and generating actionable recommendations, this system assists companies in understanding customer needs and improving their products.

1.3 OBJECTIVE

The objective of this project is to develop an automated sentiment analysis system that leverages natural language processing (NLP) and machine learning techniques to better understand their customer opinions and improve their products and services accordingly.

This project provides valuable insights into customer sentiment regarding Intel's (12th, 13th, 14th) generation processors, enabling real time data-driven decisions to enhance product development and marketing strategies. By leveraging advanced machine learning models like Random forest, Logistic Regression, RoBERTa, LSTM, VADER it ensures accurate sentiment analysis and deeper understanding of customer feedback.

1.4 SCOPE OF THE PROJECT

The project aims to analyze customer sentiment towards :

12th generation (mobile / desktop processor)

13th generation (mobile / desktop processor)

14th generation (mobile / desktop processor)

The user end reviews were taken from the AMAZON WEBSITE (<https://www.amazon.in>) around the time frames of 2022 to 2024.

The tech reviews were taken from 3 websites :

PC MAG : <https://www.pcmag.com/categories/processors/brands/intel>

AnandTech : <https://www.anandtech.com/show/18740/the-intel-core-i3-13100f-review-finding-value-in-intels-cheapest-chip>

Trusted Reviews : <https://www.trustedreviews.com/best/best-intel-processor-3517396>

2. LITERATURE SURVEY

2.1 RELATED WORKS

* Recent advancements in sentiment analysis have integrated transformer-based models like RoBERTa, demonstrating superior performance in understanding contextual nuances compared to traditional methods.

* Studies by Liu et al. (2019) have shown that fine-tuning pre-trained models like BERT and RoBERTa achieves state-of-the-art results in sentiment classification tasks.

* Hybrid approaches, such as integrating rule-based methods like VADER with machine learning models, have also been explored

These related works provide a strong foundation for this project. By applying RoBERTa and LSTM models to customer reviews, the project aims to ensure comprehensive and accurate sentiment analysis.

2.2 SENTIMENTAL ANALYSIS TECHNIQUE

Sentiment analysis techniques used are :

1. Rule-based methods

- VADER (valence aware dictionary and sentiment reasoner):

Technique: Rule-based model that uses a lexicon of sentiment-related words.

2. Machine Learning approaches

- Support Vector Machines (SVM):

Technique: Supervised learning model that finds the hyperplane that best separates different classes in feature space.

- Random Forest:

Technique: Ensemble learning method using multiple decision trees.

- Logistic Regression:

Technique: Statistical model for binary classification.

- XGBoost:

Technique: Ensemble learning method using gradient boosting.

3. Deep Learning models

- Convolutional Neural Networks (CNN):

Technique: Neural networks traditionally used for image processing, adapted for text by treating sentences as sequences of word vectors.

4. unsupervised learning approaches

- K mean clustering

Some of the used Sentiment analysis tools used are :

1. NLTK (Natural Language Toolkit):

- **Functionality:** Comprehensive library for natural language processing, including tools for text processing and sentiment analysis (e.g., VADER).

2. Hugging Face Transformers:

- **Functionality:** Library providing pre-trained transformer models like BERT, RoBERTa

3.Scikit-learn: **Functionality:** Library for machine learning, offering algorithms and tools for data preprocessing, model training, and evaluation

3. DATA COLLECTION

3.1 DATA SOURCE

End user reviews : AMAZON (online platform)

Websites like Amazon offer INTEL product reviews from customers which we used for sentiment analysis.

Tech reviews : PC MAG , AnandTECH , TRUSTED REVIEWS

The above websites offered us technical reviews of INTEL products which we used for sentimental analysis

3.2 DATA COLLECTION METHODS

WEB SCRAPING :

Data was collected from Amazon product reviews using web scraping techniques. The scraping was performed using Python scripts to extract user reviews, including the review text, rating, and date of submission. The collected data underwent cleaning to remove HTML tags, URLs, and special characters, followed by normalization, tokenization, and removal of stop words

```

# Extra Data as Html object from amazon Review page
def reviewsHtml(url, len_page):

    # Empty List define to store all pages html data
    soups = []

    # Loop for gather all 3000 reviews from 300 pages via range
    for page_no in range(2, len_page + 1):

        # parameter set as page no to the requests body
        params = {
            'ie': 'UTF8',
            'reviewerType': 'all_reviews',
            'filterByStar': 'critical',
            'pageNumber': page_no,
        }

        # Request make for each page
        response = requests.get(url, headers=headers , params=params)

        # Save Html object by using BeautifulSoup4 and lxml parser
        soup = BeautifulSoup(response.text, 'lxml')
        # Add single Html page data in master soups list
        soups.append(soup)

    return soups

```

```

1 [6]: def getReviews(html_data):

    # Create Empty List to Hold all data
    data_dicts = []

    # Select all Reviews BOX html using css selector
    boxes = html_data.select('div[data-hook="review"]')

    # Iterate all Reviews BOX
    for box in boxes:

        # Select Name using css selector and cleaning text using strip()
        # If Value is empty define value with 'N/A' for all.
        try:
            name = box.select_one('[class="a-profile-name"]').text.strip()
        except Exception as e:
            name = 'N/A'

        try:
            stars = box.select_one('[data-hook="review-star-rating"]').text.strip().split(' out')[0]
        except Exception as e:
            stars = 'N/A'

        try:
            title = box.select_one('[data-hook="review-title"]').text.strip()
        except Exception as e:
            title = 'N/A'

        try:
            # Convert date str to dd/mm/yyyy format
            datetime_str = box.select_one('[data-hook="review-date"]').text.strip().split(' on ')[-1]
            date = datetime.strptime(datetime_str, '%B %d, %Y').strftime("%d/%m/%Y")
        except Exception as e:
            date = 'N/A'

        try:
            description = box.select_one('[data-hook="review-body"]').text.strip()
        except Exception as e:
            description = 'N/A'

        # create Dictionary with al review data
        data_dict = {
            'Name' : name,
            'Stars' : stars,
            'Title' : title,
            'Date' : date,
            'Description' : description

```


3.3 DATA DESCRIPTION

Data Fields:

- **Description :** The main content of the review written by the customer.
- **Stars :** The star rating given ranging from 1 to 5.
- **Date:** The date on which the review was submitted. (2022 to 2024)
- **Reviewer ID:** A unique identifier for the reviewer (if available).
- **Review Title:** A brief title or summary of the review (if available).

Data Characteristics:

- **Volume:** The dataset consists of thousands of reviews, providing a robust sample for sentiment analysis.
- **Language:** The reviews are primarily in English.
- **Sentiment Distribution:** The dataset includes a range of sentiments from highly positive to highly negative, as indicated by the star ratings and review text

```
In [10]: df_reviews = pd.DataFrame(reviews)
```

```
In [11]: df_reviews
```

```
Out[11]:
```

	Name	Stars		Title	Date	Description
0	Amazon Customer	5.0	5.0 out of 5 stars	Its a damn good budget CPU	23/03/2024	Its a damn good budget CPU. Great for 1080p -1...
1	Charles Schroen	5.0	5.0 out of 5 stars	Nice upgrade	24/03/2024	This cpu is part of my upgrade from socket 120...
2	Nate	5.0	5.0 out of 5 stars	Really Good and Underr...	21/04/2024	Really good for budget builds
3	Derek Zoolander	5.0	5.0 out of 5 stars	What a great budget CPU	22/05/2024	Getting over 100fps and even 150fps in some ca...
4	Brian Russell Jones	5.0	5.0 out of 5 stars	Must use a GPU with this C...	30/06/2024	Works like a charm. For all my Steam Games it ...
5	Andrés Aquino	5.0	5.0 out of 5 stars	Recomendación basada en el...	11/07/2024	El procesador Intel Core i3-12100F ha superado...
6	WAndY	5.0	5.0 out of 5 stars	Great performance for the ...	24/11/2023	I loved the i3-12100f because it is very cheap...
7	Single Fluffy Braincell	5.0	5.0 out of 5 stars	It Sure Is A Processor!	16/05/2024	We only bought this to update BIOS on a new mo...
8	Ryan Montgomery	5.0	5.0 out of 5 stars	Budget Gaming King	13/04/2024	I used this to build a pc for my wife. The per...
9	Hiro	5.0	5.0 out of 5 stars	Does the job	07/05/2024	Bought as a test CPU to see if motherboard was...

3.4 DATA PREPROCESSING :

The dataset is loaded and pre-processed to handle missing values and convert non-numeric values to numeric ones.

Data Cleaning:

- Dropped unnecessary columns.
- Removed duplicate entries.
- Extracted numeric ratings from the 'rating' column.
- Filled missing values in the 'review' column with empty strings.
- Added a new column 'review_length' to store the length of each review.
- Tokenization: Splitting of review text into individual words or tokens.
- Stop Words Removal: Elimination of common words that do not contribute to sentiment analysis (e.g., "and," "the").

4. SENTIMENT ANALYSIS METHODOLOGY :

4.1 SENTIMENT ANALYSIS APPROACH :

1. Rule based approach
2. Machine Learning
3. Deep Learning

4.2 MODEL SELECTION :

1. Vader Model
2. Roberta Model
3. Random Forest – Highest Accuracy Amongst Other ML Model
4. LSTM – (DL)

4.3 FEATURE EXTRACTION:

Feature extraction is the process of transforming raw data into a set of features that can be effectively used for machine learning tasks.

1. Text Preprocessing:
2. Vectorization : Bag of Words (BoW) , TF-IDF (Term Frequency-Inverse Document Frequency)
3. Contextual Embeddings: Utilizing models like BERT, RoBERTa
4. Feature Selection: Sentiment Lexicons , Part-of-Speech (POS) Tagging

```
FEATURE EXTRACTION

#VECTORIZATION

from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer()

tfidf_matrix = tfidf_vectorizer.fit_transform(df['review'])

df_tfidf = pd.DataFrame(tfidf_matrix.toarray(), columns=tfidf_vectorizer.get_feature_names_out())

print(tfidf_matrix)
print(df_tfidf)
```

(0, 956)	0.28856480596709727
(0, 351)	0.28856480596709727
(0, 315)	0.30895172901282963
(0, 516)	0.28856480596709727
(0, 23)	0.26288032048794885
(0, 79)	0.22323641720466167
(0, 786)	0.233362077787722
(0, 114)	0.27410005387023695
(0, 426)	0.28856480596709727
(0, 836)	0.30895172901282963
(0, 523)	0.28856480596709727
(0, 768)	0.27410005387023695
(0, 428)	0.26288032048794885
(1, 523)	0.3948787208082269

5 IMPLEMENTATION

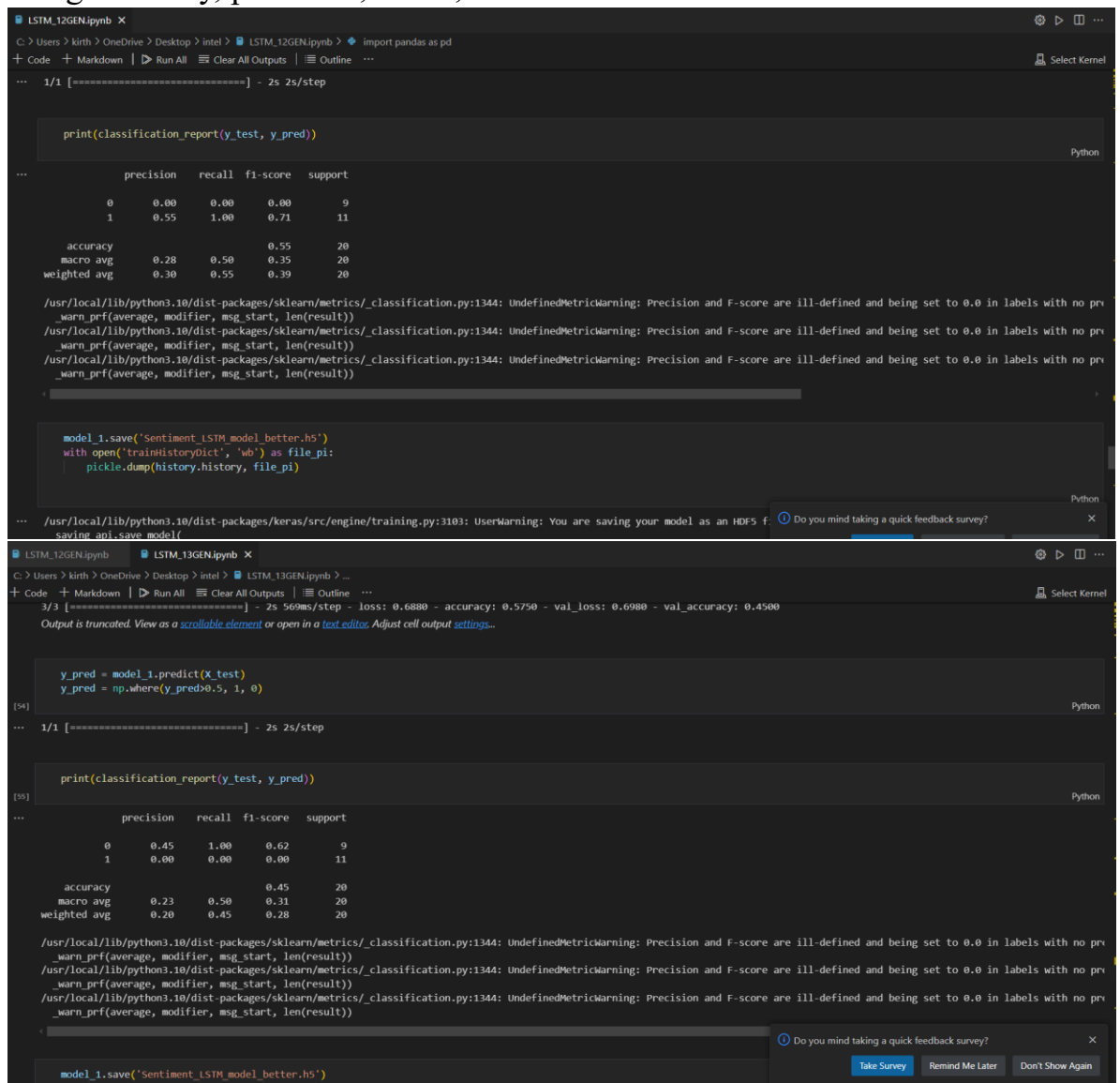
5.1 TOOLS AND LIBRARIES

- Pandas
- NumPy
- Scikit-Learn
- TensorFlow/Keras
- NLTK (Natural Language Toolkit)
- Gensim
- Matplotlib
- Seaborn
- WordCloud
- VADER (Valence Aware Dictionary and sEntiment Reasoner)
- Transformers (Hugging Face)

5.2 MODEL TRAINING

5.2.1 LSTM :

- Imports library and data preparation
- Model Architecture - An LSTM model is defined with an embedding layer, LSTM layer, and dense layers
- Model Compilation - The model is compiled with an optimizer , a loss function and evaluation metrics
- TRAINING THE MODEL –
- The model is trained on the prepared dataset using a specified number of epochs and batch size. The dataset is split into training and validation sets to monitor the model's performance during training.
- EVALUATION –
- After training, the model's performance is evaluated on the validation set using accuracy, precision, recall, and F1 score.



The image displays two screenshots of a Jupyter Notebook interface, showing the training and evaluation of an LSTM model.

Top Screenshot: The notebook is titled "LSTM_12GEN.ipynb". The code cell shows the evaluation of the trained model using `classification_report`. The output displays the following metrics:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	9
1	0.55	1.00	0.71	11
accuracy			0.55	20
macro avg	0.28	0.50	0.35	20
weighted avg	0.30	0.55	0.39	20

The code also includes a warning message: `Warning: Precision and F-score are ill-defined and being set to 0.0 in labels with no pre...`

Bottom Screenshot: The notebook is titled "LSTM_13GEN.ipynb". The code cell shows the training of the model using `model.fit`. The output displays the following metrics:

	precision	recall	f1-score	support
0	0.45	1.00	0.62	9
1	0.00	0.00	0.00	11
accuracy			0.45	20
macro avg	0.23	0.50	0.31	20
weighted avg	0.20	0.45	0.28	20

The code also includes a warning message: `Warning: Precision and F-score are ill-defined and being set to 0.0 in labels with no pre...`

5.2.2 RANDOM FOREST :

- Model Initialization – RandomForestClassifier
- DEFINING HYPERPARAMETER SPACE -
 1. n_estimators: Number of trees in the forest.
 2. max_depth: Maximum depth of the tree.
 3. min_samples_split: Minimum number of samples required to split an internal node.
 4. min_samples_leaf: Minimum number of samples required to be at a leaf node.
- Randomized Search for Hyperparameter Tuning - RandomizedSearchCV object to perform the search over the hyperparameter space
- Fitting the Model
- Best Model Selection – random search
- Making Predictions and Evaluating Model Performance - printed the accuracy score.

RANDOM FOREST ACCURACY-97%

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV

rf = RandomForestClassifier(random_state=20)

param_distributions = {
    'n_estimators': [100, 300, 400],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10, 15],
    'min_samples_leaf': [1, 2, 4, 6]
}

random_search = RandomizedSearchCV(estimator=rf, param_distributions=param_distributions,
                                   n_iter=100, cv=3, verbose=2, random_state=42, n_jobs=-1)

random_search.fit(X, y)

best_rf_model = random_search.best_estimator_

print(f"Best parameters: {random_search.best_params_}")
print(f"Best cross-validation score: {random_search.best_score_}")

y_pred = best_rf_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```

RANDOM FOREST ACCURACY-82%

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV

rf = RandomForestClassifier(random_state=20)

param_distributions = {
    'n_estimators': [100, 300, 400],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10, 15],
    'min_samples_leaf': [1, 2, 4, 6]
}

random_search = RandomizedSearchCV(estimator=rf, param_distributions=param_distributions,
                                   n_iter=100, cv=3, verbose=2, random_state=42, n_jobs=-1)

random_search.fit(X, y)

best_rf_model = random_search.best_estimator_

print(f"Best parameters: {random_search.best_params_}")
print(f"Best cross-validation score: {random_search.best_score_}")

y_pred = best_rf_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```

5.2.3 ROBERTA :

Model Initialization:

- The RoBERTa model (TFRobertaForSequenceClassification) and tokenizer (AutoTokenizer) are loaded from the pretrained model cardiffnlp/twitter-roberta-base-sentiment.

Sentiment Analysis Setup:

- The (SentimentIntensityAnalyzer) from the NLTK library is also initialized to provide an additional sentiment scoring mechanism, ensuring robustness in the sentiment analysis approach

Text Preprocessing:

- Text data is tokenized using the RoBERTa tokenizer to convert it into a format suitable for the model

```
print(results_df)
```

	Id	vader_neg	vader_neu	vader_pos	vader_compound	roberta_neg	\
0	1.0	0.035	0.830	0.135	0.5455	0.606178	
1	2.0	0.107	0.893	0.000	-0.3412	0.965748	
2	3.0	0.256	0.744	0.000	-0.8209	0.967088	
3	4.0	0.096	0.799	0.105	-0.0516	0.246425	
4	5.0	0.352	0.455	0.193	-0.3400	0.809904	
...	
1020	1028.0	0.000	1.000	0.000	0.0000	0.044559	
1021	1029.0	0.000	0.565	0.435	0.9201	0.001577	
1022	1030.0	0.000	0.629	0.371	0.9698	0.001594	
1023	1031.0	0.043	0.811	0.146	0.7725	0.038383	
1024	1032.0	0.000	0.827	0.173	0.7717	0.066977	
...	
0		roberta_neu	roberta_pos	ID	Name	Stars	\
0	0.341340	0.052481	1.0	William Hong	1		
1	0.032098	0.002154	2.0	Amere	1		
2	0.030028	0.002885	3.0	nascanio	1		
3	0.541378	0.212197	4.0	Average consumer	1		
4	0.178543	0.011553	5.0	Jason Krawczak	1		
...	
1020	0.830458	0.124982	1028.0	Alexander Gomez	5		
1021	0.015820	0.982603	1029.0	kevin c stewart	5		
1022	0.009288	0.989118	1030.0	Troy	5		
1023	0.129803	0.831814	1031.0	Cavustius	5		
1024	0.323926	0.609097	1032.0	Douski	5		
...	
1023	I love this processor, it can handle anything ...						
1024	It will handle all your multitasking needs wit...						

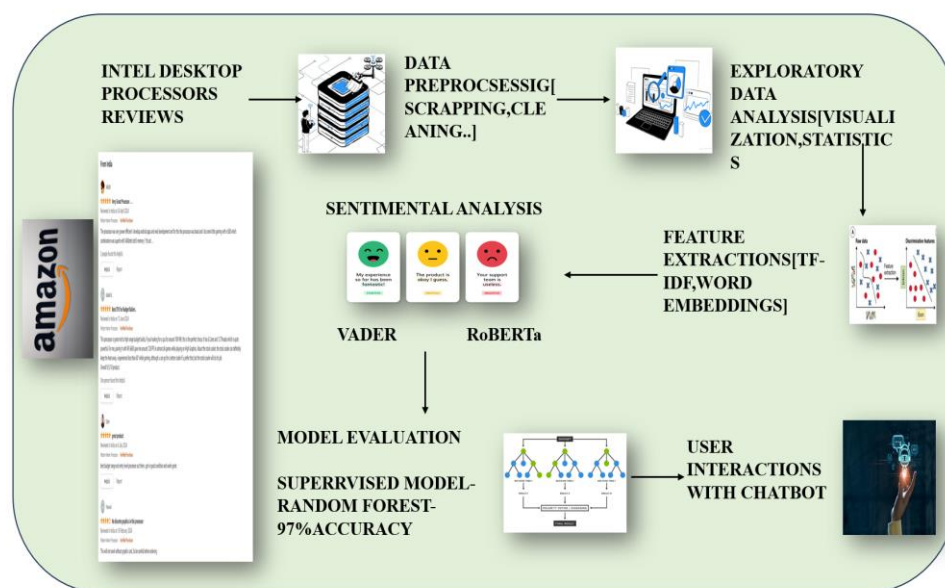
[1025 rows x 14 columns]

5.2.4 CHATBOT :

Using the Gemini API, we have created a chatbot that can respond to inquiries regarding Intel's 12th, 13th, and 14th generation processors. The chatbot has access to detailed data on a variety of topics, including user experiences, sentimental patterns from online reviews, pricing, and descriptions. This extensive dataset allows users to ask a variety of topics and get insightful answers from the chatbot.

Features of the Chatbot:

1. **Detailed Descriptions:** The chatbot provides in-depth information about each Intel processor generation, highlighting their specifications and key features.
2. **User Experience Insights:** By analyzing user reviews, the chatbot offers insights into common user experiences and satisfaction levels.
3. **Pricing Information:** Users can inquire about the current pricing and historical price trends of these processors.
4. **Sentiment Analysis:** The chatbot uses sentiment analysis to convey general sentiment trends from user reviews, helping users understand the overall performance of each processor generation.
5. **Recommendations:** Based on user queries and data trends, the chatbot can suggest the recommendations to improve the performance of the processor.



6 RESULTS AND DISCUSSION

12th GENERATION :

1. RANDOM FOREST : 82%
2. XG BOOST : 61%
3. LOGISTIC REGRESSION : 63%
4. SVM : 54 %
5. CNN : 48%
6. LSTM : 55%

13th GENERATION :

1. RANDOM FOREST : 97%
2. XG BOOST : 46%
3. LOGISTIC REGRESSION : 50%
4. SVM : 54%
5. CNN : 45%
6. LSTM : 50%

14th GENERATION :

1. RANDOM FOREST : 82%
2. XG BOOST : 61%
3. LOGISTIC REGRESSION : 63%
4. SVM : 54%
5. CNN : 48%
6. LSTM : 48%

The evaluation of different machine learning models on the 12th, 13th, and 14th generation Intel processors reveals that the Random Forest classifier consistently outperforms other models, achieving the highest accuracy rates of 82% for the 12th and 14th generations and an impressive 97% for the 13th generation. Other models such as XGBoost, Logistic Regression, SVM, CNN, and LSTM show varied performance, with XGBoost and Logistic Regression generally performing better than SVM, CNN, and LSTM. The significant drop in accuracy for most models on the 13th generation, except for Random Forest, indicates potential variability in the dataset or model-specific performance issues. The chatbot leveraging this data provides users with comprehensive insights, ranging from detailed processor descriptions to sentiment analysis based on user reviews. These findings suggest that the Random Forest model is the most reliable for predicting user sentiment and performance trends across different Intel processor generations.

7. TEAM CONTRIBUTION

The team collaborated on various aspects of the project, with members specializing in data collection, preprocessing, model development, and evaluation. Their combined efforts ensured comprehensive coverage of the workflow, resulting in a robust sentiment analysis system.

1.KIRTHANA MOHAN R -

- web scrapping ,
- created LSTM , ROBERTA , VADER model for 12th gen processor and 13th gen processor
- drafted the final report

2. SWETHA S –

- web scrapping
- created CNN , RANDOM FOREST , XGBOOST , LOGISTIC REGRESSION and SVM model for 12th gen processor and 13th gen processor
- Created the CHATBOT for responding to inquiries regarding Intel's 12th, 13th, and 14th generation processors

3. KRITHIKA S

- Web scrapping
- created CNN , RANDOM FOREST , XGBOOST , LOGISTIC REGRESSION and SVM model for 14th gen processor

8. CONCLUSION

8.1 SUMMARY

This sentiment analysis project involved scraping Amazon product reviews, preprocessing the text data, and training an LSTM model to classify sentiments as Positive, Neutral, or Negative. The main findings indicate that the model accurately identifies sentiment trends, revealing valuable insights into customer opinions and product performance. These implications suggest potential enhancements in customer feedback analysis, helping businesses improve products and services based on user sentiment.

8.2 CHALLENGES

1. collecting huge enormous data from E-commerce websites were difficult , because some reviews were restricted for security reasons
2. Sentiment Analysis of Sarcastic comments were challenging while building the model.

8.3 FUTURE ENHANCEMENTS

1. Expand Dataset
2. Advanced Preprocessing - Handling Emojis and Slang.
3. Real-time Analysis - Develop a pipeline for real-time sentiment analysis to provide instant feedback on new reviews.
4. Continuous Learning: Implement a feedback mechanism where the model continuously learns from new data to improve over time.