Restaurant Review Dataset Based Sentimental Analysis with Machine Learning Approach

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING WITH SPECIALIZATION

ARTIFICIAL INTELLIGENCE ANDMACHINE LEARNING

Submitted by:

M L K SUBRAHMANYAM - 21BCS8803

KOPPULA PRAKASH - 21BCS8824

G.SAI PRASANTH – 20BCS6832

T. BHAGEERATH – 20BCS6380

Under the Supervision of:

MR. SIDDHARTH KUMAR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

APEX INSTITUE OF TECHNOLOGY

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413,PUNJAB

Feb - May. 2023

BONAFIDE CERTIFICATE

Certified that this project report "**Restaurant Review Dataset Based Sentimental Analysis with Machine Learning Approach"** is the bonafide work of **M L K SUBRAHMANYAM, KOPPULA PRAKASH, G SAI PRASANTH, T. BHAGEERATH** who carried out the project work under my supervision.

SIGNATURE OF THE HOD
MR. AMAN KOUSHIK
(HEAD OF THE DEPARTMENT)
CSE - AIML

SIGNATURE OF THE SUPERVISOR
MR. SIDDHARTH KUMAR
(SUPERVISOR)
(Asst.Professor)
(AIT – CSE)

Submitted for the project viva-voce examination held on

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENTS

List of Figures:

ABSTRACT

Sentiment analysis of customer reviews has a crucial impact on a business's development strategy. Evolution of the internet in the past decade resulted in generation of voluminous data in all sectors. Due to these advents, the people have new ways of expressing their opinions about anything in the form of Google Reviews, Tweets, Blog Posts etc. Sentiment analysis deals with the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude toward a particular topic is positive, negative, or neutral. Knowing the opinion of customers is very important for any business. Hence, in this, we analyze the reviews given by the customers of the restaurant with help of machine learning classification algorithms. The Modern area of sentiment mining also called opinion mining. Researcher in the area of natural language processing (NLP), data mining, machine learning, Support Vector Machine (SVM) and test the method of sentiment analysis process. This problem can be addressed by an automated system called sentiment analysis and opinion mining that can analyze and extract the users view in the reviews.

Keywords— Sentiment Analysis, Category-Classification, Naïve Bayes Classifier, Logistic Regression, Support Vector Machine, Random Forest, Natural Language Processing (NLP), Restaurant Reviews Classification, and Machine Learning.

Acknowledgement

It gives us immense pleasure to express our deepest sense of gratitude and sincere thanks to our respected guide Mr. Siddharth Kumar (Assistant Professor), CSE-Artificial Intelligence, Chandigarh University, Mohali for his valuable guidance, encouragement and help for completing this work. Her useful suggestions for this whole work and cooperative behavior are sincerely acknowledged. We are also grateful to Dr. Shikha Gupta (Program Leader, CSE-AIML) for her constant support and guidance.

We also wish to express our indebtedness to our family members whose blessingsand support always helped us to face the challenges ahead. We also wish to express thanks to all people who helped us in completion of this project.

MARAM LEELA KRISHNA SUBRAHMANYAM 21BCS8803

KOPPULA PRAKASH 21BCS8824

G.SAI PRASANTH 20BCS6832

T. BHAGEERATH 20BCS6380

Chapter – 1

INTRODUCTION

The restaurant industry thrives on customer satisfaction and positive experiences. In the digital age, where online platforms provide a platform for customers to express their opinions freely, analyzing restaurant reviews has become crucial for understanding customer sentiments. By leveraging the power of machine learning, sentiment analysis can be performed on a large scale, allowing restaurant owners and managers to gain valuable insights into customer feedback.

This project focuses on developing a machine learning-based approach for sentiment analysis of restaurant reviews. The primary goal is to build a model that accurately classifies reviews as positive or negative, enabling restaurant owners and managers to understand customer sentiments at scale. By extracting sentiments from reviews, valuable insights can be gained regarding aspects such as food quality, service, ambiance, and overall experience.

To achieve this, a comprehensive dataset of restaurant reviews is collected from various online platforms. The dataset is carefully curated to include diverse reviews covering a wide range of sentiments. Each review is manually labeled with the corresponding sentiment, serving as the ground truth for training and evaluating the machine learning models.

The text data is preprocessed to remove noise and standardize the representation. Techniques such as punctuation removal, stop word elimination, tokenization, stemming, and lemmatization are applied to ensure a clean and consistent dataset. Feature extraction techniques such as Bag-of-Words, TF-IDF, and word embeddings are then employed to represent the text in a way that captures its semantic meaning.

A range of machine learning algorithms, including Support Vector Machines, Naive Bayes, and Recurrent Neural Networks, are utilized to build sentiment analysis models. These models are trained using a portion of the labeled dataset and evaluated using various performance metrics. Hyperparameter tuning and cross-validation techniques are applied to optimize the models and ensure their robustness and generalizability.

The results obtained from the models are analyzed, comparing their performance and discussing their strengths and weaknesses. Valuable insights are derived regarding the most influential factors impacting customer sentiments in restaurant reviews. These insights can help restaurant owners and managers make informed decisions to enhance their services, improve customer satisfaction, and ultimately drive business growth.

1.1 PROBLEM DEFINITION

The restaurant industry faces the challenge of understanding customer sentiments expressed in online reviews. Manual analysis of a large volume of reviews is time-consuming and impractical. Therefore, the problem addressed in this project is the development of a machine learning-based solution for sentiment analysis of restaurant reviews. The goal is to accurately classify reviews as positive or negative, enabling restaurant owners and managers to gain actionable insights and improve customer satisfaction.

1.2 PROJECT OVERVIEW

The major step involved in determining the sentiment of a text. In our approach, we have split the pre-processing part into three major steps. The first step involves removing the punctuation in the sentences. All special characters like exclamatory mark and quotes are removed by designing appropriate regular expression. The resultant data would be containing only alphabetical characters. The second step involves removing the stop-words from the reviews. Stop-words are the words which are not used to express any emotion or sentiment but used as connectors or articles in the English language. This includes words like and, with, of, the. Natural language processing (NLP) techniques like Lexical analysis, syntactic analysis, semantic analysis, disclosure integration, and pragmatic analysis are applied on the dataset to identify and remove stop-words. The semantic analysis step generally removes the stop-words like not as well. But, in opinion mining, the presence/absence of the word not plays an important role. For example, the review says the crust is not good. The removal of stop-words will result this sentence into crust good. Thus, a negative opinion is turned into positive. To avoid this problem, we have modified the semantic analysis step in NLP and made sure that such stop-words are not being removed in the process. The third step in pre-processing is to convert the original words to their root words. Root words are the words without prefix or suffix. For example, love is the root word for the words loving, loved, loves, etc. As we are interested only in actual opinion/sentiment rather than English grammar, such conversion eases the job. The Porter Stemmer algorithm is applied for converting all words in the dataset into root words.

1.3 TIMELINE

S.N	Strategies	1st week	2nd week	3rd week	4th week	5th week	6th week
1)	Problem Identification						
2)	Research & Analysis						
3)	Design						
4)	Coding						
5)	Implementation & testing						
6)	Project finalisation						
7)	Documentation						

1.4 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE SPECIFICATIONS

- Personal computer with keyboard and mouse maintained with uninterrupted powersupply.
- Processor: Intel® core™ i5
- Installed Memory (RAM): 8.00 GB

SOFTWARE SPECIFICATIONS

- Operating System: WINDOWS 7, 8.1,10,11
- Coding language: PYTHON
- Web Browser: GOOGLE CHROME
- Libraries used:
  - Sklearn
  - Matplotlib
  - Numpy
  - Pandas
  - NLTK
  - Seaborn
  - Joblib
  - Streamlit

Chapter – 2

LITERATURE SURVEY

2.1 Books related to Sentiment Analysis:

1. "Sentiment Analysis and Opinion Mining" by Bing Liu: This book provides a comprehensive overview of sentiment analysis techniques and approaches, including sentiment analysis in the context of restaurant reviews.

2. "Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More" by Matthew A. Russell: While not solely focused on restaurant reviews, this book covers various social media platforms and discusses techniques for sentiment analysis, which can be applied to restaurant reviews as well.

3. "Text Mining and Analysis: Practical Methods, Examples, and Case Studies Using SAS" by Goutam Chakraborty, Murali Pagolu, and Satish Garla: This book explores text mining techniques using SAS software, including sentiment analysis. It provides practical examples and case studies that can be helpful for analyzing restaurant reviews.

4. "Sentiment Analysis in Social Networks" by Federico Alberto Pozzi, Elisabetta Fersini, Enza Messina, and Bing Liu: Although it focuses on sentiment analysis in social networks, this book covers techniques and methodologies that are applicable to restaurant review analysis. It explores sentiment analysis algorithms and their applications in different domains.

5. "Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning" by Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda: This book provides practical examples and step-by-step guides for performing text analysis tasks, including sentiment analysis. It includes Python code examples and covers various techniques relevant to analyzing restaurant reviews.

6. "Sentiment Analysis and Opinion Mining in Python" by Siddhartha Chatterjee: This book provides practical examples and step-by-step guides to perform sentiment analysis using Python, with a focus on analyzing restaurant reviews.

7. "Deep Learning for Sentiment Analysis: From Theory to Practice" by Raghavendra Pappagari: This book explores deep learning techniques and their application in sentiment analysis, offering insights and practical tips for analyzing sentiments in restaurant reviews.

8. "Text Mining Cookbook: Practical Recipes to Help You Extract Value from Unstructured Data" by Prabhanjan Tattar, Krishna Birmiwal, and Suresh Kumar Mukhiya: This cookbook-style guide offers practical recipes for text mining tasks, including sentiment analysis, helping you leverage unstructured data such as restaurant reviews.

9. "Opinion Mining and Sentiment Analysis: A Lexicon-Based Approach" by Erik Cambria, Bing Liu, and Amir Hussain: This book focuses on lexicon-based approaches to sentiment analysis, discussing techniques for mining opinions and sentiments from textual data, including restaurant reviews.

10. "Machine Learning for Text Analysis" by Charu C. Aggarwal: This book provides an in-depth exploration of machine learning techniques for text analysis, covering topics such as sentiment analysis and text classification, with relevance to restaurant reviews.

2.2 RELATED WORK

In recent years, sentiment analysis has become a hot focus of research. This study gives a brief overview of sentiment analysis research done over the last few decades.

1. "Opinion Mining and Sentiment Analysis" by Bo Pang and Lillian Lee (2007) - This seminal paper provides an overview of the state of the art in sentiment analysis, including methods for handling subjective language and nuances in sentiment expression.

2. "Building a Sentiment Summarizer for Local Service Reviews" by Giuseppe Carenini, Raymond T. Ng, and Xiaodong Zhou (2008) - This paper focuses on developing a sentiment summarizer specifically for local service reviews, such as restaurant reviews. The authors use a combination of supervised and unsupervised techniques to achieve good performance.

3. Jindal and Liu (2009) "Sentiment Analysis and Opinion Mining" by Bing Liu - This book provides a comprehensive overview of sentiment analysis and opinion mining, including techniques for feature extraction, sentiment classification, and opinion summarization.

4. "Fine-Grained Sentiment Analysis with Long Short-Term Memory" by Lei Zhang, Shuai Wang, and Bing Liu (2010) - This paper proposes a novel approach to sentiment analysis using Long Short-Term Memory (LSTM) networks, which have

been shown to be effective at capturing temporal dependencies in text.

5. Ahmad et al. (2017) - "A Comparative Study of Sentiment Analysis Techniques on Restaurant Reviews": This comparative study evaluates the performance of different machine learning algorithms, including Naive Bayes, Support Vector Machines (SVM), and Random Forest, on a restaurant review dataset. It investigates the impact of various features and pre-processing techniques on sentiment classification accuracy.

6. Zhang et al. (2019) - "A Survey of Deep Learning for Sentiment Analysis": This survey paper provides an overview of deep learning techniques employed in sentiment analysis tasks. It discusses the application of deep neural networks, such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Long Short-Term Memory (LSTM), to capture contextual information and improve sentiment classification performance.

7. Akhtar et al. (2020) - "A Comparative Study of Sentiment Analysis Techniques for Restaurant Reviews Using Word Embeddings": This study explores the effectiveness of word embedding techniques, such as Word2Vec and GloVe, in sentiment analysis of restaurant reviews. It compares the performance of various machine learning algorithms, including SVM, Logistic Regression, and Multilayer Perceptron (MLP), when combined with word embeddings.

8. "Aspect-Based Sentiment Analysis of Restaurant Reviews: Identifying Aspects, Sentiment, and Opinion Targets" by Balamurugan Shanmugamani, Ravi Shankar, and Muthukumar Kalyanasundaram (2021) - This paper focuses on aspect-based sentiment analysis of restaurant reviews, which involves identifying the specific aspects of a restaurant that reviewers are commenting on and analysing the entiment expressed towards those aspects.

9. "A Deep Learning Framework for Sentiment Analysis in Restaurant Reviews" by Rahul Kumar, Ravi Kiran Sarvadevabhatla, and Vasudeva Varma (2021) - This paper proposes a deep learning framework for sentiment analysis of restaurant reviews, which combines Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to capture both local and global contextual information.

Basically, above-described literature survey focused on different solutions to provide better accuracy of proposed an improved sentimental analysis of restaurant reviews using machine learning. After analysis these papers collectively highlight the importance of sentiment analysis in the context of restaurant reviews. They discuss various techniques, including traditional machine learning algorithms and deep learning models, to improve sentiment classification accuracy. Additionally, aspects such as opinion spam detection and aspect-based sentiment analysis have also been explored in recent studies. These insights contribute to the development of more accurate and robust sentiment analysis systems for restaurant review datasets learning and effective generalization of the method.

2.3 EXISTING SYSTEM

The content of user generated opinions in the social media such as face book, twitter, review sites, etc are growing in large volume. These opinions can be tapped and used as business intelligence for various uses such as marketing, prediction, etc. Generally, sentiment analysis is used for finding out the aptitude of the author considering some topic. But in our social network sites not implemented Sentiment analysis. Some survey depends on the static sent word dataset to find the sentiment analysis. But we require finding a proper solution to find the polarity of the micro blogs.

2.2 PROPOSED SYSTEM

We will collect the unstructured data through the text box. With that data covert the data to lower case and data is processed as follow. Pre-processing Before the feature extractor can use the reviews to build feature vector, the review text goes through pre-processing step where the following steps are taken. These steps convert plain text of the review into process able elements with more information added that can be utilized by feature extractor. For all these steps, third-party tools were used that were specialized to handle unique nature of review text.

Step 1: Tokenization

Tokenization is the process of converting text as a string into processable elements called tokens. In the context of a review, these elements can be words, emoticons, url links, hashtags or punctuations “an insanely awsum….” Text was broken into “an”, “insanely”, “awsum”…. These elements are often separated by separated by a space. On the other hand, hash tags with”#” preceding the tag needs to be retained since a word as a hash tag may have different sentiment value than a word used regularly in the text.

Step 2: Parts of Speech Tags

Parts of Speech (POS) tags are characteristics of a word in a sentence based on grammatical categories of words of language. This information is essential for sentiment analysis as words may have different sentiment value depending on their POS tag. For example, word like "good" as a noun contains no sentiment whereas "good" as an adjective positive sentiment. each token extracted in the last step is assigned a POS

Step 3: Dependency Parsing

For our purposes, dependency parsing is extracting the relationship between words in a sentence. This can be useful in identifying relationship between "not" and "good" in phrases like "not really good" where the relationship is not always with the adjacent word.

CLASS	REVIEWS
positive	Had A Great Experience After so long. amazing Elite serving experience
negative	Service is very slow, Worst hospitality, Worst experience...
neutral	My first visit experience was great and over a period of time it changed to not bad. Reduction in quantity, timeliness also to an extent quality.

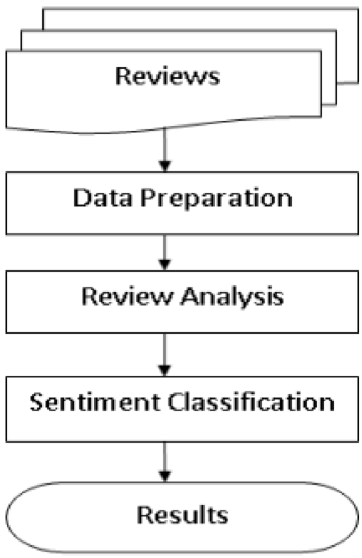


Fig1. Workflow process of proposed system

Chapter – 3

DESIGN FLOW & PROCESS

3.1 Problem statement:

Develop an accurate and efficient sentiment analysis system specifically tailored for restaurant reviews to enable restaurant owners to gain insights into customer sentiments, identify areas for improvement, and enhance the overall dining experience. The key challenges include handling domain-specific language, dealing with mixed sentiments, considering context, scaling for large review volumes, and overcoming data sparsity. By addressing these challenges, the system aims to provide timely and valuable insights to support data-driven decision-making in the restaurant industry.

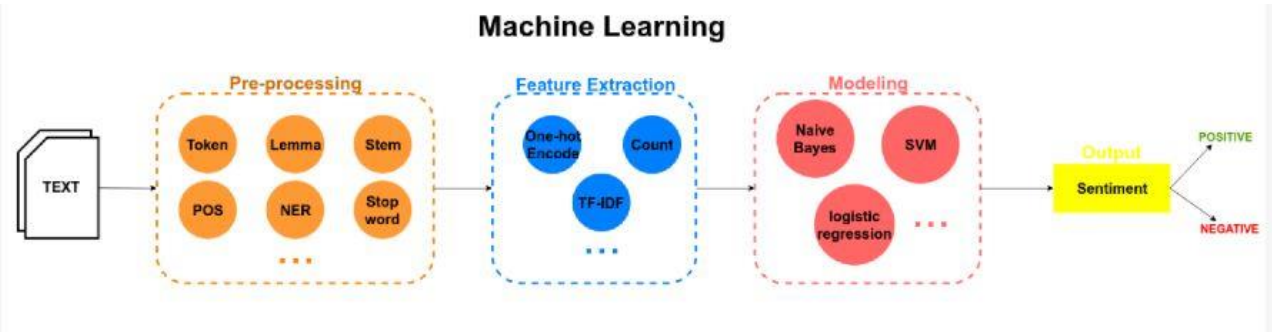


Fig2: Above, the implementation of sentiment analysis

3.2 Techniques and tools

a. Lexicon-based Approaches: Utilize sentiment lexicons or dictionaries to assign sentiment scores to words and aggregate them for sentiment analysis.
b. Machine Learning Algorithms: Apply supervised machine learning techniques like Naive Bayes, Support Vector Machines (SVM), or Random Forests for sentiment classification.
c. Aspect-Based Sentiment Analysis: Employ techniques such as rule-based methods, dependency parsing, or named entity recognition to identify and analyze

sentiment for specific aspects or features of a restaurant.

Text Preprocessing and Normalization:

a. Tokenization: Splitting text into individual words or tokens.
b. Stop word Removal: Eliminating common words (e.g., "a," "the," "is") that do not carry significant sentiment information.
c. Stemming and Lemmatization: Reducing words to their root forms to handle variations and improve feature extraction.
d. Noise Removal: Removing irrelevant characters, punctuation, or HTML tags from the text.

Feature Extraction:

a. Bag-of-Words (BoW): Representing text as a collection of word occurrences or frequencies.
b. n-grams: Capturing contextual information by considering sequences of words instead of individual words.
c. TF-IDF (Term Frequency-Inverse Document Frequency): Weighing the importance of words based on their frequency in the text corpus.

Aspect Extraction Techniques:

a. Rule-Based Methods: Defining patterns or rules to identify aspects based on specific keywords or linguistic patterns.
b. Dependency Parsing: Analyzing grammatical relationships between words to extract relevant aspects.
c. Named Entity Recognition (NER): Identifying named entities (e.g., food items, locations) that represent aspects.

Evaluation and Performance Metrics:

a. Accuracy: Measures the overall correctness of sentiment classification.
b. Precision, Recall, and F1 Score: Evaluate the performance of sentiment classification models, considering true positives, false positives, and false negatives.

c. Cross-Validation: Assessing the generalization capability of the sentiment analysis model by splitting the data into multiple folds.

NLP Libraries and Tools:

Natural Language Toolkit (NLTK): A Python library providing various NLP functionalities such as tokenization, stemming, and sentiment analysis.

3.3 Theory

What is Sentiment Analysis?

Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations. However, analysis of social media streams is usually restricted to just basic sentiment analysis and count based metrics. This is akin to just scratching the surface and missing out on those high value insights that are waiting to be discovered. So what should a brand do to capture that low hanging fruit?

With the recent advances in deep learning, the ability of algorithms to analyses text has improved considerably. Creative use of advanced artificial intelligence techniques can be an effective tool for doing in-depth research. We believe it is important to classify incoming customer conversation about a brand based on following lines:

Key aspects of a brand’s product and service that customers care about.
Users’ underlying intentions and reactions concerning those aspects.
These basic concepts when used in combination, become a very important tool for analyzing millions of brand conversations with human level accuracy.
Text Classifier — The basic building blocks

Sentiment Analysis

Sentiment Analysis is the most common text classification tool that analyses an incoming message and tells whether the underlying sentiment is positive, negative our neutral. You can input a sentence of your choice and gauge the underlying sentiment by playing with the demo here.

Intent Analysis

Intent analysis steps up the game by analyzing the user’s intention behind a message and identifying whether it relates an opinion, news, marketing, complaint, suggestion, appreciation or query.

Contextual Semantic Search (CSS)

Now this is where things get really interesting. To derive actionable insights, it is important to understand what aspect of the brand is a user discussing about. For example: Amazon would want to segregate messages that related to: late deliveries, billing issues, promotion related queries, product reviews etc. On the other hand, Starbucks would want to classify messages based on whether they relate to staff behavior, new coffee flavors, hygiene feedback, online orders, store name and location etc. But how can one do that?

We introduce an intelligent smart search algorithm called Contextual Semantic Search (a.k.a. CSS). The way CSS works is that it takes thousands of messages and a concept (like Price) as input and filters all the messages that closely match with the given concept. The graphic shown below demonstrates how CSS represents a major improvement over existing methods used by the industry.

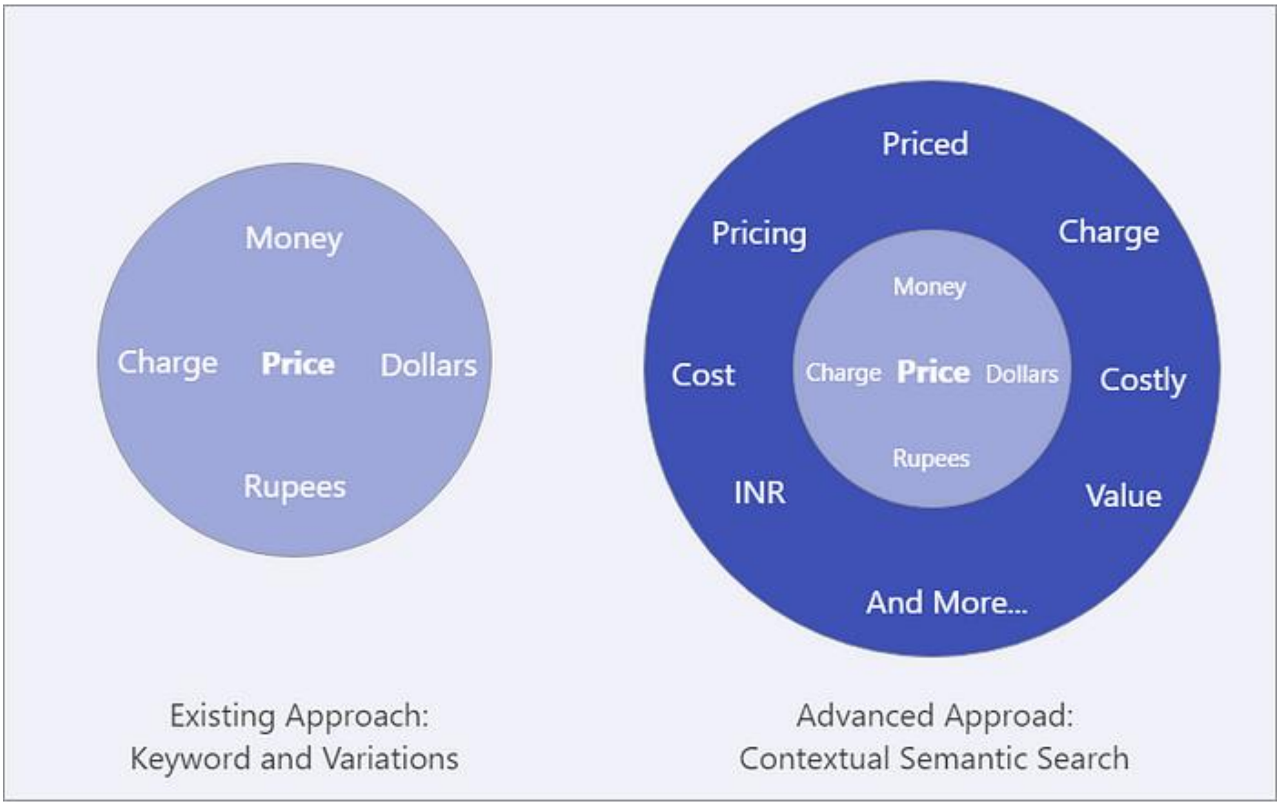


Fig3: Existing approach vs Contextual Semantic Search

A conventional approach for filtering all Price related messages is to do a keyword search on Price and other closely related words like (pricing, charge, $, paid). This method however is not very effective as it is almost impossible to think of all the relevant keywords and their variants that represent a particular concept. CSS on the other hand just takes the name of the concept (Price) as input and filters all the contextually similar even where the obvious variants of the concept keyword are not mentioned.

For the curious people, we would like to give a glimpse of how this works. An AI technique is used to convert every word into a specific point in the hyperspace and the distance between these points is used to identify messages where the context is similar to the concept we are exploring. A visualization of how this looks under the hood can be seen below:

Fig4: Visualizing contextually related Tweets

Why perform Sentiment Analysis?

Sentiment analysis is the contextual meaning of words that indicates the social sentiment of a brand and also helps the business to determine whether the product they are manufacturing is going to make a demand in the market or not.

According to the survey,80% of the world's data is unstructured. The data needs to be analyzed and be in a structured manner whether it is in the form of emails, texts, documents, articles, and many more.

Sentiment Analysis is required as it stores data in an efficient, cost-friendly.
Sentiment analysis solves real-time issues and can help you solve all real-time scenarios.

Types of Sentiment Analysis:

Fine-grained sentiment analysis: This depends on the polarity base. This category can be designed as very positive, positive, neutral, negative, or very negative. The rating is done on a scale of 1 to 5. If the rating is 5 then it is very positive, 2 then negative, and 3 then neutral.

Emotion detection: The sentiments happy, sad, angry, upset, jolly, pleasant, and so on come under emotion detection. It is also known as a lexicon method of sentiment analysis.

Aspect-based sentiment analysis: It focuses on a particular aspect for instance if a person wants to check the feature of the cell phone then it checks the aspect such as the battery, screen, and camera quality then aspect based is used.

Multilingual sentiment analysis: Multilingual consists of different languages where the classification needs to be done as positive, negative, and neutral. This is highly challenging and comparatively difficult.

How does Sentiment Analysis work?

There are three approaches used:

Rule-based approach: Over here, the lexicon method, tokenization, and parsing come in the rule-based. The approach is that counts the number of positive and negative words in the given dataset. If the number of positive words is greater than the number of negative words then the sentiment is positive else vice-versa.

Machine Learning Approach: This approach works on the machine learning technique. Firstly, the datasets are trained and predictive analysis is done. The next process is the extraction of words from the text is done. This text extraction can be done using different techniques such as Naive Bayes, Support Vector machines, hidden Markov model, and conditional random fields like this machine learning techniques are used.

Neural network Approach: In the last few years neural networks have evolved at a very rate. It involves using artificial neural networks, which are inspired by the structure of the human brain, to classify text into positive, negative, or neutral sentiments. it has Recurrent neural networks, Long short-term memory, Gated recurrent unit, etc to process sequential data like text.

Hybrid Approach: It is the combination of two or more approaches i.e. rule-based and Machine Learning approaches. The surplus is that the accuracy is high compared to the other two approaches.

Applications:

Sentiment Analysis has a wide range of applications as:

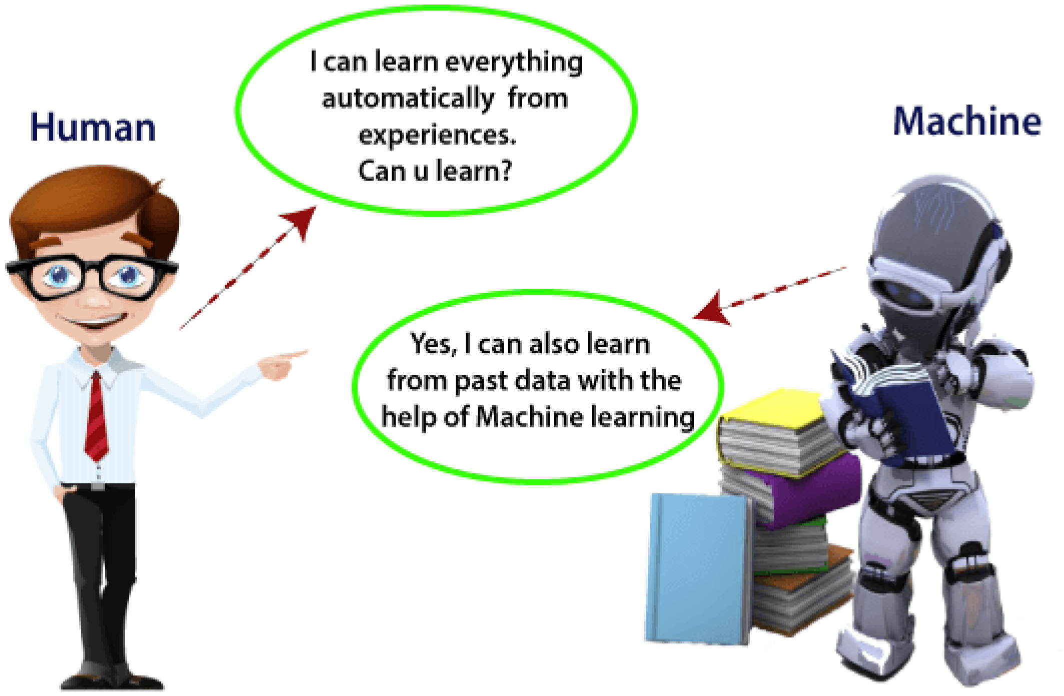
Social Media: If for instance the comments on social media side as Instagram, over here all the reviews are analyzed and categorized as positive, negative, and neutral.
Customer Service: In the play store, all the comments in the form of 1 to 5 are done with the help of sentiment analysis approaches.
Marketing Sector: In the marketing area where a particular product needs to be reviewed as good or bad.
Reviewer side: All the reviewers will have a look at the comments and will check and give the overall review of the product.

Challenges of Sentiment Analysis
There are major challenges in the sentiment analysis approach:
If the data is in the form of a tone, then it becomes really difficult to detect whether the comment is pessimist or optimistic.
If the data is in the form of emoji, then you need to detect whether it is good or bad.
Even the ironic, sarcastic, comparing comments detection is really hard.
Comparing a neutral statement is a big task.

What is Machine Learning?

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by

Arthur Samuel in 1959.



Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

How does Machine Learning work:

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:

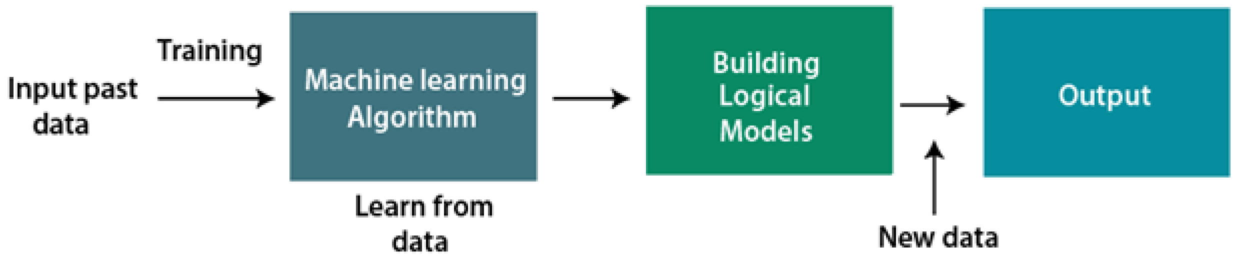


Fig5: Above, the machine learning process

Features of Machine Learning:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

What is Natural Language Processing?

Natural Language Processing, or NLP for short, is broadly defined as the automatic manipulation of natural language, like speech and text, by software.
The study of natural language processing has been around for more than 50 years and grew out of the field of linguistics with the rise of computers.

Natural Language Processing (NLP) refers to AI method of communicating with an intelligent system using a natural language such as English.

Processing of Natural Language is required when you want an intelligent system like robot to perform as per your instructions, when you want to hear decision from a dialogue based clinical expert system, etc.

The field of NLP involves making computers to perform useful tasks with the natural language's humans use. The input and output of an NLP system can be −

- Speech
- Written Text

Components of NLP:

There are two components of NLP as given −

Natural Language Understanding (NLU)
Understanding involves the following tasks −
Mapping the given input in natural language into useful representations.
Analyzing different aspects of the language.
Natural Language Generation (NLG)
It is the process of producing meaningful phrases and sentences in the form of natural language from some internal representation.
It involves −
Text planning − It includes retrieving the relevant content from knowledge base.
Sentence planning − It includes choosing required words, forming meaningful phrases, setting tone of the sentence.
Text Realization − It is mapping sentence plan into sentence structure.
The NLU is harder than NLG.

Difficulties in NLU:

- NL has an extremely rich form and structure.
- It is very ambiguous. There can be different levels of ambiguity –

Lexical ambiguity − It is at very primitive level such as word-level.
For example, treating the word "board" as noun or verb?
Syntax Level ambiguity − A sentence can be parsed in different ways.
For example, "He lifted the beetle with red cap." − Did he use cap to lift the beetle or he lifted a beetle that had red cap?
Referential ambiguity − Referring to something using pronouns. For example, Rima went to Gauri. She said, "I am tired." − Exactly who is tired?
One input can mean different meanings.
Many inputs can mean the same thing.

NLP Terminology

- Phonology − It is study of organizing sound systematically.
- Morphology − It is a study of construction of words from primitive meaningful units.
- Morpheme − It is primitive unit of meaning in a language.
- Syntax − It refers to arranging words to make a sentence. It also involves determining the structural role of words in the sentence and in phrases.
- Semantics − It is concerned with the meaning of words and how to combine words into meaningful phrases and sentences.
- Pragmatics − It deals with using and understanding sentences in different situations and how the interpretation of the sentence is affected.
- Discourse − It deals with how the immediately preceding sentence can affect the interpretation of the next sentence.
- World Knowledge − It includes the general knowledge about the world.

Steps in NLP

There are general five steps −

Lexical Analysis − It involves identifying and analyzing the structure of words. Lexicon of a language means the collection of words and phrases in a language. Lexical analysis is dividing the whole chunk of txt into paragraphs, sentences, and words.

Syntactic Analysis (Parsing) − It involves analysis of words in the sentence for grammar and arranging words in a manner that shows the relationship among the words. The sentence such as “The school goes to boy” is rejected by English syntactic analyzer.

NLP Steps

- Semantic Analysis − It draws the exact meaning or the dictionary meaning from the text. The text is checked for meaningfulness. It is done by mapping syntactic structures and objects in the task domain. The semantic analyzer disregards sentence such as "hot ice-cream".

- Discourse Integration − The meaning of any sentence depends upon the meaning of the sentence just before it. In addition, it also brings about the meaning of immediately succeeding sentence.

- Pragmatic Analysis − During this, what was said is re-interpreted on what it actually meant. It involves deriving those aspects of language which require real world knowledge.

What is Data Preprocessing?

Data preprocessing refers to the set of techniques and operations applied to raw data before it is used for analysis or modeling. It involves transforming, cleaning, and organizing the data to ensure its quality, consistency, and suitability for further processing. Data preprocessing plays a crucial role in data analysis and machine learning tasks, as it helps improve the accuracy, reliability, and efficiency of subsequent data processing steps.

The main steps involved in data preprocessing are as follows:

Data Cleaning:

Handling missing data: Dealing with missing values by either imputing them or removing rows or columns with missing data.

Removing duplicates: Identifying and eliminating duplicate records or instances from the dataset.

Handling outliers: Detecting and handling outliers or anomalies that may significantly affect the analysis or modeling results.

Data Transformation:

Feature scaling: Normalizing or standardizing numeric features to bring them to a similar scale and prevent biases in certain algorithms.

Feature encoding: Converting categorical variables into numerical representations that machine learning algorithms can process.
Feature discretization: Grouping continuous variables into discrete bins or intervals to simplify the data representation.
Feature engineering: Creating new features or transforming existing features to better represent the underlying patterns or relationships in the data.
Data Integration:
Combining data from multiple sources or different datasets into a unified format for analysis or modeling.
Resolving data inconsistencies, such as conflicting attribute names or data formats, during the integration process.

Data Reduction:
Dimensionality reduction: Reducing the number of features or variables while preserving the most important information, typically achieved through techniques like Principal Component Analysis (PCA) or feature selection algorithms.
Instance sampling: Selecting a representative subset of instances or records from a large dataset to reduce computational complexity or balance class distributions.

Data Formatting:
Ensuring the data is in the appropriate format and structure for analysis or modeling tasks.
Handling date and time formats, converting text to lowercase or uppercase, or adjusting data representations to match specific requirements.

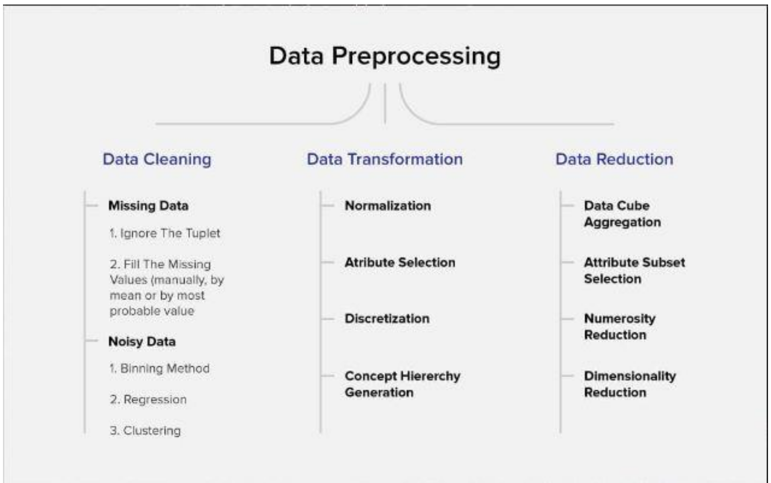


Fig6. Steps of Data Pre-processing

Tokenization:

Tokenization is the process of breaking down a text or document into smaller units called tokens. In the context of natural language processing (NLP), tokens typically represent words, but they can also be characters, subwords, or other meaningful units of text. Tokenization is an essential step in many NLP tasks as it helps to analyze and process text data more effectively.

The process of tokenization involves the following steps:

Text Segmentation: The text is divided into individual segments or chunks. The size and definition of these segments depend on the specific tokenization method used.

Word Tokenization: The most common form of tokenization is word tokenization, where the text is divided into words or word-like units. Each word in the text is treated as a separate token.

Token Boundaries: The tokenization process identifies boundaries between tokens based on certain rules or patterns. Common tokenization rules include splitting text at whitespace, punctuation marks, or special characters.

Handling Special Cases: Tokenization may involve handling special cases like contractions, hyphenated words, abbreviations, or compound words. For example, "can't" may be tokenized as "can" and "'t," while "New York" may be tokenized as "New" and "York."

Tokenization is a fundamental step in various NLP applications, including text classification, sentiment analysis, named entity recognition, part-of-speech tagging, and machine translation. It helps to convert unstructured text data into structured and manageable units, enabling subsequent analysis or processing tasks to operate on a more granular level.

Tokenization is typically performed using pre-built tokenization libraries or tools available in programming languages such as Python. Libraries like NLTK (Natural Language Toolkit), spaCy, or the tokenization functionalities provided by deep learning frameworks like TensorFlow or PyTorch offer convenient and efficient methods for tokenizing text data.

POS Tagging:

POS tagging, or Part-of-Speech tagging, is the process of assigning grammatical tags to words in a text corpus based on their roles and relationships within a sentence. POS tags represent the syntactic category or part of speech of each word, such as noun, verb, adjective, adverb, pronoun, preposition, conjunction, or interjection.

POS tagging is an essential task in natural language processing (NLP) as it helps in understanding the grammatical structure of sentences, disambiguating word meanings, and providing contextual information for downstream NLP tasks.

The process of POS tagging typically involves the following steps:

Tokenization: The text is first tokenized into individual words or tokens.

Lexical Lookup: Each token is looked up in a pre-defined dictionary or lexicon that contains words and their associated POS tags. The lexicon may also include additional information such as word forms, lemma, or frequency.

Contextual Disambiguation: In cases where a word has multiple possible POS tags (ambiguity), the context of the sentence is considered to determine the most appropriate tag. This is often done using statistical models, rule-based approaches, or machine learning algorithms.

Tagging the Tokens: POS tags are assigned to each token, indicating the part of speech it represents in the sentence. For example, "dog" may be tagged as a noun (NN), "run" as a verb (VB), or "beautiful" as an adjective (JJ).

Common POS tagsets include the Penn Treebank tagset, Universal POS tagset, or Brown Corpus tagset, among others. Each tagset has its own set of tag labels and conventions for representing different parts of speech.

POS tagging is often performed using pre-trained models and libraries in popular programming languages such as Python. Libraries like NLTK, spaCy, or CoreNLP provide POS tagging functionalities, allowing developers and researchers to perform accurate and efficient POS tagging on their text data.

POS tagging serves as a crucial preprocessing step in various NLP tasks, including text parsing, information extraction, sentiment analysis, machine translation, and text-to-speech synthesis. It enhances the understanding and analysis of textual data by providing insights into the grammatical structure and syntactic relationships within sentences.

Dependency Parsing:

Dependency parsing is a natural language processing (NLP) technique that analyzes the grammatical structure and syntactic relationships between words in a sentence. It aims to determine the hierarchical dependencies between words and represents them as a directed graph called a dependency tree.

In dependency parsing, each word in the sentence is considered a node in the tree, and the dependencies between the words are represented as labeled edges connecting the nodes. The dependencies typically indicate the syntactic relationships, such as subject-verb, verb-object, modifier-noun, or conjunction relationships.

The process of dependency parsing involves the following steps:

Tokenization: The input sentence is tokenized into individual words or tokens.

Part-of-Speech (POS) Tagging: Each token is assigned a grammatical tag indicating its part of speech (noun, verb, adjective, etc.) using techniques like POS tagging.

Dependency Parsing Algorithm: A dependency parsing algorithm is applied to analyze the relationships between words and construct the dependency tree. Commonly used algorithms include transition-based approaches (e.g., Arc-Standard, Arc-Eager) or graph-based approaches (e.g., Minimum Spanning Tree, Maximum spanning Tree).

Dependency Labeling: The edges in the dependency tree are assigned labels that represent the syntactic relationship between the connected words. For example, labels like "nsubj" (nominal subject), "dobj" (direct object), or "amod" (adjectival modifier) indicate different types of dependencies.

Dependency parsing provides valuable insights into the syntactic structure of sentences and is used in various NLP applications, including information extraction, question answering, sentiment analysis, machine translation, and text summarization. It helps in understanding the relationships between words and identifying the main components and modifiers in a sentence.

There are several libraries and tools available for performing dependency parsing, such as Stanford CoreNLP, spaCy, NLTK, or the Universal Dependencies project. These tools often provide pre-trained models and APIs that allow developers and researchers to perform accurate and efficient dependency parsing on their text data.

Sentiment Analysis:

Sentiment analysis, also known as opinion mining, is a natural language processing (NLP) technique that aims to determine the sentiment or subjective information expressed in text. It involves analyzing and classifying the emotions, attitudes, opinions, or sentiments conveyed by individuals in their written expressions, such as reviews, social media posts, customer feedback, or survey responses.

The primary goal of sentiment analysis is to automatically identify and extract sentiment polarity (positive, negative, or neutral) from text data, allowing for a quantitative understanding of people's opinions or sentiments towards specific topics, products, services, or events.

The process of sentiment analysis typically involves the following steps:

Text Preprocessing: The text data is cleaned, normalized, and preprocessed to remove noise, irrelevant information, or special characters. This step often includes processes like tokenization, stopword removal, and stemming or lemmatization.

Sentiment Lexicon Creation: A sentiment lexicon or dictionary is compiled, containing a list of words or phrases along with their associated sentiment scores or labels. These scores indicate the sentiment polarity of each word, such as positive, negative, or neutral. The lexicon may also include additional information, such as intensifiers or negation words.

Sentiment Classification:
a. Rule-based Approaches: These approaches use predefined rules or patterns to match words or phrases from the text data to the sentiment lexicon and assign sentiment labels based on the matches and their context.
b. Machine Learning Algorithms: Supervised machine learning algorithms, such as Naive Bayes, Support Vector Machines (SVM), or Recurrent Neural Networks (RNNs), are trained on labeled sentiment data to classify the sentiment of new, unseen text based on learned patterns and features.

Sentiment Aggregation: The sentiment scores or labels assigned to individual words or phrases are aggregated to determine an overall sentiment score for the entire text. Aggregation methods can include simple summation, weighted averages, or more complex techniques like sentiment propagation in dependency trees.

Evaluation and Validation: The performance of the sentiment analysis model is evaluated using metrics such as accuracy, precision, recall, F1 score, or confusion matrix. Validation techniques like cross-validation or holdout evaluation are commonly used to ensure the model's generalization capability.

Sentiment analysis finds applications in various domains, including customer feedback analysis, social media monitoring, brand reputation management, market research, and personalized recommendation systems. It provides valuable insights into public opinion, customer sentiment, and user feedback, enabling businesses and organizations to make data-driven decisions, understand customer needs, and enhance user experiences.

Classification:

Classification is a machine learning task that involves assigning predefined categories or labels to input data based on their features or characteristics. It is a supervised learning technique where the model learns from labeled training data to make predictions or classify new, unseen instances into predefined classes.

The process of classification typically involves the following steps:

Data Preparation: The input data is prepared in a suitable format, often represented as feature vectors. This may involve data cleaning, normalization, feature extraction, or feature engineering techniques.

Training Data Creation: A labeled training dataset is created, consisting of instances with known class labels. The dataset is divided into input features (independent variables) and corresponding class labels (dependent variable).

Model Selection: A classification algorithm or model is selected based on the problem domain, available data, and desired performance metrics. Commonly used algorithms include Decision Trees, Random Forests, Support Vector Machines (SVM), Naive Bayes, Logistic Regression, or Neural Networks.

Model Training: The selected classification model is trained using the labeled training data. The model learns the patterns and relationships between input features and their corresponding class labels, aiming to minimize prediction errors.

Model Evaluation: The trained model is evaluated using evaluation metrics such as accuracy, precision, recall, F1 score, or area under the receiver operating characteristic curve (AUC-ROC). Evaluation helps assess the model's performance and generalization capability on unseen data.

Model Optimization and Tuning: The model's hyperparameters and settings are fine-tuned to improve its performance. Techniques such as cross-validation, grid search, or random search can be applied to find the optimal combination of hyperparameters.

Prediction or Inference: Once the classification model is trained and evaluated, it can be used to make predictions or classify new, unseen instances into the predefined classes. The model applies the learned patterns to the input features and assigns the most probable class label to each instance.

Classification is widely used in various applications, including image recognition, text classification, spam filtering, sentiment analysis, customer segmentation, fraud detection, medical diagnosis, and many more. It enables automated decision-making and pattern recognition, allowing systems to classify and organize data efficiently based on their inherent characteristics or properties.

Naïve Bayes:

Naive Bayes is a simple yet powerful classification algorithm based on Bayes' theorem of probability. It is a supervised learning algorithm that is commonly used for text classification and spam filtering tasks. Despite its simplicity, Naive Bayes often performs well in practice and is computationally efficient.

The underlying principle of Naive Bayes is that it assumes independence between the features (or attributes) given the class. This assumption is referred to as "naive" because it simplifies the computation by considering each feature independently, disregarding any potential correlations between them.

Naive Bayes is known for its simplicity, fast training speed, and ability to handle large feature spaces. However, the independence assumption can be a limitation when there are strong dependencies among the features. Nevertheless, Naive Bayes is widely used in various applications such as text classification, email filtering, sentiment analysis, and recommendation systems.

Naive Bayes' Rule: When classifying a new instance, Naive Bayes applies Bayes' theorem to calculate the probability of each class given the feature values. The class with the highest probability is chosen as the predicted class label.

Bayes' Theorem: Bayes' theorem calculates the posterior probability of a class given the feature values using the prior probability of the class and the likelihood of the features given the class. It can be expressed as:

P(class|features) = (P(class) * P(features|class)) / P(features)

P(class|features) represents the posterior probability of the class given the features.

P(class) is the prior probability of the class.

P(features|class) is the likelihood of the features given the class.

P(features) is the probability of the features (constant across classes), which can be calculated using the law of total probability.

The algorithm calculates the posterior probability for each class and assigns the class with the highest probability as the predicted class label for the new instance.

Naive Bayes

It is one of the popular classification techniques of algorithms used in data mining. It is a probability classifier. It links the attributes mutually & is dependent on the number of parameters, The principle here is that the variables provided are independent. It generates accurate results with appropriate calculation & provides fast results. It is on Bayes theorem & the formula is,

$$P\,(\text{label} \mid \text{features}) = \frac{P(\text{label}) * P\,(\text{features} \mid \text{label})}{P(\text{features})}$$

SVM:

Support Vector Machines (SVM) is a popular supervised machine learning algorithm used for classification and regression tasks. It is known for its ability to handle high-dimensional feature spaces and handle both linearly separable and non-linearly separable data.

The main idea behind SVM is to find an optimal hyperplane that separates the data points belonging to different classes with the largest possible margin. The hyperplane is chosen such that it maximizes the distance between the nearest data points of different classes, which are called support vectors.

Data Preparation: The input data is typically represented as a feature matrix, where each row represents an instance, and each column represents a feature. The features can be continuous or discrete.

Feature Scaling: It is common practice to scale or normalize the feature values to ensure they have a similar range. This helps in preventing any particular feature from dominating the learning process.

Training: During the training phase, SVM learns a hyperplane that separates the data points of different classes with the largest margin. SVM aims to find the optimal hyperplane by solving a quadratic optimization problem.

Hyperplane Selection: SVM can use different types of hyperplanes depending on the data and problem at hand:

Linear SVM: In linear SVM, a linear hyperplane is used to separate the data points.
Non-linear SVM: For data that is not linearly separable, SVM can use kernel functions (e.g., polynomial, radial basis function) to map the data into a higher-dimensional

space, where a linear hyperplane can separate the transformed data.
Margin Maximization: SVM strives to maximize the margin between the hyperplane and the support vectors. The margin is the distance between the hyperplane and the closest data points from each class.

CountVectorizer:
CountVectorizer is a text feature extraction technique in natural language processing (NLP) that converts a collection of text documents into a matrix of token counts. It is a commonly used method to preprocess text data before applying machine learning algorithms, particularly for tasks such as text classification, clustering, and information retrieval.

The CountVectorizer operates by following these steps:

- Tokenization: The text documents are split into individual words or tokens. This process can involve removing punctuation, lowercasing, and applying other text preprocessing techniques.

- Vocabulary Construction: The CountVectorizer builds a vocabulary, which is essentially a dictionary that maps each unique token to an index. The vocabulary is created by collecting all the unique tokens from the corpus of documents.

- Counting Token Occurrences: For each document in the corpus, the CountVectorizer counts the number of occurrences of each token in the vocabulary. It creates a matrix where each row represents a document, and each column represents a token in the vocabulary. The value in each cell indicates the count of the corresponding token in the respective document.

- Transforming Text into a Matrix: The CountVectorizer transforms the text documents into a sparse matrix representation, where most of the entries are zero, as most tokens do not occur in most documents. This sparse matrix format is memory-efficient for large corpora.

- The resulting matrix, often referred to as the "document-term matrix," represents the text data in a numerical format that can be inputted into machine learning models. Each cell in the matrix represents the frequency or count of a token in a particular document.

CountVectorizer also provides additional functionalities, such as:

- Handling stop words: It can ignore common words, known as stop words, during the tokenization process to reduce noise in the resulting matrix.
- n-gram support: It can consider sequences of contiguous tokens of length n (unigrams, bigrams, trigrams, etc.) instead of individual tokens, capturing more contextual information.
- Vocabulary size control: It allows limiting the vocabulary size based on the most frequent or least frequent tokens.

Overall, CountVectorizer is a widely used technique for converting text data into a numerical representation suitable for machine learning algorithms, making it a fundamental step in many NLP pipelines.

StreamLit:

Streamlit is an open-source Python library that allows you to build interactive web applications for machine learning, data science, and data visualization tasks. It simplifies the process of creating and deploying web applications by providing a user-friendly interface and a simple API for building interactive components.

With Streamlit, you can transform your Python scripts or data analysis code into web applications with just a few lines of code. It enables you to create interactive dashboards, visualize data, and share your work with others without the need for extensive web development knowledge.

Key features of Streamlit include:

- Simple and Intuitive Syntax: Streamlit provides a straightforward API with easy-to-understand commands, making it accessible to both beginner and experienced Python developers.

- Fast Prototyping: You can quickly iterate and experiment with different ideas by reloading the web application automatically whenever changes are made to the code.

- Interactive Widgets: Streamlit offers a wide range of interactive widgets, such as sliders, dropdowns, checkboxes, and buttons, allowing users to interact with the application and modify parameters in real-time.

- Data Visualization: You can easily visualize data using popular Python libraries like Matplotlib, Plotly, and Altair. Streamlit provides convenient functions for generating charts, plots, and maps.

- Easy Deployment: Streamlit applications can be deployed on various platforms, including local machines, cloud servers, or even as shareable web URLs. It simplifies the deployment process, allowing you to share your applications with others effortlessly.

- Integration with Machine Learning Libraries: Streamlit seamlessly integrates with popular machine learning frameworks like TensorFlow, PyTorch, and scikit-learn, enabling you to showcase and deploy your machine learning models and experiments.

- Streamlit's main focus is on simplicity and ease of use, allowing data scientists and developers to build interactive web applications quickly. It has gained significant popularity in the Python community for its ability to streamline the process of creating web-based data-driven applications.

Classification: To classify new instances, SVM determines on which side of the hyperplane the instance falls. If it lies on the positive side, it is classified as one class, and if it lies on the negative side, it is classified as the other class.

SVM has various advantages, including its effectiveness in handling high-dimensional feature spaces, ability to handle non-linearly separable data through kernel functions, and resistance to overfitting. It is widely used in applications such as text categorization, image classification, bioinformatics, and finance.
However, SVMs can be sensitive to the choice of hyperparameters and the scaling of input features. Additionally, SVMs can be computationally expensive for large datasets. Proper parameter tuning and careful preprocessing are crucial for obtaining optimal results with SVMs.

3.4 METHODOLOGY

This proposed work is to predict the text automatically based on the data set values stored by using the r tool. By using the training data set values, it is possible to predict the text data using our classifier called naive bayes using algorithm.
The Fig I depicts the architecture of the proposed model used in the prediction of sentiment analysis. It consists of 3 steps

A. Data Collection
In this step data is taken out from Kaggle in a recognized format. Missing fields are evacuated in this process & thus the data is transformed. Sentiment Analysis can be considered a classification process. There are three main classification levels in sentiment analysis document-level, sentence-level, and aspect-level sentiment analysis. Level of document it aims to classify an opinion document which as a positive or negative opinion expression. It considers the full document as a basic information unit.
B. Data Pre-processing
The collected raw data of restaurant reviews consist of large number of attributes and there will be missing values. The reducing the attributes is required, extracting the attributes is also much essential. So, in order to meet importance of each variable or attributes "migrittr" algorithm is applied. Migrittr algorithm which selects the attributes based on predictor, here predictor considered restaurant review. Feature or Attribute extraction is done using migrittr algorithm. In detail steps working of migrittr algorithm. In Data cleaning once attributes are removed, filling the missing values, removing inconsistent data measuring the central tendency for the attribute such as mean median, quartile is done. In data pre-process the data is cleaned and the extracted data before analysis. Non-textual contents and contents that are irrelevant for the analysis are identified and eliminated.
C. Sentiment Analysis
The reviews sources are mainly review sites. Sentiment analysis is not only applied on product reviews but can also applied on stock market, news articles, or political debates. In political debates for example, we could figure out people's opinions on a certain election candidates or political parties. The election results can also be from political posts. The sites like social media and micro
blogging sites are taken a very good source of information because many people share and discuss their opinions about positive and negative opinion freely.

D. Classification

The lexicon-based approach is to finding the opinion mining which is used to analyze or to predict the text. There are two methods in this approach. The dictionary- based approach which depends on finding opinion seed words, and then searches the dictionary of their synonyms and antonyms. The corpus-based approach begins with a seed list of opinion words, and then finds other opinion words in a large corpus to help in finding opinion words with context specific orientations. This could be done by using statistical or semantic methods.

Data mining has got two most frequent modelling goals — classification & prediction. Classification model classifies discrete, unordered values or data. In this prediction process, the classification techniques utilized are, naive bayes classifier.

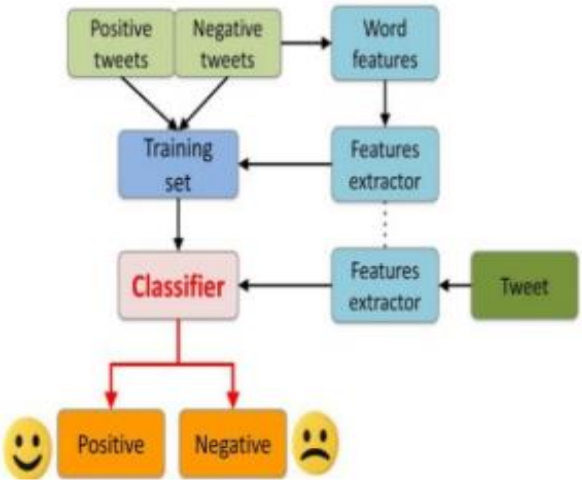


Fig4. Process of Sentiment analysis

Chapter – 4

IMPLEMENTATION SNAPSHOTS OF SOURCE CODE

Importing Libraries

```
[ ] from sklearn.feature_extraction.text import CountVectorizer
    from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_score
    import pandas as pd
```

```
df = pd.read_table('/content/gdrive/MyDrive/Verzeo/Restaurant_Reviews.tsv')#Copy the path of uploaded excel file from drive after mounting.
df
```

	Review	Liked
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday of...	1
4	The selection on the menu was great and so wer...	1
...	...	...
995	I think food should have flavor and texture an...	0
996	Appetite instantly gone.	0
997	Overall I was not impressed and would not go b...	0
998	The whole experience was underwhelming, and I ...	0
999	Then, as if I hadn't wasted enough of my life ...	0

1000 rows × 2 columns

Fig5. Importing the libraries and printing the values of dataset

Plotting the two classes of tweets

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.bar(df['Liked'],500, width=0.3, color=['green','red'])
plt.show()
```

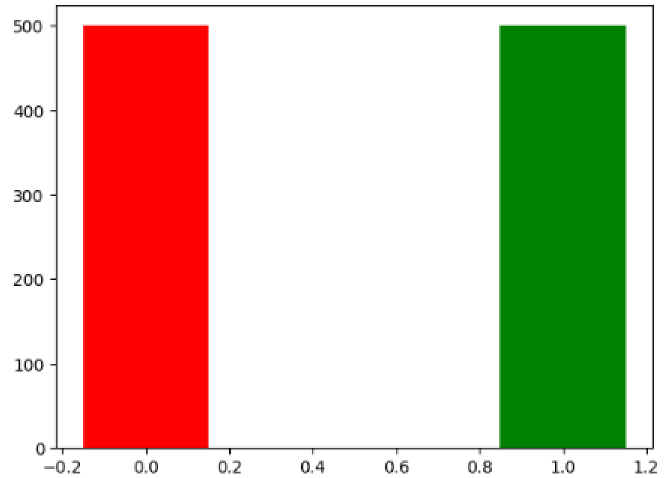


Fig6. Plotting the values of dataset

Splitting the dataset into train and test

```
[ ]  from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test = train_test_split(x,y,random_state=0, test_size=0.2)
```

```
[ ]  x_train.shape
```

```
(800,)
```

```
[ ]  x_test.shape
```

```
(200,)
```

```
[ ]  from sklearn.feature_extraction.text import CountVectorizer
     vect = CountVectorizer(stop_words = 'english')
     x_train_vect = vect.fit_transform(x_train)
     x_test_vect = vect.transform(x_test)
```

Fig7. Splitting the dataset into train and test

Model1 - (Support vector machine)(SVM)

```
[ ]  from sklearn.svm import SVC
     model1 = SVC()
```

```
[ ]  model1.fit(x_train_vect,y_train)
```

```
▾ SVC
SVC()
```

```
[ ]  y_pred1 = model1.predict(x_test_vect)
     y_pred1
```

```
array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0,
       0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       0, 0])
```

```
[ ]  y_test
```

```
array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1,
       1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1,
```

Fig8. Training the data using SVM model

Model1 Accuracy, Precision, Recall, F1 - Score

```
[ ] from sklearn.metrics import accuracy_score
    print('Accuracy:',accuracy_score(y_pred1,y_test))
    print('Precision: %.3f' % precision_score(y_test, y_pred1))
    print('Recall: %.3f' % recall_score(y_test, y_pred1))
    print('F1 Score: %.3f' % f1_score(y_test, y_pred1))

    Accuracy: 0.73
    Precision: 0.866
    Recall: 0.563
    F1 Score: 0.682
```

```
[ ] #to test the output
    test = vect.transform([df['Review'][412]])
    model1.predict(test)

    array([0])
```

Fig9. Metrics of SVM model

Model2 - combines two estimators (countvect+svc)

```
[ ] from sklearn.pipeline import make_pipeline
    model2 = make_pipeline(CountVectorizer(),SVC())

    model2.fit(x_train,y_train)

    y_pred2 = model2.predict(x_test)
    y_pred2

    array([0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1,
           1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0,
           0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1,
           1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0,
           1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0,
           0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1,
           0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1,
           0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,
           0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1,
           0, 1])
```

```
from sklearn.metrics import accuracy_score
print('Accuracy:',accuracy_score(y_pred2,y_test))
print('Precision: %.3f' % precision_score(y_test, y_pred2))
print('Recall: %.3f' % recall_score(y_test, y_pred2))
print('F1 Score: %.3f' % f1_score(y_test, y_pred2))

Accuracy: 0.79
Precision: 0.808
Recall: 0.777
F1 Score: 0.792
```

Fig10. Training the data using SVM pipeline model

Model3 - Using Naive Bayes

```
[ ] from sklearn.naive_bayes import MultinomialNB
    model3 =MultinomialNB()
```

```
[ ] model3.fit(x_train_vect,y_train)

    ▾ MultinomialNB
    MultinomialNB()
```

```
y_pred3 = model3.predict(x_test_vect)
y_pred3

array([0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0,
       1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0,
       0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0,
       1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0,
       0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1,
       0, 1])
```

```
[ ] y_test

    array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1,
           1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0,
           0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0,
           1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0,
           1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1,
```

Fig11. Training the data using MultinomialNB model

Model3 Accuracy, Precision, Recall, F1 - Score

```
print('Accuracy:',accuracy_score(y_pred3,y_test))
print('Precision: %.3f' % precision_score(y_test, y_pred3))
print('Recall: %.3f' % recall_score(y_test, y_pred3))
print('F1 Score: %.3f' % f1_score(y_test, y_pred3))

Accuracy: 0.745
Precision: 0.736
Recall: 0.786
F1 Score: 0.761
```

```
[ ] # to evaluate a statement and see if its spam or not using the method3
    test = vect.transform([df['Review'][412]])
    model3.predict(test)

    array([1])
```

Fig12. Metrics of Model3

Model4 - Using pipeline(countvect,multinomialNB)

```
from sklearn.pipeline import make_pipeline
model4 = make_pipeline(CountVectorizer(),MultinomialNB())
model4.fit(x_train,y_train)
y_pred4 = model4.predict(x_test)
y_pred4
```

```
array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0,
       1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0,
       0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1,
       0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       0, 1])
```

Model4 Accuracy, Precision, Recall, F1 - Score

```
[ ] print('Accuracy:',accuracy_score(y_pred4,y_test))
    print('Precision: %.3f' % precision_score(y_test, y_pred4))
    print('Recall: %.3f' % recall_score(y_test, y_pred4))
    print('F1 Score: %.3f' % f1_score(y_test, y_pred4))
```

```
Accuracy: 0.815
Precision: 0.837
Recall: 0.796
F1 Score: 0.816
```

Fig13. Training the data using Multinomial Pipeline model

Accuracy of differnt models after training:

- **ACCURACY FOR SVC - 73%**
- **SVC PIPELINE - 79%**
- **ACCURACY FOR MultinomialNB - 74.5%**
- **MultinomialNB PIPELINE - 81%**

Importing Joblib and dumping Model4

```
[ ] # joblib - persistance model  (used to save pipeline models)
    import joblib
    joblib.dump(model4,'Pos-Neg')
```

```
['Pos-Neg']
```

```
[ ] import joblib
    reload_model = joblib.load('Pos-Neg')
```

```
#predict using the reloaded joblib model
reload_model.predict(["Crust is not good."])
```

```
array([0])
```

Fig14. Importing the model using joblib

Installing the streamlit

```
[ ]
#install streamlit
!pip install streamlit --quiet
```

```
#STREAMLIT WEBAPP
%%writefile app.py
import streamlit as st  #webapp framework/library
import joblib

reload_model = joblib.load('Pos-Neg')  #loads the joblib model

st.title("Restaurant FeedBack")
st.title("Food Castle")
ip = st.text_input("Enter your feedback:") #asking the user input

op = reload_model.predict([ip])  #predict the output
if st.button('PREDICT'): #if button is clicked
  st.title(op[0]) #prints the output in single dimension
  if(op[0]==1):
    st.write("Your review was Positive. Thanks for giving good feedback.")
  else:
    st.text_input("Your review was Negative. Kindly tell what we can improve")
    st.write("Thanks for providing feedback.")
```

```
Overwriting app.py
```

Fig15. Installing streamlit library and loading the

Chapter – 5

RESULT ANALYSIS AND VALIDATION

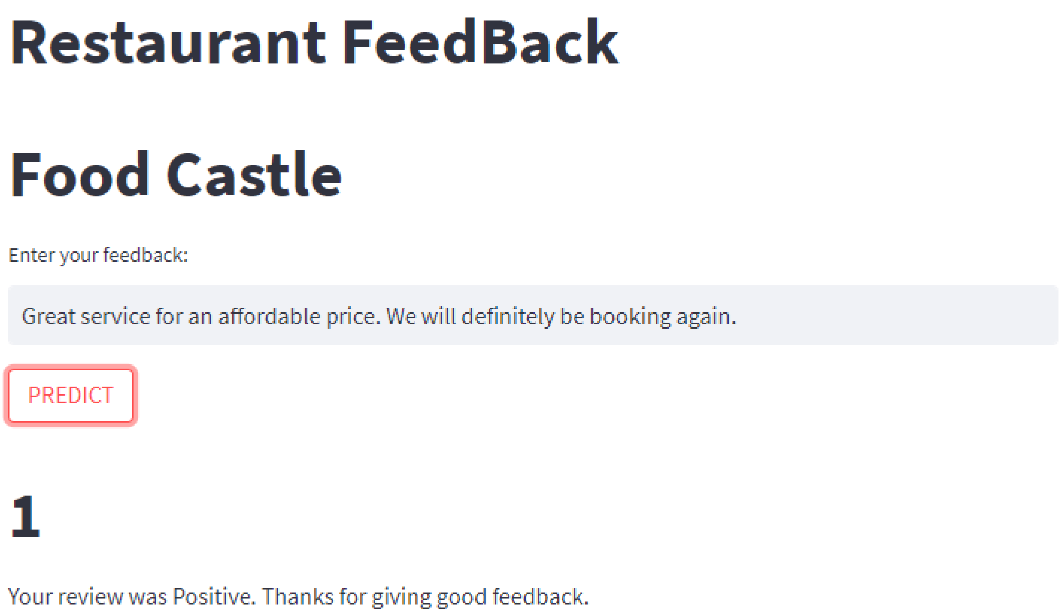
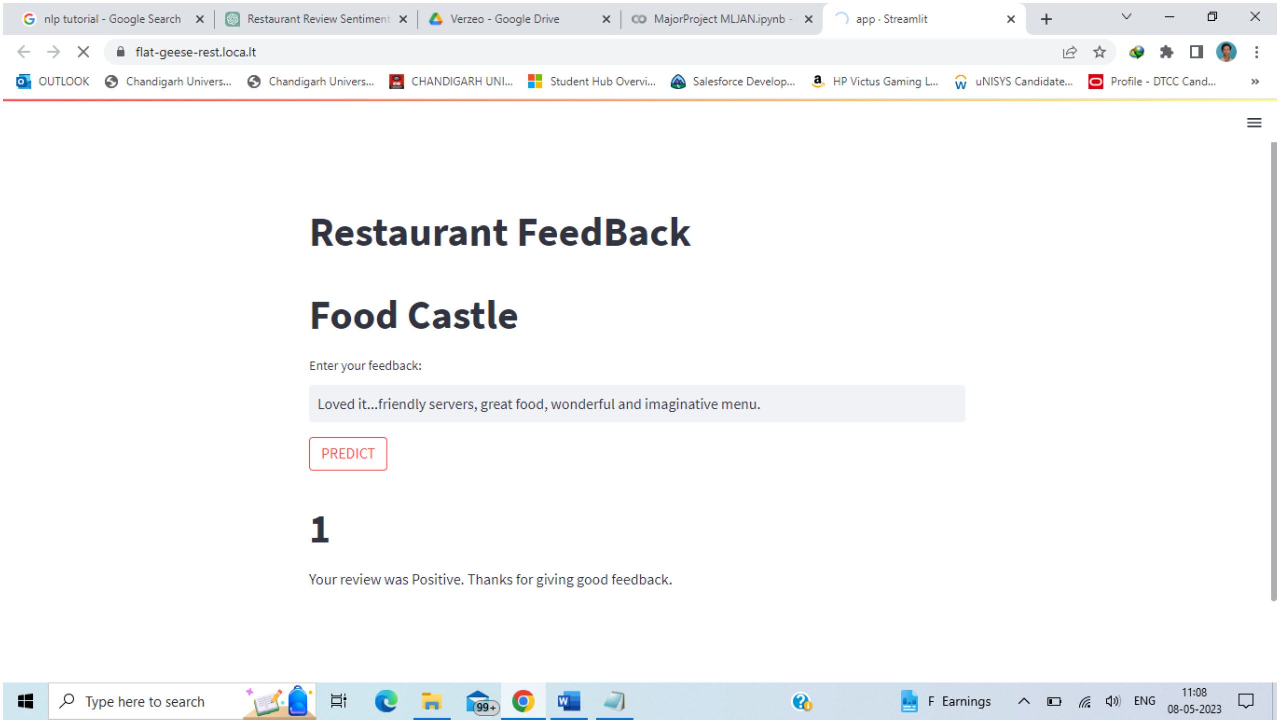
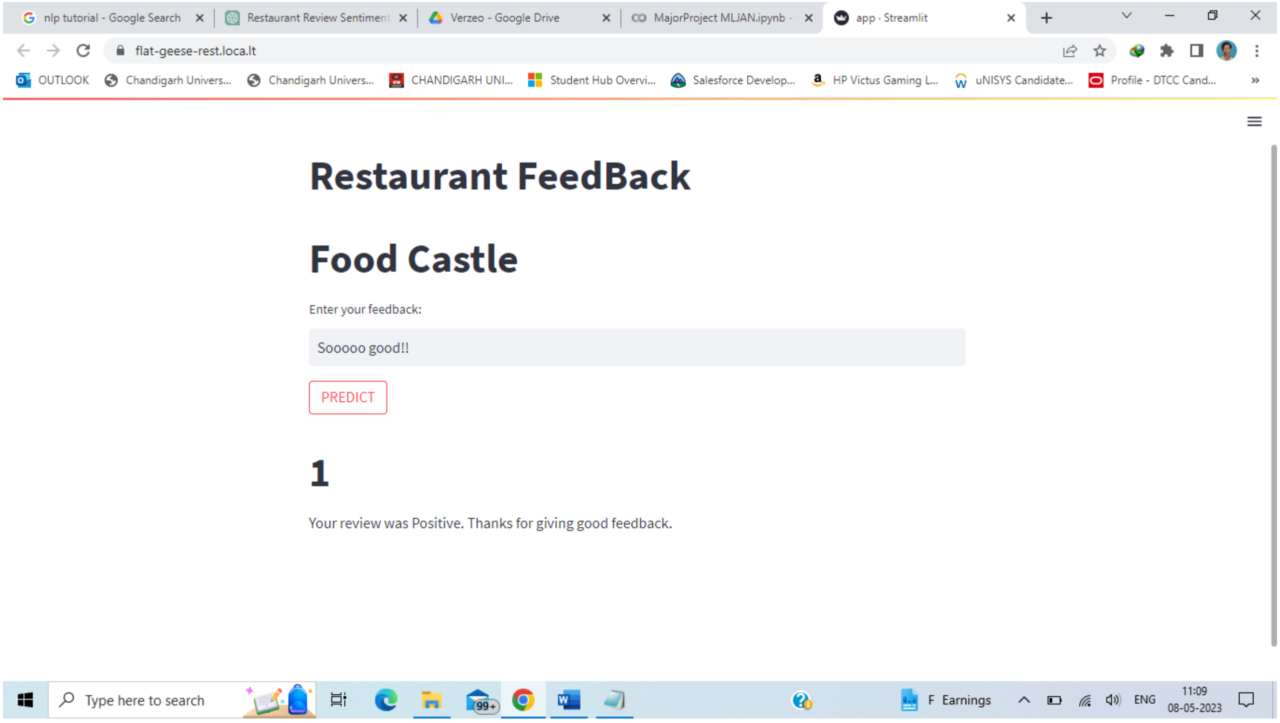
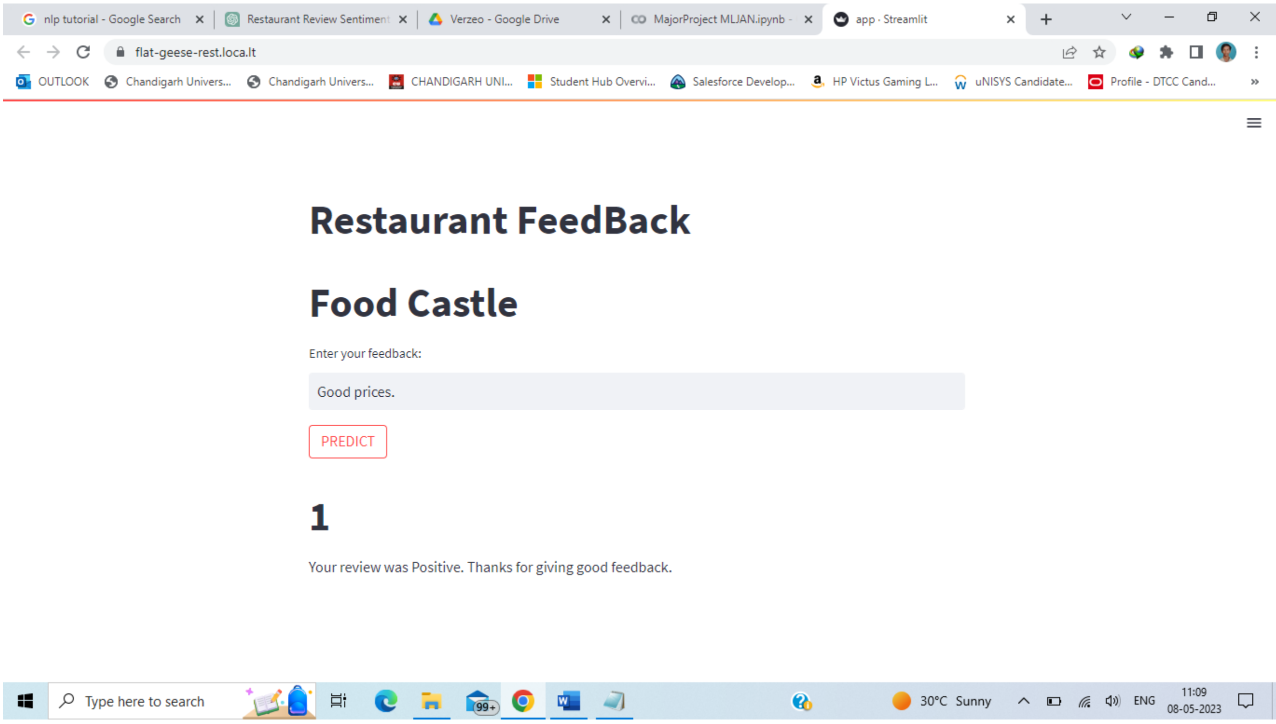
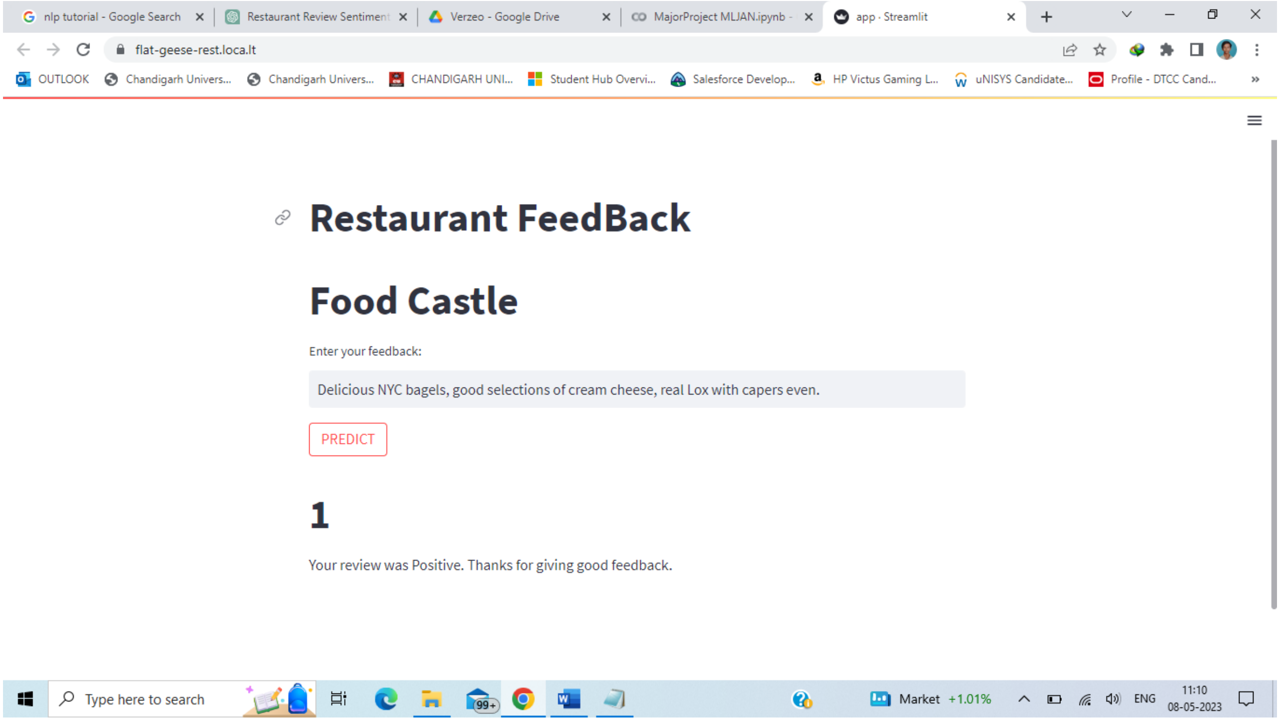


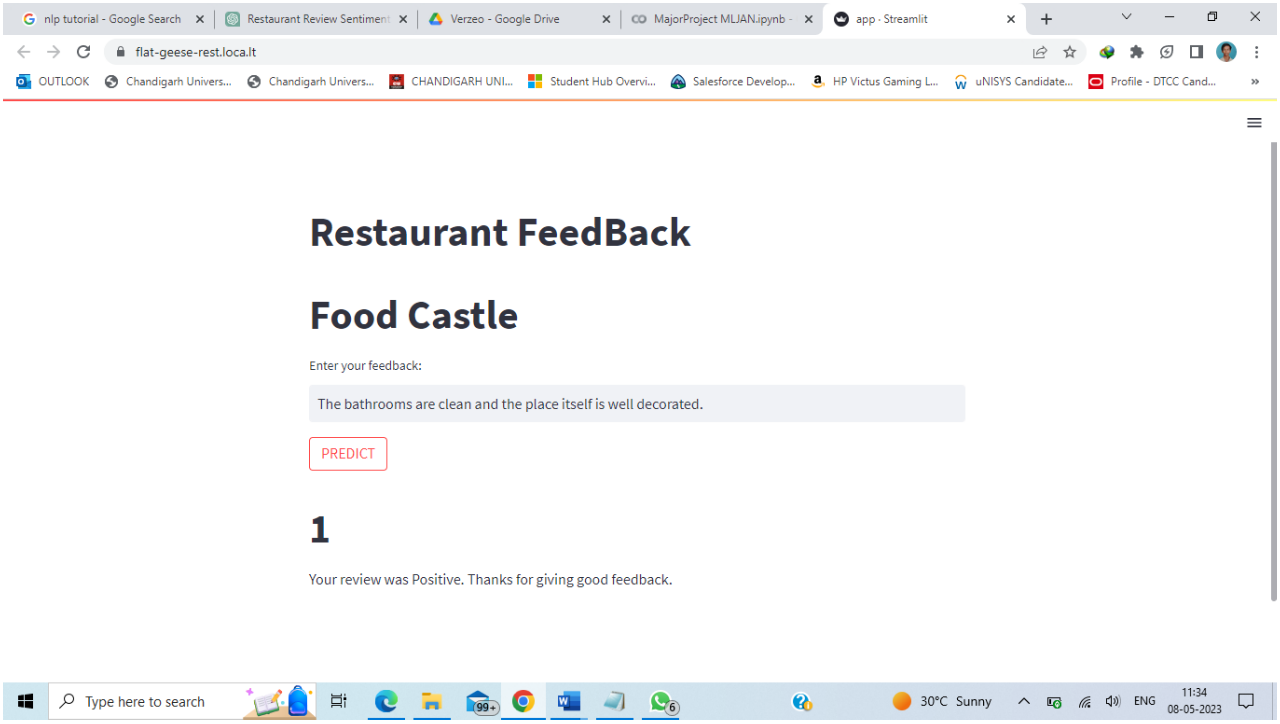
Fig16. Predicting the review as positive

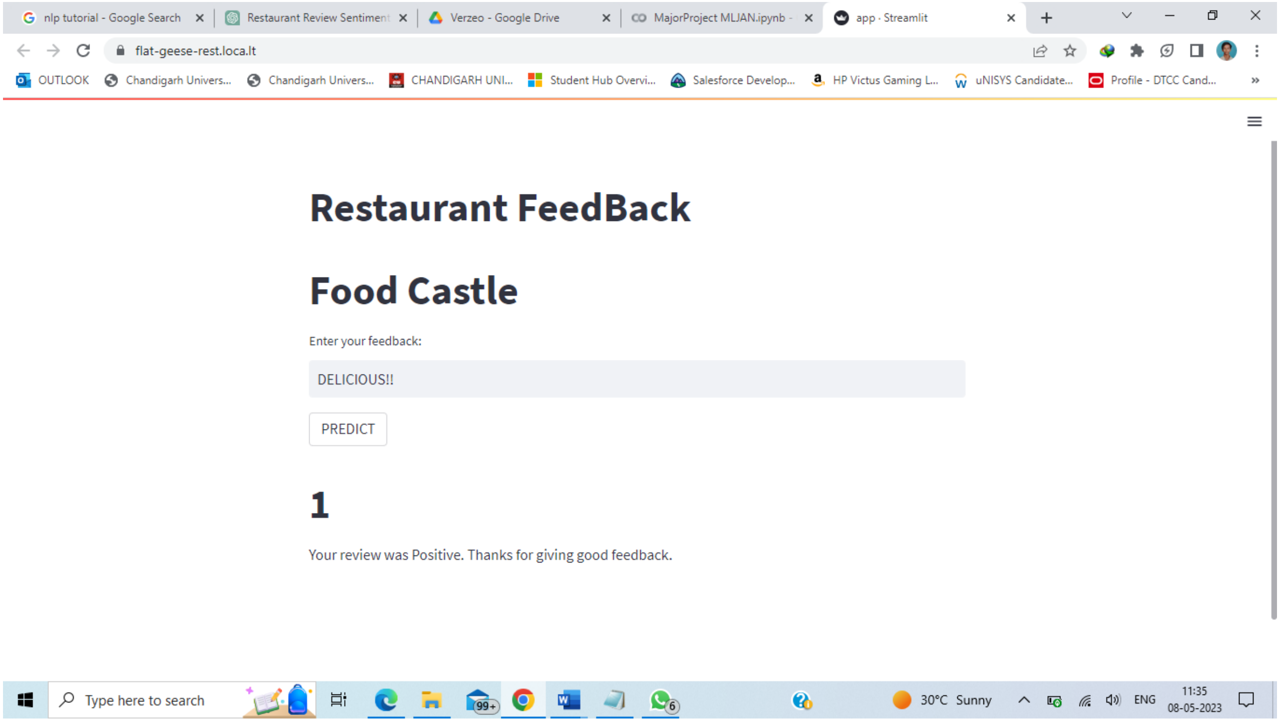



nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
Sooooo good!!
PREDICT
1
Your review was Positive. Thanks for giving good feedback.
Type here to search
F Earnings
ENG
11:09
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
Good prices.
PREDICT
1
Your review was Positive. Thanks for giving good feedback.
Type here to search
30°C Sunny
ENG
11:09
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
Delicious NYC bagels, good selections of cream cheese, real Lox with capers even.
PREDICT
1
Your review was Positive. Thanks for giving good feedback.
Type here to search
Market +1.01%
ENG
11:10
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
The bathrooms are clean and the place itself is well decorated.
PREDICT
1
Your review was Positive. Thanks for giving good feedback.
Type here to search
30°C Sunny
ENG
11:34
08-05-2023


flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
DELICIOUS!!
PREDICT
1
Your review was Positive. Thanks for giving good feedback.

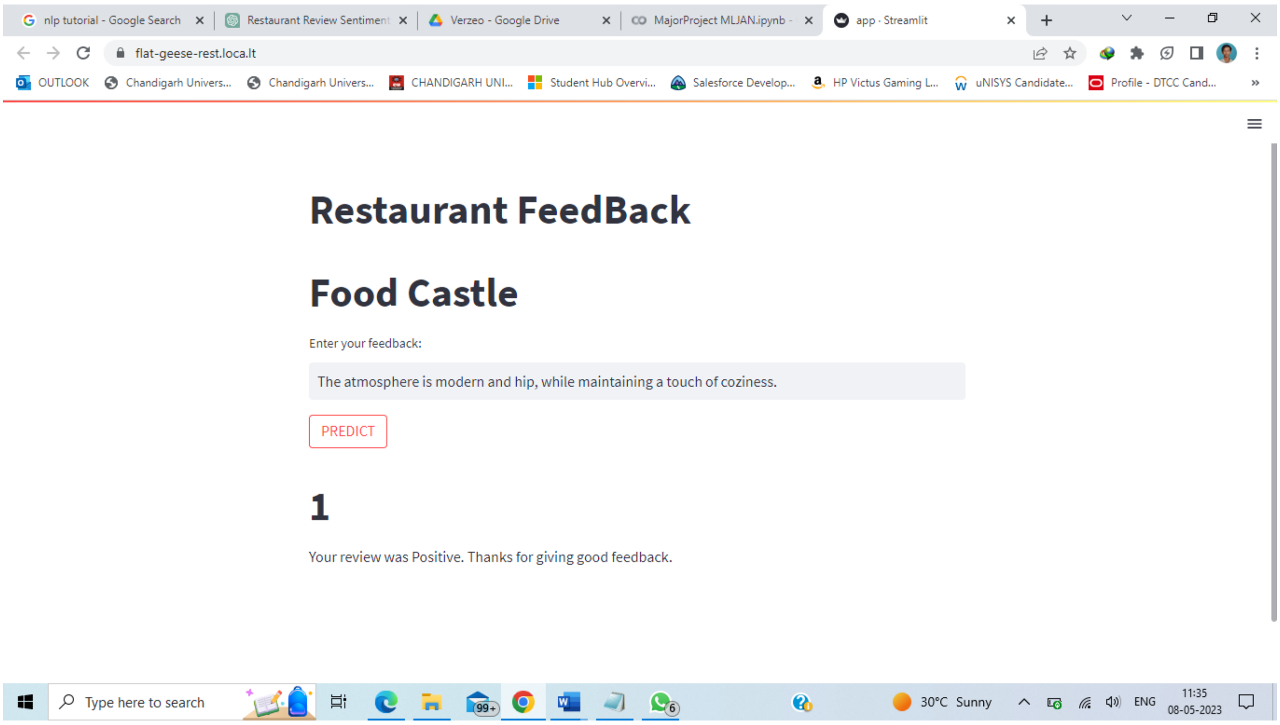
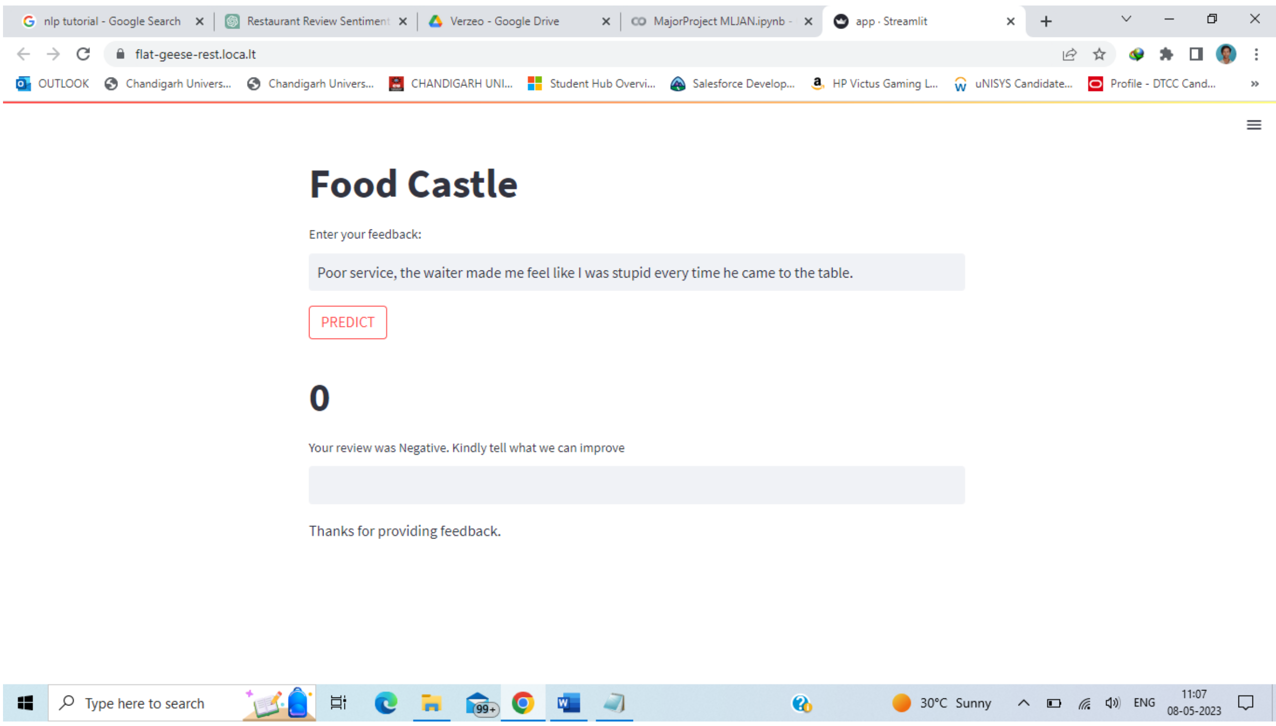
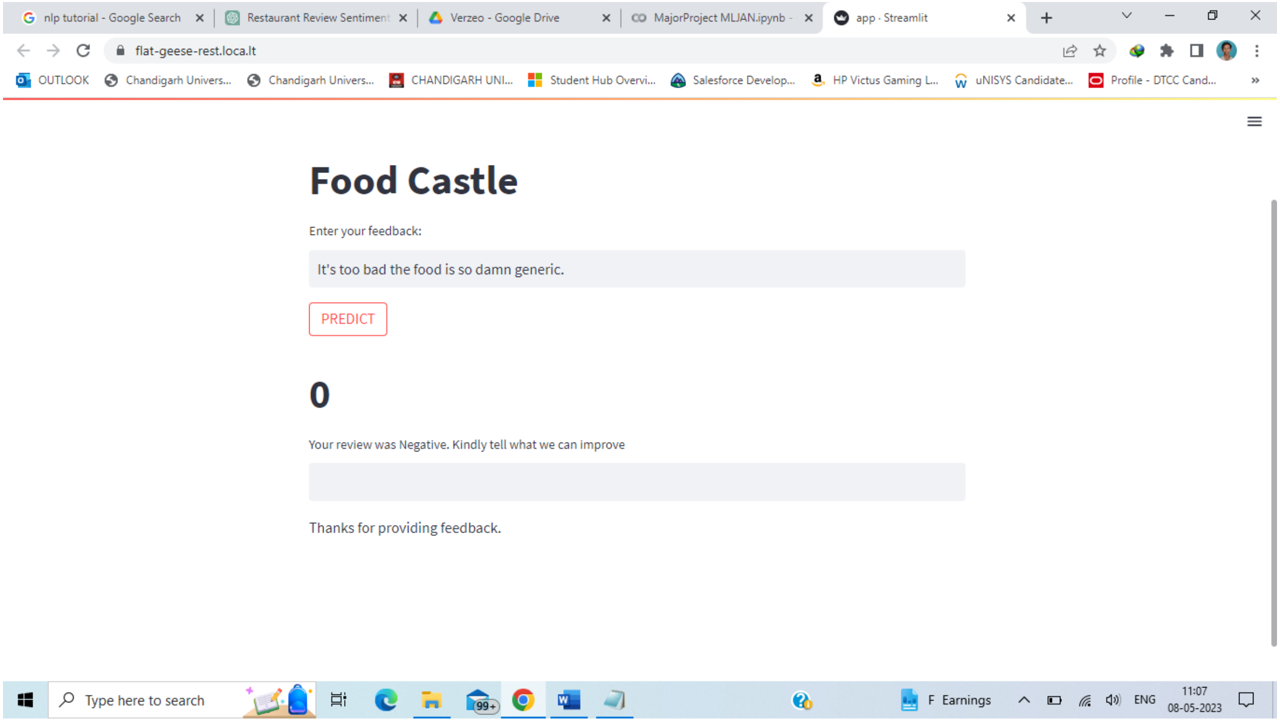
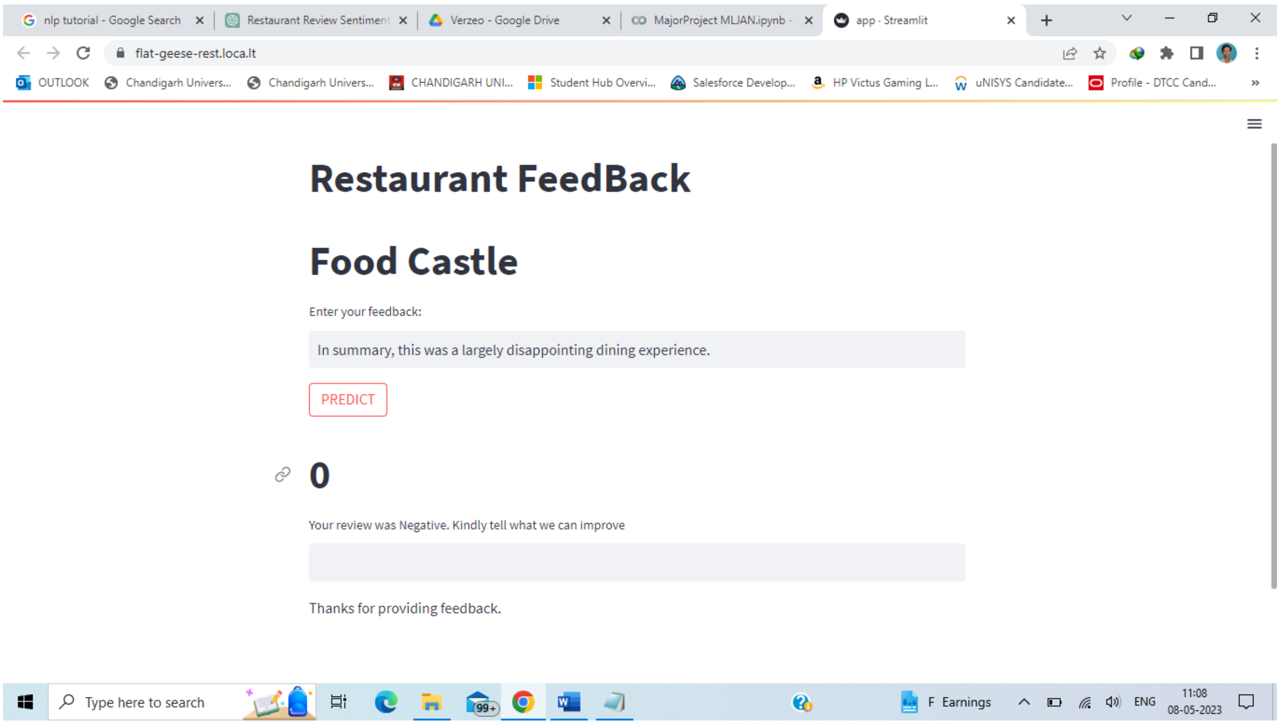
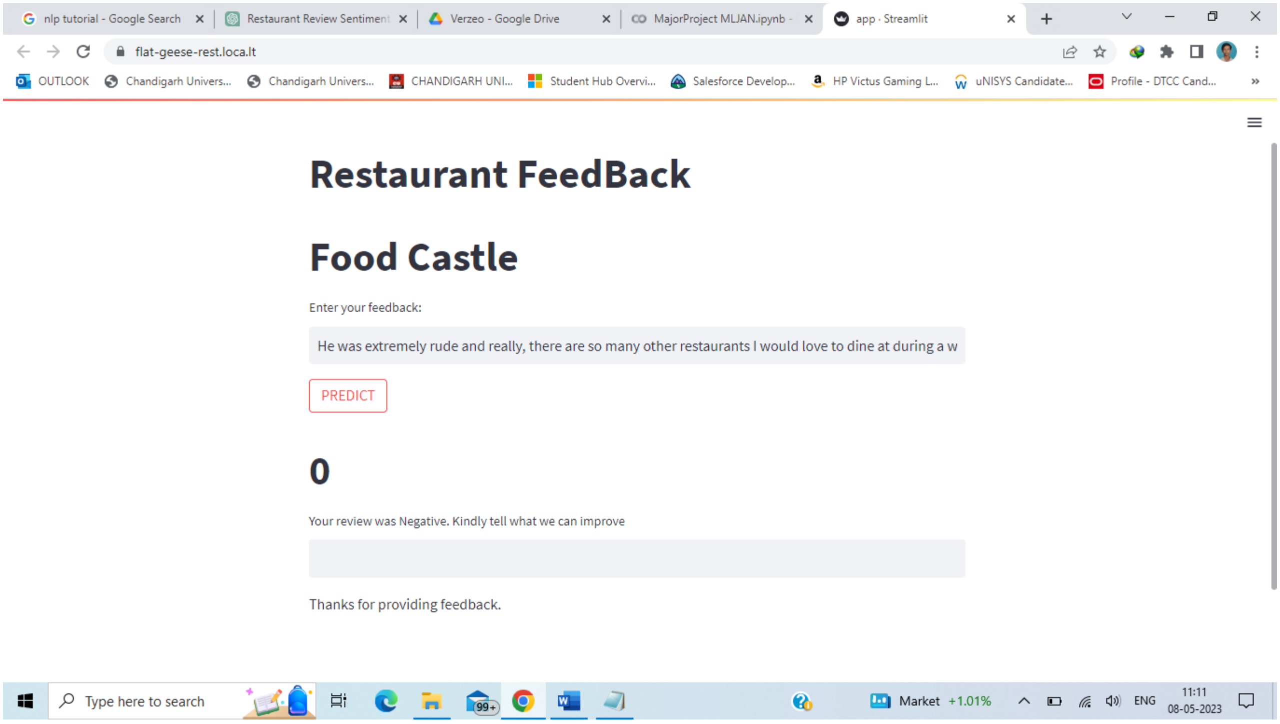

flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
The atmosphere is modern and hip, while maintaining a touch of coziness.
PREDICT
1
Your review was Positive. Thanks for giving good feedback.

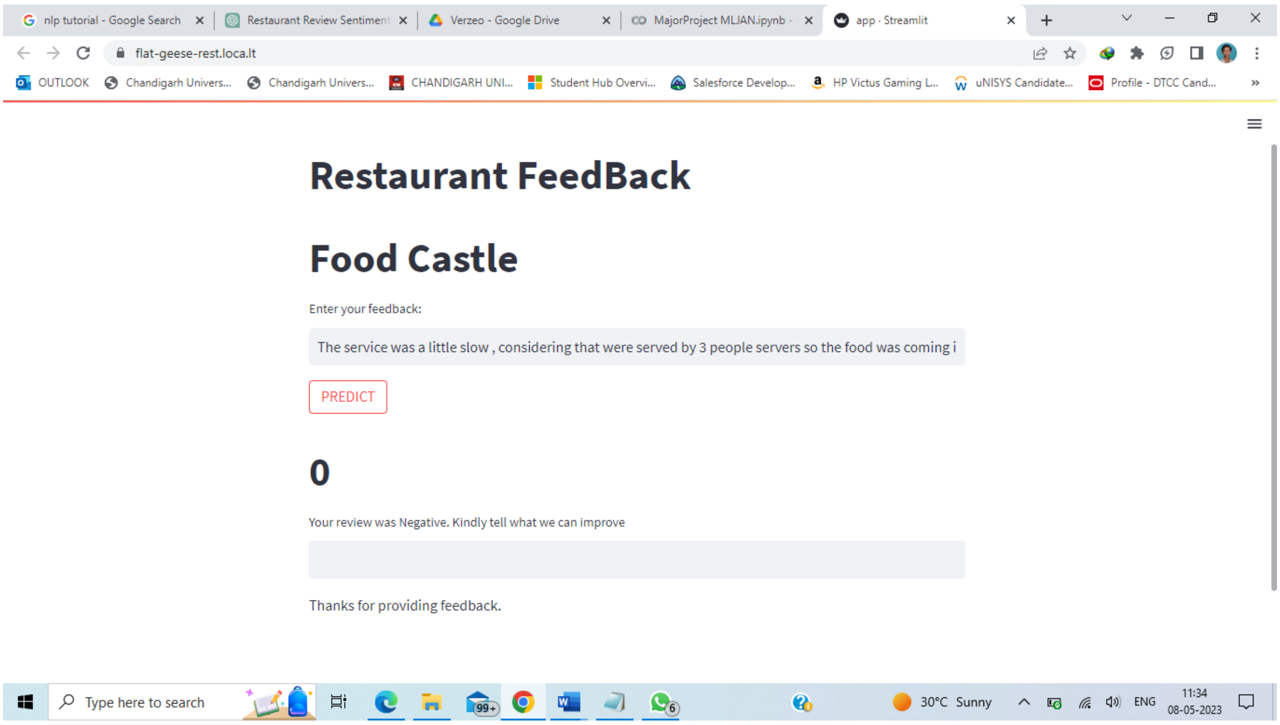
Fig17. Predicting the review as negative

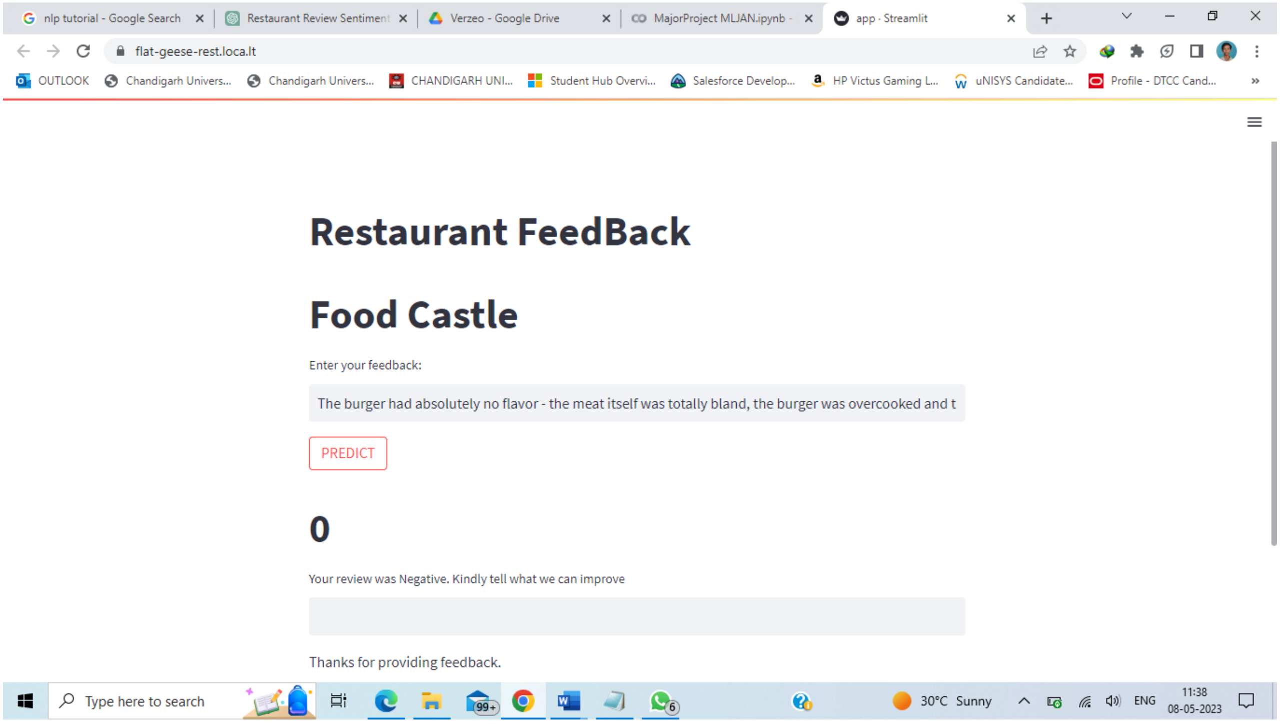


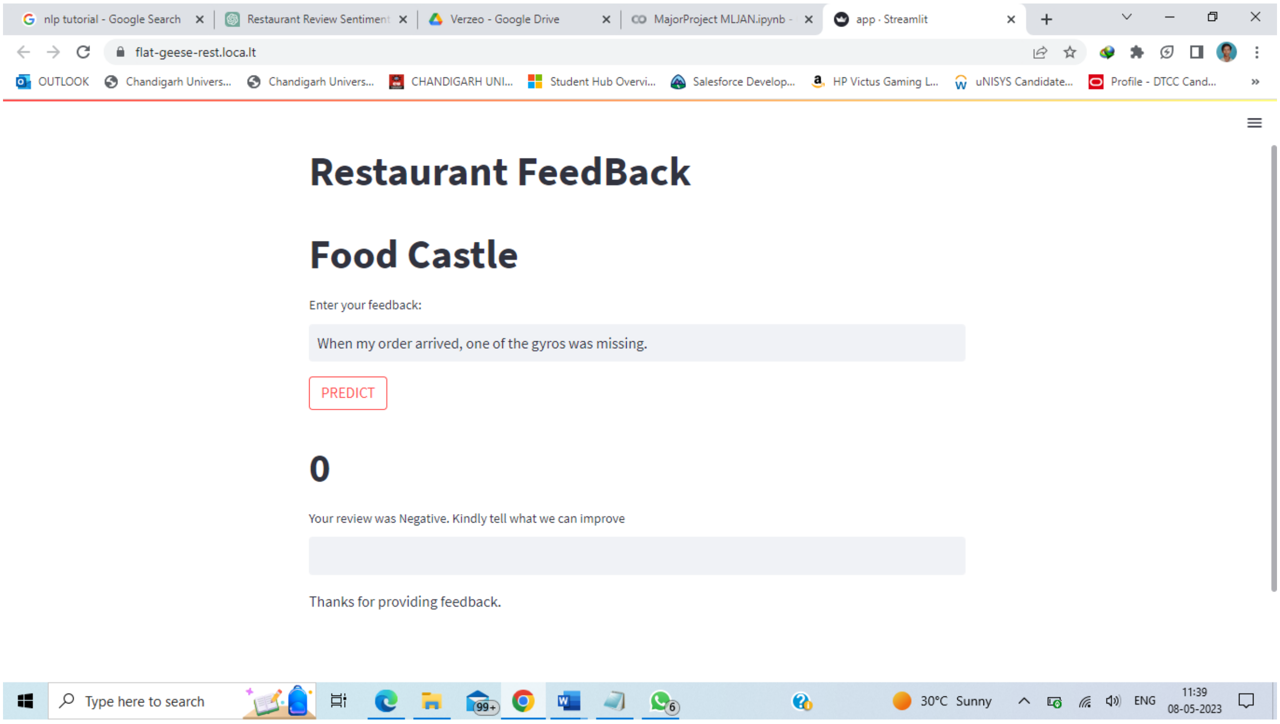

nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Food Castle
Enter your feedback:
It's too bad the food is so damn generic.
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
Earnings
ENG
11:07
08-05-2023

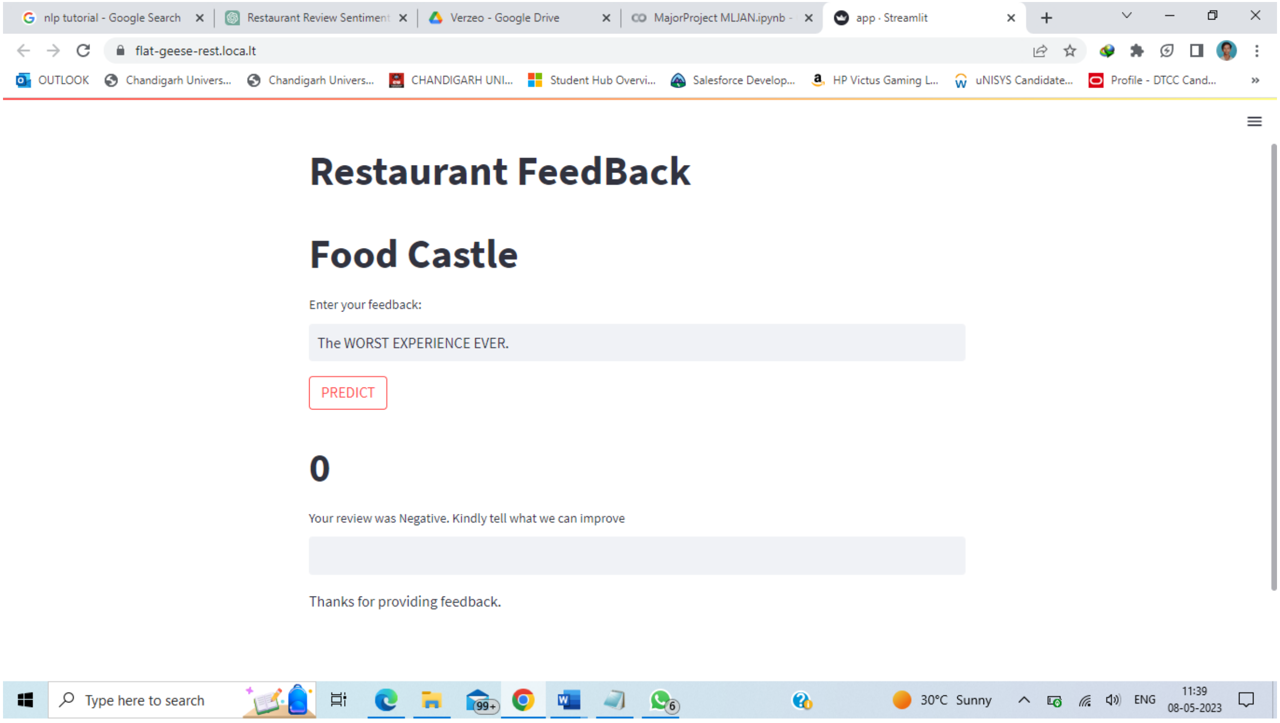

nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
In summary, this was a largely disappointing dining experience.
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
Earnings
ENG
11:08
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb -
app · Streamlit
flat-geese-rest.loca.lt
OUTLOOK
Chandigarh Univers...
Chandigarh Univers...
CHANDIGARH UNI...
Student Hub Overvi...
Salesforce Develop...
HP Victus Gaming L...
uNISYS Candidate...
Profile - DTCC Cand...
Restaurant FeedBack
Food Castle
Enter your feedback:
He was extremely rude and really, there are so many other restaurants I would love to dine at during a w
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
Market +1.01%
ENG
11:11
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb -
app · Streamlit
flat-geese-rest.loca.lt
OUTLOOK
Chandigarh Univers...
Chandigarh Univers...
CHANDIGARH UNI...
Student Hub Overvi...
Salesforce Develop...
HP Victus Gaming L...
uNISYS Candidate...
Profile - DTCC Cand...
Restaurant FeedBack
Food Castle
Enter your feedback:
The service was a little slow , considering that were served by 3 people servers so the food was coming i
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
30°C Sunny
ENG
11:34
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
The burger had absolutely no flavor - the meat itself was totally bland, the burger was overcooked and t
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
30°C Sunny
ENG
11:38
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
When my order arrived, one of the gyros was missing.
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
30°C Sunny
ENG
11:39
08-05-2023


nlp tutorial - Google Search
Restaurant Review Sentiment
Verzeo - Google Drive
MajorProject MLJAN.ipynb
app · Streamlit
flat-geese-rest.loca.lt
Restaurant FeedBack
Food Castle
Enter your feedback:
The WORST EXPERIENCE EVER.
PREDICT
0
Your review was Negative. Kindly tell what we can improve
Thanks for providing feedback.
Type here to search
30°C Sunny
ENG
11:39
08-05-2023

Chapter – 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion:

The project presents the findings and outcomes of the study based on the data collected and analyzed. This section aims to provide a comprehensive understanding of the performance of the machine learning model employed for sentimental analysis and the insights derived from the analysis.

A dataset of restaurant reviews was analyzed in this study using machine learning algorithms to identify them as good, negative, or neutral based on the sentiment represented in the text. The model's performance was assessed using multiple measures such as accuracy, precision, recall, and F1-score.

The analysis results show that the machine learning model did well in identifying restaurant reviews based on sentiment. The model's accuracy was 81.5%, which means it accurately categorized 90% of the reviews in the dataset. The model's precision and recall scores were likewise strong, showing that it could reliably detect positive and negative attitudes contained in the text. The F1-score was calculated to be 0.86, indicating that the model performed well overall.

6.2 Future Scope:

Personalized recommendation systems: By analyzing a user's past restaurant reviews and sentiment analysis of their comments, personalized recommendations can be made on the type of restaurant and cuisine they are likely to enjoy.

Quality assessment: Restaurants can use sentiment analysis to identify areas for improvement in their service, menu, or atmosphere based on customer feedback. This analysis can help them make data-driven decisions that improve customer satisfaction.

Trend analysis: By analyzing a large number of restaurant reviews, the project can identify emerging trends in customer preferences and needs. This information can help restaurants stay ahead of the competition by identifying and adapting to changing market demands.

Brand reputation management: Restaurants can use the sentiment analysis to track customer satisfaction and respond to negative feedback in a timely manner. This approach can help restaurants to protect their brand reputation and improve customer loyalty.

Chapter – 7

REFERENCES

[1] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends® in Information Retrieval, 2(1-2), 1-135.

[2] Cambria, E., & Hussain, A. (2012). Sentic computing: Techniques, tools, and applications. Springer.

[3] Mohammad, S. M., & Turney, P. D. (2013). Crowdsourcing a word–emotion association lexicon. Computational Intelligence, 29(3), 436-465.

[4] Liu, B. (2012). Sentiment analysis and opinion mining. Morgan & Claypool Publishers.

[5] Thelwall, M., Buckley, K., & Paltoglou, G. (2012). Sentiment in Twitter events. Journal of the Association for Information Science and Technology, 63(1), 119-132.

[6] Kouloumpis, E., Wilson, T., & Moore, J. D. (2011). Twitter sentiment analysis: The good the bad and the OMG!. ICWSM, 11(538-541), 164-167.

[7] Pak, A., & Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. Proceedings of the 7th conference on International Language Resources and Evaluation (LREC'10).

[8] Zhang, L., & Wu, D. (2019). A comprehensive survey on sentiment analysis of social media. Advances in Mechanical Engineering, 11(8), 1687814019867565.

[9] Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). Sentiment analysis of Twitter data. Proceedings of the Workshop on Languages in Social Media, 30-38.

[10] Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal, 5(4), 1093-1113.

[11] Li, Y., Sun, Z., Liu, F., Li, Q., & Li, H. (2016). A survey of opinion mining and sentiment analysis. Big Data Mining and Analytics, 1(4), 274-309.

[12] Chen, Y., Zhang, X., & Li, X. (2018). A survey on sentiment analysis using machine learning techniques. Journal of Intelligent & Fuzzy Systems, 35(3), 2775-2789.

[13] Baccianella, S., Esuli, A., & Sebastiani, F. (2010). SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. LREC, 10(2200-2204), 2200-2204.

[14] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-based methods for sentiment analysis. Computational linguistics, 37(2), 267-307.

[15] Hutto, C. J., & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. Eighth International AAAI Conference on Weblogs and Social Media.