

SENTIMENT ANALYSIS ON GOOGLE PAY REVIEWS

A Mini project Report

Submitted for the Partial fulfillment for the award of degree of

### MASTER OF SCIENCE IN

**DATA SCIENCE AND BUSINESS ANALYTICS**

|  |  |  |
| --- | --- | --- |
| By | | |
| **NAME** | **:** | **GOWTHAM K** |
| **REGISTER NO** | **:** | **22235136** |

Under the guidance of

## DR.K.DHARMARAJAN M.Sc, M.Phil,Ph.D

**School of Computing Sciences Department of Information Technology (BCA & IT)**



**CERTIFICATE**

This is to certify that the project titled “**SENTIMENT ANALYSIS ON GOOGLE PAY REVIEWS”** is the original record work done by **GOWTHAM K (REGISTER NO. 22235136)** Submitted by me during the period from 2022 TO 2024, under my guidance and supervision for the partial fulfillment of award of degree in Master of Science in **DATA SCIENCE AND BUSINESS ANALYTICS** (M.Sc. DS & BA), as per the syllabus prescribed by the Vels Institute of Science, Technology and Advanced Studies (VISTAS

GUIDE HEAD OF THE DEPARTMENT

Submitted for the Viva-Voice examination held on at Vels Institute of Science, Technology and Advanced Studies (VISTAS).

Place: Chennai EXAMINERS

Date:



## DECLARATION

I, **GOWTHAM K** **(REGISTER NO. 22235136)**, declares that the project entitled “**SENTIMENT ANALYSIS ON GOOGLE PAY REVIEWS**” submitted by me during the period from 2022 TO 2024 under the guidance **DR.K.DHARMARAJAN,** and has not formed the basis for the award of any degree diploma, associate-ship, fellowship, titles in this or any other University or other similar institutions of higher learning.

Place: Chennai Signature of the Student

Date:



## BONAFIDE CERTIFICATE

I certify that the project titled “**SENTIMENT ANALYSIS ON GOOGLE PAY REVIEWS”** for the Master of Science in Data Science and Business Analytics (M.Sc. Ds & BA) done by **GOWTHAM K (REGISTER NO. 22235136)** is the project work carried out by he during the period from 2022 TO 2024 under my guidance and supervision and that this work has not formed the basis for the award of any degree, diploma, associate-ship, fellowship, titles in this or any other University or other similar institutions of higher learning. He fulfills the eligibility criteria for submission of this project as per rules and regulations of the University.

Place: Chennai

Date: Guide

## ACKNOWLEDGEMENT

I thank our **Honorable Founder & Chancellor Dr Ishari K. Ganesh** , **Honorable Vice-Chancellor, Dr. S. Sriman Narayanan** and **Honorable Vice President Dr. Preethaa Ganesh** VISTAS, for allowing me to pursue my M.Sc. Programme.

I thank our **Registrar, Dr. P. Saravanan** and **The Controller of Examinations**, **Dr. A. Udhayakumar,** VISTAS for their support.

I sincerely **thank Dr. P. SUJATHA**,Professor and Head of the Department of Information Technology, VIATAS for his continued support and motivation.

I profoundly thank my research supervisor **DR.K.DHARMARAJAN**, Associate Professor for he total encouragement, inspiration and advice from the beginning till the end of this project / dissertation.

I am grateful to the faculties of the department of Information Technology, VISTAS for their assistance for the successful completion of this dissertation.

I thank VISTAS librarians for providing me access to the useful textual materials available in VISTAS which supported my research genre.

I am grateful to my friends and family for rendering me their recommendations and moral support.

Name of the student / scholar

### GOWTHAM K

**ABSTRACT**

Google Pay is a widely used digital payment platform that has garnered a vast user base globally. Understanding user sentiment towards this platform is of paramount importance for Google and other stakeholders in the digital payment industry. This abstract provides an overview of a comprehensive study that employs Natural Language Processing (NLP) techniques to analyse the sentiment of user reviews of Google Pay.

The study focuses on gathering user reviews and feedback from various online platforms such as app stores, review websites, and social media. These reviews are a rich source of information regarding user experiences, satisfaction, and concerns. NLP techniques are applied to extract valuable insights from these reviews by determining the sentiment expressed within them. Sentiment analysis involves classifying each review as positive, negative, or neutral, allowing for a quantitative assessment of user sentiment.

The analysis involves multiple steps, including data collection, pre-processing, sentiment classification, and results interpretation. We use state-of-the-art NLP models to ensure the accuracy and reliability of sentiment classification. Additionally, the study explores the application of advanced NLP techniques such as aspect-based sentiment analysis, which helps in understanding specific aspects of Google Pay that users are satisfied or dissatisfied with.

TABLE OF CONTENTS

1. **INTRODUCTION**
   1. Objectives …………………………………………………………………………………4
   2. Challenges in the Domain ……………………………………………………………….5
   3. Significance of Mobile Payment Application …………………………………………….7
   4. Significance of Sentiment Analysis in Understanding User Feedback …………...............8

## SYSTEM SEPIFICATION

* 1. Hardware Specification ………………………………………………………..................12
  2. System Requirements …………………………………………………………….............12

## MODEL ANALYSIS

* 1. Problem Statement …………………………………………………………….................25
  2. Challenges and Limitations in Existing Systems for Sentiment Analysis ………………26
  3. Proposed Model ………………………………………………………………….............27

## MODEL DESIGN

* 1. Modules …………………………………………………………………………..............29
  2. Data Collection …………………………………………………………………...............30
  3. Data pre-processing …………………………………………………………….….…....33
  4. Sentiment Labeling …………………………………………………………….….….….35

## SENTIMENT CLASSIFICATION

* 1. Vectorization …………………………………………………………………….….……38
  2. Model ………………………………………………………………………….….……...39
  3. Logistic Regression ……………………………………………………………....………41
  4. Model Testing ……………………………………………………………………….….43

## SOURCE CODE

* 1. Coding …………………………………………………………………………….…..….46

## CONCLUSION ………………………………………………………………….………67

**CHAPTER 1**

1. **INTRODUCTION**

### OBJECTIVES

In today's digital age, mobile payment applications have become an integral part of our financial landscape, simplifying the way we transact and manage our money. Google Pay is one such prominent player in the field, offering users a convenient and secure platform to make payments, transfer money, and manage their finances. User feedback is invaluable for any technology company, and Google Pay is no exception. Therefore, the objective of this project is to analyse Google Pay customer reviews to assess user satisfaction and identify areas for improvement. This two-page document outlines the significance and scope of the project and the methodology to be employed.

### Significance of the Project

The significance of this project lies in the value of user feedback as a source of critical insights for Google Pay. Customer reviews, which are freely available on various platforms such as app stores and social media, provide a window into the experiences and opinions of Google Pay users. By conducting a systematic analysis of these reviews, Google Pay can achieve several important objectives:

### User Satisfaction Assessment

The project aims to gauge the overall satisfaction levels of Google Pay users. By understanding the sentiments expressed in reviews, Google Pay can identify the aspects of its service that users appreciate and those that may lead to dissatisfaction.

### Quality Improvement

The project will help Google Pay pinpoint areas for improvement. By identifying recurring issues or concerns in user feedback, Google Pay can take targeted actions to enhance the quality of its service, leading to a more positive user experience.

### Competitive Positioning

Analysing user reviews also involves understanding how Google Pay stacks up against its competitors. By comparing the sentiment of Google Pay reviews with those of other mobile payment apps, the project can provide insights into market positioning and potential areas where Google Pay can outperform rivals.

### CHALLENGES IN THE DOMAIN

The field of sentiment analysis and its integration with ensemble methods and data visualization present several complex challenges that researchers and practitioners must address. These challenges underscore the intricacies and evolving nature of this domain.

### Ambiguity and Context

Sentiment analysis often grapples with the ambiguity of human language. Words and phrases can have multiple meanings, and the interpretation of sentiment can vary based on context. Understanding context is a persistent challenge, especially when dealing with sarcasm, irony, or cultural nuances.

### Data Quality and Noise

The quality of the data used for sentiment analysis is a recurring challenge. User-generated content is often noisy, containing typos, misspellings, grammatical errors, and slang. Distinguishing genuine sentiment from noise is essential for accurate analysis.

### Multilingual and Cross-Cultural Analysis

The global nature of the internet means that sentiment analysis often needs to accommodate multiple languages and cultural contexts. Different languages and cultures express sentiment differently, making it challenging to create universal sentiment models.

### Handling Negations and Modifiers

Sentences with negations (e.g., "not good") and modifiers (e.g., "very bad" vs. "slightly bad") require advanced models to correctly interpret sentiment. Handling these linguistic nuances remains a challenge.

### Sentiment Evolution

Public sentiment can change rapidly due to external events, news, or viral content. Sentiment models need to adapt to these shifts and provide real-time analysis.

### Lack of Labeled Data

Creating labeled datasets for training sentiment analysis models is a resource-intensive task. The availability of labeled data that covers a wide range of domains and languages is often limited, making it challenging to build robust models.

### Model Interpretability

As sentiment analysis models become more complex, model interpretability becomes a concern. Users and decision-makers often require insight into why a model classified a text in a particular way, which is challenging with certain ensemble methods and deep learning models.

### Ethical Concerns

Sentiment analysis has ethical considerations, especially when used in applications like automated decision-making or profiling. It's crucial to address potential biases in data and models and ensure fairness.

### Scalability

For real-world applications, sentiment analysis must be scalable to process vast amounts of data. Developing high-performance models that can handle big data efficiently is a continuous challenge.

### Visualization Complexity

Data visualization, while powerful, can become complex when dealing with multidimensional sentiment analysis results. Creating intuitive, informative visualizations without oversimplification is a challenge.

Addressing these challenges requires a combination of advanced NLP techniques, machine learning, domain expertise, and ongoing research. As the field of sentiment analysis evolves, researchers and practitioners must continuously adapt and innovate to overcome these obstacles and extract valuable insights from textual data.

### SIGNIFICANCE OF MOBILE PAYMENT APPLICATIONS

In recent years, the world has witnessed a rapid transformation in the way people conduct financial transactions. The advent of mobile payment applications has revolutionized the way we make payments, from traditional cash and card-based systems to seamless digital transactions. Mobile payment applications have become an integral part of our daily lives, offering a wide array of benefits that extend beyond just convenience. This shift towards mobile payments has transformed the way businesses and individuals handle their finances, impacting various aspects of the economy, society, and individual lifestyles.

### Convenience and Accessibility

One of the primary reasons for the growing importance of mobile payment applications is the unparalleled convenience and accessibility they offer. These apps allow users to make payments or transfer money with just a few taps on their smartphones. Whether it's splitting a restaurant bill with friends, paying for groceries, or settling utility bills, mobile payment apps eliminate the need for physical cash or credit/debit cards. This convenience is particularly vital in a fast-paced world where time is of the essence.

Moreover, mobile payment applications break down geographical barriers. They enable users to make transactions from anywhere, reducing the need to visit physical banks or ATMs. As a result, people in rural areas, where traditional banking infrastructure may be lacking, gain access to financial services. This increased accessibility is a crucial step toward financial inclusion, ensuring that more individuals have the opportunity to participate in the modern economy.

### Enhanced Security

Mobile payment applications have also significantly improved the security of financial transactions. Traditional payment methods, such as cash or physical cards, are susceptible to theft and loss. In contrast, mobile payment apps employ advanced security features, such as biometric authentication, tokenization, and encryption, to protect user data and transactions. Users can remotely lock or wipe their devices in case they are lost or stolen, further safeguarding their financial information.

### Efficient Budget Management

Mobile payment applications have also become powerful tools for personal finance management. They offer features like transaction categorization, expenditure tracking, and detailed statements, enabling users to gain insights into their spending habits. This empowers individuals to make informed financial decisions and establish budgeting goals. In addition, some apps offer automatic savings features, rounding up purchases to save the spare change, or setting up recurring transfers to savings accounts, making it easier for individuals to save and invest.

For businesses, mobile payment applications provide valuable data on customer spending patterns. This data can be analysed to understand consumer behaviour and preferences, facilitating more effective marketing and product development strategies. This, in turn, can help businesses optimize their operations and increase revenue.

### Economic Impact

The widespread adoption of mobile payment applications has significant economic implications. For businesses, these apps expand their customer base by allowing them to cater to a broader audience. With mobile payments, online businesses can tap into global markets without the need for complex international payment systems. This not only stimulates economic growth but also fosters entrepreneurship as new businesses can be established more easily with access to a global customer base.

Additionally, the reduction in the use of physical cash has an impact on the cost of currency management for governments and financial institutions. The digital nature of mobile payments reduces the need for printing, transporting, and securing physical currency. This can lead to substantial cost savings and increased efficiency in the financial system.

### SIGNIFICANCE OF SENTIMENT ANALYSIS IN UNDERSTANDING USER FEEDBACK

Sentiment analysis, also known as opinion mining, is a powerful natural language processing (NLP) technique that involves the automated extraction and analysis of subjective information from text data to determine the sentiment or emotional tone expressed by the author. It plays a crucial role in understanding user feedback, whether it's from customer reviews, social media posts, surveys, or any form of written communication. The significance of sentiment analysis in understanding user feedback

### Insight into User Satisfaction

Sentiment analysis helps organizations gauge the overall satisfaction levels of their customers or users. By analysing feedback, companies can determine whether their products or services are well-received, which aspects are appreciated, and where improvements are needed. This insight is invaluable for product development and marketing strategies.

### Quick and Scalable Feedback Processing

With the vast amount of user-generated content available online, it is practically impossible for companies to manually sift through and analyse all user feedback. Sentiment analysis automates this process, allowing organizations to process and categorize large volumes of feedback quickly and accurately.

### Identifying Emerging Trends

Sentiment analysis enables organizations to identify emerging trends and issues by tracking shifts in sentiment over time. By monitoring user feedback, companies can detect changes in customer preferences or concerns and respond proactively.

### Competitor Analysis

Understanding user sentiment not only involves analysing feedback related to one's own products or services but also includes monitoring how users feel about competitors. This information can be critical for market positioning and competitive strategy.

### Brand Reputation Management

Sentiment analysis allows companies to manage their brand reputation effectively. By tracking online sentiment, they can respond to negative comments or issues promptly, potentially mitigating reputational damage and demonstrating responsiveness to customer concerns.

### Customer Support Improvement

Analysing the sentiment of customer support interactions can help companies identify areas where their support teams excel and where they need improvement. This leads to better customer service and enhanced customer experiences.

### Product Development and Innovation

User feedback often contains valuable suggestions and ideas for product enhancements or new features. Sentiment analysis can help companies pinpoint these suggestions, facilitating innovation and product development based on customer preferences.

**Personalization and Customer Segmentation**

Sentiment analysis can contribute to more effective personalization and customer segmentation. By understanding individual preferences and emotions, companies can tailor their communications, offerings, and marketing strategies to specific customer groups.

### Enhanced Customer Engagement

Sentiment analysis can also be used to identify brand advocates and loyal customers. Recognizing and engaging with these customers can foster brand loyalty and word-of-mouth marketing.

# CHAPTER 2

## SYSTEM SPECIFICATION

* 1. **HARDWARE SPECIFICATION (GOOGLE COLABORATORY)** The Project is developed in the system having following configuration Processor Intel(R) Xeon(R) CPU @ 2.20GHz

Ram 12.7 GB

Monitor 15” COLOR

Hard Disk 107.7 GB

### SYSTEM REQUIREMENTS

Operation System Ubuntu (Google Colaboratory)

Programming Language Python

Environment IPYNB (Python Notebook)

### Python Environment

You should have a Python environment installed on your system. The code is written in Python, so you'll need Python to execute it. You can download Python from the official Python website (https://[www.python.org/downloads/)](http://www.python.org/downloads/)) and choose a version compatible with your operating system**.**

### Python Libraries

The code relies on several Python libraries. You can install these libraries using `pip`. You mentioned specific libraries in your original code, but here's a complete list for reference:

* pandas
* re
* emoji
* nltk

scikit-learn (for machine learning)

* wordcloud
* matplotlib

You can install these libraries with the following commands:

pip install pandas re emoji contractions nltk scikit-learn transformers wordcloud matplotlib

**NLTK Data:** The code uses the Natural Language Toolkit (NLTK) for various natural language processing tasks. You will need to download some NLTK data using the following commands

import nltk nltk.download('punkt') nltk.download('stopwords') nltk.download('words') nltk.download('vader\_lexicon')

### Operating System

The code should work on most major operating systems, including Windows, macOS, and Linux.

### Hardware

The code should run on standard desktop or laptop computers. There are no specific hardware requirements, but if you're working with very large datasets, you may benefit from a computer with more memory and processing power.

### IDE or Text Editor

You can run the code in a Python IDE (e.g., PyCharm, VSCode, or Jupyter Notebook) or a simple text editor, depending on your preference. In this research, Google Colaboratory

### HOW TO INSTALL PYTHON IDE

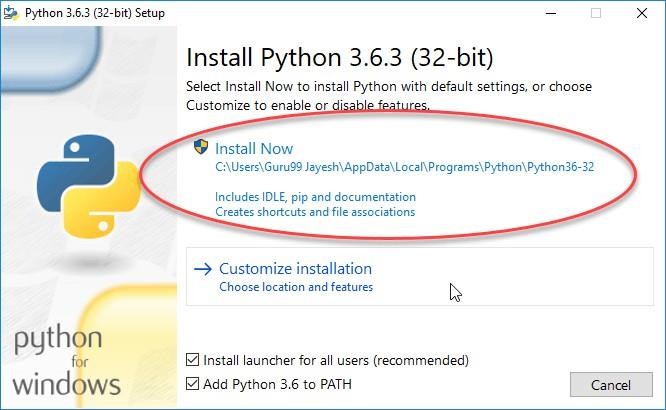
Below is a step-by-step process on how to download and install Python on Windows: Step 1: To download and install

Data: You need to provide a CSV file with the data you want to process. The code expects the data to be in a specific format, so make sure your CSV file follows that format.

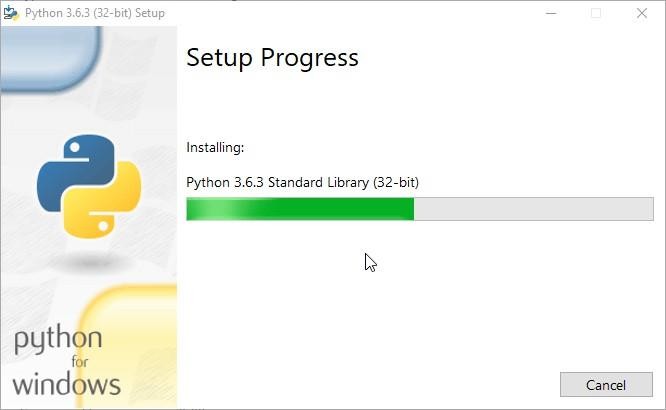
For Python, visit the official website of Python at <https://www.python.org/downloads/> and choose your version. We have chosen Python version 3.6.3.



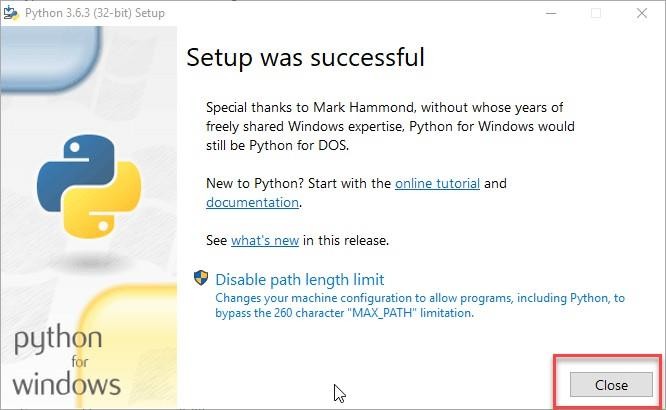
Step 2: Once the download is completed, run the .exe file to install Python. Now click on Install Now.



Step 3: You can see Python installing at this point.



Step 4: When it finishes, you can see a screen that says the Setup was successful. Now click on “Close”.



### Google Colaboratory

Google Colaboratory, or Colab, is an as-a-service [version of Jupyter Notebook](https://www.techtarget.com/searchaws/video/Set-up-a-Jupyter-notebook-on-AWS-with-this-tutorial) that enables you to write and execute [Python](https://www.techtarget.com/whatis/definition/Python) code through your browser.

Jupyter Notebook is a free, open source creation from the Jupyter Project. A Jupyter notebook is like an interactive laboratory notebook that includes not just notes and data, but also code that can manipulate the data. The code can be executed within the notebook, which, in turn, can capture the code output. Applications such as Matlab and Mathematica pioneered this model, but unlike those applications, Jupyter is a browser-based web application.

Google Colab is built around Project Jupyter code and hosts Jupyter notebooks without requiring any local software installation. But while Jupyter notebooks support multiple languages, including Python, Julia and R, Colab currently only supports Python.

Colab notebooks are stored in a Google Drive account and can be shared with other users, similar to other Google Drive files. The notebooks also include an autosave feature, but they do not support simultaneous editing, so collaboration must be serial rather than parallel.

Colab is free, but has limitations. There are some code types that are forbidden, such as media serving and crypto mining. Available resources are also limited and vary depending on demand, though Google Colab offers a pro version with more reliable resourcing. There are other cloud services based on Jupyter Notebook, including Azure Notebooks from Microsoft and [SageMaker Notebooks](https://www.techtarget.com/searchcloudcomputing/tip/6-Amazon-SageMaker-capabilities-developers-should-know-about) from Amazon.

### The Benefits of Google Colab

Enterprise data analysts and analytics developers can use Colab to work through data analytics and manipulation problems in collaboration. They can write, execute and revise core code in a tight loop, developing the documentation in Markdown format, LaTeX or HTML as they go.

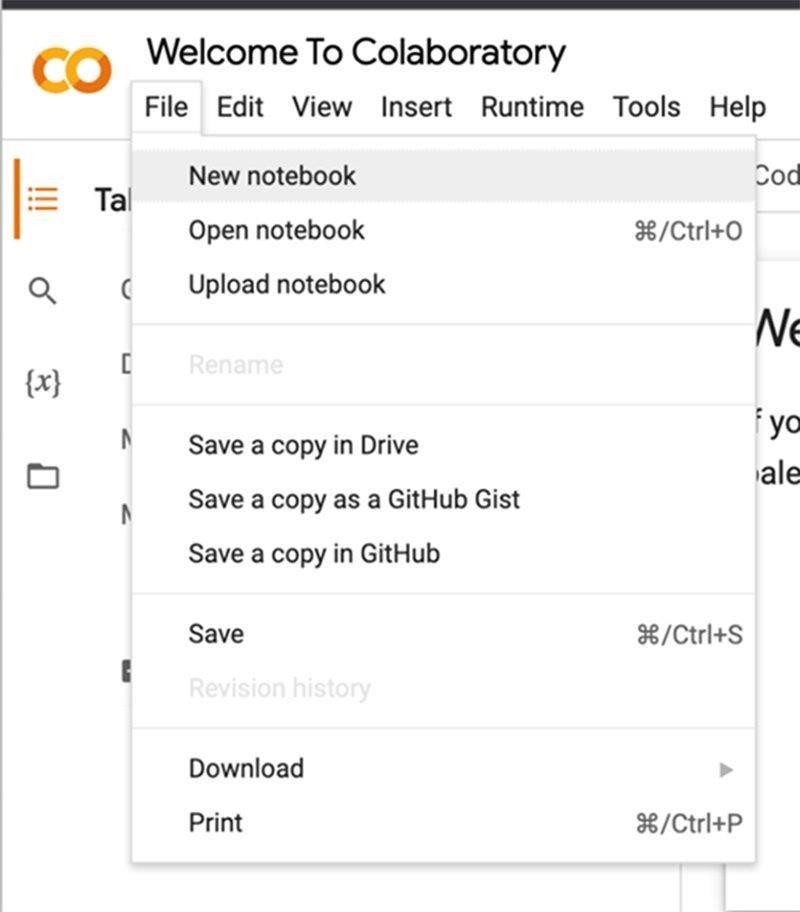
Notebooks can include embedded images as part of the documentation or as generated output. In addition, you can copy finished analytics code, with documentation, into other platforms for production use once sufficiently tested and debugged.

Google Colab eliminates the need for complex configuration setup and installation, as it runs right in the browser. It also includes pre-installed Python libraries that require no setup to use.

### How to Use Colaboratory

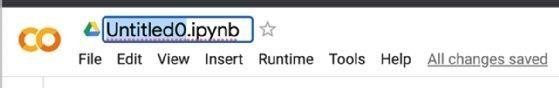
To use Colaboratory, you must have a Google account.

On your first [visit](https://colab.research.google.com/), you will see a Welcome To Colaboratory notebook with links to video introductions and basic information on how to use Colab.



If you are not yet logged in to a Google account, the system will prompt you to log in.

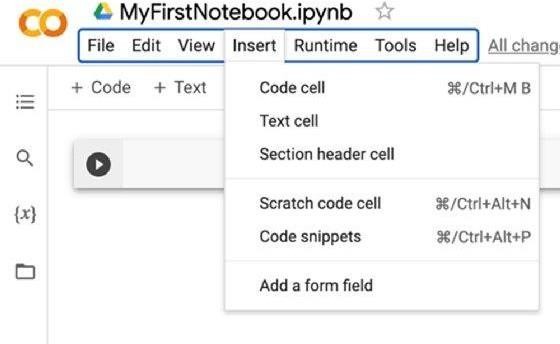
The notebook will by default have a generic name; click on the filename field to rename it.





The file type, IPYNB, is short for "IPython notebook" because IPython was the forerunner of Jupyter Notebook.

The interface allows you to insert various kinds of cells, mainly text and code, which have their own shortcut buttons under the menu bar via the Insert menu.

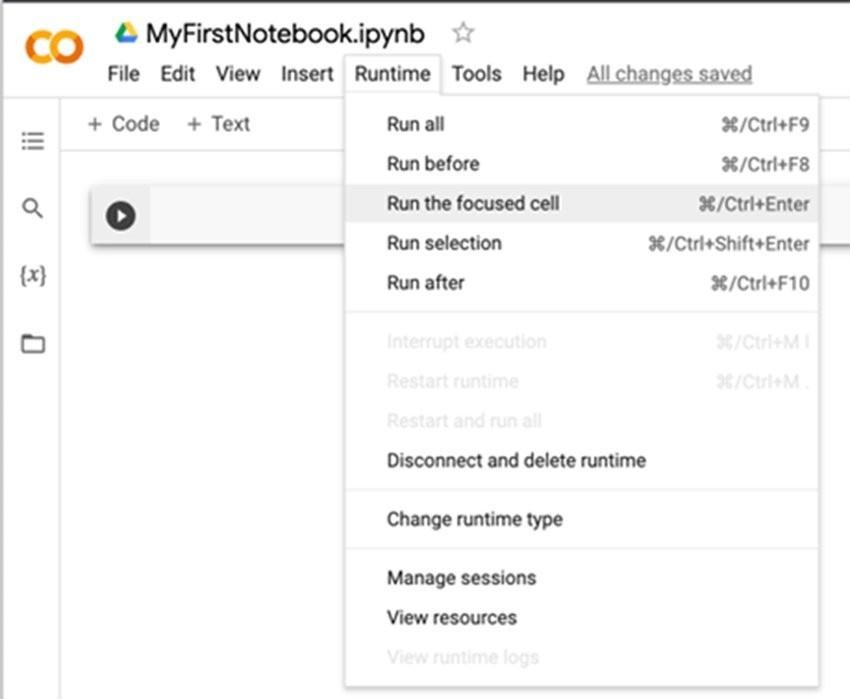


Because notebooks are meant for sharing, there are accommodations throughout for structured documentation.

### Code, Debug, Repeat

You can insert Python code to execute in a code cell. The code can be entirely standalone or imported from various Python libraries.

A notebook can be treated as a rolling log of work, with earlier code snippets being no longer executed in favor of later ones, or treated as an evolving set of code blocks intended for ongoing execution. The Runtime menu offers execution options, such as Run all, Run before or Run the focused cell, to match either approach.



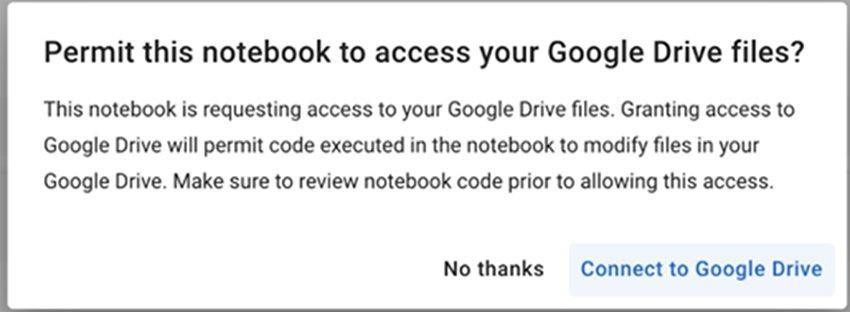
### Incorporating Data into the Notebook

After getting comfortable with the interface and using it for initial test coding, you must eventually provide the code with data to analyse or otherwise manipulate.

Colab can mount a user's Google Drive to the VM hosting their notebook using a code cell.



Once you hit run, Google will ask for permission to mount the drive.



If you allow it to connect, you will then have access to the files in your Google Drive via the /my\_drive path.

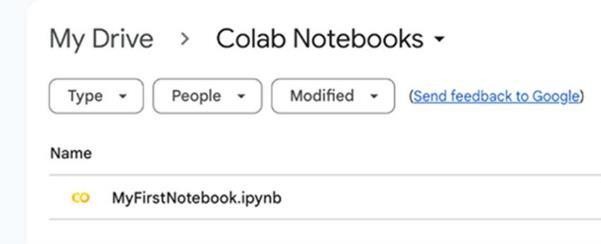
If you prefer not to grant access to your Drive space, you can upload files or any network file space mounted as a drive from your local machine instead.



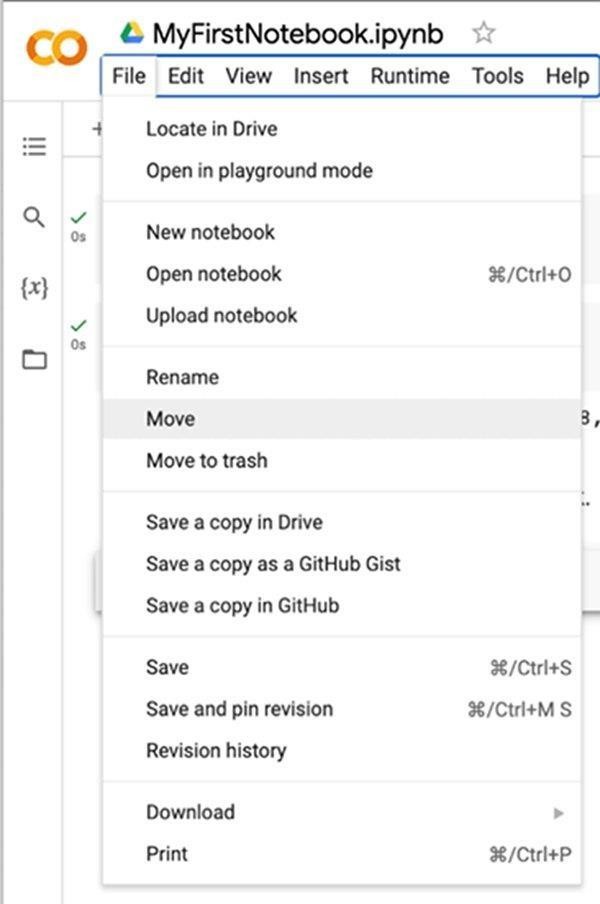
With file access, many functions are available to read data in various ways. For example, importing the Pandas library gives access to functions such as read\_csv and read\_json.

### Save and Share

By default, Colab puts notebooks in a Colab Notebooks folder under My Drive in Google Drive.



The File menu enables notebooks to be saved as named revisions in the version history, relocated using Move, or saved as a copy in Drive or GitHub. It also allows you to download and upload notebooks. Tools based on Jupyter provide broad compatibilities, so you can create notebooks in one place and then upload and use them in another.



You can use the Share button in the upper right to grant other Google users access to the notebook and to copy links.

Google also [provides](https://colab.google/notebooks) example notebooks illustrating available resources, such as pre-trained image classifiers and language transformers, as well as addressing common business problems, such as working with BigQuery or performing time series analytics. It also provides links to introductory Python coding notebooks.

# CHAPTER 3

## MODEL ANALYSIS

### PROBLEM STATEMENT

The problem at the heart of this research project lies in the effective extraction, classification, and communication of sentiments from textual data, with the specific objectives of:

### Sentiment Analysis Accuracy

Developing a sentiment analysis model that can accurately classify sentiments within textual data This encompasses the correct identification of positive, negative, and neutral sentiments, as well as handling nuances such as sarcasm and negations.

### Ensemble Method Integration

Integrating ensemble methods, including Support Vector Machines (SVM), decision trees, and naive Bayes, to enhance the precision and robustness of sentiment classification The challenge is to create an ensemble that leverages the strengths of these individual methods and produces superior sentiment predictions.

### Data Visualization for Insightful Communication

Utilizing data visualization techniques to effectively convey the patterns, trends, and insights derived from the sentiment analysis The aim is to ensure that complex sentiment analysis results are communicated in a comprehensible and engaging manner.

### Real-World Application

Applying the developed sentiment analysis model to the Uber Review dataset, a real-world corpus of diverse user-generated content The model should effectively compute sentiment scores, categorize the data into distinct labels, and provide valuable insights for practical applications.

### CHALLENGES AND LIMITATIONS IN EXISTING SYSTEMS FOR SENTIMENT ANALYSIS

**Ambiguity and Contextual Understanding**

Existing sentiment analysis systems struggle with disambiguating words or phrases that can carry multiple meanings based on context. The inability to comprehend the context effectively hampers accurate sentiment classification.

### Handling Sarcasm and Irony

Sentiment analysis models often falter in recognizing and correctly classifying sarcastic or ironic statements. The reliance on literal interpretations can lead to misclassification.

### Negations and Modifiers

Existing systems may not adequately handle negations (e.g., "not good") and sentiment modifiers (e.g., "extremely happy" vs. "slightly happy"), resulting in sentiment misclassification.

### Subjectivity and Multimodal Data

Sentiment analysis usually focuses on textual data, but emotions and sentiments are expressed through multiple modalities, including images and videos. Existing systems have limitations in analysing and integrating these diverse data types.

### Cross-Linguistic Challenges

Language and cultural differences pose substantial challenges. Many existing systems are designed for English, and adapting them to other languages and cultures remains an obstacle.

### Data Quality and Noise

User-generated content often contains errors, slang, and colloquialisms. Ensuring high-quality data for training and testing sentiment analysis models is a persistent issue.

### Lack of Contextual Understanding

Current sentiment analysis systems may not fully capture the broader context of a document or conversation. Understanding the overall context is crucial for accurate sentiment interpretation.

### Bias and Fairness

Sentiment analysis models may inherit biases present in their training data. Addressing bias and ensuring fairness in sentiment analysis is a critical concern.

### Real-Time Analysis

Many systems lack real-time capabilities, limiting their utility in applications where rapid sentiment analysis is essential.

### PROPOSED MODEL

The proposed Model aims to overcome the challenges and limitations of existing sentiment analysis methods by integrating ensemble learning techniques and dynamic data visualization. This system is designed to improve the accuracy of sentiment classification, adapt to evolving sentiment trends, and provide accessible, real-time insights. Here are the key components and features of the proposed Model

# CHAPTER 4

1. **MODEL DESIGN**

### MODULES

* + - Data Collection
    - Data Preprocessing
    - sentiment labeling
    - Vectorization
    - Model Training
    - Model Testing
    - model evaluation
    - model deployment

DATA COLLECTION

DATA PREPROCESSING

SENTIMENT LABELING

VECTORIZATION

MODEL TRAINING

MODEL TESTING

MODEL

### DATA COLLECTION

Collecting data from the Google Play Store for sentiment analysis is a common practice, especially when analysing user reviews of mobile applications. Here is an outline of the steps involved in data collection from the Google Play Store:

**Select the target mobile application:** Choose the mobile application for which you want to collect user reviews and perform sentiment analysis. It can be any app available on the Google Play Store.

**Web scraping or API access:** There are two primary methods for collecting data from the Google Play Store:

**Web Scraping:** Use web scraping tools, libraries, or frameworks (e.g., BeautifulSoup, Scrapy, Selenium) to extract user reviews and associated data from the app's Google Play Store page.

**API Access:** The Google Play Developer API provides an official way to programmatically access user reviews and other app-related data. You'll need to set up API access through the Google Play Console.

**Specify Data Fields:** Determine the specific data fields you want to collect. Typical data fields include the review text, rating, review date, and reviewer username. You might also consider collecting additional information like review titles or the number of upvotes or downvotes.

# Data-set description

**Data Set Name**: googlepay Customer Reviews.csv

**Source**: Google Play Store

**Purpose:** Sentiment Analysis

**Format:** CSV (Comma-Separated Values)

**Size:** 40000 rows × 6 columns(24000 datas)

# Attributes:

1."source" The source of the review.

2."review\_id" A unique identifier for each review.

3."user\_name" The username of the person who left the review.

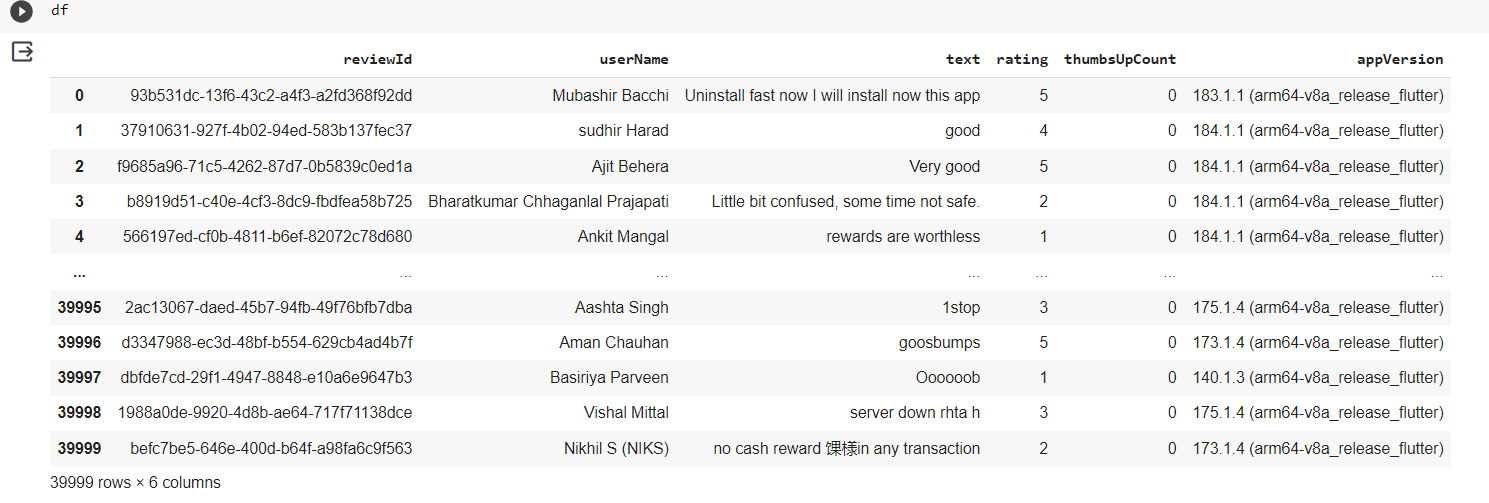
4." text " The main text of the review.

5."rating" The rating given by the user (e.g., 1 to 5 stars).

6."thumbs\_up" The number of thumbs-up (likes) the review received.

7."appVersion" The version of the Google pay app that the user was using when they left the review.

**Datasets attributes are shown an below**:

****

**Link**

<https://www.kaggle.com/>

**Data Collection Parameters:** When scraping reviews, specify parameters such as the number of reviews to collect, the sorting order (e.g., most recent or highest rated), and filters (e.g., reviews from a specific date range).

**Data Storage:** As you collect the data, store it in a structured format, such as a CSV file or a database. Ensure that you record the source of each review (i.e., the app it pertains to) to maintain data integrity.

**Handling User Agreements and Permissions:** Be aware of Google Play's terms of service and any legal requirements related to web scraping. Some methods of data collection may require permission from the app developer or store owner.

**Rate Limiting and Throttling:** If using web scraping, consider implementing rate limiting and throttling mechanisms to avoid overloading the Google Play Store servers, which can result in IP bans or other restrictions.

**Data Preprocessing:** After collecting the data, perform preprocessing tasks to clean and prepare it for sentiment analysis. This includes text normalization, the removal of special characters, and handling missing data.

**Anonymization and Privacy:** Ensure that you follow data privacy regulations and consider anonymizing user data to protect reviewers' privacy.

**Ethical Considerations:** Use the data collected for sentiment analysis responsibly and ethically. Ensure that you have the necessary rights and permissions to use the data for research or analysis purposes.

Remember that web scraping practices may change over time due to updates in website structure or changes in terms of service, so it's essential to stay informed about any relevant legal or technical developments. Additionally, always respect users' privacy and use the collected data for legitimate purposes.

### DATA PRE-PROCESSING

Text preprocessing is a crucial step in designing a sentiment analysis module. It involves cleaning and transforming the raw text data to prepare it for analysis. Here are the key text preprocessing steps I consider when designing my sentiment analysis module:

**Lowercase:** Convert all text to lowercase to ensure consistency in the text data.

**Tokenization:** Split the text into individual words or tokens. Tokenization helps in breaking down the text into smaller units for analysis.

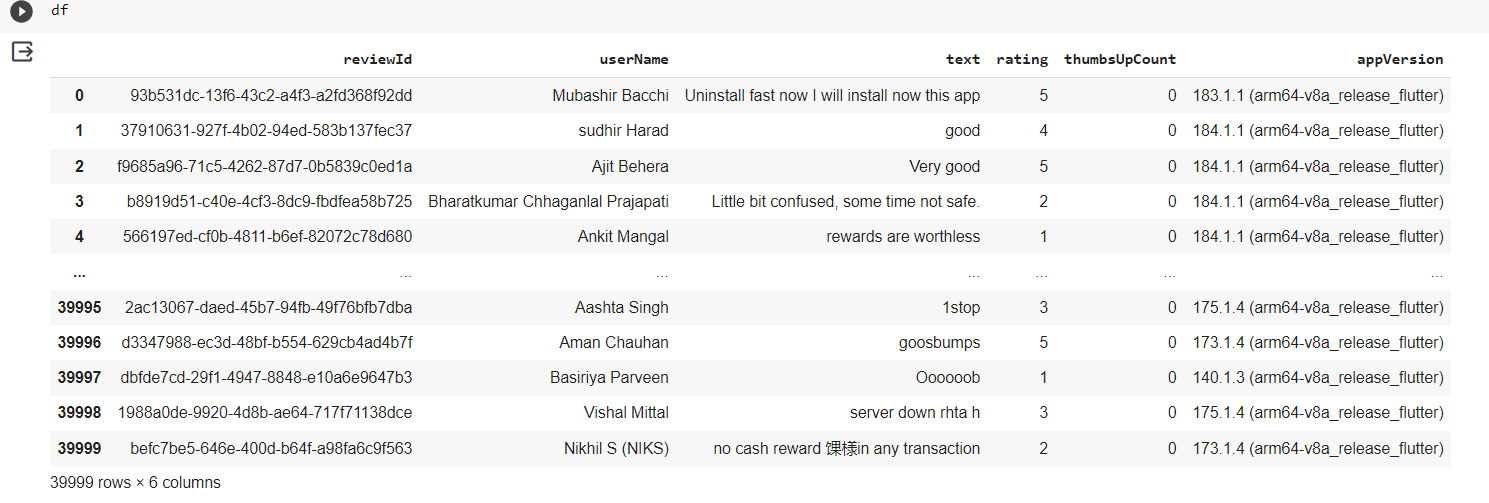
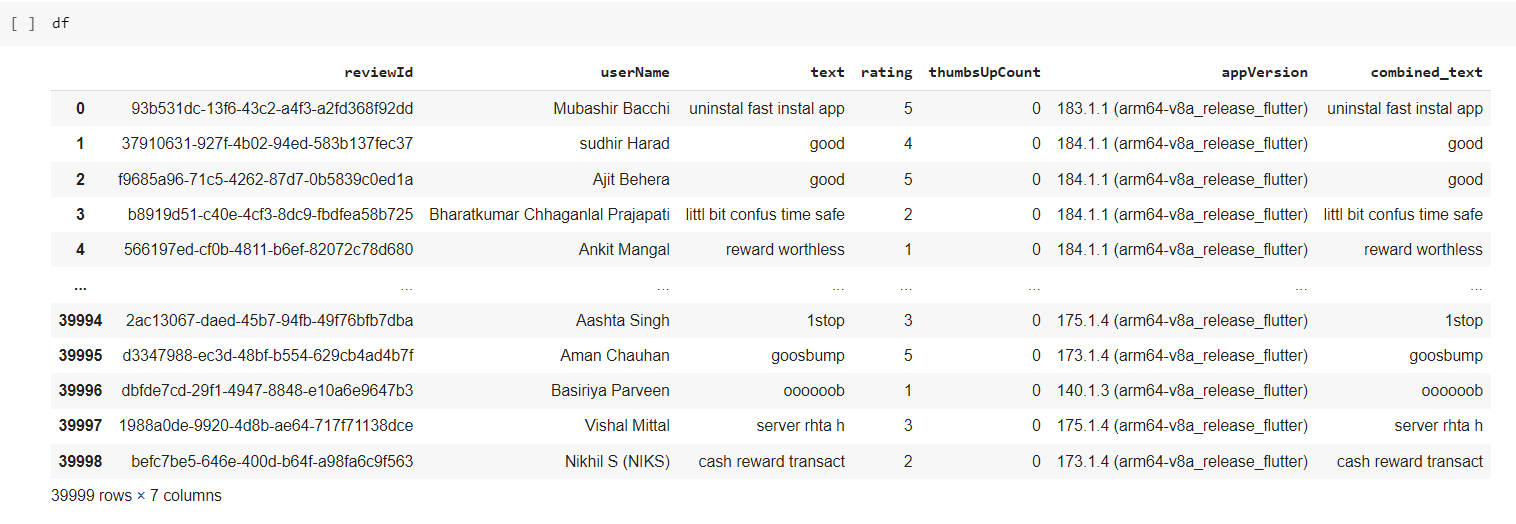
**Removing Punctuation:** Remove punctuation marks like commas, periods, and special characters. Punctuation often doesn't carry sentimental information and can be safely removed.

**Stop Word Removal:** Remove common stop words like "and," "the," "in," etc. These words are frequent but often do not contribute much to sentiment analysis.

**Stemming or Lemmatization:** Reduce words to their base or root form. For example, "running" and "ran" might be reduced to "run." You can choose between stemming (more aggressive) and lemmatization (more accurate but computationally expensive). Removing Numerical Values: If numerical values are not relevant to your sentiment analysis, consider removing them from the text.

**Removing HTML Tags:** If your text data contains HTML tags (common in web scraping), strip them from the text.

**Handling Emojis and Emoticons**: Emojis and emoticons can carry sentiment. Decide whether to remove, replace with text labels, or keep them in your text data.

**DATA SET ET BEFORE PRE-PROCESSING**

**DATA SET AFTER PRE-PROCESSING**



Input

Removing HTML tags

Lowercasing

Emoji handling

Stop word removal

Tokenization



Stemming

Spell checking



Out put

Non-English word removing

### SENTIMENT LABELING

Sentiment labeling using VADER (Valence-Aware Dictionary and Entiment Reasoner) is a valuable approach to quickly assessing and categorizing the sentiment of text data. VADER is a lexicon and rule-based sentiment analysis tool that is particularly useful for short text, social media content, and user reviews. Here's how you can label sentiments using VADER:

**Installation and Library Import:** Install the VADER library using pip and import it into your Python script.

**Initialization:** Create an instance of the VADER sentiment analyser by using

`SentimentIntensityAnalyzer()`.

**Sentiment Labeling:** For each text entry you want to label, follow these steps:

**Text Entry:** Input the text data you wish to analyse. VADER works well with short text, social media content, and user reviews.

**Sentiment Scores**: Use the VADER analyser to obtain sentiment scores for the text entry. VADER provides four scores: positive, negative, neutral, and a compound score representing the overall sentiment.

**Compound Score Categorization:** Categorize the text's sentiment based on the compound score. Common categorizations are:

Positive: Compound score > 0.1

Neutral: -0.1 >= Compound Score <= 0.1 Negative: Compound score < -0.1

**Display or Store the Result:** You can print or store the sentiment label for each text entry.

**Repetition:** Repeat this process for as many text entries as needed for sentiment analysis.

**Considerations:** Remember that while VADER is a valuable tool for quick sentiment labeling, it may not be perfectly accurate for every context. Complex or context-dependent sentiments may require additional analysis.

**Practical Application:** This approach can be applied to various use cases, including analysing user reviews, social media sentiment, or any text data where sentiment assessment is required.

Input text

Finding compound

Finding plolarity\_score

Sentiment labeling

Setting threshold value



Out put

By following these steps, you can efficiently label sentiments in your text data using the VADER sentiment analysis tool.

# CHAPTER 5

## SENTIMENT CLASSIFICATION

### VECTORIZATION

Term Frequency-Inverse Document Frequency (TF-IDF) vectorization is a text processing technique commonly used in natural language processing (NLP) and information retrieval. It's particularly useful for transforming text data into a numerical format suitable for machine learning models. TF-IDF vectorization assigns numerical values to each term (word) in a document to represent its importance in the context of a collection of documents (corpus). Here's an overview of TF-IDF vectorization

### Term Frequency (TF)

Term frequency measures how often a term appears in a document. It's calculated as the number of times a term appears in a document divided by the total number of terms in that document.

TF (t, d) = (Number of times term t appears in document d) / (Total number of terms in document d)

TF values are specific to each document and represent how important a term is within that document.

### Inverse Document Frequency (IDF)

IDF measures the importance of a term in the entire corpus. It's calculated as the logarithm of the total number of documents in the corpus divided by the number of documents containing the term, plus one.

IDF (t, D) = log ((total number of documents in the corpus D) / (number of documents containing term t in D)) + 1

IDF values are the same for all terms across the entire corpus.

### TF-IDF Score

The TF-IDF score for a term in a document is the product of its TF and IDF values. TF-IDF (t, d, D) = TF (t, d) \* IDF (t, D)

The TF-IDF score represents the importance of a term within a specific document relative to its importance in the entire corpus.

### Vectorization

After calculating the TF-IDF scores for each term in each document, you can represent each document as a vector where each dimension corresponds to a term in the corpus, and the value of each dimension is the TF-IDF score for the corresponding term in the document.

The result is a matrix where rows represent documents and columns represent terms, with each cell containing the TF-IDF score of a term in a document.

### Benefits of TF-IDF Vectorization

* TF-IDF vectorization is effective for converting text data into a numerical format suitable for machine learning models.
* It assigns higher values to terms that are important within a document and less important in the overall corpus.
* It helps capture the specificity of terms in individual documents, making it valuable for tasks like text classification and information retrieval.
* It reduces the dimensionality of the data, which can be beneficial for model training and efficiency.

### Use Cases

TF-IDF vectorization is commonly used in information retrieval systems, document clustering, text classification (including sentiment analysis), and search engines.

When working with text data for sentiment analysis, TF-IDF vectorization can be a valuable preprocessing step to convert textual information into a format that machine learning models can work with effectively. It helps to represent the text data in a way that preserves the importance of individual words or terms within the context of the entire dataset.

### MODEL

Selecting the right model for sentiment analysis is crucial, and the choice between Logistic Regression should be based on the characteristics of the data and the specific requirements of your project.

Machine learning is important because it gives enterprises a view of trends in customer behaviour and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google, and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies.

Machine learning Types

### Supervised Learning

In this type of machine learning, data scientists supply algorithms with labeled training data and define the variables they want the algorithm to assess for correlations. Both the input and the output of the algorithm is specified.

### Unsupervised Learning

This type of machine learning involves algorithms that train on unlabeled data. The algorithm scans through datasets looking for any meaningful connection. The data that algorithms train on as well as the predictions or recommendations they output are predetermined.

### Semi-supervised Learning

This approach to machine learning involves a mix of the two preceding types. Data scientists may feed an algorithm mostly labeled training data, but the model is free to explore the data on its own and develop its own understanding of the data set.

### Reinforcement Learning

Data scientists typically use reinforcement learning to teach a machine to complete a multi- step process for which there are clearly defined rules. Data scientists program an algorithm to complete a task and give it positive or negative cues as it works out how to complete a task. But for the most part, the algorithm decides on its own what steps to take along the way.

Supervised machine learning requires the data scientist to train the algorithm with both labeled inputs and desired outputs. Supervised learning algorithms are good for the following tasks:

**Binary Classification**: Dividing data into two categories

**Multi-class Classification**: Choosing between more than two types of answers.

**Regression Modelling**: Predicting continuous values.

**Ensembling:** Combining the predictions of multiple machine learning models to produce an accurate prediction.

### LOGISTIC REGRESSION

Logistic regression is a statistical method used for modelling the relationship between a binary dependent variable and one or more independent variables. It is a fundamental technique in the field of machine learning and statistics, frequently employed in various applications such as classification, prediction, and risk assessment. In this explanation, we will delve into the key concepts of logistic regression, how it works, and its applications in real-world scenarios.

### Introduction to Logistic Regression

Logistic regression is a type of regression analysis, but unlike linear regression, which is used to predict continuous outcomes, logistic regression is designed for binary outcomes, where the dependent variable is categorical with two possible values (e.g., 0 or 1, Yes or No, True or False). The goal of logistic regression is to model the probability that a given observation belongs to a particular category based on one or more predictor variables.

### How Logistic Regression Works

Data Preparation: To use logistic regression, you need a dataset with a binary dependent variable and one or more independent variables. The data should be cleaned and pre-processed to handle missing values and outliers.

Model Building: The logistic regression model is built by estimating the coefficients that relate the independent variables to the log-odds of the dependent variable. This is done through an iterative optimization process, typically using algorithms like gradient descent.

Model Evaluation: The performance of the logistic regression model is assessed using various metrics such as accuracy, precision, recall, F1 score, and the receiver operating characteristic

(ROC) curve. These metrics help you determine how well the model predicts the binary outcome.

### Applications of Logistic Regression

Logistic regression has a wide range of applications, including but not limited to:

Medical Diagnosis: It can be used to predict the likelihood of a patient having a disease based on their symptoms, medical history, and test results.

Credit Scoring: Logistic regression is used to assess the creditworthiness of individuals by predicting whether they are likely to default on loans.

Customer Churn Prediction: In industries like telecommunications, logistic regression helps identify customers who are likely to switch to a competitor's service.

Email Spam Detection: Logistic regression can classify emails as either spam or not spam based on various features of the email content.

Market Research: It's used to determine factors influencing consumer choices, such as the probability of buying a product based on price, advertising, and other variables.

### Considerations for Model Selection

Data Characteristics: Consider the nature of your text data. Is it highly structured, or does it contain complex language and nuances? This can guide your choice.

Resource Constraints: If you have limited computational resources, Naive Bayes is a lightweight and efficient option.

Accuracy Requirements: If high accuracy is a primary goal, Logistic Regression

Ensemble Methods: Consider whether you might want to explore ensemble methods in combination with these models to improve performance. Ensemble methods can often mitigate the weaknesses of individual models.

Data Preprocessing: Preprocessing your data, such as text normalization and feature engineering, can significantly impact model performance. Make sure to carry out these steps effectively.

Cross-Validation: Regardless of the model you choose, use cross-validation to assess its performance on your specific dataset. This helps you determine which model works best in your context.

In summary, Logistic Regression should be based on the nature of your data, your project's requirements, and your computational resources.

### MODEL TESTING

Model testing in the context of sentiment analysis is the process of evaluating the performance of your trained sentiment analysis model using a separate test dataset. This step is essential to assess how well your model generalizes unseen data and to ensure its accuracy and reliability in practical applications. Here's a step-by-step guide to conducting model testing:

### Prediction

Use the trained model to make predictions on the test dataset. For each text sample in the test dataset, the model will produce a predicted sentiment label (e.g., positive, negative, or neutral).

### Evaluation Metrics

Employ appropriate evaluation metrics to assess the model's performance. Common evaluation metrics for sentiment analysis include:

* Accuracy: the proportion of correctly classified instances.
* Precision: the number of true positive predictions divided by the total number of positive predictions.
* Recall: The number of true positive predictions divided by the total number of actual positive instances
* F1-Score: The harmonic mean of precision and recall, which balances precision and recall.
* Confusion Matrix: A table that displays the number of true positives, true negatives, false positives, and false negatives.

### Model Assessment

Analyse the results obtained from the evaluation metrics to determine how well the model is performing. Look for patterns and areas of improvement. For example, identify cases where the model struggled to classify sentiments correctly.

### Error Analysis

Examine the misclassified instances to gain insights into why the model made incorrect predictions. This analysis can guide further improvements or adjustments to the model.

Model testing is a critical step in the machine learning workflow to ensure that your sentiment analysis model is effective, reliable, and suitable for the intended use case. The results from testing provide valuable feedback for model improvement and future development.

True Positive: You predicted positive, and it’s true. True Negative: You predicted negative, and it’s true.

False Positive: (Type 1 Error): You predicted positive, and it’s false. False Negative: (Type 2 Error): You predicted negative, and it’s false.

In this module we test the trained machine learning model using the test dataset.

### Accuracy

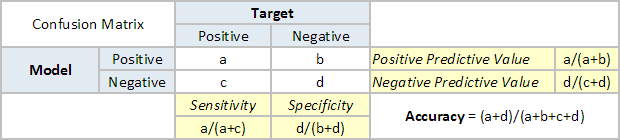


The most commonly used metric to judge a model is actually not a clear indicator of the performance. The worst happens when classes are imbalanced.

### Precision



Percentage of positive instances out of the total predicted positive instances.



### ENSEMBLE CLASSIFIER RESULTS Accuracy: 0.9614988411022405

**Classification Report:**

### Logistic Regression Performance Metrics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **precision** | **recall** | **f1-score** | **support** |
| **negative** | 0.92 | 0.82 | 0.87 | 712 |
| **neutral** | 0.92 | 0.98 | 0.95 | 2380 |
| **positive** | 0.99 | 0.97 | 0.98 | 4674 |

**Accuracy**

The accuracy of approximately 96.61% is a measure of how often the model correctly predicts the sentiment labels. It indicates that the majority of predictions are accurate.

### Classification Report

Precision: Precision measures the accuracy of positive predictions made by the model. In this case, it's quite high for all three classes (negative, neutral, and positive), indicating that when the model predicts a specific sentiment label, it is often correct.

Recall: Recall measures how many of the actual positive instances the model correctly identifies. The recall values are relatively high for all three classes, indicating that the model is effective at capturing actual positive instances.

F1-Score: The F1-score is the harmonic mean of precision and recall and provides a balance between the two. The F1-scores are also high for all three classes, indicating a good balance between precision and recall.

Support: The "support" column indicates the number of instances in each class in your test dataset.

### Macro Avg and Weighted Avg

The "macro avg" represents the average of precision, recall, and F1-score calculated for each class independently. In your case, the macro average is around 0.93, indicating good overall performance.

The "weighted average" takes into account class imbalances by considering the weighted average of precision, recall, and F1-score based on the number of instances in each class. The weighted average is approximately 0.96, showing that the model performs well while considering class distribution.

These results suggest that the ensemble classifier is effective at classifying sentiment across different classes (negative, neutral, and positive). The high precision, recall, and F1-scores demonstrate a good balance between correctly identifying each sentiment class and minimizing false positives and false negatives. The high accuracy indicates that the model is making correct predictions in the majority of cases.

Overall, the sentiment analysis model appears to be performing well and can be considered a reliable tool for sentiment classification. However, it's essential to consider the specific requirements and constraints of your application to determine if this level of performance is suitable for your needs.

# CHAPTER 6

**6 SOURCE CODE**

### 6.1 CODING

The coding phase brings the actual system into action by converting the design of the system into code in a given programming language. Therefore, a good coding style has to be taken whenever changes are required and easily screwed into the system.

### CODING STANDARDS

Coding standards are guidelines for programming that focus on the physical structure and appearance of the program. They make the code easier to read, understand, and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standards needed to achieve the above-mentioned objectives are as follows:

The program should be simple, clear, and easy to understand.

* Naming conventions
* Value conventions
* Script and comment procedure
* Message box format
* Exception and error handle

**Mini-Project**

### PROJECT TITLE

SENTIMENT ANALYSIS ON GOOGLE PAY REVIEW



Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).

## TOOLS & TECHNIQUES

1**.Python** Python progamming language as the primary tool for data perprocessing,analysis.

**2.NLTK(Natural Language Toolkit)Libary**:NLTK will be used for various natural language prooessing tasks,including text preprocessing,tokenization,and potentially for sentiment analysis.

**3.VADER Sentiment Analysis**: VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool. It will help me calculate sentiment scores for the googlepay reviews.

4.**Logistic Regression**: This type of statistical model (also known as logit model) is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, such as voted or didn’t vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1

## DATA-SET DESCRIPTION

**Data Set Name**: googlepay Customer Reviews.csv

**Source**: Google Play Store

**Purpose**: Sentiment Analysis

**Format**: CSV (Comma-Separated Values)

**Size**:40000 rows × 6 columns(24000 datas)

[5]:

## Columns:

1. “source” The source of the review. 2.“review\_id” A unique identifier for each review

3.“user\_name” The username of the person who left the review. 4.“content” The main text of the review.

5.“rating” The rating given by the user (e.g., 1 to 5 stars). 6.“thumbs\_up” The number of thumbs-up (likes) the review received.

1. “appVersion” The version of the Google pay app that the user was using when they left the review.



df=pd.read\_csv("/content/drive/MyDrive/Data/googlepay Customer Reviews.csv")

[6]:



[6]: (40000, 6)

[7]:



* 1. : reviewId object

userName object

text object

rating int64

thumbsUpCount int64 appVersion object dtype: object

[8] :



* 1. : reviewId 0

userName 0

text 1

rating 0

thumbsUpCount 0

appVersion 3758

dtype: int64

[9] :



* 1. : rating thumbsUpCount count 40000.000000 40000.000000 mean 3.870850 0.637000

std 1.600542 55.511205

min 1.000000 0.000000

25% 3.000000 0.000000

50% 5.000000 0.000000

75% 5.000000 0.000000

max 5.000000 11020.000000

[10] :



df = pd.read\_csv("/content/drive/MyDrive/Data/googlepay Customer Reviews.csv")

df.to\_csv("/content/drive/MyDrive/Data/preprocess1\_reviews.csv", index=**False**)

[11] :

[11] :



reviewId userName \

0 93b531dc-13f6-43c2-a4f3-a2fd368f92dd Mubashir Bacchi

1 37910631-927f-4b02-94ed-583b137fec37 sudhir Harad

2 f9685a96-71c5-4262-87d7-0b5839c0ed1a Ajit Behera

3 b8919d51-c40e-4cf3-8dc9-fbdfea58b725 Bharatkumar Chhaganlal Prajapati

4 566197ed-cf0b-4811-b6ef-82072c78d680 Ankit Mangal

… … …

39995 2ac13067-daed-45b7-94fb-49f76bfb7dba Aashta Singh

39996 d3347988-ec3d-48bf-b554-629cb4ad4b7f Aman Chauhan

39997 dbfde7cd-29f1-4947-8848-e10a6e9647b3 Basiriya Parveen

39998 1988a0de-9920-4d8b-ae64-717f71138dce Vishal Mittal

39999 befc7be5-646e-400d-b64f-a98fa6c9f563 Nikhil S (NIKS)

text rating thumbsUpCount\

1. Uninstall fast now I will install now this app 5 0
2. good 4 0
3. Very good 5 0
4. Little bit confused, some time not safe. 2 0
5. rewards are worthless 1 0

… … … …

39995 1stop 3 0

183.1.1 (arm64-v8a\_release\_flutter)

|  |  |  |  |
| --- | --- | --- | --- |
| 39996  39997 | goosbumps Oooooob | 5  1 | 0  0 |
| 39998  39999 | server down rhta h no cash reward in any transaction | 3  2 | 0  0 |
| 0 | appVersion |  |  |
| 1 | 184.1.1 (arm64-v8a\_release\_flutter) |  |  |
| 2 | 184.1.1 (arm64-v8a\_release\_flutter) |  |  |
| 3 | 184.1.1 (arm64-v8a\_release\_flutter) |  |  |
| 4  … | 184.1.1 (arm64-v8a\_release\_flutter) |  |  |
| 39995  39996 | 175.1.4 (arm64-v8a\_release\_flutter) |  |  |
| 39997 | 140.1.3 (arm64-v8a\_release\_flutter) |  |  |
| 39998 | 175.1.4 (arm64-v8a\_release\_flutter) |  |  |
| 39999 | 173.1.4 (arm64-v8a\_release\_flutter) |  |  |

…

173.1.4 (arm64-v8a\_release\_flutter)

[39999 rows x 6 column

[12] :

[12] : reviewId 0

userName 0

text 0

rating 0

thumbsUpCount 0

appVersion 3758

dtype: int64

[13] :

[14] :



Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)

Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)

Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.3.2)

Requirement already satisfied: regex>=2021.8.3 in

/usr/local/lib/python3.10/dist-packages (from nltk) (2023.6.3)

Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.1)



Collecting emoji

Downloading emoji-2.8.0-py2.py3-none-any.whl (358 kB)

358.9/358.9

kB 5.2 MB/s eta 0:00:00

Installing collected packages: emoji

Successfully installed emoji-2.8.0

[15] :

**import pandas as pd import nltk**

**import re**

**from nltk.corpus import** stopwords **from nltk.stem import** PorterStemmer **from nltk.tokenize import** word\_tokenize **import emoji**

*# Download NLTK data* nltk.download('punkt') nltk.download('stopwords')

*# Load the CSV file*

csv\_file = "/content/drive/MyDrive/Data/preprocess1\_reviews.csv" df = pd.read\_csv(csv\_file)

*#Define a function for text preprocessing*

**def** preprocess\_text(text):

*# Check for NaN and empty strings*

**if** pd.isna(text) **or** text == "":

**return** ""

*# Step 1: Lowercasing*

text = text.lower()

*# Step 2: Removing HTML Tags*

text = re.sub(r'<.\*?>', '', text)

*# Step 3: Tokenization and Symbol Removal*

tokens = [re.sub(r'[^a-zA-Z0-9]', '', word) **for** word **in** word\_tokenize(text)]

*# Step 4: Stop Word Removal*

stop\_words = set(stopwords.words('english'))

tokens = [word **for** word **in** tokens **if** word **not in** stop\_words]

*# Step 5: Handling Emojis and Emoticons*

tokens = [emoji.demojize(word) **for** word **in** tokens]

*# Step 6: Stemming*

stemmer = PorterStemmer()

tokens = [stemmer.stem(word) **for** word **in** tokens]

*# Rejoin the processed tokens into a single string*

**return** ' '.join(tokens)

*# Specify the columns containing text data to preprocess*

columns\_to\_preprocess = ['text']

*# Apply the preprocessing function to the specified columns*

**for** col **in** columns\_to\_preprocess:

df[col] = df[col].apply(preprocess\_text)

*# Combine the preprocessed text columns into a single text column*

df['combined\_text'] = df[columns\_to\_preprocess].apply(**lambda** x: ' '.join(x),␣

𝗌axis=1)

*# Save the preprocessed data to a new CSV file if needed*

df.to\_csv("/content/drive/MyDrive/Data/text\_preprocessed.csv", index=**False**)

[nltk\_data] Downloading package punkt to /root/nltk\_data…

[nltk\_data] Unzipping tokenizers/punkt.zip.

[nltk\_data] Downloading package stopwords to /root/nltk\_data…

[nltk\_data] Unzipping corpora/stopwords.zip.

[



[17] :

**import pandas as pd**

**import matplotlib.pyplot as plt**

*# Load the preprocessed data*

csv\_file = "/content/drive/MyDrive/Data/text\_preprocessed.csv" df = pd.read\_csv(csv\_file)

*# Handle NaN values in the 'combined\_text' column and convert them to empty*␣

𝗌*strings*

df['combined\_text'] = df['combined\_text'].fillna('')

*# Combine preprocessed text from all reviews into a single string*

all\_reviews\_text = ' '.join(df['combined\_text'])

*# Tokenize the text into words*

words = all\_reviews\_text.split()

*# Create a frequency distribution of words*

word\_freq = pd.Series(words).value\_counts()

*# Set the number of top keywords you want to display in the bar chart*

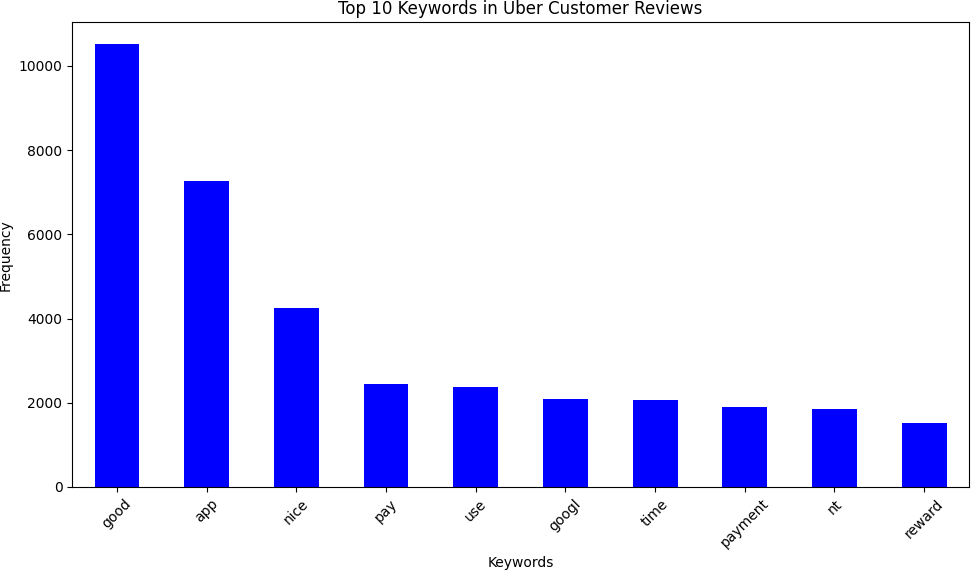
top\_n\_keywords = 10 *# Adjust this number as needed*

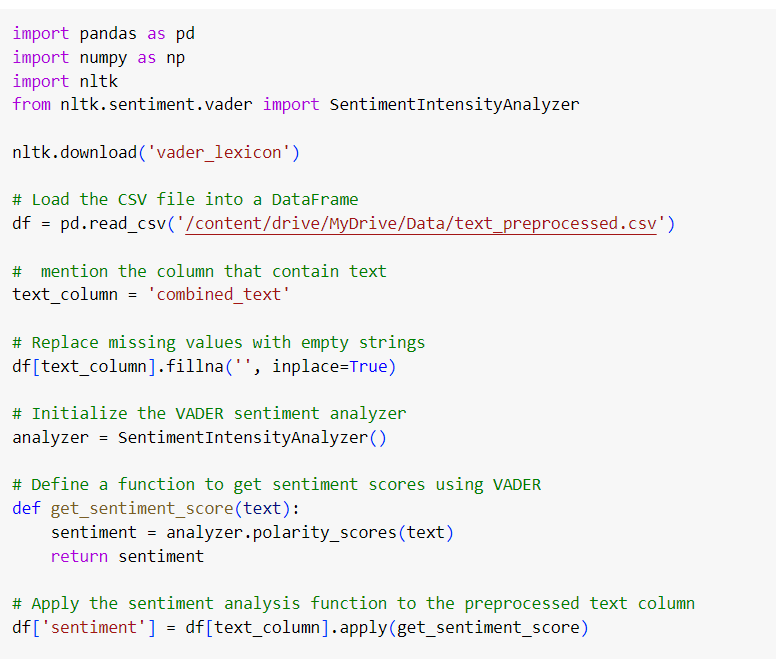
*# Get the top keywords*

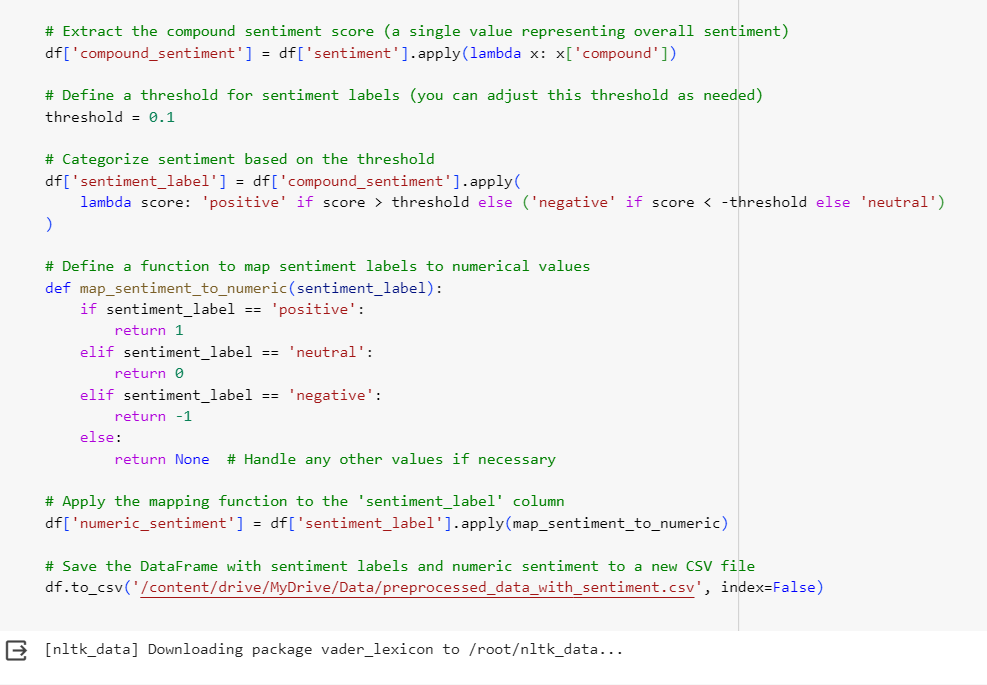
top\_keywords = word\_freq.head(top\_n\_keywords)

*# Create a bar chart*





[18





Mubashir Bacchi Ajit Behera Ankit Mangal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [19]: | 0  1  2  3  4  … | reviewId 93b531dc-13f6-43c2-a4f3-a2fd368f92dd 37910631-927f-4b02-94ed-583b137fec37 f9685a96-71c5-4262-87d7-0b5839c0ed1a  b8919d51-c40e-4cf3-8dc9-fbdfea58b725 566197ed-cf0b-4811-b6ef-82072c78d680  … | userName sudhir Harad  Bharatkumar Chhaganlal Prajapati  … | \ |
|  | 39994 | 2ac13067-daed-45b7-94fb-49f76bfb7dba | Aashta Singh |  |
|  | 39995 | d3347988-ec3d-48bf-b554-629cb4ad4b7f | Aman Chauhan |  |
|  | 39996  39997  39998 | dbfde7cd-29f1-4947-8848-e10a6e9647b3 1988a0de-9920-4d8b-ae64-717f71138dce befc7be5-646e-400d-b64f-a98fa6c9f563 | Basiriya Parveen  Nikhil S (NIKS) |  |
|  | 0  1 | text rating uninstal fast instal app 5  good 4 | thumbsUpCount \  0 |  |
|  | 2 | good 5 | 0 |  |
|  | 3  4  …  39994 | littl bit confus time safe 2  reward worthless 1  … …  1stop 3 | 0  … |  |
|  | 39995  39996 | goosbump 5  oooooob 1 | 0 |  |
|  | 39997  39998 | server rhta h 3  cash reward transact 2 | 0 |  |

Vishal Mittal

0

0

0

0

0

appVersion combined\_text \

* 1. 183.1.1 (arm64-v8a\_release\_flutter) uninstal fast instal app
  2. 184.1.1 (arm64-v8a\_release\_flutter) good
  3. 184.1.1 (arm64-v8a\_release\_flutter) good
  4. 184.1.1 (arm64-v8a\_release\_flutter) littl bit confus time safe

4

…

39994

184.1.1

175.1.4

(arm64-v8a\_release\_flutter)

…

(arm64-v8a\_release\_flutter)

reward worthless

…

1stop

39995 173.1.4 (arm64-v8a\_release\_flutter) goosbump

39996 140.1.3 (arm64-v8a\_release\_flutter) oooooob

39997 175.1.4 (arm64-v8a\_release\_flutter) server rhta h

39998 173.1.4 (arm64-v8a\_release\_flutter) cash reward transact

sentiment compound\_sentiment \ 0 {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound… 0.0000

1 {'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound… 0.4404

2 {'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound… 0.4404

3 {'neg': 0.0, 'neu': 0.58, 'pos': 0.42, 'compou… 0.4404

4

…

39994

{'neg':

{'neg':

0.439, 'neu': 0.0, 'pos': 0.561, 'comp…

…

0.0, 'neu': 1.0, 'pos': 0.0, 'compound…

…

[39999 rows x 11 columns]

[20] :



sentimnumeric\_

**from pandas.core.series import** Frequency

**import pandas as pd**

**import matplotlib.pyplot as plt**

*# Load the preprocessed data with sentiment labels*

csv\_file = "/content/drive/MyDrive/Data/preprocessed\_data\_with\_sentiment.csv" df = pd.read\_csv(csv\_file)

*# Count the number of reviews in each sentiment category*

sentiment\_counts = df['sentiment\_label'].value\_counts()

*# Create a bar chart*

plt.figure(figsize=(8, 6))

ax = sentiment\_counts.plot(kind='bar', color='black')

*# Set chart title and labels*

plt.title('Sentiment Distribution in Google Pay Customer Reviews') plt.xlabel('Sentiment Labels')

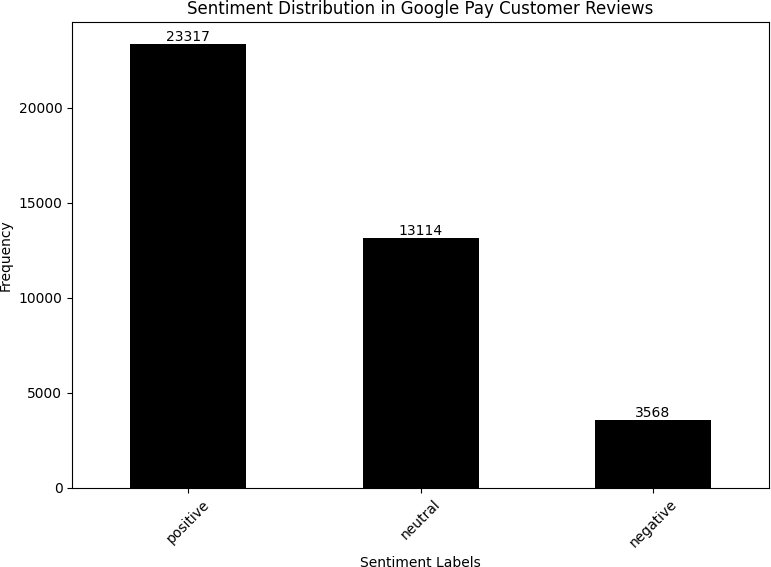
plt.ylabel('Frequency')

*# Rotate x-axis labels for better readability*

plt.xticks(rotation=45)

*# Ensure that the labels fit within the figure area*

plt.tight\_layout()





*# Split the data into training and testing sets*

X = df['combined\_text']

y = df['numeric\_sentiment']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2,␣

𝗌random\_state=42)

*# Create a TF-IDF vectorizer for text data*

tfidf\_vectorizer = TfidfVectorizer()

X\_train\_tfidf = tfidf\_vectorizer.fit\_transform(X\_train) X\_test\_tfidf = tfidf\_vectorizer.transform(X\_test)

*#Initialize and train the Logistic Regression model* logistic\_reg\_model = LogisticRegression() logistic\_reg\_model.fit(X\_train\_tfidf, y\_train)

*# Predict sentiment on the test data*

y\_pred = logistic\_reg\_model.predict(X\_test\_tfidf)

*# Evaluate the model*

print(classification\_report(y\_test, y\_pred))

0 0.92 0.98 0.95 2380

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | f1-score | support |
| -1 | 0.92 | 0.82 | 0.87 | 712 |

1 0.99 0.97 0.98 4674

accuracy 0.96 7766

macro avg 0.94 0.93 0.93 7766

weighted avg 0.96 0.96 0.96 7766

[25]:



/usr/local/lib/python3.10/dist-packages/sklearn/linear\_model/\_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):

STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in: https://scikit-learn.org/stable/modules/preprocessing.html

Please also refer to the documentation for alternative solver options: https://scikit-learn.org/stable/modules/linear\_model.html#logistic-

regression

n\_iter\_i = \_check\_optimize\_result(

[25]: 0.9614988411022405

# CHAPTER 7

## CONCLUSION:

The project titled "Sentiment Analysis on Google Pay Reviews" involved the analysis of customer reviews from Google Play Store for the Google Pay app. This analysis aimed to determine the sentiment of the reviews and gain insights into the opinions and experiences of users. Here's a summary of the key steps and findings:

### Data Collection and Preprocessing

The dataset, named "googlepay Customer Reviews.csv," was obtained from Google Play Store. It contains 40,000 rows and 6 columns, with information such as review source, review ID, user name, review text, rating, and thumbs-up count. Data preprocessing involved addressing missing values, particularly in the 'text' and 'appVersion' columns. Rows with missing 'text' values were dropped, resulting in a dataset with 40,000 rows.

The text data in the 'text' column underwent several preprocessing steps, including lowercase conversion, HTML tag removal, tokenization, symbol removal, stop word removal, emoji handling, and stemming. The preprocessed text was then combined into a new column named 'combined\_text.'

### Exploratory Data Analysis

A word cloud was created to visualize the most frequent words in the preprocessed reviews. This gave an overview of the prominent themes and terms in the reviews.

A bar chart displayed the top keywords in the reviews, providing insights into the most frequently mentioned terms in the dataset.

### Sentiment Analysis

Sentiment analysis was performed using the VADER (Valence Aware Dictionary and sentiment Reasoner) tool, a lexicon and rule-based sentiment analysis tool. It calculates sentiment scores for each review.

The sentiment analysis process resulted in a 'compound\_sentiment' score, which represented the overall sentiment of each review. Sentiment labels ('positive,' 'negative,' or 'neutral') were assigned based on a predefined threshold.

The sentiment labels were mapped to numerical values to facilitate machine learning model training.

### Machine Learning Model

A machine learning model was built to predict sentiment based on the preprocessed text data. The model used logistic regression, a classification algorithm.

The data was split into training and testing sets, and a TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer was used to convert text data into numerical features.

The logistic regression model was trained on the training data and evaluated on the test data. The evaluation metrics included precision, recall, F1-score, and accuracy.

The model achieved an accuracy score of approximately 96.15% on the test data, indicating strong performance in predicting sentiment

In conclusion, the sentiment analysis on Google Pay reviews revealed that the majority of the reviews were positive or neutral, with very few negative sentiments. Users seem to have a favorable opinion of the Google Pay app based on the analysis.

The machine learning model demonstrated high accuracy in classifying sentiment, which could be valuable for automating the analysis of a large volume of reviews. This project provides valuable insights for Google Pay app developers and can help them make data-driven decisions to improve user satisfaction and address potential issues

**Future Scope**

**Enhancing Data Preprocessing**

You can explore more advanced techniques for data preprocessing, such as handling misspelled words, expanding contractions, and dealing with slang and abbreviations. This will help improve the quality of the data and, consequently, the sentiment analysis results.

**Fine-Tuning Sentiment Analysis Model**

You can experiment with different machine learning models, such as support vector machines, random forests, or deep learning models, to see if they can provide better sentiment analysis performance compared to logistic regression.

**Aspect-Based Sentiment Analysis**

Extend the project to perform aspect-based sentiment analysis, where you analyze not only the overall sentiment of reviews but also sentiments related to specific aspects or features of the Google Pay app (e.g., user interface, payment reliability, customer support). This can provide more detailed insights for app improvement.

**Leveraging Deep Learning**

You can explore deep learning techniques, such as recurrent neural networks (RNNs) or transformers (e.g., BERT), for sentiment analysis. These models have shown remarkable performance in natural language processing tasks.

**Real-Time Sentiment Analysis**

Create a real-time sentiment analysis system that can continuously analyze incoming Google Pay reviews from the Google Play Store. This would allow you to monitor user sentiment in real-time and respond promptly to issues or concerns.

**Sentiment Visualization**

Develop visualizations to present the sentiment analysis results in a more user-friendly and informative manner. Word clouds, sentiment trend charts, and interactive dashboards can help stakeholders understand **Multilingual Sentiment Analysis**

If Google Pay has a global user base, consider expanding the sentiment analysis to cover multiple languages. This requires using language-specific models and sentiment lexicons.

**Feedback Categorization**

Apart from sentiment, categorize reviews into topics or themes. For example, identify common topics like "payment issues," "user experience," or "security concerns." This can help prioritize and address specific areas of improvement.

**Sentiment Integration**

Integrate sentiment analysis results into Google Pay's feedback and improvement pipeline. This would enable the development team to automatically prioritize and categorize reviews for further action.

**A/B Testing**

Implement A/B testing based on user feedback and app improvements to measure the impact of changes on user sentiment and overall app ratings.

**User Engagement**

Consider conducting user surveys or feedback solicitation within the app to gather more structured and detailed feedback from users, which can complement the sentiment analysis of reviews.

**Social Media Sentiment Analysis**

Extend sentiment analysis to social media platforms where users discuss Google Pay. This can provide insights into user sentiment outside of app store reviews.

**BIBLIOGRAPHY**

**Online Tutorial Reference’s:**

Javatpoint sentiment-analysis-in-python, References

<https://www.javatpoint.com/sentiment-analysis-in-python>

GeeksforGeeks create-a-simple-sentiment-analysis

https://www.geeksforgeeks.org/create-a-simple-sentiment-analysis

[kaggle nlp-sentiment-analysis-in-python](https://www.kaggle.com/code/valmetisrinivas/nlp-sentiment-analysis-in-python)

<https://www.kaggle.com/nlp-sentiment-analysis-in-python>

[Tutorialspoint python text processing, python sentiment analysis](https://www.tutorialspoint.com/python_text_processing/python_sentiment_analysis.htm)

<https://www.tutorialspoint.com/python_text_processing/python_sentiment_analysis.htm>

Python programming.net sentiment analysis module nltk

<https://pythonprogramming.net/sentiment-analysis-module-nltk-tutorial/>

**Books :**

[Dipanjan Sarkar](https://www.google.co.in/search?sca_esv=579769985&hl=en&q=inauthor:%22Dipanjan+Sarkar%22&tbm=bks) . “Text Analytics with Python”, 2nd ed, 2016

[Vivian Siahaan](https://www.google.co.in/search?sca_esv=579769985&hl=en&q=inauthor:%22Vivian+Siahaan%22&tbm=bks), [Rismon Hasiholan Sianipar](https://www.google.co.in/search?sca_esv=579769985&hl=en&q=inauthor:%22Rismon+Hasiholan+Sianipar%22&tbm=bks) .“TEXT PROCESSING AND SENTIMENT ANALYSIS USING MACHINE LEARNING AND DEEP LEARNING WITH PYTHON GUI” , 1 st ed, 2023

[Bing Liu](https://www.google.co.in/search?hl=en&gbpv=1&dq=book+for+python+for+sentiment+analysis&printsec=frontcover&q=inauthor:%22Bing+Liu%22&tbm=bks&sa=X&ved=2ahUKEwiJgv_5ta-CAxVja2wGHSDgAngQmxMoAHoECAgQAg) . “Sentiment Analysis Mining Opinions, Sentiments, and Emotions”, 2 nd ed, 2020

[Jalaj Thanaki](https://www.google.co.in/search?hl=en&gbpv=1&dq=book+for+nltk&printsec=frontcover&q=inauthor:%22Jalaj+Thanaki%22&tbm=bks&sa=X&ved=2ahUKEwj8m7Xvt6-CAxXGcmwGHTvXDbwQmxMoAHoECA4QAg) . “Python Natural Language Processing” ,2nd ed, 2017