INSTITUTE OF SCIENCE, TECHNOLOGY & ADVANCED STUDIES (VISTAS)
(Deemed to be University Estd. u/s 3 of the UGC Act, 1956)
PALLAVARAM - CHENNAI
ACCREDITED BY NAAC WITH 'A' GRADE
Marching Beyond 30 Years Successfully
INSTITUTION WITH UGC 12B STATUS

ENSEMBLE LEARNING FOR SENTIMENT ANALYSIS

A Mini project Report

Submitted for the Partial fulfillment for the award of degree of

MASTER OF SCIENCE

IN

DATA SCIENCE AND BUSINESS ANALYTICS

NAME : DEEPAN CHAKARAVARTHI J

REGISTER NO : 22235106

c

Under the guidance of

DR.K.KASTURI

School of Computing Sciences

Department of Information Technology (BCA & IT)

VELS

INSTITUTE OF SCIENCE, TECHNOLOGY & ADVANCED STUDIES (VISTAS)
(Deemed to be University Estd. u/s 3 of the UGC Act, 1956)
PALLAVARAM - CHENNAI
ACCREDITED BY NAAC WITH 'A' GRADE
Marching Beyond 30 Years Successfully
INSTITUTION WITH UGC 12B STATUS

This is to certify that the project titled "**ENSEMBLE LEARNING FOR SENTIMENT ANALYSIS"** is the original record work done by **DEEPAN CHAKARAVARTHI J 22235106**, under my guidance and supervision for the partial fulfillment of award of degree in Master of Science in data science and business analytics(Msc 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: 1.

INSTITUTE OF SCIENCE, TECHNOLOGY & ADVANCED STUDIES (VISTAS)
(Deemed to be University Estd. u/s 3 of the UGC Act, 1956)
PALLAVARAM - CHENNAI
ACCREDITED BY NAAC WITH 'A' GRADE
Marching Beyond 30 Years Successfully
INSTITUTION WITH UGC 12B STATUS

DECLARATION

I, **DEEPAN CHAKARAVARTHI J (REGISTER NO. 22235106)**, declares that the project entitled "**ENSEMBLE LEARNING FOR SENTIMENT ANALYSIS**" submitted by me during the period from **2021-2023** under the guidance **DR.K.KASTURI** , 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:

INSTITUTE OF SCIENCE, TECHNOLOGY & ADVANCED STUDIES (VISTAS)
(Deemed to be University Estd. u/s 3 of the UGC Act, 1956)
PALLAVARAM - CHENNAI
ACCREDITED BY NAAC WITH 'A' GRADE
Marching Beyond 30 Years Successfully
INSTITUTION WITH UGC 12B STATUS

BONAFIDE CERTIFICATE

I certify that the project titled "**ENSEMBLE LEARNING FOR SENTIMENT ANALYSIS"** for the Master of Science in data science and business analytics(Msc ds&ba) done by **DEEPAN CHAKARAVARTHI J(REGISTER NO. 22235106)** is the project work carried out by her during the period from **2021-2023** 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/She 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. T. Kamala Kannan**, Associate Professor and **Head of the Department of Information Technology**, VIATAS for his continued support and motivation.

I profoundly thank my research supervisor **DR.K.KASTURI** for her 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

Deepan chakaravarthi j

TABLE OF CONTENTS

ABSTRACT

Sentiment Analysis, also referred to as Opinion Mining, represents a fundamental Natural Language Processing (NLP) task dedicated to discerning, extracting, and assessing sentiments, opinions, and emotional nuances expressed in textual data. The significance of this field has exponentially grown in recent years, primarily driven by the exponential surge in user-generated content across social media platforms, product evaluations, and customer feedback. Consequently, it has evolved into a crucial instrument for comprehending public sentiment and guiding decision-making processes.

The primary objective of this research endeavor is to offer an extensive and insightful exploration of the essential principles and methodologies intrinsic to sentiment analysis. The study's inception involves introducing the foundational elements of sentiment analysis, encompassing crucial aspects such as text preprocessing techniques, feature extraction methodologies, and the workings of sentiment classification algorithms.

In this project, we embark on a practical application of sentiment analysis, employing the Uber Review dataset as our foundation. Our overarching aim is to compute sentiment scores for the reviews and subsequently categorize the dataset into distinct sentiment labels. The intention is to harness this meticulously labeled dataset to construct a logistic regression model, a powerful tool for predicting the sentiment of new reviews effectively.

Furthermore, our project endeavors to augment its analytical prowess through data visualization techniques. These visual representations serve as a dynamic means of conveying critical patterns, trends, and sentiments embedded within the dataset. By presenting findings in a visually compelling manner, we aim to enhance comprehension, facilitate decision-making, and foster effective communication of insights gleaned from the data. This approach not only aids in unveiling hidden

narratives within the dataset but also adds a layer of accessibility and engagement to the analysis.

In summary, this research venture showcases the transformative potential of sentiment analysis, both as a theoretical concept and as a practical tool for real-world applications. By combining a comprehensive understanding of sentiment analysis principles with hands-on data analysis using the Uber Review dataset, our project strives to illuminate the intricate dynamics of sentiment within textual data and empower informed decision-making.

CHAPTER -1

INTRODUCTION

1.1 OVERVIEW

In an era dominated by an unprecedented influx of textual data, discerning the underlying sentiments, opinions, and emotional nuances has become not only a challenge but also a crucial necessity. The field of Sentiment Analysis, often synonymous with Opinion Mining, has emerged as a beacon of understanding in the realm of Natural Language Processing (NLP). Its significance has grown exponentially in recent years, driven by the surge in user-generated content across diverse platforms, ranging from the labyrinthine threads of social media to the meticulous evaluations of products and services. Within this vast expanse of textual data lies a wealth of information, waiting to be harnessed to comprehend public sentiment and guide critical decision-making processes.

This research endeavor embarks on a voyage to explore the multifaceted landscape of Sentiment Analysis, unveiling its essential principles and methodologies. Our journey commences with an introduction to the foundational elements that underpin sentiment analysis, encompassing the intricacies of text preprocessing, the intricacies of feature extraction, and the inner workings of sentiment classification algorithms. These bedrock concepts are pivotal to grasp the core of sentiment analysis.

Our project then transitions into a practical realm, where we harness the Uber Review dataset as the canvas for our explorations. Our overarching objective is to decode and quantify the sentiments embedded in textual reviews and to categorize this corpus into distinct sentiment labels. With meticulous labeling as our guide, we navigate the construction of a logistic regression model, a potent instrument for predicting the sentiment of new reviews with precision.

Not content with mere quantitative assessments, our research aspires to elevate its analytical prowess through the incorporation of ensemble methods. Within the ensemble, Support Vector Machines (SVM), decision trees, and naive Bayes work in harmony, each contributing its unique strengths to enhance the efficacy of our sentiment analysis model. The amalgamation of these diverse algorithms yields a holistic approach to sentiment classification, delivering robust results.

Furthermore, we recognize the power of data visualization as a beacon of insight. It transcends the boundaries of numbers and words, bringing to life critical patterns, trends, and sentiments concealed within the data. Through the art of visualization, we aim to empower decision-makers with accessible and engaging insights, fostering a deeper understanding of the data's narrative.

In summary, this research endeavor represents a journey through the transformative potential of sentiment analysis, both as a theoretical framework and as a practical tool for real-world applications. Our project integrates a deep understanding of

sentiment analysis principles with hands-on data analysis using the Uber Review dataset. It strives to illuminate the intricate dynamics of sentiment within textual data and empowers informed decision-making. With the ensemble methods and data visualization techniques as our companions, our exploration promises a holistic and valuable contribution to the world of sentiment analysis.

Machine learning is important because it gives enterprises a view of trends in customer behavior 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.

- 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 modeling: Predicting continuous values.
- Ensembling: Combining the predictions of multiple machine learning models to produce an accurate prediction.

1.2 OBJECTIVE

- To be measured with the help of graphical notations such as pie chart, bar chart and line chart. The data can be given in dynamical data.
- The main objective of this research is to develop an Intelligent System using machine learning techniques, namely logistic regression, svm and Random forest.

1.3 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:

1. 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.

2. 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

1.4 MOTIVATION FOR THE WORK

The motivation behind embarking on this research project stems from several key factors that underscore the significance and relevance of sentiment analysis,

ensemble methods, and data visualization in the context of today's information-rich landscape.

1. Proliferation of User-Generated Content: In the digital age, user-generated content has reached an unprecedented scale, inundating social media platforms, e-commerce websites, and review forums. This deluge of data represents a valuable source of public sentiment and opinion. Understanding and harnessing this vast volume of unstructured text data have never been more critical.

2. Decision-Making in a Data-Driven World: The ability to make informed decisions, whether in business, governance, or social contexts, depends on a comprehensive understanding of public sentiment. Sentiment analysis equips decision-makers with the tools to gauge public perception, anticipate trends, and respond effectively to evolving situations.

3. Improving Classification Accuracy: Sentiment analysis, when bolstered by ensemble methods, has the potential to significantly enhance classification accuracy. Traditional sentiment analysis algorithms, while effective, often have limitations. Combining multiple methods can lead to more robust and accurate sentiment classification, improving the quality of insights derived from the data.

4. Data Visualization as a Communicative Medium: In an age where data reigns supreme, effective communication of insights is crucial. Data visualization techniques serve as a bridge between raw data and human understanding. They enable us to convey complex patterns and trends in a digestible and engaging manner, fostering better communication and decision-making.

5. Practical Real-World Applications: Beyond the theoretical aspects of sentiment analysis and ensemble methods, this project is grounded in practicality. We intend to apply these concepts to the Uber Review dataset, a real-world corpus of user-generated content. The application of sentiment analysis and ensemble methods in this context demonstrates their tangible impact in solving real-world problems.

In light of these motivations, our work seeks to not only explore the theoretical underpinnings of sentiment analysis but also to apply this knowledge practically. By employing ensemble methods and data visualization, we aim to improve the accuracy of sentiment classification and to communicate our findings effectively. Ultimately, our research endeavors to empower individuals, organizations, and decision-makers with the insights needed to navigate the data-rich landscape of the 21st century.

1.5 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:

1. 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.

2. 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.

3. 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.

4. 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.

In addressing this problem, the research seeks to enhance the capacity to harness textual data for understanding public sentiment, making informed decisions, and facilitating effective communication. Additionally, it endeavors to contribute to the ongoing development of sentiment analysis methodologies, ensemble methods, and data visualization techniques in the field of Natural Language Processing.

CHAPTER 2

2.1 LITERATURE SURVEY

The following literature survey provides an overview of relevant studies and research in the areas of sentiment analysis, ensemble methods, and data visualization, highlighting key contributions, trends, and challenges in each field.

Sentiment Analysis:

1. Pang and Lee, 2008 - The work by Pang and Lee is a seminal paper in sentiment analysis. It discusses the challenges of sentiment classification, introduces a dataset

for movie reviews, and presents early approaches to sentiment analysis, including lexicon-based methods and machine learning techniques.

2. Socher et al., 2013 - This study introduced recursive neural tensor networks, a deep learning approach that improved sentiment analysis accuracy by capturing syntactic structures and compositionality in textual data.

3. VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text - This paper presents the Valence Aware Dictionary and sEntiment Reasoner (VADER), a lexicon and rule-based sentiment analysis tool that performs well with short texts, such as those found on social media platforms.

4. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding - The introduction of BERT (Bidirectional Encoder Representations from Transformers) revolutionized sentiment analysis by pre-training on vast text data and fine-tuning on specific tasks, achieving state-of-the-art results.

2.2 Ensemble Methods in Sentiment Analysis:

1. Ensemble Methods for Sentiment Analysis - This review paper provides an overview of various ensemble methods, including bagging, boosting, and stacking, and their application in sentiment analysis.

2. Heterogeneous Ensemble for Sentiment Analysis - This study explores the potential of combining heterogeneous classifiers, such as Support Vector Machines, decision trees, and neural networks, to improve sentiment classification accuracy.

The literature survey illustrates the evolution of sentiment analysis techniques, the increasing prominence of deep learning, the effectiveness of ensemble methods, and the significance of data visualization in making sentiment analysis results accessible and actionable. It provides a strong foundation for the research project by drawing on the knowledge and insights from previous works.

CHAPTER 3

3.1 Challenges and Limitations in Existing Systems for Sentiment Analysis:

1. 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.

2. 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.

3. 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.

4. 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 analyzing and integrating these diverse data types.

5. 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.

6. 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.

7. 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.

8. 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.

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

3.2 Limitations in Ensemble Methods:

1. Complexity: Building and maintaining ensemble models can be complex and resource-intensive, requiring expertise in multiple algorithms and model integration techniques.

2. Selection of Base Classifiers: Choosing appropriate base classifiers for an ensemble can be challenging, as the performance of the ensemble heavily relies on the diversity and quality of the base classifiers.

3. Ensemble Size: Determining the ideal number of base classifiers in an ensemble is not always straightforward. Too few may result in underfitting, while too many can lead to overfitting.

4. Data Imbalance: Imbalanced datasets can affect the performance of ensemble methods. Ensuring a balance between the classes in the training data is crucial for effective ensemble learning.

3.3 Challenges in Data Visualization:

1. Multidimensional Data: Visualizing sentiment analysis results, which are often multidimensional, can be challenging. Combining aspects like sentiment score, topic, and time in a single visualization is not always straightforward.

2. Interactivity and Real-Time Updates: Developing interactive visualizations that allow users to explore and interact with sentiment data in real-time can be complex and resource-intensive.

3. Avoiding Misinterpretation: Poorly designed visualizations can lead to misinterpretation of sentiment analysis results. Ensuring that the visual representation accurately reflects the data is crucial.

4. Scalability: Visualizing sentiment data on a large scale, such as monitoring sentiments across social media platforms, can strain the capacity of visualization tools.

5. Accessibility: Ensuring that data visualizations are accessible to all users, including those with disabilities, is a challenge in design and development.

Addressing these challenges and limitations is essential for advancing the field of sentiment analysis, improving the accuracy of sentiment classification, enhancing the utility of ensemble methods, and making sentiment analysis results more accessible through effective data visualization techniques.

CHAPTER 4

4.1 PROPOSED SYSTEM:

The proposed system 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 system:

4.2. Ensemble Learning for Enhanced Sentiment Analysis:

Diverse Base Classifiers: The system will incorporate diverse base classifiers, including Support Vector Machines (SVM), decision trees, and naive Bayes, within an ensemble framework. The diversity of base classifiers will help capture a wide range of sentiment patterns and enhance classification accuracy.

Ensemble Model: These base classifiers will be combined using ensemble techniques such as bagging or boosting to create a robust and accurate sentiment

analysis model. The ensemble model will leverage the strengths of individual classifiers, improving overall performance.

Machine Learning for Sentiment Trends: The system will employ machine learning models to detect and adapt to evolving sentiment trends. This adaptive capability ensures that the sentiment analysis remains accurate even as sentiment expressions change over time.

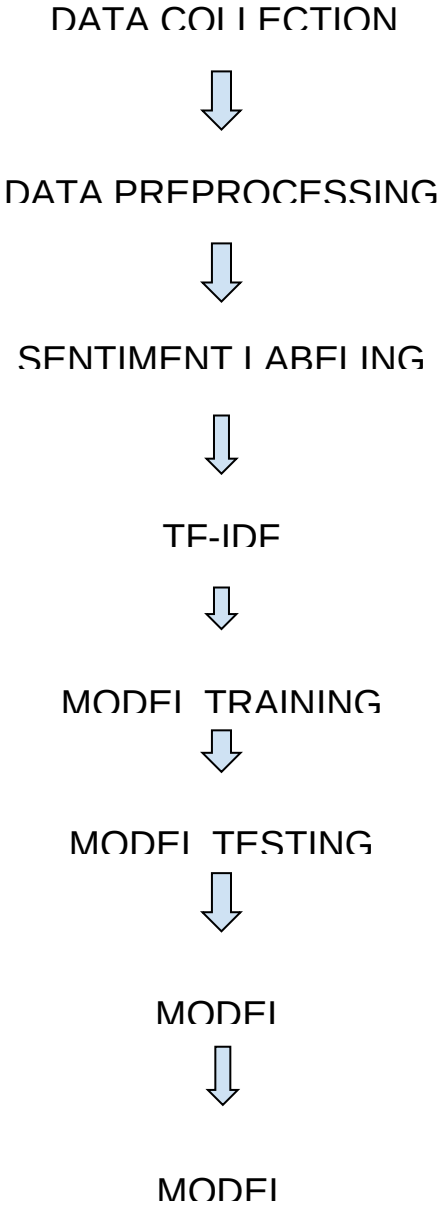
Visualization: The system will employ data visualization techniques to present sentiment analysis results in a visually engaging Users can explore sentiment trends, see sentiment distributions, and identify influential topics through interactive visualizations.

The proposed system offers a comprehensive solution for sentiment analysis, addressing the challenges of accuracy,and dynamic visualization. By integrating ensemble learning techniques, and interactive data visualization, the system empowers users to gain deeper insights into sentiment trends and make informed decisions based on data.

CHAPTER 5

5.1 MODULES

- Data Collection
- Data Preprocessing
- sentiment labeling
- TF-IDF
- Model Training
- Model Testing
- model evaluation
- model deployment



5.2 DATA COLLECTION

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

1. 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.

2. Web Scraping or API Access: There are two primary methods for collecting data from the Google Play Store:

a. 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.

b. API Access: 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.

3. 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.

4. 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).

5. 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.

6. 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.

7. 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.

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

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

10. 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.

5.3 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:

1.Lowercasing: Convert all text to lowercase to ensure consistency in the text data.

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

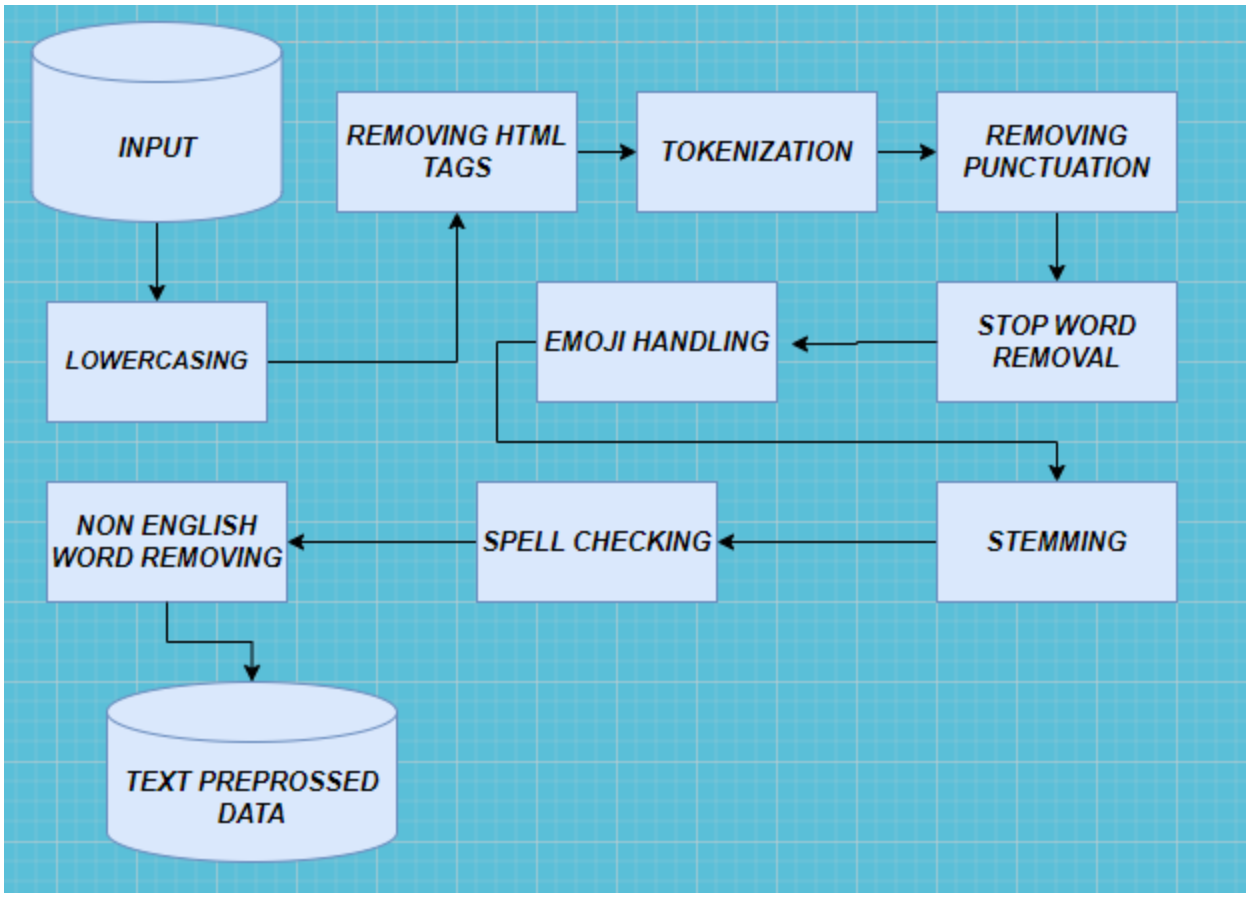
3. Removing Punctuation: Remove punctuation marks like commas, periods, and special characters. Punctuation often doesn't carry sentiment information and can be safely removed.

4. 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.

5. 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.

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

7. 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.



5.4 SENTIMENT LABELING

Sentiment labeling using VADER (Valence Aware Dictionary and sEntiment Reasoner) is a valuable approach to quickly assess and categorize 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:

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

2. Initialization: Create an instance of the VADER sentiment analyzer by using `SentimentIntensityAnalyzer()`.

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

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

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

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

- Positive: Compound score >= 0.05

- Neutral: -0.05 < Compound score < 0.05

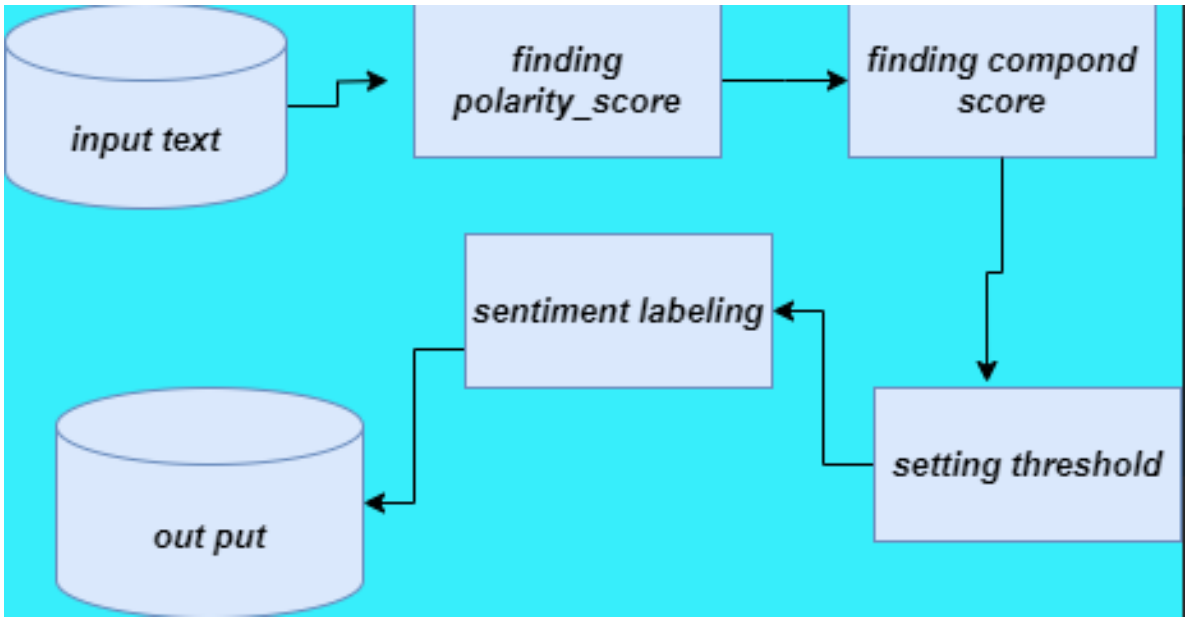
- Negative: Compound score <= -0.05

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

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

5. 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.

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



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

CHAPTER - 6

6.1 TF-IDF(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:

1. 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.

2. 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.

3. 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.

4. 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.

5. 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 to 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.

6. 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.

6.2 MODEL

Selecting the right model for sentiment analysis is crucial, and your choice between Support Vector Machines (SVM), Decision Trees, and Naive Bayes should be based on the characteristics of your data and the specific requirements of your project. Here's a brief comparison to help you decide which model to use:

Support Vector Machines (SVM):

Strengths:

- SVMs are effective for binary and multiclass sentiment classification.
- They can handle both linear and non-linear data separation.
- SVMs often perform well when you need a high level of accuracy.
- They are less prone to overfitting, making them a good choice when you have a limited amount of training data.

Considerations:

- SVMs can be computationally expensive, particularly when the dataset is large.
- Model tuning, including selecting appropriate kernel functions and regularization parameters, can be challenging.

Decision Trees:

Strengths:

- Decision trees are interpretable and easy to understand. They provide insights into the decision-making process.
- They are efficient and can handle both numerical and categorical data.
- Decision trees are less sensitive to outliers and missing values.

Considerations:

- Decision trees may not perform as well as other models in capturing complex patterns in the data.
- They are prone to overfitting, especially when the tree is deep. Pruning is often necessary to mitigate overfitting.

Naive Bayes:

Strengths:

- Naive Bayes is simple and efficient, making it a good choice for sentiment analysis when you have limited computational resources.
- It can handle large datasets and performs well with text data.
- Naive Bayes is based on probabilistic principles and can provide interpretable results.

Considerations:

Naive Bayes makes a strong independence assumption between features, which might not hold in some cases. However, it often performs surprisingly well despite this simplification.

Considerations for Model Selection:

1. 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.

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

3. Accuracy Requirements: If high accuracy is a primary goal, SVMs may be a suitable choice. However, decision trees and Naive Bayes can also perform well in certain scenarios.

4. Interpretability: Decision trees and Naive Bayes are more interpretable models. If you need insights into how the model makes decisions, these may be preferable.

5. 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.

6. **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.

7. 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, the choice between SVM, Decision Trees, and Naive Bayes should be based on the nature of your data, your project's requirements, and your computational resources. It's also worth experimenting with different models and evaluating their performance to make an informed decision.

6.3 ENSEMBLE MODEL

Ensemble models are a powerful approach in machine learning that combine the predictions of multiple base models to improve overall performance, accuracy, and generalization. When it comes to sentiment analysis or any other classification task, ensemble methods can be highly effective. Here are some popular ensemble methods and an overview of how they can be applied to sentiment analysis:

1. Voting Ensemble: In a voting ensemble, multiple base models make predictions, and the final prediction is determined by a majority vote (for classification tasks). The mode of the predictions is taken as the final prediction.

You can combine multiple sentiment analysis models (e.g., SVM, Decision Trees, Naive Bayes) and have them vote on the sentiment label.

2. Bagging (Bootstrap Aggregating): Bagging is a technique where multiple subsets of the training data are created through resampling (bootstrap sampling). Multiple base models are trained on these subsets, and their predictions are averaged or voted upon.

It can be applied by training multiple sentiment analysis models on different bootstrapped samples of your training data and averaging their predictions.

3. Random Forest: Random Forest is a specific type of bagging ensemble method that uses decision trees as base models. It adds an element of randomness during training to reduce overfitting.

You can create a sentiment analysis model using a Random Forest ensemble with decision trees as base models.

4. Boosting: Boosting is an ensemble method where multiple base models are trained sequentially. Each model is trained to correct the errors of the previous one.

You can apply boosting to sentiment analysis by training a series of models, each focusing on the misclassified instances of the previous model.

5. Stacking: Stacking combines the predictions of multiple models by using another model (meta-learner) to learn how to best weight or combine their predictions.

In the context of sentiment analysis, you can stack multiple base models (e.g., SVM, Decision Trees, Naive Bayes) and use a meta-learner to make the final prediction.

6. Gradient Boosting: Gradient Boosting is a boosting technique that builds an ensemble of decision trees in a way that minimizes the prediction error.

You can create a gradient boosting model for sentiment analysis using decision trees as base models.

7. AdaBoost: AdaBoost is a specific boosting algorithm that assigns weights to training instances, giving more importance to the instances that were misclassified by the previous models.

You can apply AdaBoost to sentiment analysis to give higher weights to challenging sentiment instances.

Considerations for Ensemble Models in Sentiment Analysis:

- Ensure that base models in your ensemble are diverse and have different strengths and weaknesses to benefit from the ensemble's collective power.
- Carefully tune hyperparameters for both the base models and the ensemble method to optimize performance.
- Cross-validation is crucial to evaluate and fine-tune your ensemble.

Ensemble methods can significantly enhance the accuracy and robustness of sentiment analysis models. The choice of the ensemble method and the base models should be based on the characteristics of your data and specific project requirements. Experimentation and thorough evaluation are key to achieving the best results with ensemble models.

6.4 HYBRID ALGORITHM

A hybrid algorithm in the context of sentiment analysis refers to a combination of multiple algorithms or approaches to improve the accuracy and robustness of sentiment classification. The idea is to leverage the strengths of different methods to address the limitations of individual algorithms. Hybrid sentiment analysis models often yield better results by exploiting the diversity of techniques. Here's how you can create a hybrid sentiment analysis algorithm:

1. Data Preprocessing: Start with data preprocessing, which may include text cleaning, tokenization, stop-word removal, and stemming or lemmatization to prepare your text data for analysis.

2. Feature Engineering: Create features from the preprocessed text data. Common features used in sentiment analysis include TF-IDF (Term Frequency-Inverse Document Frequency) vectors, word embeddings (e.g., Word2Vec or GloVe), and n-grams.

3. Hybrid Algorithm Components: Select multiple sentiment analysis algorithms or approaches that you want to combine. Common choices include rule-based methods (e.g., VADER), machine learning models (e.g., Naive Bayes, SVM, Decision Trees), and deep learning models (e.g., neural networks like LSTM or Transformer-based models).

4. Model Training: Train each selected sentiment analysis algorithm separately on your labeled training data.

5. Individual Predictions: Use each algorithm to make predictions on your test or validation data. Each algorithm will provide its sentiment prediction for each input.

6. Integration of Predictions: Combine the predictions from different algorithms in a way that leverages their collective wisdom. Some common methods for combining predictions include:

- Majority Voting: The final sentiment label is determined by the majority vote of the algorithms.
- Weighted Averaging: Assign weights to each algorithm's prediction based on their performance on the validation set and average their predictions with those weights.
- Stacking: Use another machine learning model (meta-learner) to learn how to best combine the predictions of individual algorithms.

7. Final Sentiment Prediction: The integrated predictions from the hybrid algorithm are used to determine the final sentiment label for each input text.

8. Model Evaluation: Evaluate the performance of your hybrid sentiment analysis algorithm using appropriate metrics (e.g., accuracy, precision, recall, F1-score). Perform cross-validation to ensure robustness.

9. Fine-Tuning and Iteration: Depending on the performance, you may need to fine-tune the algorithm, adjust the weights, or add more algorithms to the hybrid model for further improvement.

Considerations for Hybrid Sentiment Analysis:

- The choice of algorithms to include in the hybrid model should be based on the characteristics of your data and the desired level of accuracy.
- Experiment with different combination techniques and weight assignments to optimize the hybrid model's performance.
- Ensure that you have a validation set for each algorithm to assess their individual performance before combining them.
- Regularly update your hybrid model as new sentiment analysis algorithms and techniques become available.

A well-designed hybrid sentiment analysis algorithm can provide more accurate and reliable results compared to using a single method, especially when dealing with complex and diverse text data.

CHAPTER - 7

7.1 MODEL TRAINING

Training sentiment analysis models like Support Vector Machines (SVM), Decision Trees, and Naive Bayes involve specific steps and considerations. Here's an overview of how to train these models for sentiment analysis:

1. Data Preparation:

- Gather a labeled dataset that includes text samples and their associated sentiment labels (e.g., positive, negative, neutral).
- Preprocess the text data by cleaning, tokenizing, and converting it into a numerical format suitable for machine learning.

2. Split the Dataset:

Divide your dataset into training, validation, and test sets. The training set is used for model training, the validation set helps tune hyperparameters, and the test set is used for final evaluation.

3. Feature Extraction:

Depending on the model, you may need to extract features from the text data. For traditional machine learning models like SVM and Naive Bayes, common features include TF-IDF vectors or word embeddings.

4. Model Training:

Training an SVM (Support Vector Machine):

- Initialize an SVM classifier, selecting the appropriate kernel function (e.g., linear, polynomial, radial basis function).
- Train the SVM using the training dataset, and make predictions on the validation set to assess performance.
- Optimize hyperparameters, such as the regularization parameter (C) and kernel-specific parameters, using cross-validation.
- Once you're satisfied with performance on the validation set, evaluate the SVM on the test set.

Training a Decision Tree:

- Create a decision tree classifier, specifying optional hyperparameters like the maximum depth or minimum samples per leaf node.
- Train the decision tree using the training dataset.
- Evaluate the decision tree on the validation set, and adjust hyperparameters to optimize performance (e.g., to prevent overfitting).
- Finally, assess the decision tree on the test set to obtain the model's performance.

Training Naive Bayes:

- For Naive Bayes, choose the specific variant you want to use, such as Multinomial Naive Bayes, which is often suitable for text data.
- Train the Naive Bayes model using the training dataset.
- Evaluate the model on the validation set and adjust any smoothing parameters if necessary.
- Ultimately, assess the Naive Bayes model on the test set for the final evaluation.

6. Hyperparameter Tuning (Optional):

- For SVM and Decision Trees, it's essential to tune hyperparameters to optimize model performance. Cross-validation is often used to find the best hyperparameters.
- Naive Bayes usually requires less hyperparameter tuning, but you can experiment with smoothing parameters if needed.

7.2 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:

1. 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, neutral).

2. 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.

3. Model Assessment:

Analyze 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.

4. 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

$$\frac{TP + TN}{TP + FP + TN + FN}$$

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

$$\frac{TP}{TP + FP}$$

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

Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	*Positive Predictive Value*	a/(a+b)
	Negative	c	d	*Negative Predictive Value*	d/(c+d)
		Sensitivity	*Specificity*	**Accuracy** = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		

ENSEMBLE CLASSIFIER RESULTS:

Accuracy: 0.9461895656924337

Classification Report:

	precision	**recall**	**f1-score**	**support**
negative	0.80	0.92	0.85	471
neutral	0.92	0.87	0.90	581
positive	0.98	0.97	0.97	3661

accuracy			0.95	3661
macro avg	0.90	0.92	0.91	3661
weighted avg	0.95	0.95	0.95	3661

1. Accuracy:

The accuracy of approximately 94.62% is a measure of how often your model correctly predicts the sentiment labels. It indicates that the majority of predictions are accurate.

2. 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, 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.

3. 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 0.90, indicating a good overall performance.

The "weighted avg" 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 0.95, 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 - 8

8.1 system requirements

1. **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/) and choose a version compatible with your operating system.

2. 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
- contractions
- nltk
- scikit-learn (for machine learning)
- transformers (for BERT models)
- spellchecker
- wordcloud
- matplotlib

You can install these libraries with the following commands:

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

3. 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')
```

4. Pre-Trained Models: The code uses pre-trained models for text summarization. You should have an internet connection to download these models. In the example code, the "facebook/bart-large-cnn" model is used. You can change the model name according to your requirements and download the corresponding model.

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

7. 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.

8. 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 colab was used:

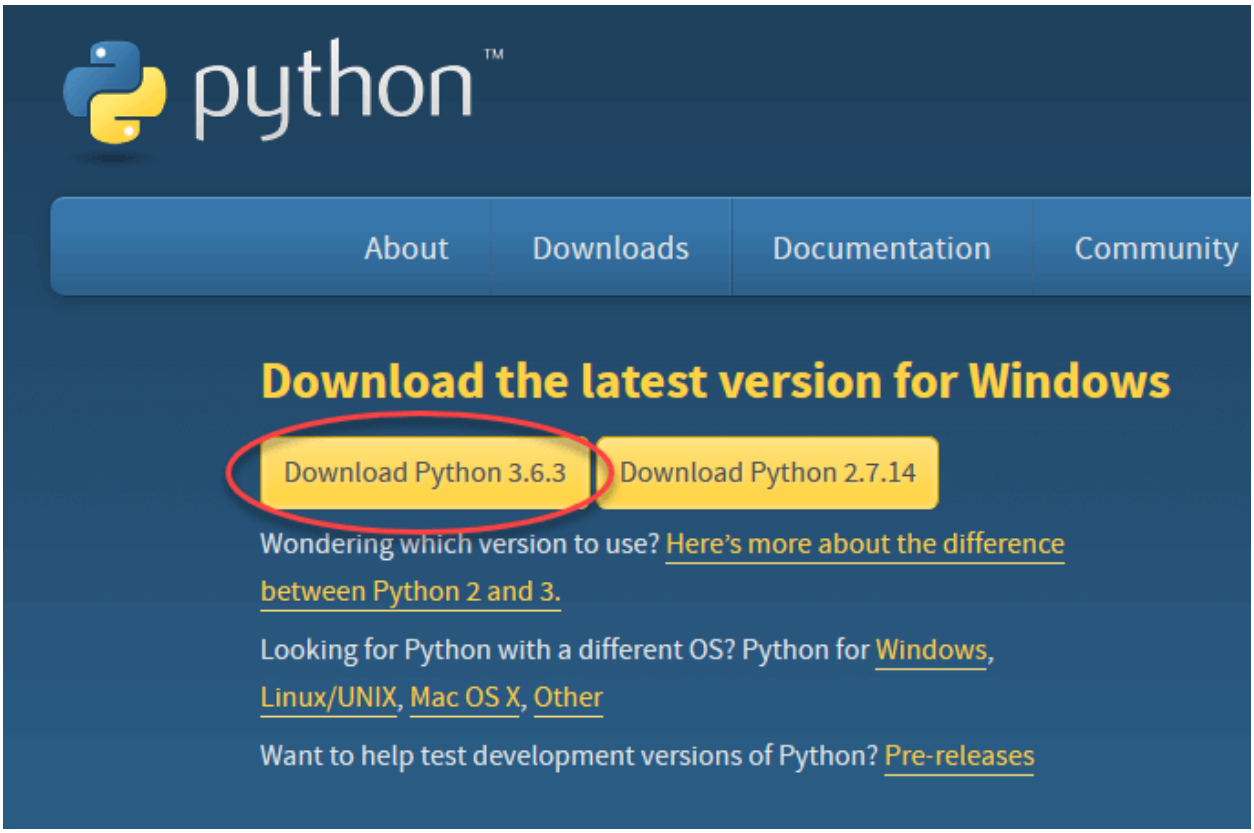
8.2 HOW TO INSTALL PYTHON IDE

Below is a step by step process on how to download and install Python on Windows:

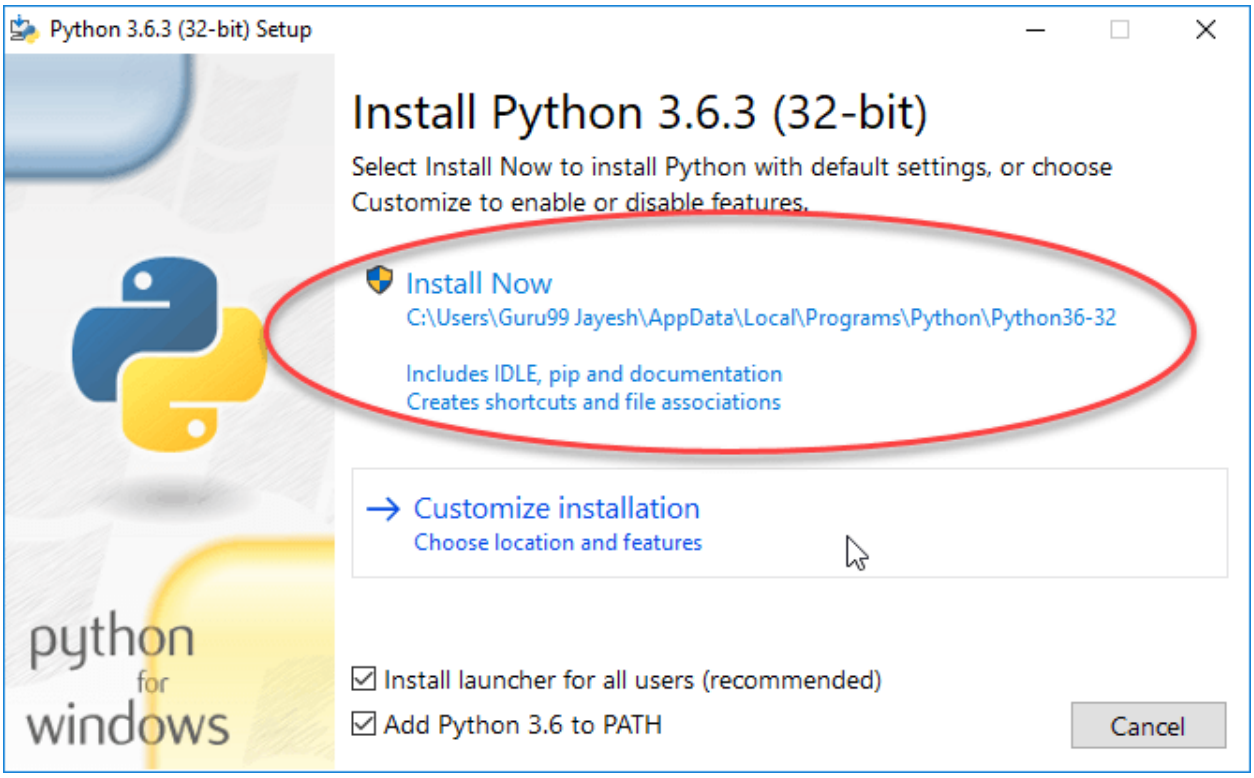
Step 1) To download and instal

5. Data: You need to provide the 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.

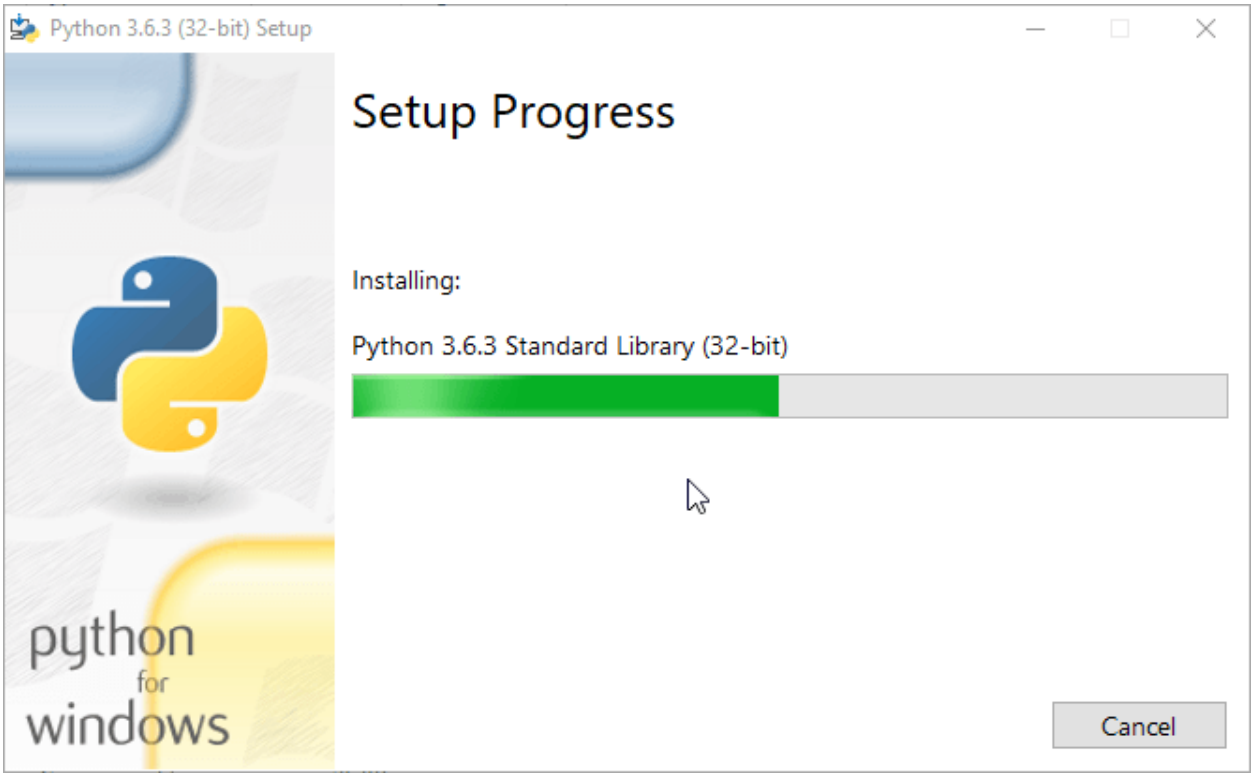
l Python, visit the official website of Python https://www.python.org/downloads/ and choose your version. We have chosen Python version 3.6.3



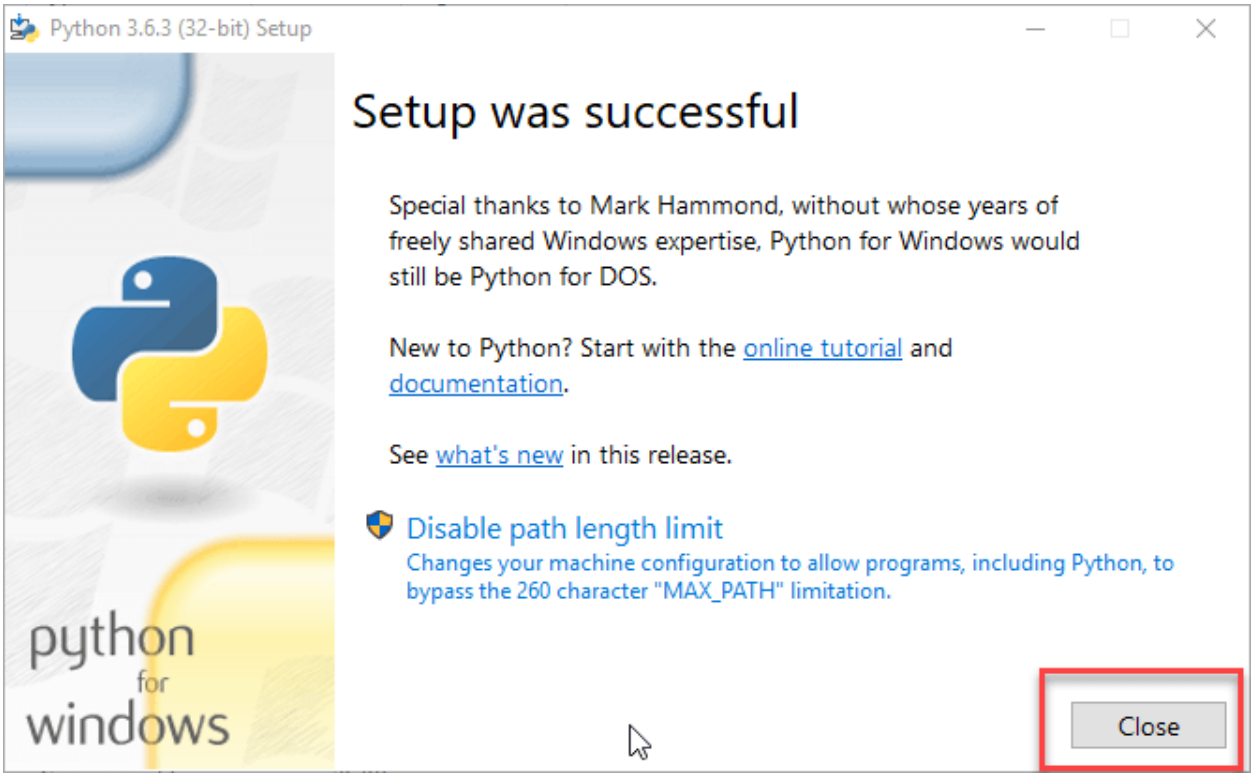
Step 2) Once the download is completed, run the .exe file to install Python. Now click on Install Now.



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



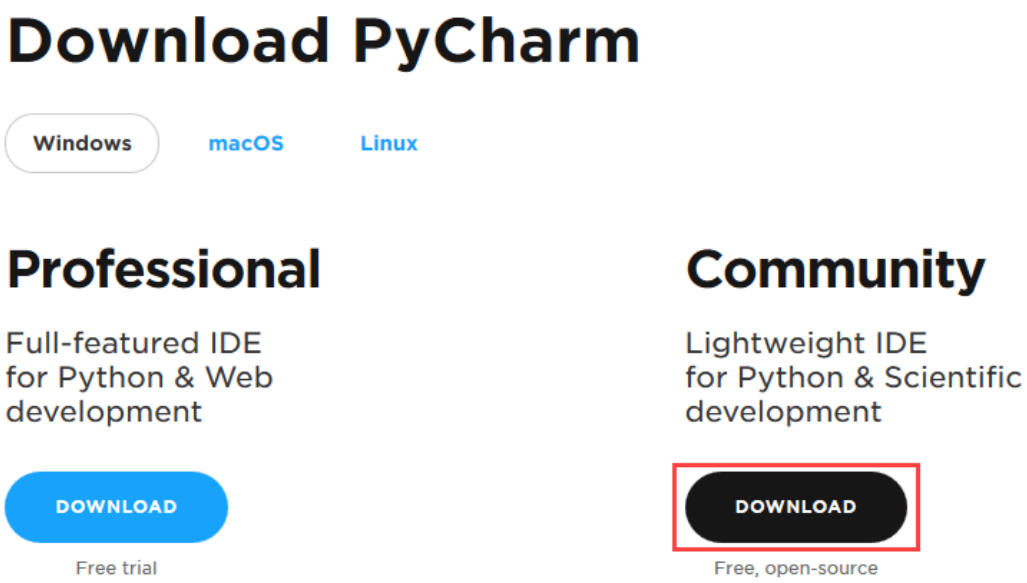
Step 4) When it finishes, you can see a screen that says the Setup was successful. Now click on “Close”.



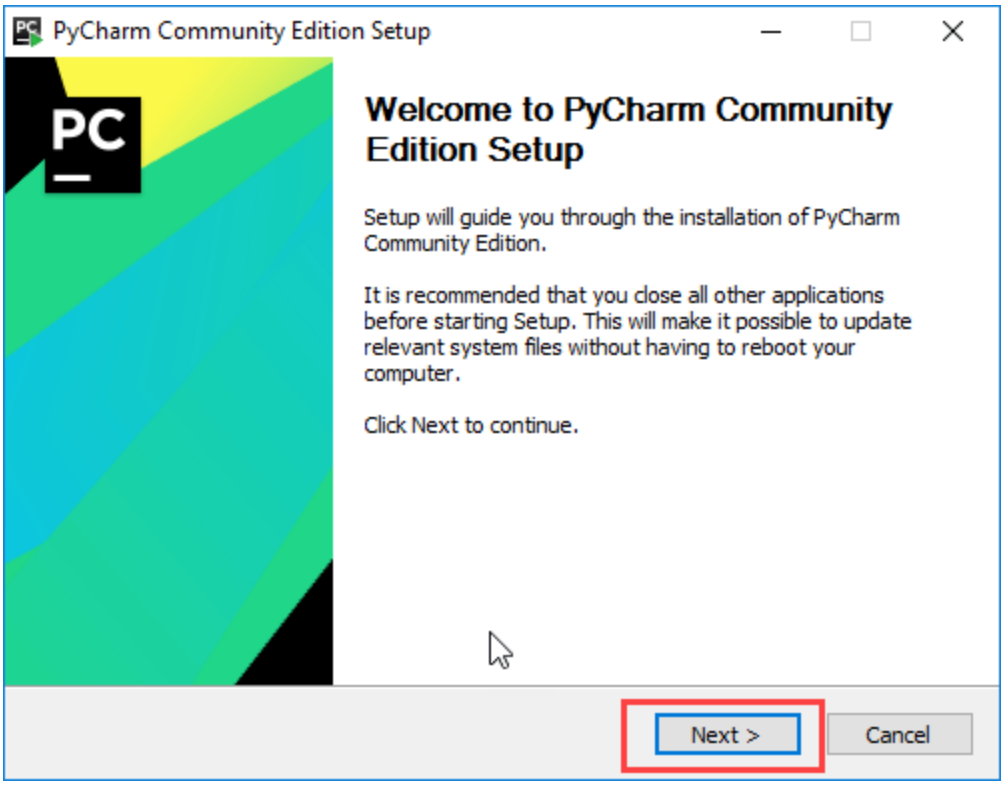
HOW TO INSTALL PYCHARM

Here is a step by step process on how to download and install Pycharm IDE on Windows:

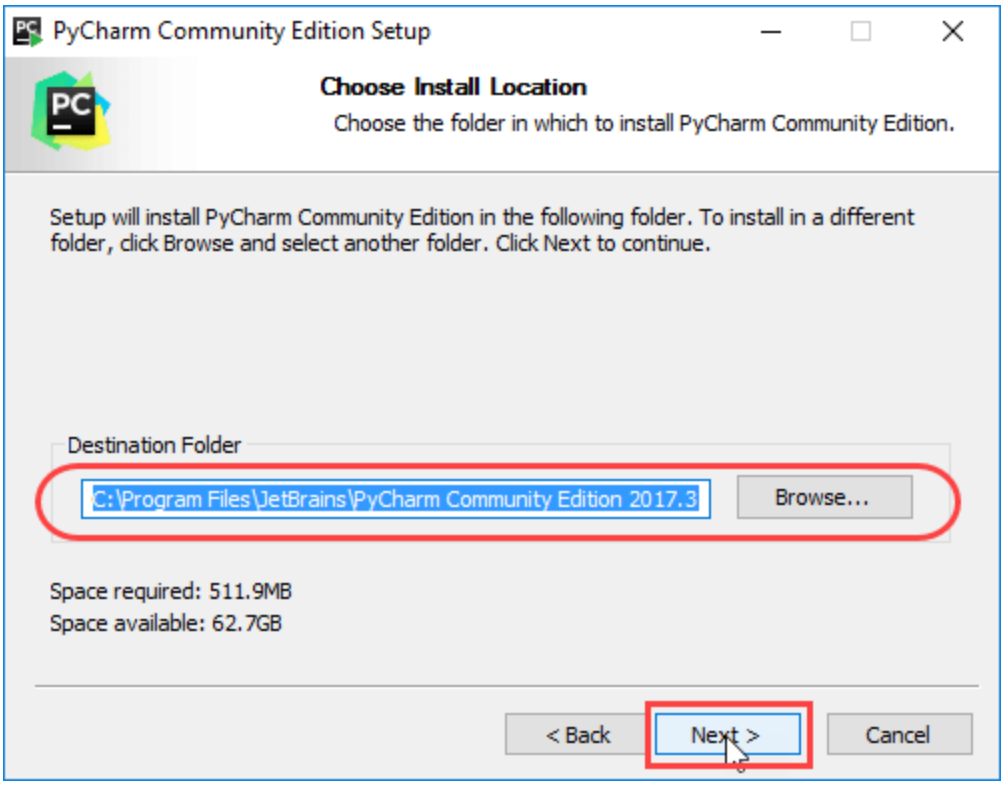
Step 1) To download PyCharm visit the website https://www.jetbrains.com/pycharm/download/ and Click the “DOWNLOAD” link under the Community Section.



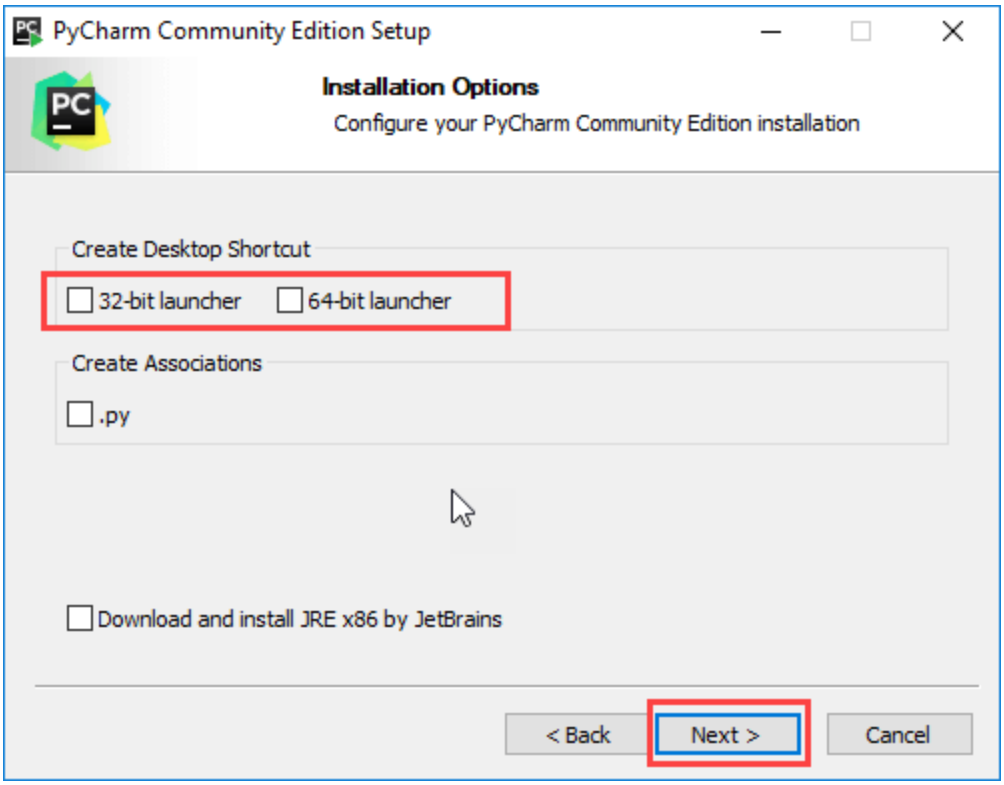
Step 2) Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.



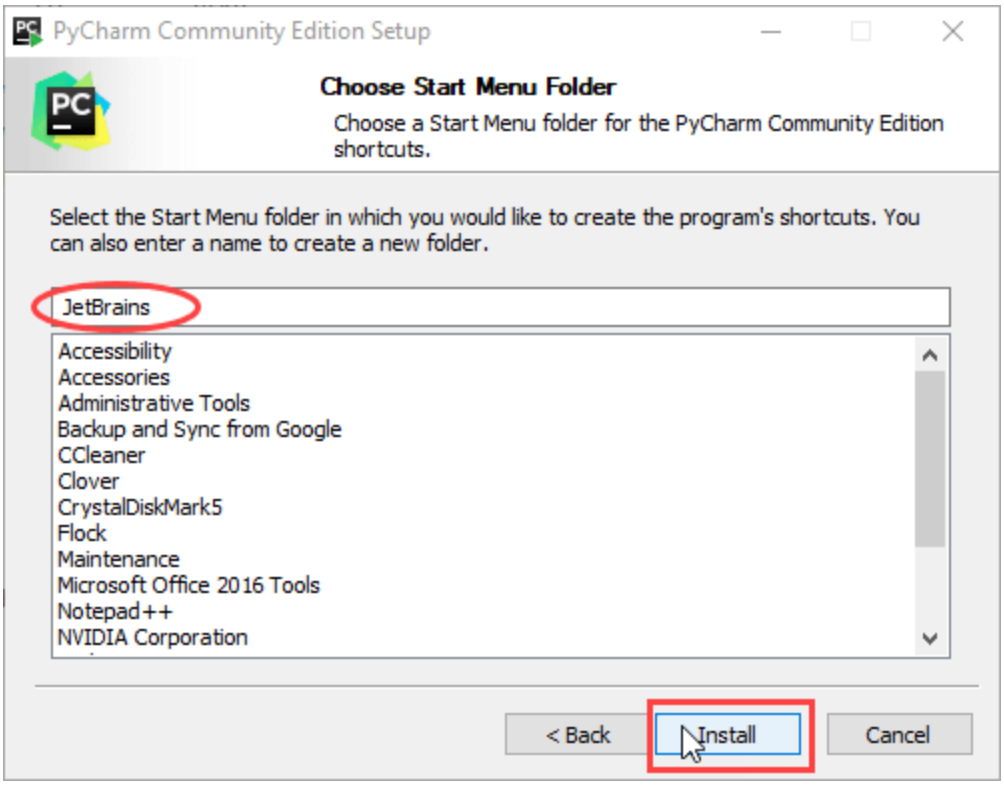
Step 3) On the next screen, Change the installation path if required. Click "Next".



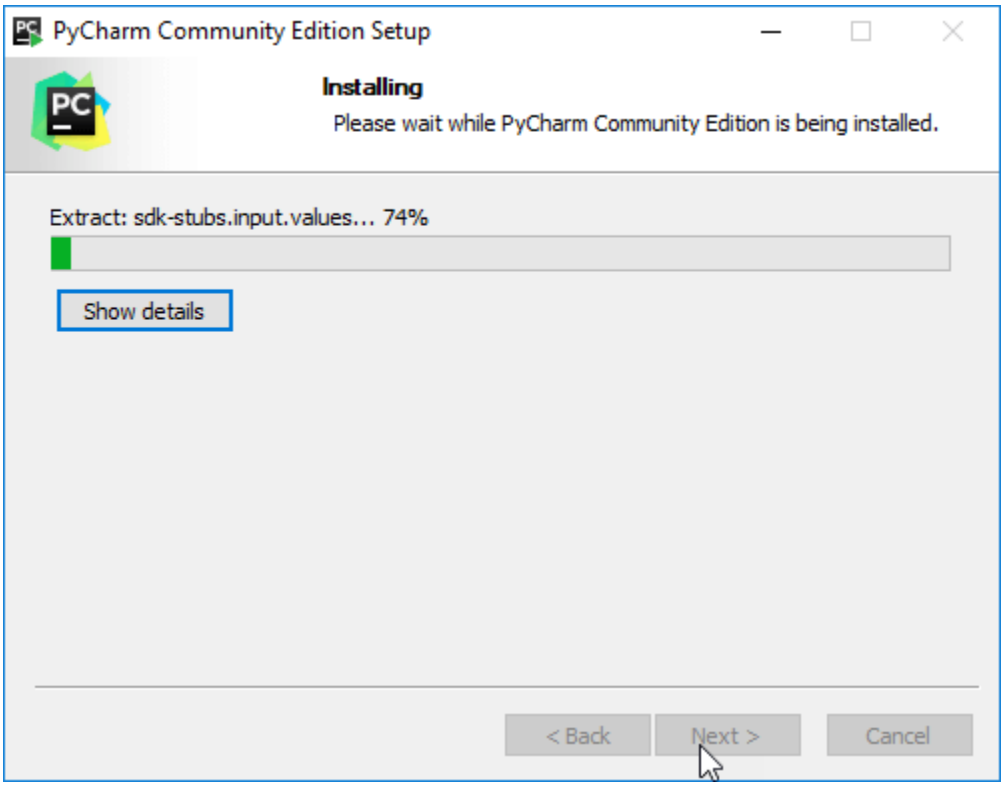
Step 4) On the next screen, you can create a desktop shortcut if you want and click on “Next”.



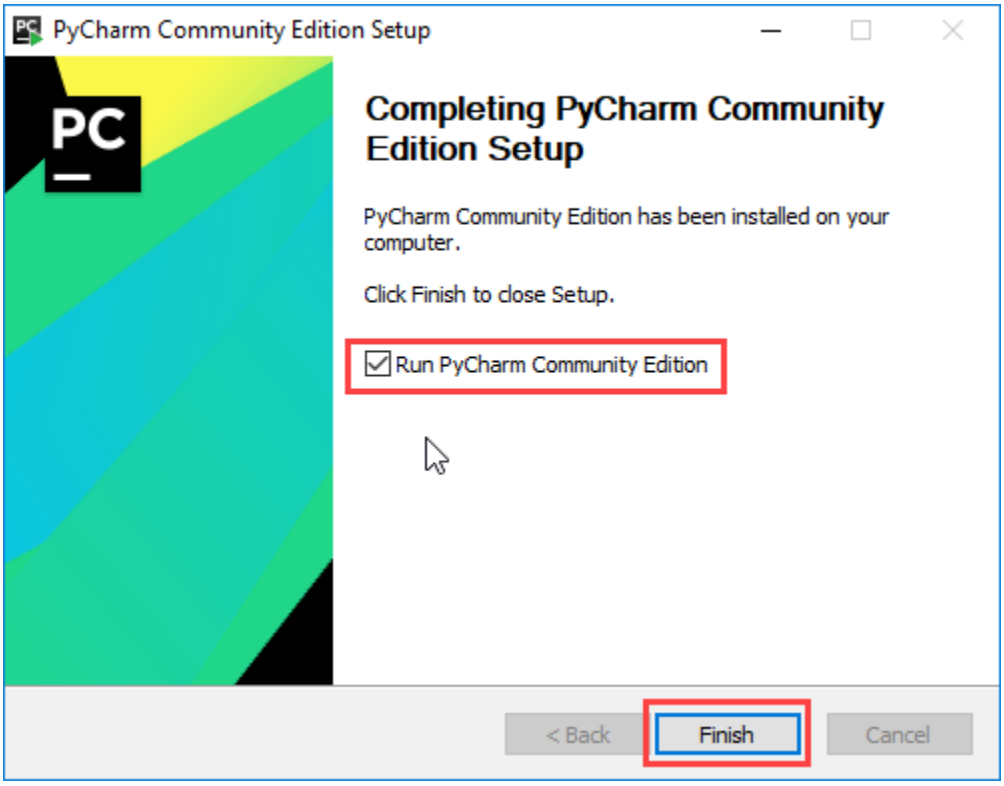
Step 5) Choose the start menu folder. Keep selected JetBrains and click on "Install".



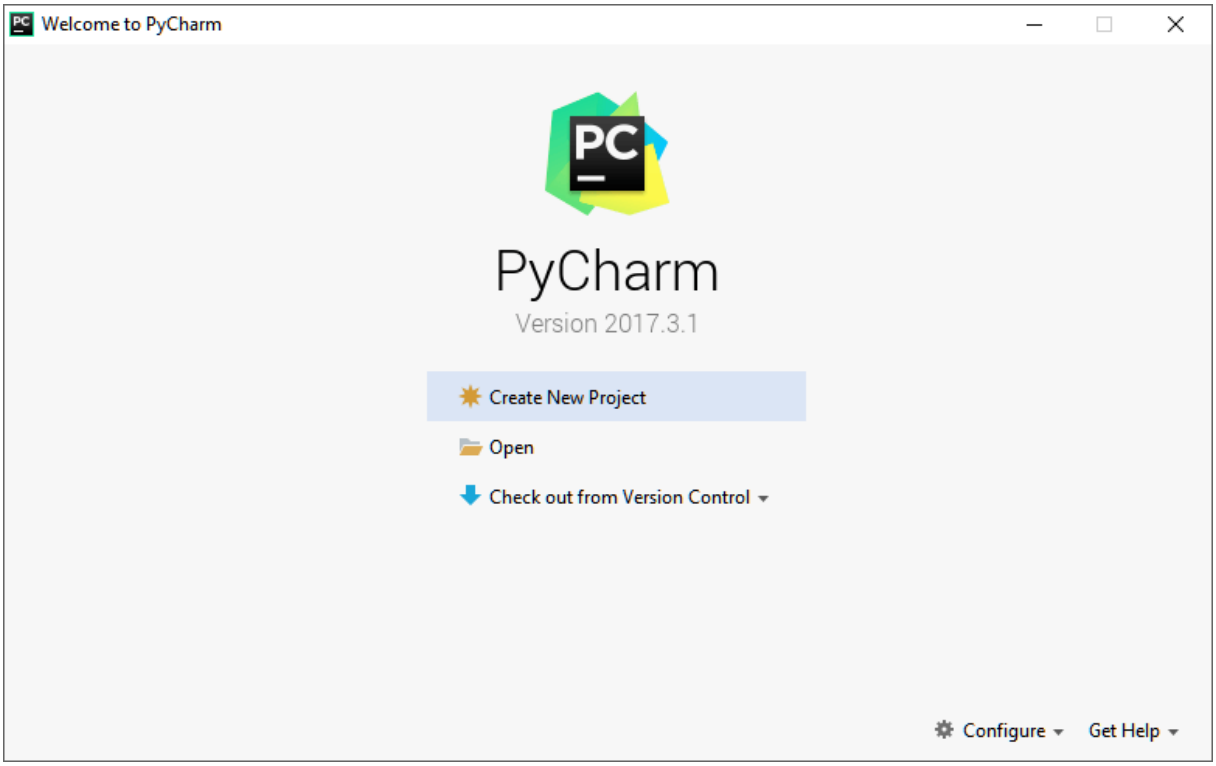
Step 6) Wait for the installation to finish.



Step 7) Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the "Run PyCharm Community Edition" box first and click "Finish".

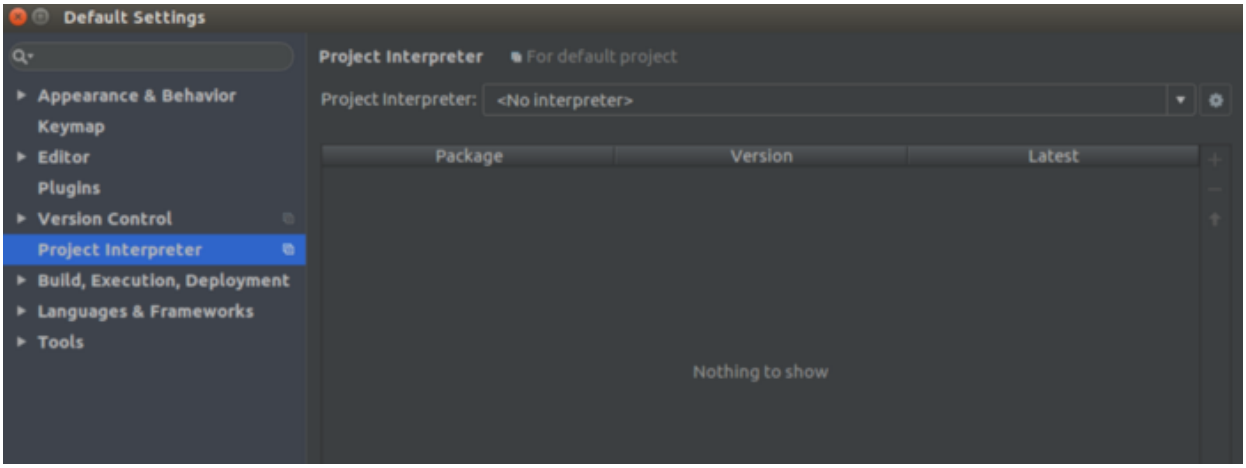


Step 8) After you click on "Finish," the Following screen will appear.

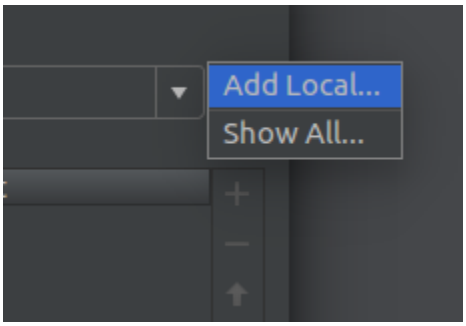


Click on Configure > Settings to open up settings in PyCharm

Search for “Project Interpreter”. My PyCharm looks like this



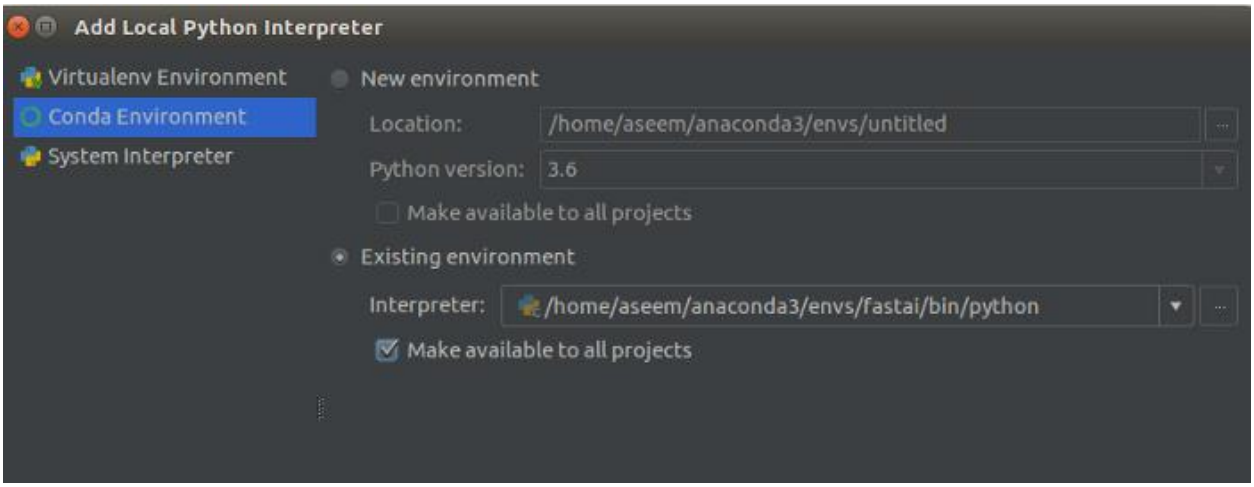
Click on Add local via the settings on the right side



Select “conda environment”

Click on “Existing environment” and navigate to the environment that you want to use. Note that you have to select the bin/python file inside the conda environment for PyCharm to be able to recognise the environment

Make sure to click the “Make available to all projects” if you want the interpreter to be used by multiple projects



Click ok and you are done

Before running the code, ensure that you have set up your Python environment and installed the necessary libraries. Additionally, prepare your data in a suitable CSV file, and make sure the data format matches the code's expectations.

8.3 CODING

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

8.4 CODING STANDARDS

Coding standards are guidelines to programming that focuses 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 standard needed to achieve the above-mentioned objectives are as follows:

Program should be simple, clear and easy to understand.

- Naming conventions
- Value conventions
- Script and comment procedure
- Message box format
- Exception and error handling

8.5 CODE