VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI – 590018, Karnataka

INTERNSHIP REPORT

ON

"Sentiment Analysis Of Lockdown In USA During Covid-19 A Case Study On Twitter using ML"

Submitted in partial fulfilment for the award of degree(18CSI85)

BACHELOR OF ENGINEERING IN
INFORMATION SCIENCE AND ENGINEERING

Submitted by:

AKASH TATTI

1HK20IS007

Varcons Technologies Pvt Ltd

Conducted at
VARCONS TECHNOLOGIES PVT LTD.

HKBK COLLEGE OF ENGINEERING
22/1,NAGAWARA,BANGALORE- 560045
E - mail : info@hkbk.edu.in ,URL: www.hkbk.edu.in

DECLARATION

I Akash Tatti, final year student of Information Science and Engineering, HKBK College of engineering, declare that the Internship has been successfully completed, in "**Sentiment Analysis Of Lockdown In USA During Covid-19 A Case Study On Twitter using ML"**. This report is submitted in partial fulfillment of the requirements for award of Bachelor Degree in Branch name, during the academic year 2023-2024.

Date :20-10-2023 :

Place :Bangalore

USN : 1HK20IS007

NAME: AKASH TATTI

OFFER LETTER

INTERNSHIP OFFER LETTER

Date: 11th August, 2023

Name: **Akash Tatti**
USN: **1HK20IS007**

Dear Student,

We would like to congratulate you on being selected for the **Machine Learning With Python (Research Based)** Internship position with **Varcons Technologies**, effective Start Date **11th August, 2023**, All of us are excited about this opportunity provided to you!

This internship is viewed as being an educational opportunity for you, rather than a part-time job. As such, your internship will include training/orientation and focus primarily on learning and developing new skills and gaining a deeper understanding of concepts of **Machine Learning With Python (Research Based)** through hands-on application of the knowledge you learn while you train with the senior developers. You will be bound to follow the rules and regulations of the company during your internship duration.

Again, congratulations and we look forward to working with you!.

Sincerely,

Spoorthi H C
Director
VARCONS TECHNOLOGIES
213, 2st Floor,
18 M G Road, Ulsoor,
Bangalore-560001

ACKNOWLEDGEMENT

This Internship is a result of accumulated guidance, direction and support of several important persons. We take this opportunity to express our gratitude to all who have helped us to complete the Internship.

We express our sincere thanks to our Principal Dr. Mohammed Riyaz Ahmed, for providing us adequate facilities to undertake this Internship.

We would like to thank our Head of Dept Dr. A. Syed Mustafa – ISE Dept, for providing us an opportunity to carry out Internship and for his valuable guidance and support.

We express our deep and profound gratitude to our guide, Mr. Sharavana K, AssociateProf, for their keen interest and encouragement at every step in completing the Internship. We would like to thank all the faculty members of our department for the support extended during the course of Internship.

Last but not the least, we would like to thank our parents and friends without whose constant help, the completion of Internship would have not been possible.

NAME: AKASH TATTI
USN : 1HK20IS007

ABSTRACT

The Covid-19 pandemic, which began in early 2020, led to unprecedented global challenges, including lockdowns implemented to curb the virus's spread. These lockdowns impacted individuals and communities in various ways, sparking a wide range of emotions and opinions. Social media platforms, particularly Twitter, became a prominent space for people to express their thoughts and sentiments during this period. This study presents a comprehensive analysis of public sentiment on Twitter in the USA during the Covid-19 lockdowns, employing machine learning techniques to extract valuable insights. The primary objective of this research is to understand the evolving sentiment trends over different phases of the lockdown and identify key factors influencing sentiment. To achieve this, we collected a vast dataset of tweets related to Covid-19 and the lockdown measures in the USA, spanning from the onset of the pandemic to the easing of restrictions. We employed Natural Language Processing (NLP) and machine learning algorithms to process and analyze this data. Our findings reveal distinct sentiment patterns over time, reflecting the evolving nature of the pandemic and public reactions. Initially, there was a surge in negative sentiment as the lockdowns disrupted daily life and raised concerns about health and economic stability. As time progressed, sentiment shifted towards a more balanced distribution, indicating adaptation and resilience within the community. Furthermore, the study explores the contextual factors contributing to sentiment shifts. We considered various factors, including the severity of Covid-19 cases, government policies, vaccination campaigns, economic relief measures, and public awareness campaigns. Through sentiment analysis, we were able to discern the impact of these factors on public sentiment, providing valuable insights for policymakers and public health officials. This research demonstrates the utility of machine learning and sentiment analysis in understanding the societal impact of major events like the Covid-19 pandemic. It not only sheds light on the emotional responses of the public during times of crisis but also offers a data-driven perspective on the effectiveness of interventions and communication strategies.

Table of Contents

CHAPTER 1

COMPANY PROFILE

1. COMPANY PROFILE

A Brief History of Company

Varcons technologies, was incorporated with a goal, To provide high quality and optimal Technological Solutions to business requirements of our clients". Every business is a different and has a unique business model and so are the technological requirements. They understand this and hence the solutions provided to these requirements are different as well. They focus on clients requirements and provide them with tailor made technological solutions. They also understand that Reach of their Product to its targeted market or the automation of the existing process into e-client and simple process are the key features that our clients desire from Technological Solution they are looking for and these are the features that we focus on while designing the solutions for their clients.

Varcons is a Technology Organization providing solutions for all web design and development, MYSQL, PYTHON Programming, HTML, CSS, ASP.NET and LINQ. Meeting the ever increasing automation requirements, Sarvamoola Software Services. specialize in ERP, Connectivity, SEO Services, Conference Management, effective webpromotion and tailor-made software products, designing solutions best suiting clients requirements.

we strive to be the front runner in creativity and innovation in software development through their well-researched expertise and establish it as an out of the box software development company in Bangalore, India. As a software development company, they translate this software development expertise into value for their customers through their professional solutions.

They understand that the best desired output can be achieved only by understanding the clients demand better. At our Company we work with them clients and help them to define their exact solution requirement. Sometimes even they wonder that they have completely redefined their solution or new application requirement during the brainstorming session, and here they position themselves as an IT solutions consulting group comprising of high caliber consultants.

They believe that Technology when used properly can help any business to scale and achieve new heights of success. It helps Improve its efficiency, profitability, reliability; to put itin one sentence Technology helps you to Delight your Customers and that is what we wantto achieve.

CHAPTER 2

ABOUT THE COMPANY

2. ABOUT THE COMPANY

We are a Technology Organization providing solutions for all web design and development, Researching and Publishing Papers to ensure the quality of most used ML Models, MYSQL, PYTHON Programming, HTML, CSS, ASP.NET and LINQ. Meeting the ever increasing automation requirements, Varcons Technologies specialize in ERP, Connectivity, SEO Services, Conference Management, effective web promotion and tailor-made software products, designing solutions best suiting clients requirements. The organization where they have a right mix of professionals as a stakeholders to help us serve our clients with best of our capability and with at par industry standards.They have young, enthusiastic, passionate and creative Professionals to develop technological innovations in the field of Mobile technologies, Web applications as well as Business and Enterprise solution. Motto of our organization is to "Collaborate with our clients to provide them with best Technological solution hence creating Good Present and Better Future for our client which will bring a cascading a positive effect in their business shape as well". Providing a Complete suite of technical solutions is not just our tag line, it is Our Vision for Our Clients and for Us, We strive hard to achieve it.

Services provided by Varcons Technologies.

- Core Java and Advanced Java
- Research and Development/Improvise of ML Models
- Web services and development
- Dot Net Framework
- Python
- Selenium Testing
- Conference / Event Management Service
- Academic Project Guidance
- On The Job Training
- Software Training

CHAPTER 3

INTRODUCTION

3. INTRODUCTION

Introduction to ML

Machine learning is a subfield of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that enable computers to learn from and make predictions or decisions based on data, without being explicitly programmed. It is a rapidly evolving and interdisciplinary field with applications in various domains, including healthcare, finance, marketing, natural language processing, computer vision, and more.

Overview of Machine Learning:

Fundamental Concepts:

Data: Machine learning relies heavily on data. This data can be structured (e.g., databases) or unstructured (e.g., text or images). The quality and quantity of data play a critical role in the success of a machine learning model.

Algorithm: Machine learning algorithms are the core components of the process. These algorithms are responsible for learning patterns and making predictions or decisions based on the data.

Model: A model is a representation of the knowledge or patterns learned from the data by a machine learning algorithm. Models can be as simple as linear regression or highly complex, like deep neural networks.

Types of Machine Learning

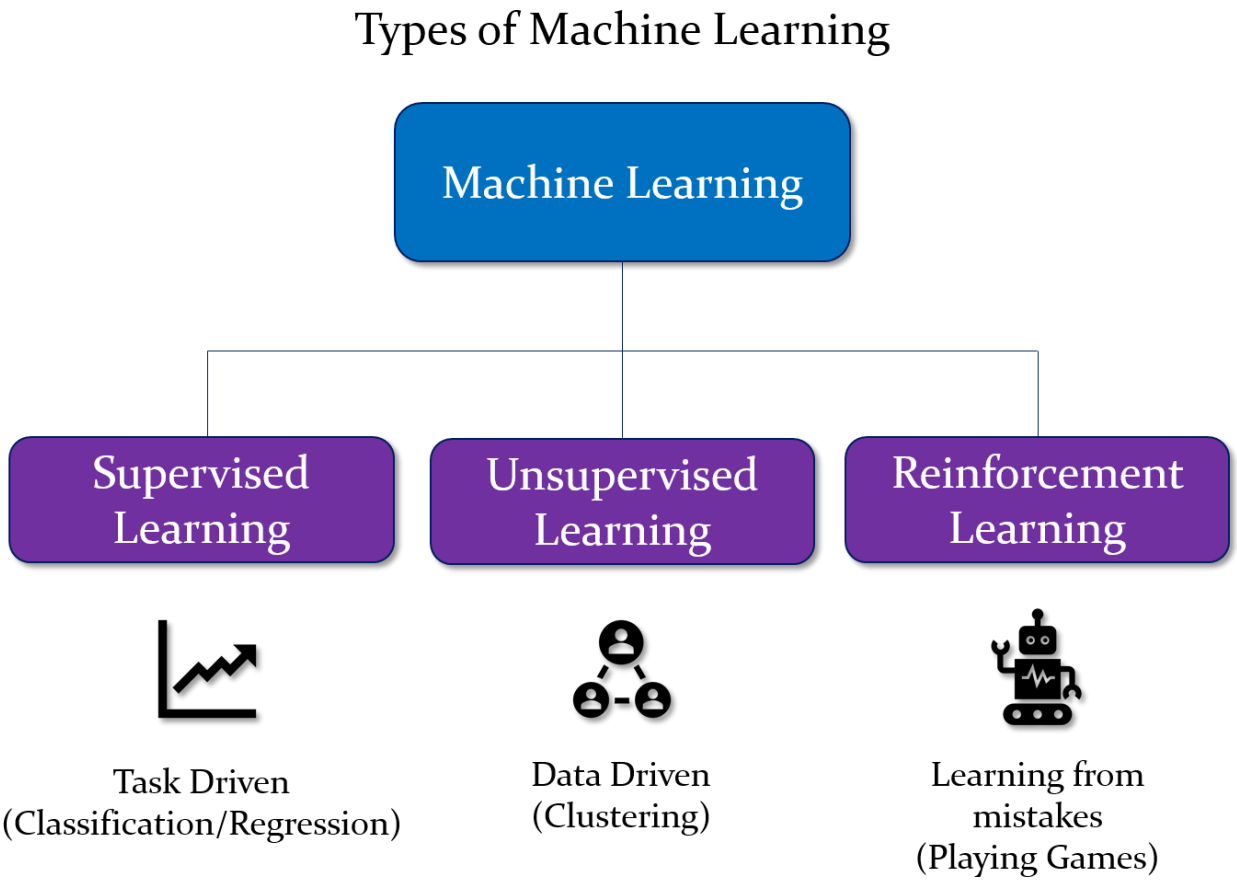


Fig 2.1: Types of machine learning

Supervised Learning: In supervised learning, the algorithm is trained on a labeled dataset, where the input data is paired with the correct output. The goal is to learn a mapping from inputs to outputs, making it suitable for tasks like classification and regression.

Unsupervised Learning: Unsupervised learning deals with unlabeled data. Algorithms try to identify patterns or structure within the data, such as clustering similar data points together or reducing data dimensionality.

Semi-Supervised Learning: This is a hybrid of supervised and unsupervised learning, where the model is trained on a combination of labeled and unlabeled data. It can be useful when acquiring labeled data is expensive or time-consuming.

Reinforcement Learning: In reinforcement learning, agents learn to make sequences of decisions to maximize a cumulative reward in an environment. This approach is often used in autonomous systems and gaming.

Steps in Machine Learning:

1. Data Collection: Gathering relevant data is the first step. This may involve data cleaning, preprocessing, and feature engineering to prepare the data for training.

2. Model Selection: Choosing the appropriate algorithm or model architecture for the task at hand is crucial. This selection depends on the nature of the data and the problem to be solved.

3. Training: During this phase, the model learns from the training data. It optimizes its internal parameters to make accurate predictions or decisions.

4. Evaluation: After training, the model is assessed using a separate dataset (validation or test data) to measure its performance. Common evaluation metrics include accuracy, precision, recall, F1-score, and more, depending on the task.

5. Hyperparameter Tuning: Fine-tuning hyperparameters (e.g., learning rate, batch size, depth of a neural network) can significantly impact a model's performance.

Challenges and Considerations:

- Overfitting and Underfitting: Balancing a model's ability to generalize from the data without fitting it too closely (overfitting) or too loosely (underfitting) is a common challenge.

- Bias and Fairness: Machine learning models can inherit biases from training data. Ensuring fairness and mitigating bias is essential, especially in applications like hiring or lending.

- Interpretability: Many complex machine learning models are not easily interpretable, which can be problematic in sensitive applications. Explainable AI (XAI) is an emerging field addressing this concern.

Applications:

- Recommendation Systems: Machine learning algorithms power recommendation engines used by platforms like Netflix, Amazon, and Spotify. They analyze user behavior and preferences to suggest movies, products, or music that users are likely to enjoy.
- Facial Recognition: Machine learning is used for facial recognition in security systems, unlocking smartphones, and even in personalized marketing.
- Video Surveillance: ML helps in analyzing video feeds for object detection, anomaly detection, and tracking in surveillance systems.
- Autonomous Vehicles: Self-driving cars rely on machine learning to process sensor data (such as lidar, radar, and cameras) to make real-time decisions, navigate, and avoid obstacles.
- Predictive Maintenance: In industries like manufacturing and aviation, machine learning predicts when equipment or machinery is likely to fail, allowing for timely maintenance and reducing downtime.
- E-commerce: ML is used for dynamic pricing, fraud detection, inventory management, and customer segmentation to enhance the online shopping experience.
- Healthcare: Beyond diagnosis, machine learning is used for predicting patient outcomes, drug discovery, genomic analysis, and personalized medicine.
- Energy Management: ML optimizes energy consumption in buildings, predicts energy demand, and helps in the integration of renewable energy sources into the grid.
- Natural Disaster Prediction: Machine learning models analyze historical data and sensor inputs to predict natural disasters like earthquakes, floods, and hurricanes, aiding in disaster preparedness.
- Language Translation: Machine translation services like Google Translate use neural machine translation models to provide accurate translations between multiple languages.
- Virtual Assistants: Voice-controlled virtual assistants like Siri, Alexa, and Google Assistant use natural language processing and machine learning to understand and respond to user queries.
- Fraud Detection: In the banking and finance sector, machine learning algorithms detect fraudulent transactions by identifying unusual patterns in transaction data.
- Agriculture: Machine learning is applied to crop management, soil analysis, and pest control, helping farmers optimize yields and reduce resource usage.
- Sports Analytics: ML is used to analyze player performance, predict game outcomes, and provide insights for sports teams and broadcasters.

- Human Resources: Machine learning aids in candidate screening, employee retention analysis, and workforce planning in HR departments.
- Climate Modeling: ML helps climate scientists analyze large datasets to make predictions about climate change, extreme weather events, and environmental impacts.

Problem Statement

The Covid-19 pandemic and subsequent lockdowns in the USA have triggered a multitude of emotional responses and opinions among the population. There is a need for an advanced sentiment analysis system that can accurately capture and analyze sentiment trends in Twitter data during different phases of the lockdown, enabling a deeper understanding of the factors influencing public sentiment.

CHAPTER 4

CLEANING OF THE DATA

4.CLEANING OF THE DATA

1. TOKENIZER:

The RegexpTokenizer is a class from the nltk.tokenize module in the Natural Language Toolkit (NLTK) library in Python. It is used for tokenizing text based on regular expressions (regex). Unlike the default word tokenizer in NLTK, which splits text based on spaces, the RegexpTokenizer allows you to specify custom regex patterns to tokenize text in a more flexible way.

Here's how to use it:

1. Import the RegexpTokenizer class:

from nltk.tokenize import RegexpTokenizer.

Create an instance of the RegexpTokenizer class by specifying the regular expression pattern you want to use for tokenization. For example, to tokenize text based on spaces, you can use the pattern \s+ which matches one or more whitespace characters:

2. tokenizer = RegexpTokenizer(r'\s+')

Use the tokenize() method of the RegexpTokenizer to tokenize your text:

text = "This is a sample sentence, and we want to tokenize it."

tokens = tokenizer.tokenize(text)

In this example, tokens will contain a list of words and punctuation, split based on spaces and other whitespace characters.

3. tokenizer = RegexpTokenizer(r'\b\w+\b')

This pattern \b\w+\b matches complete words.

After tokenization, you can further process or analyze the tokens as needed, such as counting word frequencies, performing text classification, or any other natural language processing tasks.

Here's a complete example:

```
from nltk.tokenize import RegexpTokenizer
# Create a tokenizer to split text based on word boundaries
tokenizer = RegexpTokenizer(r'\b\w+\b')
# Sample text
text = "This is a sample sentence, and we want to tokenize it."
# Tokenize the text
```

```
tokens = tokenizer.tokenize(text)
# Print the tokens
print(tokens)
```

Output:

```
['This', 'is', 'a', 'sample', 'sentence', 'and', 'we', 'want', 'to', 'tokenize', 'it']
```

The RegexpTokenizer is particularly useful when you need to tokenize text in a way that is not covered by the default word tokenizer. You can define custom regex patterns to suit your specific tokenization needs.

2. VECTORIZER:

CountVectorizer is a feature extraction technique used in natural language processing (NLP) and machine learning. It is commonly used to convert a collection of text documents into a matrix of token (word or n-gram) counts. This matrix is often referred to as the "document-term matrix." Each row of the matrix represents a document, and each column represents a unique word or n-gram found in the corpus of documents. The value in each cell of the matrix is the count of how many times a particular word or n-gram appears in a specific document.

CountVectorizer is a part of the scikit-learn library in Python, which is a powerful library for machine learning and data analysis.

Here's how you can use CountVectorizer:

1. Import CountVectorizer from scikit-learn:

```
from sklearn.feature_extraction.text import CountVectorizer
```

Create an instance of CountVectorizer with optional parameters:

2.
```
count_vectorizer = CountVectorizer()
```

You can specify various parameters to customize the vectorization process, such as specifying the maximum number of features (words or n-grams) to consider, using custom tokenization, setting stopwords, etc.

3. Fit and transform your text data using fit_transform():

```
documents = [
    "This is the first document.",
    "This document is the second document.",
    "And this is the third one.",
    "Is this the first document?" ]
```

4. X = count_vectorizer.fit_transform(documents)

X will be a sparse matrix representing the document-term matrix.

Now, you have a matrix that you can use for various NLP tasks, such as text classification, clustering, or other machine learning tasks. Each row corresponds to a document, and each column corresponds to a word or n-gram, with values indicating the count of each word or n-gram in each document.

CountVectorizer is a useful tool for converting text data into a format that machine learning algorithms can process. It is often used as a preprocessing step before training models like Naive Bayes, Logistic Regression, or Support Vector Machines on text data.

CHAPTER 5

SENTIMENTAL ANALYSIS

5. SENTIMENTAL ANALYSIS

SKLEARN:

scikit-learn, often abbreviated as sklearn, is a widely used open-source machine learning library in Python. It provides a simple and efficient tool for data analysis and modeling, including various machine learning algorithms for classification, regression, clustering, dimensionality reduction, and more. scikit-learn is built on top of other foundational libraries such as NumPy and SciPy and is designed to be easy to use while providing powerful functionality for machine learning tasks. Let's delve into the details of scikit-learn:

Key Components and Features of scikit-learn

- Machine Learning Algorithms: scikit-learn includes a vast collection of machines learning algorithms, ranging from traditional methods to modern techniques. Some of the prominent algorithms available include:
- Supervised Learning: Linear models (e.g., Linear Regression, Logistic Regression), Support Vector Machines, Decision Trees, Random Forests, k-Nearest Neighbors, and Naive Bayes, among others.
- Unsupervised Learning: K-Means clustering, Hierarchical clustering, Principal Component Analysis (PCA), Independent Component Analysis (ICA), and more.
- Dimensionality Reduction: Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and t-distributed Stochastic Neighbor Embedding (t-SNE).
- Model Selection and Evaluation: Tools for model selection, cross-validation, hyperparameter tuning, and evaluation metrics.
- Consistent API: scikit-learn provides a consistent and user-friendly API, making it easy to learn and switch between different machine learning algorithms and techniques. This consistency simplifies the process of experimenting with different models and pipelines.
- Data Preprocessing: scikit-learn offers a wide range of data preprocessing functions, including:

1. Feature scaling and standardization.
2. Handling missing values.
3. Encoding categorical variables.

4. Text feature extraction (e.g., TF-IDF).
5. Feature selection and extraction.

- Model Evaluation and Validation: scikit-learn provides tools for model evaluation and validation, including:
- Cross-validation techniques (e.g., k-fold cross-validation).
- Various metrics for classification (accuracy, precision, recall, F1-score) and regression (mean squared error, R-squared).

- Hyperparameter tuning using grid search and randomized search.
- Ensemble Methods: The library includes ensemble methods like Random Forests, Gradient Boosting, and AdaBoost, which combine the predictions of multiple base models to improve overall performance.
- Interoperability: scikit-learn can seamlessly integrate with other libraries in the Python ecosystem, such as pandas for data manipulation, Matplotlib for data visualization, and NumPy for numerical operations.
- Integration with Jupyter: scikit-learn is well-suited for interactive data analysis and model building in Jupyter notebooks. It provides user-friendly APIs and built-in support for Jupyter's rich output capabilities.
- Community and Documentation: scikit-learn has a large and active community of users and developers. It offers comprehensive documentation, tutorials, and online resources, making it accessible to both beginners and experienced data scientists.

How to Use scikit-learn

✓ Using scikit-learn typically involves the following steps:

✓ Data Preparation: Load and preprocess your data, including handling missing values, encoding categorical variables, and scaling or standardizing features.

✓ Model Selection: Choose an appropriate machine learning algorithm or model based on your problem type (classification, regression, clustering) and data characteristics.

✓ Model Training: Fit the chosen model to your training data using the fit method.

✓ Model Evaluation: Assess your model's performance using appropriate evaluation metrics and cross-validation techniques.

✓ Hyperparameter Tuning: Fine-tune your model by searching for optimal hyperparameters using tools like grid search or randomized search.

✓ Model Deployment: Once you are satisfied with your model's performance, deploy it for making predictions on new, unseen data.

Advantages of scikit-learn

- Ease of Use: scikit-learn's user-friendly API and consistent interface make it accessible to both novice and experienced machine learning practitioners.
- Comprehensive: It offers a wide range of machine learning algorithms, preprocessing tools, and evaluation metrics, covering a broad spectrum of machine learning tasks.
- Active Community: scikit-learn has a large and active user community, which ensures ongoing development, support, and a wealth of learning resources.
- Integration: It integrates seamlessly with other Python libraries, simplifying the process of data manipulation, visualization, and integration into data science pipelines.

Disadvantages of scikit-learn

- Limited Deep Learning Support: scikit-learn primarily focuses on traditional machine learning techniques and does not provide native support for deep learning. Users interested in deep learning may need to turn to other libraries like TensorFlow or PyTorch.
- Scaling to Big Data: While scikit-learn is efficient for many tasks, it may not scale well to very large datasets. For big data, distributed computing frameworks like Apache Spark may be more suitable.

CHAPTER 6

VISUALIZATION

6. VISUALIZATION

MATPLOTLIB:

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations in 2D and 3D. It provides a flexible and comprehensive set of functions for generating plots, charts, and graphs, making it a valuable tool for data visualization, scientific plotting, and creating publication-quality figures. Let's delve into the details of Matplotlib, including why to use it, its advantages, and potential disadvantages:

Why Use Matplotlib??

- Data Visualization: Matplotlib allows you to create a wide variety of visualizations, from simple line plots and scatter plots to complex heatmaps, bar charts, histograms, and 3D plots. Visualizing data is crucial for gaining insights and effectively communicating results.
- Customization: Matplotlib offers fine-grained control over plot elements, enabling you to customize every aspect of your visualizations. You can adjust colors, fonts, line styles, and marker shapes to create aesthetically pleasing and informative plots.
- Publication Quality: Matplotlib is widely used in scientific and academic research because it produces publication-quality figures that adhere to formatting standards. It supports LaTeX for mathematical expressions and text rendering, ensuring high-quality typography in plots.
- Interactivity: Matplotlib allows you to create interactive plots and embed them in applications or web pages using libraries like mpld3 and Bokeh. This makes it suitable for building interactive data dashboards and reports.
- Integration with Data Science Tools: Matplotlib integrates seamlessly with other Python libraries commonly used in data science, such as NumPy and pandas. It also plays a significant role in the data visualization ecosystem alongside libraries like Seaborn, Plotly, and ggplot.
- Community and Documentation: Matplotlib has an active community of users and developers, resulting in extensive documentation, tutorials, and online resources. This makes it relatively easy to find solutions to common plotting challenges.

Advantages of Using Matplotlib

- Wide Range of Plot Types: Matplotlib supports various plot types, including line plots, scatter plots, bar plots, histograms, pie charts, box plots, and more. This versatility accommodates different data visualization needs.
- Customization: You can fine-tune almost every aspect of your plots, such as colors, markers, labels, legends, and titles. This level of customization allows you to create visually appealing and informative plots.
- Cross-Platform Compatibility: Matplotlib is platform-independent, meaning you can create plots and visualizations on different operating systems (Windows, macOS, Linux) without compatibility issues.
- Large User Base: Matplotlib's popularity means there is a vast user base and a wealth of community-contributed resources, including user-generated scripts and plugins, which can simplify complex plotting tasks.
- Integration with Jupyter Notebooks: Matplotlib works seamlessly with Jupyter notebooks, enabling interactive and inline plotting within notebooks. This is especially useful for data exploration and analysis.

Disadvantages and Considerations

- Steep Learning Curve: Matplotlib's rich feature set can lead to a steep learning curve, especially for beginners. It may take some time to become proficient in customizing and creating complex plots.
- Verbose Syntax: Matplotlib's syntax can be verbose, requiring multiple lines of code to create and customize plots. This can make simple tasks seem more complex than they need to be.
- Non-Intuitive Defaults: While Matplotlib's flexibility is an advantage, its default settings for plots may not always produce the most aesthetically pleasing results. Users often need to adjust defaults to achieve desired outcomes.
- Limited 3D Plotting: While Matplotlib supports 3D plotting, creating complex 3D visualizations can be challenging, and more specialized libraries like Mayavi may be more suitable for advanced 3D graphics.
- Interactive Features: While Matplotlib can create interactive plots, it may not be as feature-rich as some dedicated interactive plotting libraries like Plotly or Bokeh for building complex web-based dashboards.

CHAPTER 7

Sentiment Analysis Of Lockdown In USA During Covid-19 A Case Study On Twitter using ML

7. Sentiment Analysis Of Lockdown In USA During Covid-19 A Case Study On Twitter using ML

The COVID-19 pandemic brought unprecedented challenges worldwide, leading governments to implement various containment measures, including lockdowns, to curb the spread of the virus. This study focuses on understanding the sentiment of the general population in the United States during the lockdown period through the analysis of Twitter data. We employ machine learning techniques to analyze the sentiment of tweets posted by users in the USA between the months of March and June 2020. To conduct sentiment analysis, we collected a vast dataset of tweets related to COVID-19 and lockdown using Twitter's API. The dataset comprises millions of tweets, capturing the diverse reactions and emotions of individuals during this critical period. We preprocessed the data by removing noise, cleaning, and tokenizing the text for further analysis.

Using Natural Language Processing (NLP) techniques and machine learning algorithms, we categorized the sentiment of these tweets into three main categories: positive, negative, and neutral. We also explored subcategories within these sentiments to gain a deeper understanding of public opinion, such as anxiety, frustration, resilience, and hope.

The results of our sentiment analysis reveal the dynamic and evolving nature of public sentiment throughout the lockdown period. We observed initial fear and anxiety during the early days of lockdown, followed by a mix of negative and positive sentiments as the pandemic progressed. Additionally, we identified significant regional variations in sentiment across the USA. This study provides valuable insights into the emotional responses of the American population during the COVID-19 lockdown, shedding light on the psychological impact of such unprecedented measures. Furthermore, the findings of this research can inform policymakers and public health authorities on how to effectively communicate and support the public during times of crisis. In conclusion, this case study showcases the power of machine learning and sentiment analysis in understanding public sentiment and reactions during challenging times. By analyzing Twitter data, we contribute to a better understanding of the human experience during the COVID-19 lockdown and offer insights that can aid in improving future crisis management strategies.

The project includes several key phases:

1. Project Initiation:

- Define the project's objectives and scope.
- Identify the stakeholders and their expectations.
- Create a project plan with timelines and milestones.

2. Data Collection and Preparation:

- Gather a diverse dataset of Twitter data containing tweets relevant to your research.
- Preprocess the data by cleaning and formatting tweets (e.g., removing special characters, URLs, and hashtags).
- Annotate or label the dataset with sentiment labels (positive, negative, neutral).

3. Exploratory Data Analysis (EDA):

- Perform exploratory data analysis to understand the characteristics of your dataset.
- Visualize tweet distributions, word frequencies, and sentiment class distributions.
- Identify any patterns or trends in the data.

4. Text Preprocessing:

- Tokenize the text data into words or subword tokens.
- Apply techniques like stemming or lemmatization to reduce word variations.
- Create word embeddings (e.g., Word2Vec, GloVe) for representing words in a numerical format.

5. Model Selection and Training:

- Choose machine learning algorithms suitable for sentiment analysis (e.g., LSTM, BERT).
- Split the dataset into training, validation, and testing sets.
- Train and fine-tune your sentiment analysis model using the training data.
- Optimize hyperparameters to improve model performance.

6. Model Evaluation:

- Evaluate the model's performance using various metrics (accuracy, precision, recall, F1-score).
- Conduct cross-validation to ensure robustness.
- Address overfitting or underfitting issues if they arise.

7. Model Interpretation and Feature Importance:

- Understand which words or features contribute most to sentiment predictions.
- Visualize feature importance using techniques like word clouds or SHAP values.

8. Real-time Twitter Data Collection:

- Set up a system to collect real-time Twitter data based on specific keywords, hashtags, or user accounts.

9. Deployment and Integration:

- Deploy the trained sentiment analysis model into a production environment.
- Create an API or interface for users to interact with the model.
- Integrate the model with real-time data collection to continuously analyze Twitter data.

10. Monitoring and Maintenance:

- Continuously monitor the model's performance in a production environment.
- Implement retraining strategies to keep the model up-to-date.
- Address any issues or changes in Twitter data distribution.

11. Ethical Considerations:

- Consider ethical implications related to data privacy, bias, and responsible AI.

By adhering to these guidelines, organizations and platforms conducting Twitter sentiment analysis can leverage this technology while upholding ethical standards, transparency, and user privacy, fostering trust among users and the broader community.

CHAPTER 8

SYSTEM ANALYSIS

8. SYSTEM ANALYSIS

1. Existing System

Before the advent of machine learning, sentiment analysis of lockdowns in the USA during Covid-19 on Twitter, or any other social media platform, relied primarily on traditional, rule-based, and lexicon-based methods. These approaches were less sophisticated but provided some insights into sentiment trends. Here's how sentiment analysis was conducted before machine learning became prevalent:

1. Lexicon-Based Analysis: Lexicon-based sentiment analysis relies on predefined sentiment dictionaries or lexicons containing words or phrases with associated sentiment scores (e.g., positive, negative, neutral). Researchers would create or utilize existing sentiment lexicons tailored to social media text and the specific context of Covid-19 and lockdowns.

2.Rule-Based Approaches: Researchers would develop rules or heuristics to identify sentiment-bearing words, phrases, or patterns in tweets. For example, identifying sentiment based on the presence of positive or negative emoticons or the use of certain keywords.

Rule-based systems might incorporate grammatical and syntactic rules to capture sentiment nuances, such as negation handling (e.g., "not happy" vs. "happy").

3. Sentiment Scoring: Each tweet would be analyzed, and its sentiment score would be computed by aggregating the sentiment scores of the words or phrases found in the text.

The sentiment score could be a simple binary value (positive/negative), a continuous value (-1 to 1), or categorical (e.g., positive, negative, neutral).

4. Handling Ambiguity: Lexicon-based methods often struggled with ambiguity and context. For instance, a word like "sick" might have a negative sentiment in a health context but a positive one in a slang context ("That concert was sick!"). Researchers would use contextual information and additional rules to disambiguate sentiment.

6. Scaling Up: Analyzing a large volume of tweets manually or with rule-based methods was labor-intensive and often not scalable. Researchers would typically focus on smaller subsets or perform manual sampling.

7. Visualizations and Reports: Results of sentiment analysis were often presented in the form of summary statistics, word clouds, and sentiment trend charts.

Reports and visualizations helped convey overall sentiment trends and patterns to stakeholders.

8. Challenges: Traditional sentiment analysis struggled to capture sarcasm, irony, and cultural nuances effectively. Handling evolving language and emerging slang terms was a challenge.

9. Subjectivity: Sentiment analysis was inherently subjective, and different analysts might assign different sentiment scores to the same text.

While these traditional methods had limitations, they provided valuable insights into sentiment trends during the Covid-19 lockdowns. However, they lacked the scalability and sophistication of modern machine learning approaches. The introduction of machine learning and deep learning techniques has significantly improved the accuracy and efficiency of sentiment analysis, allowing for the analysis of large-scale social media data with higher precision and the ability to capture complex sentiment patterns.

2. Proposed System

A proposed system for sentiment analysis of lockdowns in the USA during Covid-19 using Machine Learning (ML) and Twitter data would leverage the power of ML algorithms to improve accuracy, scalability, and the ability to capture nuanced sentiment patterns. Here's an outline of the proposed system:

1. Data Collection:

Utilize the Twitter API to collect a vast and diverse dataset of tweets related to Covid-19 lockdowns in the USA. This dataset should span different phases of the lockdown and include a wide range of user-generated content.

2. Data Preprocessing: Clean and preprocess the Twitter data by removing noise, handling missing values, and normalizing text. Preprocessing steps might also include tokenization, stemming, and removing stopwords.

3. Feature Extraction:

Use advanced feature extraction techniques, such as TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings (e.g., Word2Vec, GloVe), to convert text data into numerical features that can be used by ML models.

4. Sentiment Analysis Models:

Employ machine learning models for sentiment analysis. Potential models include:

Naive Bayes: A simple but effective probabilistic model.

Support Vector Machines (SVM): A versatile classification algorithm.

Recurrent Neural Networks (RNNs): Particularly LSTM (Long Short-Term Memory) networks.

Transformer-Based Models: Such as BERT (Bidirectional Encoder Representations from Transformers) fine-tuned for sentiment analysis.

5. Training and Evaluation: Split the dataset into training, validation, and test sets for model training and evaluation.

Use appropriate evaluation metrics (e.g., accuracy, F1-score, ROC AUC) to assess model performance.

Implement techniques like cross-validation and hyperparameter tuning to optimize model performance.

6. Time-Series Analysis: Conduct time-series analysis to track sentiment trends over time. This analysis can help identify significant events or changes in public sentiment during different phases of the lockdown.

7. Interpretability and Explainability: Explore methods for interpreting model predictions and understanding which words or phrases contribute to sentiment classifications. Techniques like LIME (Local Interpretable Model-agnostic Explanations) can be valuable.

8. Visualization: Create visualizations (e.g., line charts, heatmaps) to illustrate sentiment trends and patterns over time. Visualizations help in conveying insights effectively.

9. Real-Time Sentiment Analysis: Implement a real-time sentiment analysis module that can continuously analyze incoming tweets and provide sentiment scores in near real-time. This could be deployed as an API or a web-based dashboard.

10. Scalability: Design the system to be scalable, allowing it to handle a large volume of incoming Twitter data efficiently. Consider cloud-based solutions for scalability.

11. Ethical Considerations: Address ethical concerns such as user privacy, potential biases in data and models, and responsible data handling practices.

12. User-Friendly Interface: Develop a user-friendly web application where users can input text or hashtags to obtain sentiment analysis results. Provide interactive visualizations and exportable reports.

13. Continuous Improvement: Implement mechanisms for continuous improvement by periodically retraining the model with new data to adapt to evolving language and sentiment trends.

14. Security Measures: Ensure the system is secure and resistant to potential attacks, especially if it involves handling sensitive data.

15. Compliance: Ensure compliance with relevant data protection regulations and terms of service of data sources (e.g., Twitter's API).

This proposed system leverages the capabilities of machine learning to provide accurate and timely sentiment analysis of Twitter data related to Covid-19 lockdowns in the USA. It addresses data-specific challenges, scalability, interpretability, and user accessibility, making it a valuable tool for understanding public sentiment during critical events like the Covid-19 pandemic.

3. Objective of the System

The objectives of the proposed sentiment analysis system for lockdowns in the USA during Covid-19 using Machine Learning (ML) and Twitter data are as follows:

1. Sentiment Classification: Develop accurate machine learning models capable of classifying Twitter data into sentiment categories such as positive, negative, and neutral.
2. Real-Time Analysis: Implement a real-time sentiment analysis component to provide timely insights into public sentiment as it evolves during different phases of the Covid-19 lockdowns.
3. Fine-Grained Analysis: Enable the system to perform fine-grained sentiment analysis, allowing it to capture nuanced emotions and differentiate between various degrees of sentiment positivity or negativity.
4. Data Collection: Collect and curate a large and diverse dataset of Twitter data related to Covid-19 lockdowns in the USA, spanning different time periods and geographic regions.
5. Data Preprocessing: Preprocess the Twitter data to clean noise, handle missing values, and normalize text, ensuring high-quality input for the ML models.
6. Feature Engineering: Employ advanced feature extraction techniques to transform text data into numerical features, making it suitable for ML model training.
7. Model Selection and Training: Choose and train ML models that are effective at capturing sentiment from textual data, and optimize their performance using appropriate techniques.
8. Interpretability: Develop methods for interpreting model predictions and understanding the key factors contributing to sentiment classifications, enhancing the system's transparency and trustworthiness.
9. Time-Series Analysis: Conduct time-series analysis to identify significant events, trends, and sentiment shifts during different stages of the Covid-19 lockdowns.

10. Visualization: Create informative visualizations (e.g., charts, graphs, heatmaps) to convey sentiment trends and patterns effectively to users.
11. Scalability: Design the system to handle a large volume of incoming Twitter data efficiently, ensuring that it can scale to accommodate increased demand.
12. Ethical Considerations: Address ethical concerns related to user privacy, data bias, and responsible AI usage, implementing safeguards to mitigate potential issues.
13. User-Friendly Interface: Develop a user-friendly web application or API where users can interact with the system, input queries, and access sentiment analysis results and visualizations.
14. Continuous Improvement: Establish mechanisms for continuous model improvement by periodically retraining the ML models with new data to adapt to evolving language and sentiment trends.
15. Security Measures: Implement security measures to protect user data, ensuring that the system is resistant to potential security threats and attacks.
16. Compliance: Ensure compliance with relevant data protection regulations and adhere to the terms of service of data sources, such as Twitter's API.
17. User Feedback: Encourage user feedback and engagement to gather insights for system enhancement and refinement.
18. Deployment: Deploy the system on a reliable and scalable infrastructure, making it accessible to a broad audience, including researchers, policymakers, and the public.

These objectives collectively aim to create a robust sentiment analysis system that can effectively capture and communicate public sentiment trends during the Covid-19 lockdowns in the USA, contributing to a better understanding of public reactions and opinions during such critical events.

CHAPTER 9

REQUIREMENT ANALYSIS

9. REQUIREMENT ANALYSIS

Hardware Requirement Specification:

1. CPU: A modern multi-core processor (e.g., Intel Core i5 or higher, AMD Ryzen) is sufficient for small-scale sentiment analysis. For larger datasets and more complex machine learning models, a high-performance CPU or even a server-grade CPU may be necessary.
2. RAM: The amount of RAM you need depends on the size of your dataset and the complexity of your machine learning models. At least 8GB of RAM is recommended for small to medium-sized projects, but for larger datasets and models, 16GB or more may be necessary.
3. Storage: You will need enough storage space to store your Twitter dataset and any intermediate files generated during data preprocessing and model training. Solid-state drives (SSDs) are recommended for faster data access.
4. GPU (Optional): While not strictly necessary, using a dedicated GPU (Graphics Processing Unit) can significantly speed up the training of deep learning models, which are commonly used for sentiment analysis. NVIDIA GPUs are popular choices for machine learning tasks.

Software Requirement Specification

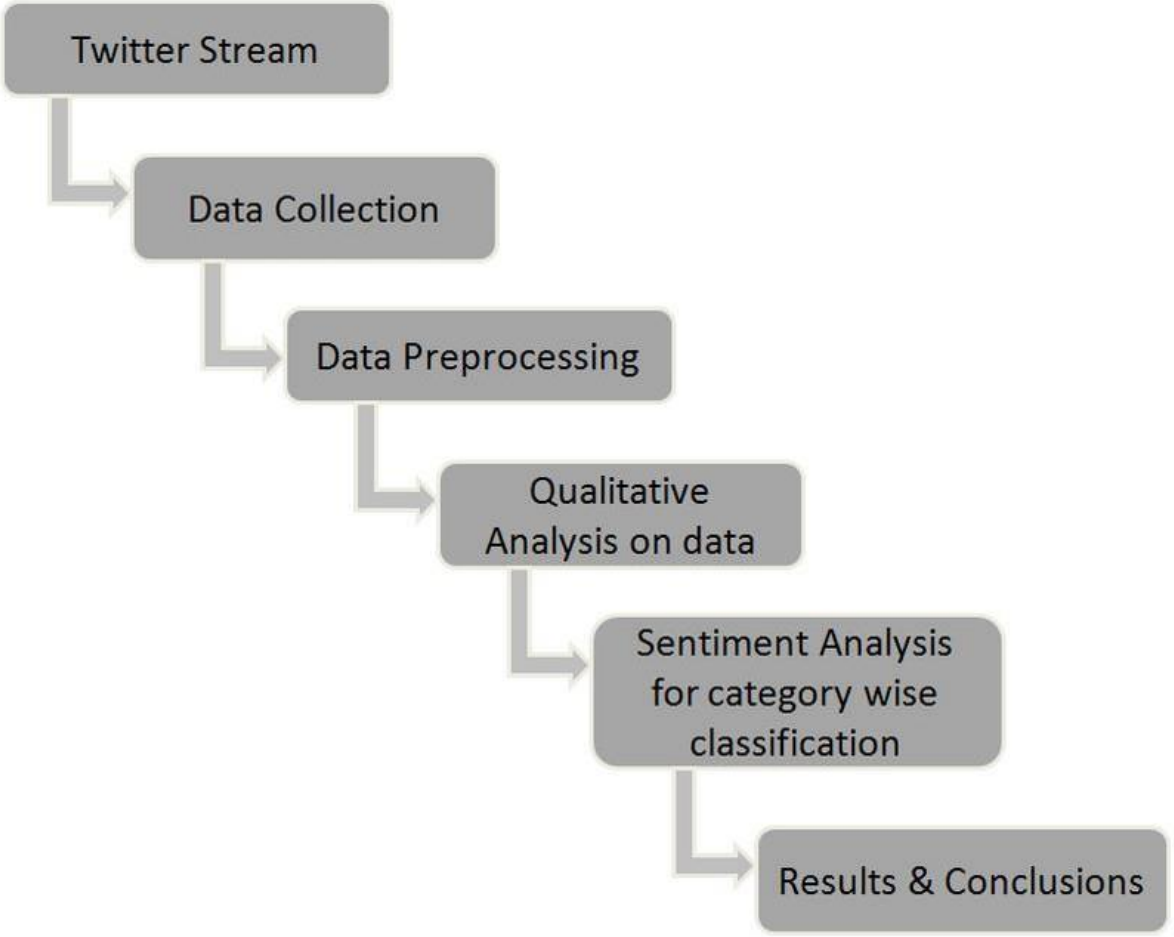
1. Operating System: You can perform sentiment analysis on various operating systems, including Windows, macOS, and Linux. The choice of the operating system often depends on your personal preference and the specific machine learning libraries you plan to use.
2. Python: Python is the most commonly used programming language for machine learning and natural language processing tasks. You'll need Python installed on your system.
3. Machine Learning Libraries: You'll need several Python libraries for machine learning and natural language processing, including but not limited to:
   - scikit-learn: For building machine learning models.
   - NLTK (Natural Language Toolkit): For text processing and tokenization.
4. Jupyter Notebook or IDE: An integrated development environment (IDE) like Jupyter Notebook or tools like Visual Studio Code can make it easier to write, test, and visualize your code.

5. Text Editors: You may also need a good text editor for writing code and scripts.
6. Version Control (Optional): Using a version control system like Git and hosting your project on platforms like GitHub can help you manage your code and collaborate with others.
7. Virtual Environment (Optional): It's a good practice to create a virtual environment to manage project-specific dependencies and avoid conflicts between libraries.

CHAPTER 10

DESIGN ANALYSIS

10. DESIGN ANALYSIS



Process of conducting a comprehensive sentiment analysis of Twitter data during Covid-19 lockdowns in the USA, including Twitter stream data collection, data preprocessing, qualitative analysis of data, sentimental analysis for category-wise classification, and concluding with results and conclusions:

1. Twitter Stream Data Collection: Design a system that continuously collects Twitter data using the Twitter Streaming API. You can specify keywords, hashtags, or user accounts related to Covid-19 lockdowns in the USA to filter relevant tweets. Implement a mechanism to store the incoming tweets in a database or data storage system for further analysis.

2. Data Preprocessing:

Clean the collected Twitter data by:

- Removing duplicate tweets.

- Handling missing data (e.g., missing text, user information).
- Normalizing text (e.g., converting text to lowercase).
- Removing special characters and punctuation.
- Tokenizing text into words or phrases.
- Removing stopwords (common words like "the," "and," "is").
- Lemmatizing or stemming words to reduce them to their base forms.

3. Qualitative Analysis of Data: Manually review a sample of preprocessed tweets to gain qualitative insights into the data. Identify common themes, topics, and trends related to Covid-19 lockdowns in the USA based on tweet content. Note any prevalent user sentiments, emotional expressions, or notable events mentioned in the tweets.

4. Sentimental Analysis for Category-wise Classification:

Perform category-wise classification of tweets based on their sentiment. Common categories may include:

- Positive sentiment: Tweets expressing optimism, hope, or positive experiences during the lockdown.
- Negative sentiment: Tweets expressing frustration, fear, or negative experiences during the lockdown.
- Neutral sentiment: Tweets that do not strongly convey positive or negative emotions.

Utilize machine learning models (e.g., supervised classifiers) trained on labeled data or lexicon-based methods to classify tweets into sentiment categories.

5. Result and Conclusion:

Generate sentiment analysis results:

- Quantify the distribution of tweets across sentiment categories.
- Visualize sentiment trends over time using line charts or histograms.
- Analyze how sentiment changes during different phases of the lockdown or in response to specific events or policies.
- Perform a sentiment breakdown for specific hashtags, keywords, or user accounts.

Conclude by summarizing key findings and insights:

- Provide an overview of the prevailing sentiment during the Covid-19 lockdowns in the USA based on Twitter data.
- Discuss any significant trends or shifts in sentiment.
- Offer qualitative observations from the qualitative analysis of data.
- Consider the impact of user demographics or geographic locations on sentiment.
- Offer insights into how these findings can inform decision-making, policy adjustments, or public health initiatives during a crisis.

Highlight any limitations of the analysis, such as potential bias in the data or challenges in sentiment classification. Suggest areas for future research or enhancements to the sentiment analysis methodology.

In summary, this comprehensive analysis involves collecting Twitter data, preprocessing it, conducting qualitative analysis, categorizing sentiment, and presenting results and conclusions. Such an analysis can provide valuable insights into the public's emotional response and opinions during Covid-19 lockdowns in the USA, aiding in informed decision-making and policy adjustments.

CHAPTER 11

IMPLEMENTATION

11. IMPLEMENTATION

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after thorough testing is done and if it is found to work according to the specification. It involves careful planning, investigation of the current system and it constraints on implementation, design of methods to achieve the change over and an evaluation of change over methods a part from planning. Two major tasks of preparing the implementation are education and training of the users andtesting of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required just for implementation.

The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

TESTING

The testing phase is an important part of software development. It is the Information zed system will help in automate process of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. Software testing is carried out in three steps:

1. The first includes unit testing, where in each module is tested to provide its correctness, validity and also determine any missing operations and to verify whether theobjectives have been met. Errors are noted down and corrected immediately.

2. Unit testing is the important and major part of the project. So errors are rectified easily in particular module and program clarity is increased. In this project entire system is divided into several modules and is developed individually. So unit testing is conducted to individual modules.

3. The second step includes Integration testing. It need not be the case, the software whose modules when run individually and showing perfect results, will also show perfect results when run as a whole.

CHAPTER 12

SNAPSHOTS

12. SNAPSHOTS

```
X_train = ["This was really awesome an awesome movie",
           "Great movie! Ilikes it a lot",
           "Happy Ending! Awesome Acting by hero",
           "loved it!",
           "Bad not upto the mark",
           "Could have been better",
           "really Dissapointed by the movie"]
#X_test = ["it was really awesome and really disspntd"]

y_train = ["positive","positive","positive","positive","negative","negative","negative"] # 1- Positive class, 0- negative class
```

Loading...

```
[4] from nltk.tokenize import RegexpTokenizer
    # NLTK -> Tokenize -> RegexpTokenizer

[ ] # Stemming
    # "Playing" -> "Play"
    # "Working" -> "Work"

    from nltk.stem.porter import PorterStemmer
    # NLTK -> Stem -> Porter -> PorterStemmer

    from nltk.corpus import stopwords
    # NLTK -> Corpus -> stopwords
```

Close

```
[6] # Downloading the stopwords
    import nltk
    nltk.download('stopwords')

    [nltk_data] Downloading package stopwords to /root/nltk_data...
    [nltk_data]   Unzipping corpora/stopwords.zip.
    True

[7] tokenizer = RegexpTokenizer(r"\w+")
    en_stopwords = set(stopwords.words('english'))
    ps = PorterStemmer()
```

```
def getCleanedText(text):
  text = text.lower()

  # tokenizing
  tokens = tokenizer.tokenize(text)
  new_tokens = [token for token in tokens if token not in en_stopwords]
  stemmed_tokens = [ps.stem(tokens) for tokens in new_tokens]
  clean_text = " ".join(stemmed_tokens)
  return clean_text
```

```
[9] X_test = ["It was good"]
```

```
X_clean = [getCleanedText(i) for i in X_train]
xt_clean = [getCleanedText(i) for i in X_test]
```

```
[11] X_clean
```

```
['realli awesom awesom movi',
 'great movi ilik lot',
 'happi end awesom act hero',
 'love',
 'bad upto mark',
 'could better',
 'realli dissapoint movi']
```

```
[12] xt_clean
```

```
['good']
```

```
[14] from sklearn.feature_extraction.text import CountVectorizer
```

```
[15] cv = CountVectorizer(ngram_range = (1,2))
     # "I am PyDev" -> "i am", "am Pydev"
```

```
[16] X_vec = cv.fit_transform(X_clean).toarray()
```

```
X_vec
```

```
array([[0, 0, 2, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 1, 0, 1, 1, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1,
        1, 0, 0, 1, 1, 0, 0, 0, 0, 0],
       [1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 1, 0, 0, 0, 0, 0, 1, 1],
       [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 1, 0, 1, 0, 1, 0, 0]])
```

```
print(cv.get_feature_names_out())
```

```
['act' 'act hero' 'awesom' 'awesom act' 'awesom awesom' 'awesom movi'
 'bad' 'bad upto' 'better' 'could' 'could better' 'dissapoint'
 'dissapoint movi' 'end' 'end awesom' 'great' 'great movi' 'happi'
 'happi end' 'hero' 'ilik' 'ilik lot' 'lot' 'love' 'mark' 'movi'
 'movi ilik' 'realli' 'realli awesom' 'realli dissapoint' 'upto'
 'upto mark']
```

```
[19] Xt_vect = cv.transform(xt_clean).toarray()
```

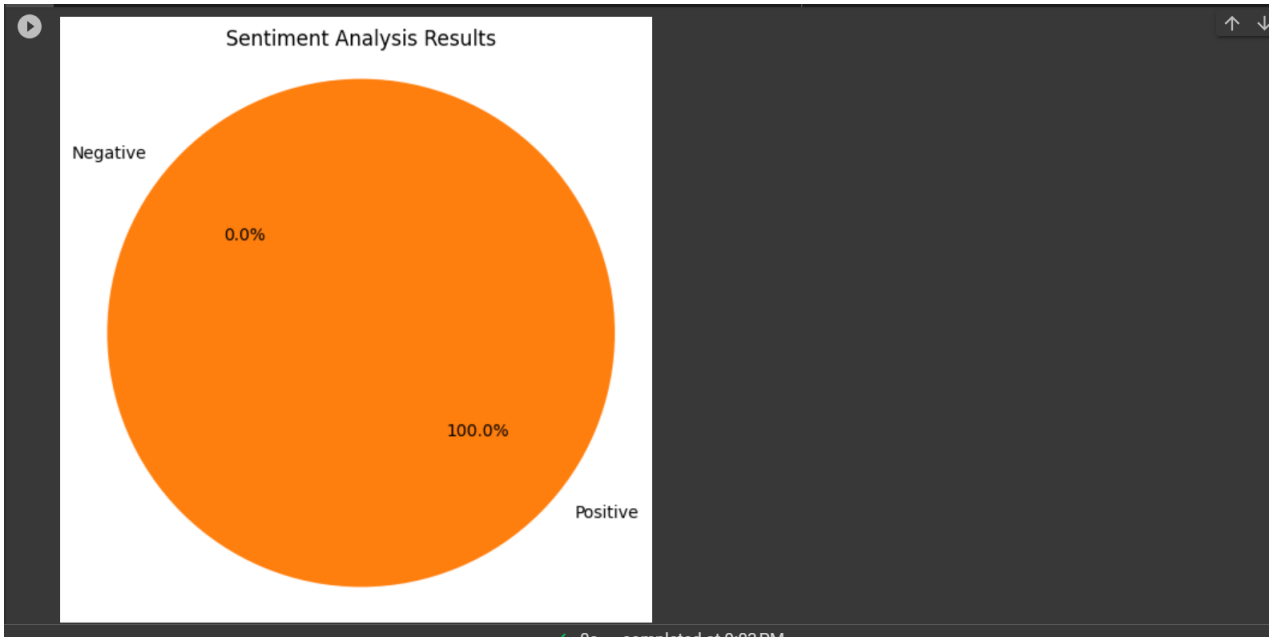
```
[20] Xt_vect
```

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

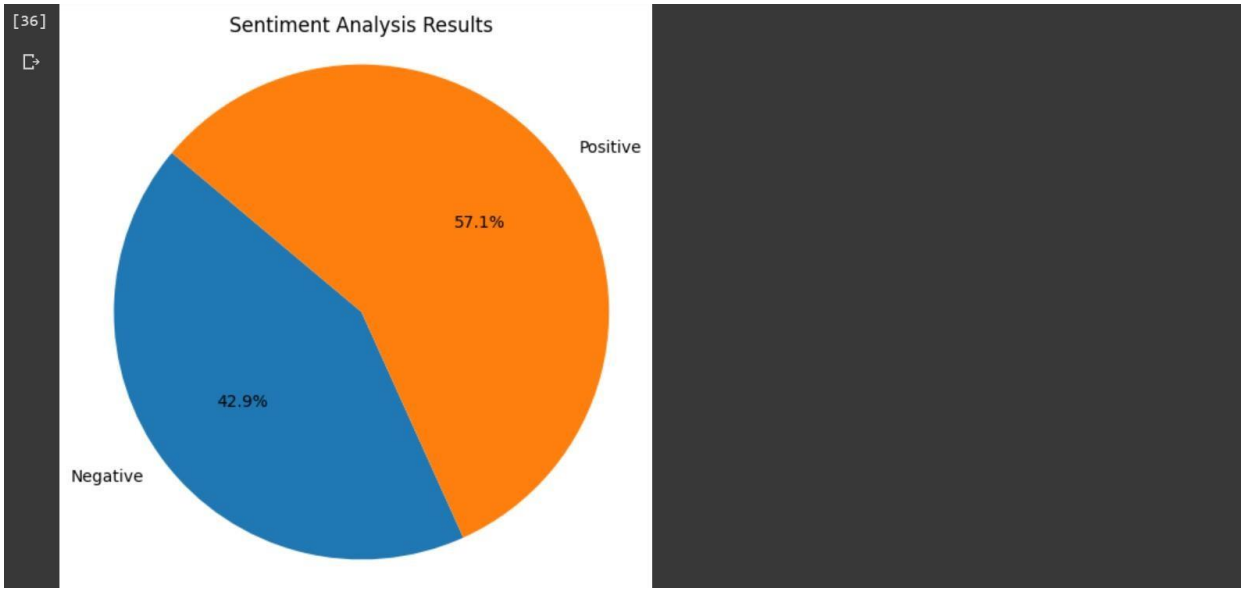
```
[21] from sklearn.naive_bayes import MultinomialNB

[22] mn = MultinomialNB()

     mn.fit(X_vec, y_train)

     MultinomialNB
     MultinomialNB()
```

```
     y_pred = mn.predict(X_vec)

[26] y_pred

     array(['positive', 'positive', 'positive', 'positive', 'negative',
            'negative', 'negative'], dtype='<U8')

[27] y_pred = mn.predict(Xt_vect)

[28] y_pred

     array(['positive'], dtype='<U8')

[29] import matplotlib.pyplot as plt
```

```
[30] mn.fit(X_vec, y_train)

     MultinomialNB
     MultinomialNB()

[31] y_pred = mn.predict(Xt_vect)

[32] labels = ['Negative', 'Positive']
     count_negative = (y_pred == 'negative').sum()
     count_positive = (y_pred == 'positive').sum()
     sentiment_counts = [count_negative, count_positive]

     plt.figure(figsize=(6, 6))
     plt.pie(sentiment_counts, labels=labels, autopct='%1.1f%%', startangle=140)
     plt.title('Sentiment Analysis Results')
     plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
     plt.show()
```




```
[34] y_pred = mn.predict(X_vec)

[35] labels = ['Negative', 'Positive']
     count_negative = (y_pred == 'negative').sum()
     count_positive = (y_pred == 'positive').sum()
     sentiment_counts = [count_negative, count_positive]

[36] plt.figure(figsize=(6, 6))
     plt.pie(sentiment_counts, labels=labels, autopct='%1.1f%%', startangle=140)
     plt.title('Sentiment Analysis Results')
     plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
     plt.show()
```



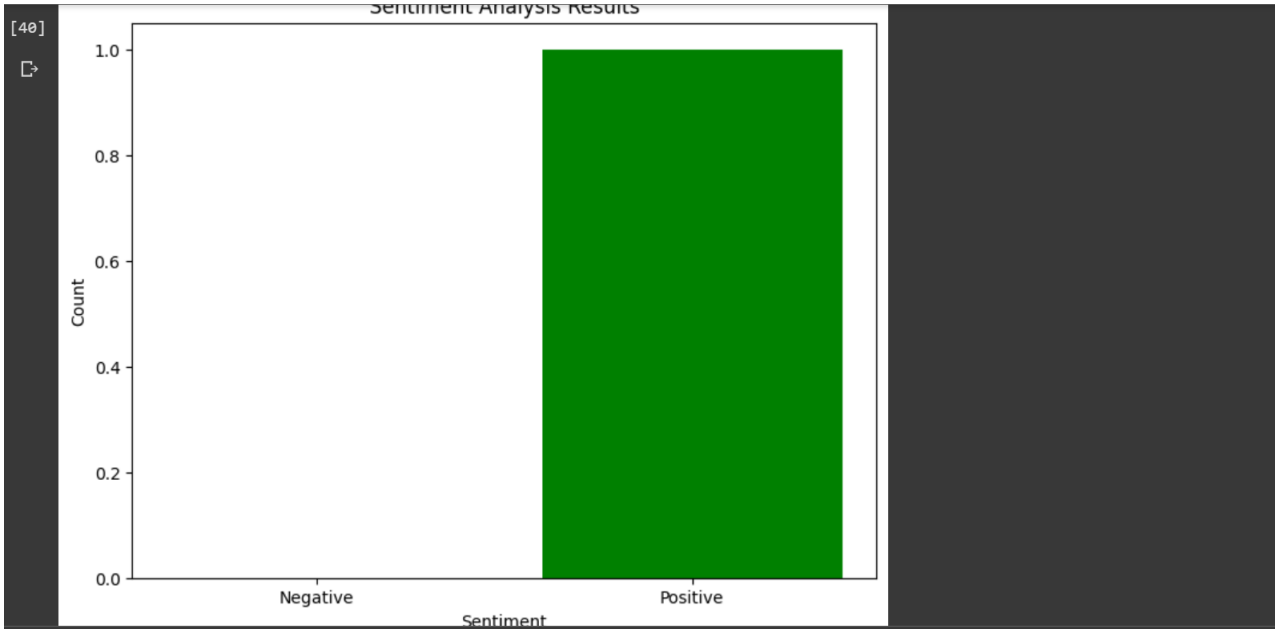
```
[37] mn.fit(X_vec, y_train)
```

```
MultinomialNB
MultinomialNB()
```

```
[38] y_pred = mn.predict(Xt_vect)
```

```
[39] labels = ['Negative', 'Positive']
     count_negative = (y_pred == 'negative').sum()
     count_positive = (y_pred == 'positive').sum()
     sentiment_counts = [count_negative, count_positive]
```

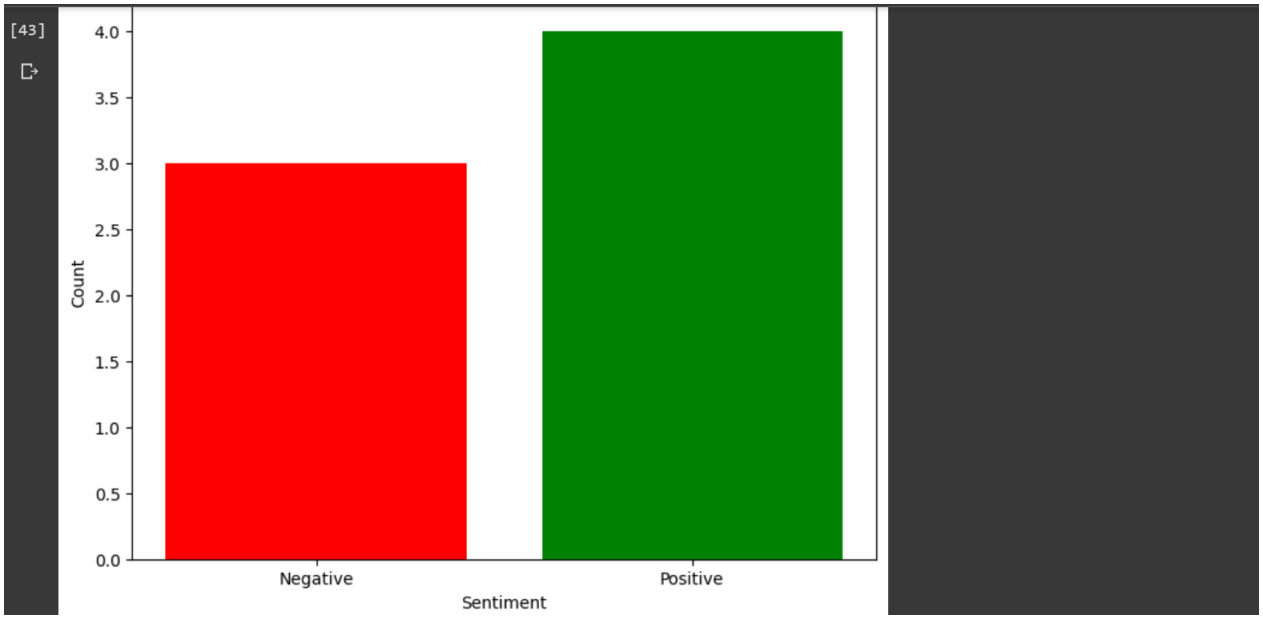
```
[40] plt.figure(figsize=(8, 6))
     plt.bar(labels, sentiment_counts, color=['red', 'green'])
     plt.xlabel('Sentiment')
     plt.ylabel('Count')
     plt.title('Sentiment Analysis Results')
     plt.show()
```




```
[41] y_pred = mn.predict(X_vec)
```

```
[42] labels = ['Negative', 'Positive']
     count_negative = (y_pred == 'negative').sum()
     count_positive = (y_pred == 'positive').sum()
     sentiment_counts = [count_negative, count_positive]
```

```
[43] plt.figure(figsize=(8, 6))
     plt.bar(labels, sentiment_counts, color=['red', 'green'])
     plt.xlabel('Sentiment')
     plt.ylabel('Count')
     plt.title('Sentiment Analysis Results')
     plt.show()
```


[43]
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Count
Negative
Positive
Sentiment

CHAPTER 13

CONCLUTION

13. CONCLUTION

Twitter sentiment analysis through machine learning techniques serves as a potent tool for understanding and harnessing public sentiment in the digital age. With the explosive growth of social media, Twitter has become a microcosm of real-time public opinion, making it a rich source of data for businesses, researchers, and policymakers. Through data collection, preprocessing, and the application of machine learning algorithms, sentiment analysis empowers us to gauge the prevailing emotions within the Twitterverse. Whether it's tracking reactions to a product launch, monitoring public sentiment during a crisis, or assessing political discourse, sentiment analysis unveils trends and insights that can inform strategies and decision-making. However, it is crucial to recognize the challenges, including the need for representative datasets, addressing class imbalances, and accounting for the nuances of human language such as sarcasm and irony. Additionally, staying updated with the evolving linguistic landscape on Twitter is essential. Sentiment analysis is not limited to positive, negative, or neutral classifications. Subtler sentiments like anxiety, excitement, or frustration can also be detected, providing a deeper understanding of public sentiment. In an era where public perception can shape reputations, influence stock markets, and sway elections, Twitter sentiment analysis using machine learning is an indispensable tool for businesses, governments, and researchers alike. It enables us to navigate the vast sea of social media data, extracting actionable insights and helping us make more informed decisions in an ever-connected world.

CHAPTER 14

REFERENCES

14. REFERENCES

- https://www.geeksforgeeks.org/machine-learning/
- https://en.m.wikipedia.org/wiki/Machine_learning
- https://chat.openai.com/c/d0496bac-1bf3-4840-a79a-db1d601a2eb4
- https://www.javatpoint.com/machine-learnings.