InfosysSpringboard Internship 4.0 Project

Documentation

TEXT CLASSIFICATION AND SENTIMENT ANALYSIS USING MACHINE LEARNING

*Submitted by*

|  |  |
| --- | --- |
| INTERN NAME | LAKSHITA M |
| MENTOR | SUDHEER KUMAR Y |



**APRIL-MAY 2024**

# ABSTRACT

This study investigates the performance of various machine learning models for text classification tasks. We evaluate the effectiveness of Support Vector Machine (SVM), Logistic Regression, Random Forest, XGBoost, LightGBM, Naive Bayes, K-Nearest Neighbors (KNN), and Decision Tree on two datasets: one balanced and one imbalanced.

The project explores the trade-off between accuracy and handling imbalanced data. While models generally achieve high accuracy on the balanced dataset, imbalanced data necessitates a shift towards metrics like precision, recall, and F1-score to assess performance, particularly for the minority class.

Our findings indicate that SVM and Logistic Regression achieve high accuracy on the balanced dataset. For the imbalanced dataset, models like SVM, Random Forest, and XGBoost are promising candidates based on their ability to handle complex data. We recommend further evaluation with precision, recall, and F1-score to identify the model that best classifies the minority class in the imbalanced scenario. Additionally, techniques like cost-sensitive learning and data balancing (oversampling/undersampling) might be necessary to improve performance.

This project provides insights into selecting appropriate text classification models for balanced and imbalanced datasets, highlighting the importance of considering both accuracy and the specific data distribution when making a final decision

## TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO.** | **TITLE** | **PAGE**  **NO** |
|  | **ABSTRACT** | ***ii*** |
|  | **LIST OF FIGURES** | ***v*** |
|  | **LIST OF TABLE** | ***1*** |
| **1.** | **INTRODUCTION** | **1** |
|  | 1.1 Requirements | 1 |
|  | 1.2 Goal and Objective | 1 |
|  | 1.3 Data Science Process | 2 |
|  | 1.3.1 Data Preparation | 2 |
|  | 1.3.2 Data Exploration | 2 |
|  | 1.3.3 Data Modelling | 3 |
|  | 1.3.4 Presentation and automation | 4 |
|  | 1.4 Introduction to Sentiments | 5 |
|  | 1.5 Sentiment Analysis Using Lexicon | 6 |
|  | 1.6 Emotion Analyzer | 7 |
| **2.** | **TEXT CLASSIFICATION** | **8** |
| **3.** | **PROBLEM STATEMENT AND USE CASES** | **9** |
| **4.** | **PROPOSED SOLUTION** | **12** |
|  | **4.1 Building Text Classifiers** | 15 |
|  | **4.1.1 Data Overview** | 16 |
|  | **4.1.2 Text Preprocessing** | 17 |
|  | 4.1.2.1 Text Preprocessing | 17 |
|  | 4.1.2.2 Text Preprocessing Steps | 18 |
|  | **4.1.3 Text Representation** | 19 |
|  | 4.1.3.1 Need of Representation | 19 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 4.1.3.2 TF-IDF Vectorizer  **4.1.4 Handling Multi-Class Imbalanced** | 20 |
| 22 |
|  |  | **4.1.5 Train , Test , Validate Division** | 23 |
|  |  | **4.1.6 ML Classification Algorithms** | 26 |
|  |  | 4.2.1.1 Support Vector Machine | 26 |
|  |  | 4.2.1.2 Logistic Regression | 27 |
|  |  | * + - 1. Naives Bayes Classifier       2. K-Nearest Neighbour | 27 |
| 28 |
|  |  | 4.2.1.5 Decision Tree | 27 |
|  |  | 4.2.1.7 Boosting Classifiers(XGBoost, LightGBM) | 27 |
|  |  |  |  |
| **5.** | **RESULTS AND DISCUSSION** | | **28** |
|  | 5.1 Obtained Results | | 29 |
|  | 5.2 Interference | | 33 |
| **6.** | **CONCLUSION AND FUTURE WORKS** | | **34** |
|  | 6.1 Conclusion | | 34 |
|  | 6.2 Future Works | | 34 |
|  | **REFRENCES** | | **38** |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **S. NO.** | **FIGURES** |
| 1.1 | Lexicon Score Formula |
| 1.2 | Example Distribution of Emotions in Reviews With Score |
| 4.1 | Working of Text Classifier |
| 4.2 | Project Flow |
| 4.3 | Data set Info |
| 4.4 | Words Distribution in Text column |
| 4.5 | Labels Distribution using bar graph |
| 4.6 | Text Preprocessing flow |
| 4.15 | Formulation of TF\_IDF frequency value |
| 4.16 | Resampling of Data |
| 4.17 | SMOTE Visualization |
| 4.18 | Train , Test , Validate Split |
| 4.19 | Working of ML models |
| 4.20 | SVM -Binary classification |
| 4.21 | Logistic Regression cost function |
| 4.22 | Naive Bayes Probability formula |
| 4.23 | Distance Formula for KNN |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **TABLES** | **PAGE NO** |
| 5.1 | Classifiers and Results | 27 |
| 5.2 | Weighted Values ,Best score , parameter for balanced set | 36 |
| 5.3 | Weighted Values ,Best score , parameter for Imbalanced set | 36 |

# CHAPTER 1 INTRODUCTION

## REQUIREMENTS

This internship project centers around building robust text classification models for sentiment analysis. To achieve this goal, the project demands a comprehensive understanding of sentiment analysis techniques, the core data science project lifecycle, and proficiency in machine learning algorithms for text data

\* In-depth knowledge of sentiment analysis techniques, encompassing both lexicon-based approaches and machine learning methods.

\* Familiarity with the entire data science project process, including data acquisition, pre-processing, model development, evaluation, and potential deployment.

\* Strong skills in implementing machine learning classification algorithms specifically tailored for text data.

## GOAL AND OBJECTIVE

This internship project aims to equip with the skills to build text classification models for sentiment analysis. Diving into both lexicon-based methods, learning to construct sentiment dictionaries and apply them for classification, and explore machine learning algorithms suited for text data. The project will provide hands-on experience with the entire data science lifecycle, from data acquisition and pre-processing to model development and evaluation. Ultimately, building and comparing sentiment classification models using machine learning approaches, gaining a well-rounded understanding of this field and fulfilling the mentor's goal of building robust text classification models.

## DATA SCIENCE PROCESS

Leaf deficiency progress through several stages each displaying a distinct symptoms as the scarcity of essential nutrition intensifies :

* + 1. Data Preparation
    2. Data Exploration
    3. Data Modeling
    4. Presentation and Automation

## Data Preparation

In the realm of data science, data preparation is a crucial initial step that lays the foundation for successful analysis and modeling. It involves meticulously transforming raw data into a clean and usable format. This often entails tasks like handling missing values, removing outliers and inconsistencies, and transforming text data into numerical representations suitable for machine learning algorithms. By meticulously preparing your data, you ensure your models can effectively learn from the information and generate accurate results.

## Data Exploration

Data exploration, a cornerstone of data science projects, involves delving into the dataset to gain a deep understanding of its characteristics and potential hidden patterns. This initial phase employs various techniques to uncover trends, identify outliers, and assess data quality. Exploratory methods often involve visualizing data distributions through histograms and scatter plots, calculating summary statistics like mean and standard deviation, and potentially performing dimensionality reduction for high-dimensional datasets. Through this exploration, we can refine the research questions, guide feature engineering decisions, and ultimately prepare our data for robust modeling and analysis.

## Data Modeling

Data modeling in data science bridges the gap between exploration and actionable insights. It involves selecting or creating a mathematical representation of the relationships within your data. This can take many forms, such as building regression models to predict continuous values, classification models to categorize data points, or clustering algorithms to group similar data together. The chosen model is then trained on a portion of your data, learning the underlying patterns and relationships. Once trained, the model can be used to make predictions or classifications on unseen data, uncovering valuable insights and informing decision-making processes.

## Presentation and Automation

Data science thrives on clear communication and efficiency. Data presentation involves transforming complex findings into easily digestible visuals and narratives, using tools like dashboards and interactive reports to effectively convey insights to both technical and non-technical audiences. Automation plays a vital role in streamlining repetitive tasks within the data science lifecycle. By leveraging scripting languages and automation tools, data scientists can automate data cleaning, model training, and even report generation, freeing up valuable time for deeper analysis and exploration. This focus on clear presentation and efficient workflows ensures that data science findings have a tangible impact and can be readily applied to solve real-world problems.

## INTRODUCTION TO SENTIMENTS

Sentiment analysis is a valuable technique in data science that focuses on uncovering the emotional tone (positive, negative, or neutral) within textual data. This ability to understand user sentiment plays a vital role in various data science projects, offering significant benefits across different domains.

One of the most crucial applications of sentiment analysis is gauging user opinions and reactions. By analyzing sentiment in text reviews, social media posts, or customer surveys, data scientists can gain valuable insights into how users perceive products, services, brands, or even social events. This information is invaluable for businesses seeking to improve customer satisfaction. Companies can identify areas for improvement in their offerings, understand user needs and preferences, and gain valuable insights into market trends to stay ahead of the curve.

Sentiment analysis goes beyond simply understanding user opinions; it can also be used to personalize the customer experience. By analyzing sentiment in real-time interactions with customers (e.g., chatbots, support emails), companies can tailor their responses based on the user's emotional tone. Positive feedback can be met with expressions of gratitude and appreciation, while negative sentiment might necessitate proactive support or addressing user concerns in a timely and empathetic manner. This personalized approach to customer interactions fosters loyalty and builds stronger relationships.

The power of sentiment analysis extends to the realm of marketing. By identifying sentiment in social media data or customer reviews, data scientists can inform targeted marketing campaigns. Understanding user emotions allows companies to craft messages that resonate with the audience and drive engagement. For instance, analyzing positive sentiment towards a specific product feature can be leveraged to create targeted marketing campaigns highlighting that feature. Conversely, negative sentiment surrounding a particular aspect can be used to identify areas for improvement and adjust marketing strategies accordingly.

## SENTIMENT ANALYSIS USING LEXICON BASED APPROACH

Lexicon means the vocabulary of a person, language or branch of knowledge. Here, in lexicon based sentiment analysis we already have a given set of dictionary of words with each labelled as positive negative, neutral sentiments along with polarity , parts of speech and subjectivity classifiers, mood, modality and the like.A sentence is tokenized and each token is matched with the available words in the model to find out its context and sentiment (if any). A combining function such as sum or average is taken to make the final prediction regarding the total text component.

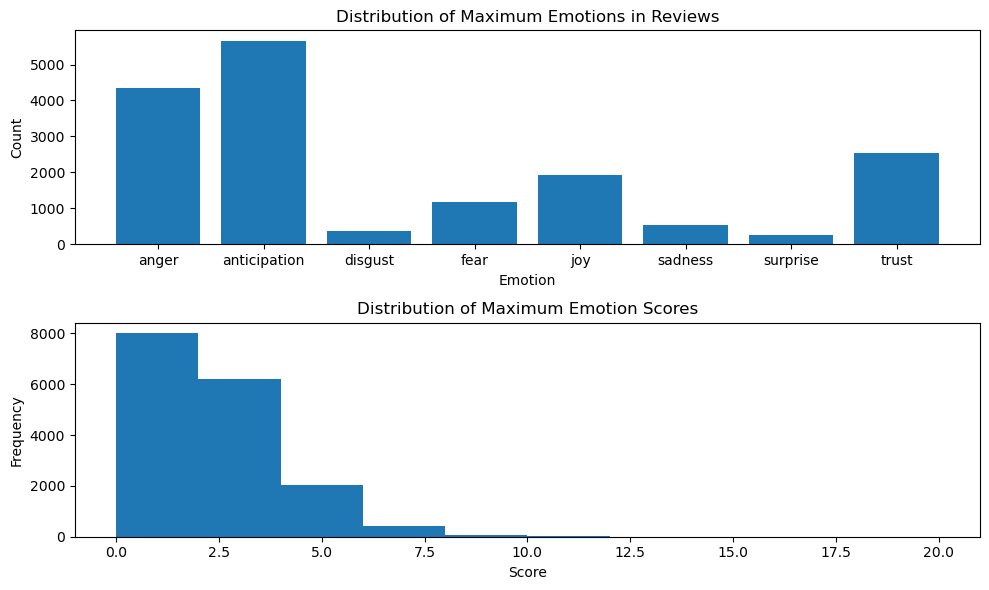
.

**Figure 1.1 Lexicon Score Formula**

## EMOTION ANALYZER - EMOLEX

EmoLex, also known as the NRC Emotion Lexicon, is a powerful resource for lexicon-based sentiment analysis. This pre-built dictionary offers a vast collection of words (around 14,000) along with their emotional associations. Unlike basic sentiment analysis with positive, negative, or neutral labels, EmoLex delves deeper, assigning scores to eight basic emotions (anger, fear, etc.) and two sentiment polarities (positive/negative) for each word. This multi-dimensional approach provides a richer understanding of emotional tone. Despite limitations in capturing language nuances and customization, EmoLex's comprehensiveness, efficiency, and transparent logic make it a valuable tool for initial analysis, smaller datasets, or when a simpler approach is preferred.

**Figure 1.2 Example-Distribution of Emotions in Reviews With Score**



## CHAPTER ORGANIZATION

This report provides an in-depth explanation of the project titled, "TEXT CLASSIFICATION ON SENTIMENTS " and its related technologies are explained in detail.

The investigation begins in Chapter 1 with a broad overview of Data Science Projects process and About Sentiment datasets.

In order to understand ”Text classification”, Chapter 2 provides insights about uses and application of classifying textual data

In Chapter 3, Discussion of Problem statements and potential Business use cases by addressing the problems

The suggested methodology is described in Chapter 4, which also offers an overview of the method used for classification .

In Chapter 5, the findings are presented and thoroughly discussed, with an emphasis on interpreting the results in light of the project's goals.

The report's six and final chapter summarizes the main findings, outlines the implications, and recommends directions for further study.

A reference section at the end of the report guarantees that the studies and sources cited throughout it are properly credited. A logical progression through the project's development, analysis, and conclusions is ensured by this structured approach.

# CHAPTER 2

# TEXT CLASSIFICATION

Classification, a fundamental concept in data science, refers to the process of organizing data points into predefined categories. Imagine a large collection of unsorted documents - classification helps us categorize them as news articles, emails, or social media posts. This organization becomes crucial for extracting valuable insights from data.

Data science projects often deal with vast amounts of unstructured data. Classification helps us transform this chaos into a structured format, enabling further analysis and modeling.

* **Pattern Recognition:** By classifying data points, we can identify underlying patterns and trends within the data. This allows us to make predictions about future data points and uncover hidden relationships.
* **Decision Making**: Classification empowers data-driven decision making. By classifying data points like customer purchases or loan applications, businesses can make informed decisions about targeted marketing campaigns or risk assessments.
* **Predictive Modeling:** Classification forms the foundation for building powerful predictive models. These models can learn from classified data and predict future outcomes, such as customer churn or creditworthiness.

Text classification, a specific type of classification, focuses on categorizing textual data. This could involve classifying emails as spam or not spam, news articles by topic (sports, politics, etc.), or social media comments as positive, negative, or neutral. Text classification unlocks a wealth of insights from textual data sources.

# CHAPTER 3

# **PROBLEM STATEMENT**

# **AND USE CASES**

* **Inaccurate Overall Sentiment:**

Text data often mixes opinions with factual statements. Without classification, the overall sentiment can be misinterpreted.(eg) .A product review praising features but ending with disappointment about a specific issue might be misclassified as entirely positive, leading to a skewed understanding of customer sentiment.

* **Misguided Targeting and Decision-Making:**

Data-driven decisions based on sentiment analysis are crucial.

Without classification, businesses may target the wrong audience with marketing campaigns (eg) Classifying all social media mentions of a brand as positive could lead to neglecting negative feedback and missing opportunities to address customer concerns.

* **Wasted Resources and Inefficiency:**

Text data often contains irrelevant or neutral information. Without classification, valuable resources might be wasted on analyzing irrelevant text. (eg) A system analyzing all customer support tickets, including greetings and closings, instead of focusing on the core issue within the message.

**USE CASES:**

Text classification in sentiment analysis acts as a solution to these problems by:

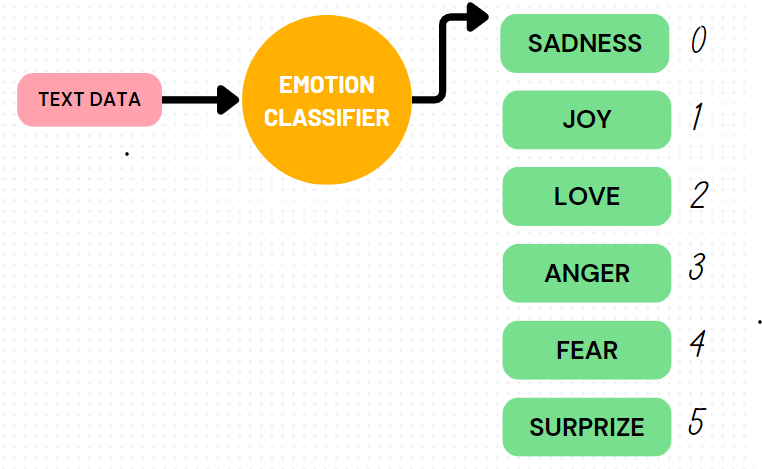
* Categorizing data for focused analysis.
* Extracting a clearer understanding of overall sentiment.
* Enabling targeted marketing and improved decision-making.
* Optimizing resource allocation by focusing on relevant text.
* Identifying trends and patterns within specific categories.
* Facilitating customization and scalability for diverse contexts.

# CHAPTER 4 PROPOSED METHODOLOGY

## BUILDING TEXT CLASSIFIERS

Text classifiers act like emotional detectives, analyzing textual data to understand feelings like anger, joy, surprise, and more. They work by transforming text into a format computers can understand, then using techniques like pre-built emotion dictionaries or machine learning algorithms to identify emotional cues within the words and sentence structures. These classifiers are constantly evolving, offering businesses and researchers valuable tools to decode the emotional undercurrents hidden within written communication.

## Figure 4.1 Working of Text Classifier



**Modeling Architecture**

Building a text classification model for sentiment analysis involves a step-by-step process. First, we explore the data to understand its characteristics. Then, we clean the text by removing unnecessary elements and converting it into a format usable by machines. Next, we transform the text into numerical values to allow for analysis. To address potential biases in the data, we might adjust the balance between positive, negative, and neutral examples. Finally, we build a machine learning model using algorithms like Support Vector Machines, train it on a portion of the data, and evaluate its performance on unseen data to ensure it accurately classifies sentiment.

## Figure 4.2 **Project flow**

## 

Performing EDA(Exploratory Data analysis) requires the well known understanding of the data and gathering insights on it . Hence the following Dataset overview and identifying the target and feature variables will be driven by performing EDA on the dataset.

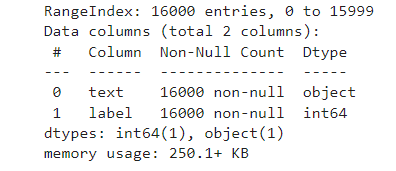
## DATA OVERVIEW

The dataset appears well-suited for sentiment analysis with an emotional classification twist. It contains 16,000 entries, each with two key columns: "text" and "label". The "text" column holds comments from customers, providing the raw data for analysis. The "label" column assigns an emotion label to each comment using a numerical code. These codes range from 0 to 5, corresponding to specific emotions:

* 0: sadness
* 1: joy
* 2: love
* 3: anger
* 4: fear
* 5: surprise

With this structure, the dataset allows to explore the emotional sentiment expressed within customer comments and categorize them into six distinct emotional states..

## Figure 4.3 **Dataset Info**



* A visual representation of text data using word cloud. The largest words likely represent the most frequent terms in your text. These might be common words or content-specific terms depending on the data source.

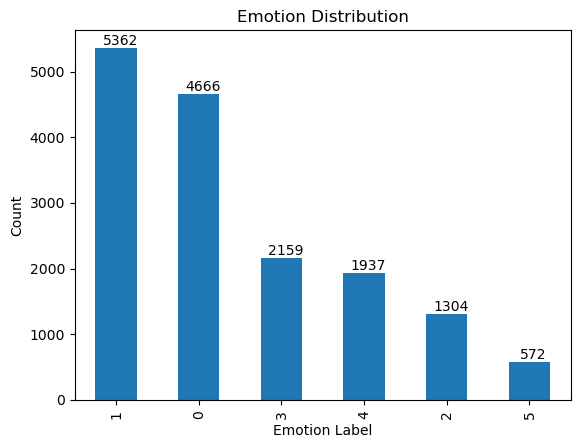
## Figure 4.4 Words Distribution in Text column

## IMG_256

* From the class labels distribution we can see there exist multi class imbalance in

them ,so there is a chance of our classifier being bias or gives False predictionmaximum. To avoid it we need to remove the imbalance

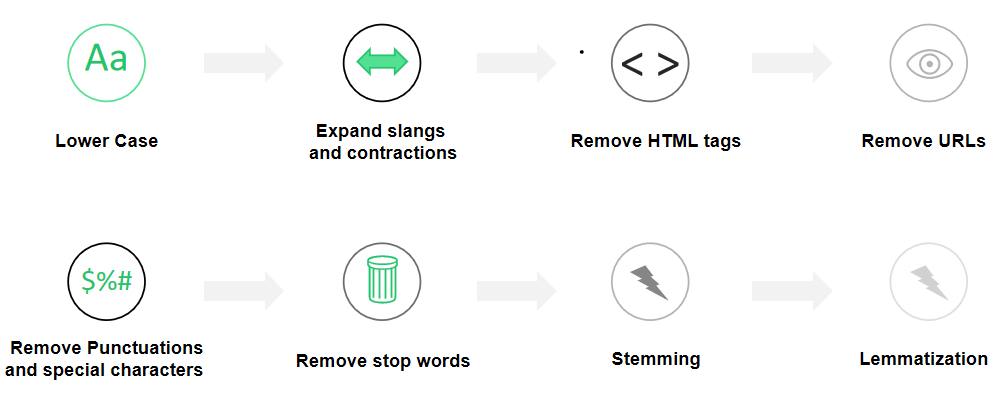
## Figure 4.5 labels distribution using bar graph



## TEXT PREPROCESSING

Text classification models struggle to understand raw text data directly. Text preprocessing is a crucial step in preparing text data for machine learning models to improve their performance and accuracy in classification tasks.

## Figure 4.6 Text preprocessing flow



* + - 1. **TEXT PREPROCESSING STEPS**

## Lower Case Conversions:

* Converts all text to lowercase (e.g., "Hello" becomes "hello").
* This eliminates the need for the model to learn the importance of case sensitivity, which can be irrelevant for many classification tasks. or LAB).

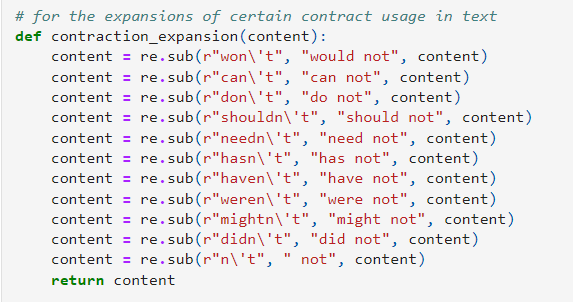
## Figure 4.7 Lower Case Conversion code



## Removal of Slangs and contractions :

* This step expands shortened words like "don't" to "do not" or "can't" to "cannot."
* It ensures consistency in the data and avoids confusion for the model, especially when dealing with informal text.

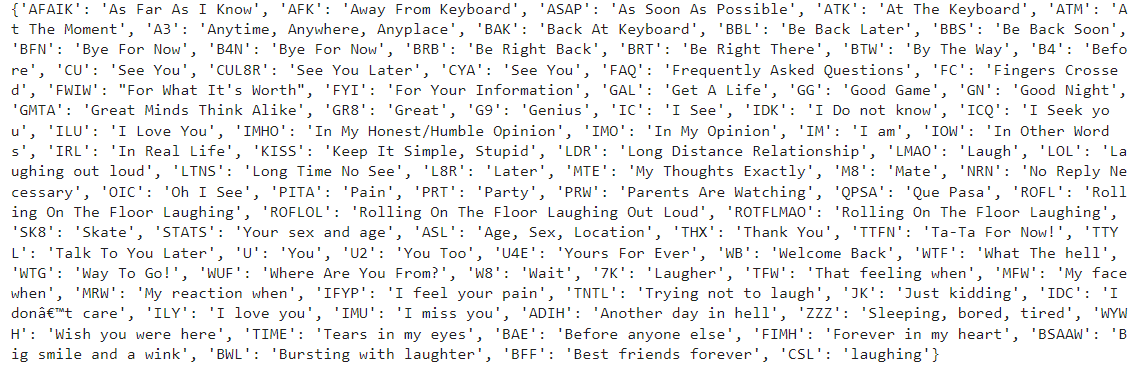
## Figure 4.8 Contract expandtion code



## Figure 4.9 Abberivation of Slangs

*These abberivation slangs are applied on the text column , where the removal of slangs*

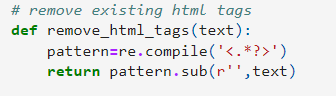
*is excuted using it*



## Remove HTML tags:

* This removes HTML tags like <p>, <b>, and others that might be present in the text data.
* These tags don't contribute to the meaning and can be safely removed for text classification.

## Figure 4.10 Regex Expression to remove tags



## Remove URL (http links):

* This removes website URLs from the text data.
* URLs generally don't provide semantic meaning for classification and might introduce irrelevant information.

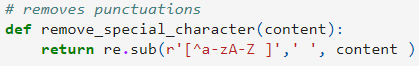
## Figure 4.11 Regex Expression to remove links



## Remove punctuation and special character:

* This removes punctuation marks like commas, periods, exclamation points, and digits , white spaces , special characters like @,#$,%,^.. are all removed
* While punctuation can be helpful in some NLP tasks like sentiment analysis, it's often irrelevant for classification based on content.

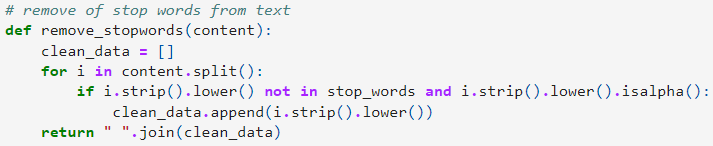
## Figure 4.12 Regex Expression to remove punctuation and special character



## Remove Stop words

* This removes common words like "the", "a", "an", "is", etc. that don't carry much meaning on their own.
* Removing stopwords reduces noise in the data and allows the model to focus on more content-specific terms.

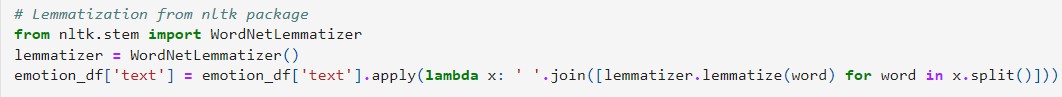
## Figure 4.13 From nltk package , using the built-in stopwords , to eliminate presence of stop words in text Column



## Stemming and Lemmatization

* These techniques aim to reduce words to their base form.
* Stemming uses simple rules to chop off suffixes (e.g., "running" becomes "run"). Lemmatization uses a dictionary to map words to their dictionary form (e.g., "running" becomes "run").This helps capture the semantic similarity between words with different variations

## Figure 4.14 From nltk package , using the WordNetLemmatizer.



## TEXT REPRESENTATION

In text classification projects, raw text data cannot be directly processed by machine learning models. we need text representation and how vectorizers come into play:

* **Challenges of Raw Text:**
* *Unstructured Data:* Text data is a sequence of characters, lacking the inherent structure of numerical data.
* *Variable Length:* Text documents can vary greatly in length, making them difficult to compare directly.
* *Semantic Meaning:* Models need a way to understand the meaning conveyed by the words and their relationships.

Text representation is the process of converting textual data into a numerical format that a machine learning model can understand. This allows the model to identify patterns and relationships within the text for classification tasks. Some common text representation techniques:

* **Bag-of-Words (BoW):** This method represents a document as a simple collection of words, ignoring their order or relationships. Each word is treated as a feature, and its frequency in the document is counted. The resulting representation is a vector with a dimension equal to the vocabulary size, where each element indicates the word's frequency.
* **TF-IDF (Term Frequency-Inverse Document Frequency**): This builds upon BoW by weighting words based on their importance. Words that appear frequently within a document (high TF) but rarely across all documents (low IDF) are considered more informative for classification.
* **Word Embeddings:** These techniques represent words as dense vectors in a high-dimensional space. Words with similar meanings are positioned closer together in this space, capturing semantic relationships between words. Popular methods include Word2Vec and GloVe.
* **The Role of Vectorizers:**

Vectorizers are tools that automate the process of converting text data into a numerical representation. They handle tasks like:

**Tokenization:** Breaking down text into individual words or phrases (tokens).

**Vocabulary Building**: Creating a dictionary of all unique words encountered in the training data.

**Feature Extraction:** Applying the chosen text representation technique (e.g., BoW, TF-IDF) to convert each document into a numerical vector.

Popular libraries like scikit-learn in Python offer various vectorizer implementations, such as CountVectorizer for BoW and TfidfVectorizer for TF-IDF.

Benefits of Text Representation:

**Enables Machine Learning:** By converting text into a numerical format, vectorizers allow machine learning models to process and analyze text data effectively.

**Feature Extraction:** Text representation techniques extract relevant features from the text that can be used for classification.

**Comparison and Similarity**: Documents can be compared and classified based on their vector representations, identifying similar topics or themes.

## 4.1.3.1 TF-IDF(Term Frequency-Inverse Document Frequency)

**1. Term Frequency (TF):**

This measures how frequently a particular word appears within a single document.Documents with a higher frequency of a specific word are considered to have a stronger association with that term.

**2. Inverse Document Frequency (IDF):**

This component focuses on the overall importance of a word across the entire dataset.Words that appear very frequently in all documents (e.g., "the", "a") have a low IDF score, indicating they are not very informative for classification.

Words that are specific and appear less frequently across documents (e.g., technical terms) have a higher IDF score, suggesting they are more discriminative for classification.

**3.Combining TF-IDF:**

The TF-IDF vectorizer multiplies the TF and IDF scores for each word in a document. This results in a weighted vector where words with high relevance within a document and low prevalence across the dataset receive higher weights.

## Figure 4.15 Formulation of TF-IDF frequency value

## 

The TF-IDF value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.

.

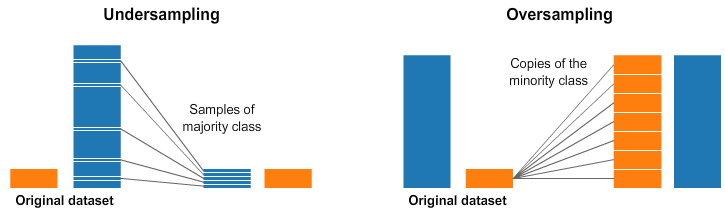
## HANDLING MULTICLASS IMBALANCE

When the target classes (two or more) of classification problems are not equally distributed, then we call it Imbalanced data. If we failed to handle this problem then the model will become a disaster because modeling using class-imbalanced data is biased in favor of the majority class.

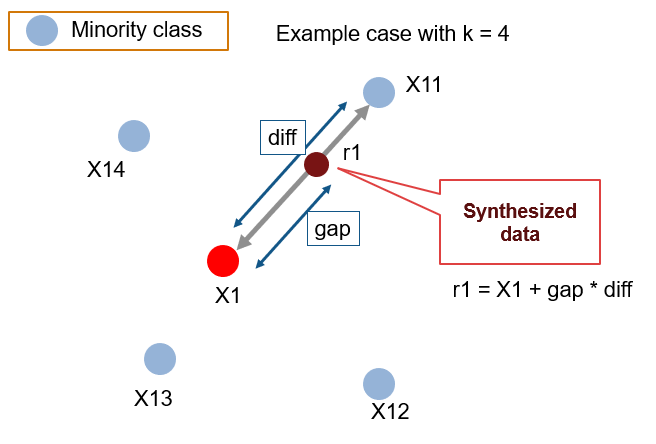
**Methods to Handle class Imbalance**

1. *Random Over-Sampling:* Duplicates data points from the minority class randomly. This is simple but might lead to overfitting.
2. *SMOTE (Synthetic Minority Over-Sampling Technique):* Creates synthetic data points for the minority class by interpolating between existing minority class examples. This helps create variations within the feature space and can be more effective than random oversampling.
3. *Undersampling:* It involves reducing the number of examples in the majority class. It's generally used in conjunction with other techniques like SMOTE to avoid discarding potentially valuable information from the majority class. There are various methods for undersampling, such as random undersampling, selecting informative samples from the majority class, or using techniques like NearMissnt.

## Figure 4.16 Resampling of Data Visualization



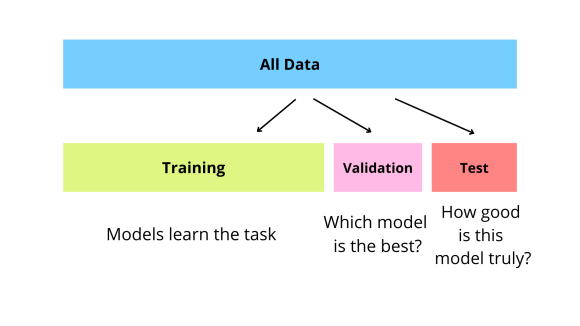
## Figure 4.17 SMOTE Visualization



## 4.1.5. TRAIN , TEST , VALIDATE DATA SPLIT

Splitting the data into train, validation, and test sets is essential for reliable machine learning models. The train set teaches the model, but evaluating it on the same data can lead to overfitting (memorizing specifics rather than learning patterns). The test set, completely unseen by the model during training, provides a realistic assessment of generalizability. The validation set acts as a middle ground for hyperparameter tuning, where the experiment with model settings to find the best configuration without using the test set. Common splits are 80/10/10 or **70/10/20 (train/validation/test)**, with the latter offering more data for evaluation. This approach ensures our model can learn effectively from the data and perform well on unseen data in real-world scenarios.

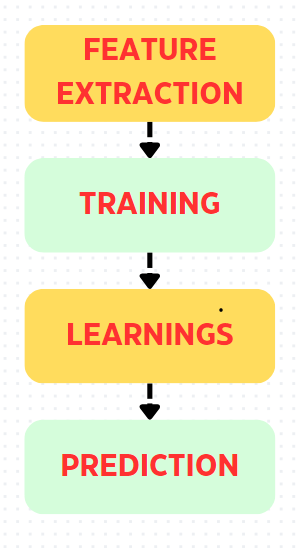
## Figure 4.18 Train , Test , Validate Split



## 4.1.6 ML CLASSIFICATION MODELS

Machine learning classification models are algorithms that learn to categorize data points into predefined classes. In text classification, these models are specifically trained to analyze textual data and assign documents or sentences to relevant categories

## Figure 4.19 Working of ML models



## SUPPORT VECTOR MACHINE

An SVM is a supervised machine learning algorithm that excels at classification tasks. It can effectively separate data points belonging to different classes by finding an optimal hyperplane in the feature space.

* **Hyperplane:** This is a decision boundary that divides the feature space into regions. In text classification, features might represent word frequencies or other characteristics extracted from text data.
* **Support Vectors:** These are the data points closest to the hyperplane from each class. They essentially "support" the hyperplane in defining the maximum margin for separation.
* **Margin:** This refers to the distance between the hyperplane and the closest support vectors from each class. A larger margin indicates a better separation between the classes, leading to more robust classification. meticulous manual mapping, making it valuable for applications with time constraints.

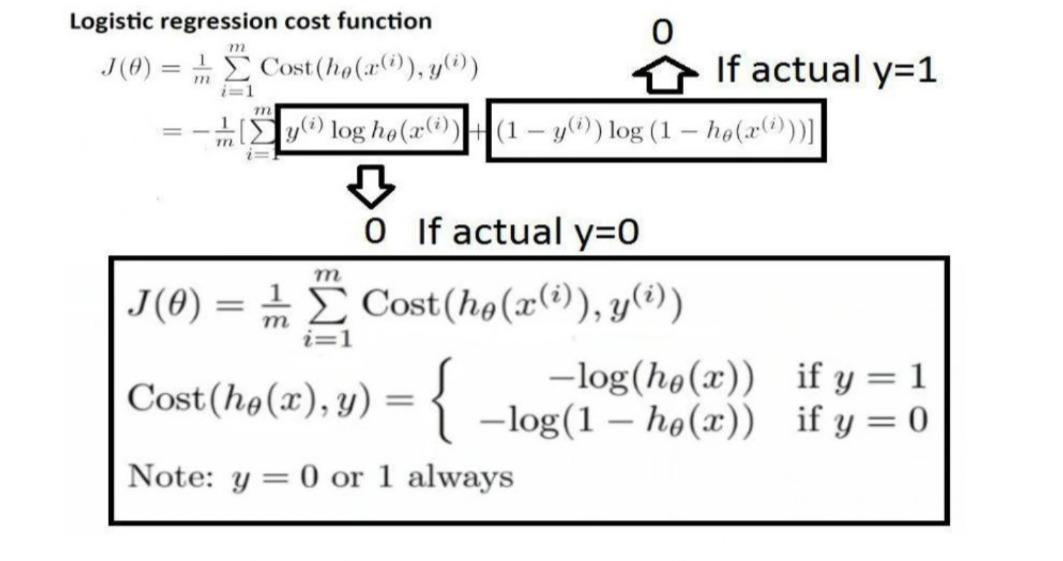
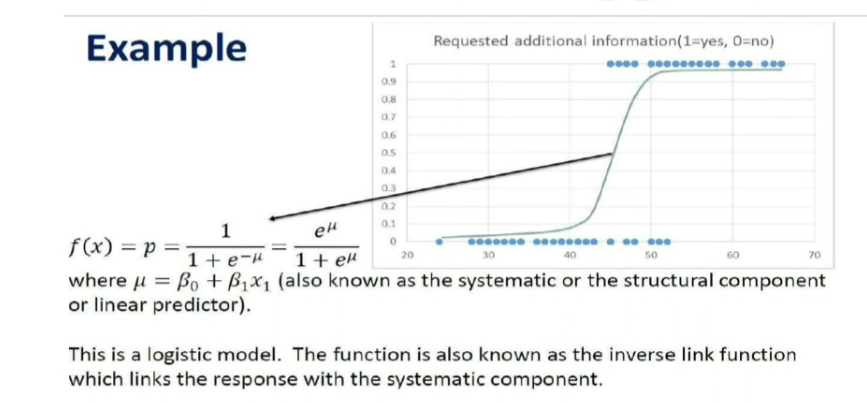
## Figure 4.20 SVM - Example for binary classification

## 

## LOGISTIC REGRESSION

Logistic regression utilizes a linear model and the sigmoid function to predict class probabilities. By minimizing the cost function and adjusting model parameters, it learns to effectively classify data points. The provided formulas offer a deeper understanding of the calculations behind the model.The classification *threshold (often 0.5) converts probabilities to class labels (above 0.5 for class 1, below for class 0).*Logistic regression is interpretable through *coefficients and odds ratios*. It's efficient and performs well with large datasets

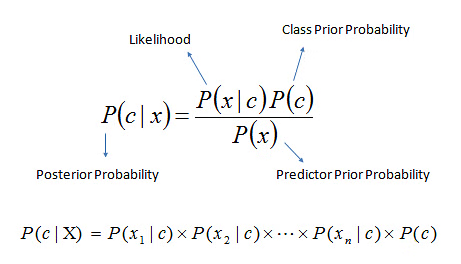
## Figure 4.21 Logistic Regression visualization and cost function descriptions



## NAVIE BAYES CLASSIFIER

Naive Bayes is the simple algorithm that classifies text based on the probability of occurrence of events. This algorithm is based on the Bayes theorem, which helps in finding the conditional probabilities of events that occurred based on the probabilities of occurrence of each individual event.

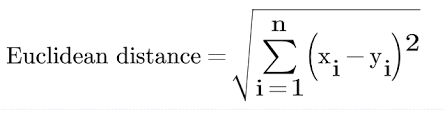
## Figure 4.22 Naive Bayes probability formula



## K-NEAREST NEIGHBOUR CLASSIFIER

The K-Nearest Neighbors (KNN) classifier predicts labels for new data points based on their similarity to existing labeled data. It uses *distance metrics (like Euclidean distance) t*o find the K closest neighbors (data points) in the training set to the new point. The majority class label among these neighbors becomes the predicted class for the new point. KNN is easy to understand and works with various data types, but choosing the optimal K value (number of neighbors) and mitigating the "curse of dimensionality" (performance decline with high feature counts) are important aspects for successful KNN implementation.

## Figure 4.23 Distance formula used in KNN algorithm



## DECISION TREE CLASSIFIER

Decision trees are a powerful machine learning algorithm used for both classification and regression tasks. They work by building a tree-like structure that resembles a flowchart, where each internal node represents a question based on a feature of the data, and each branch represents a possible answer that leads to a child node.

***Entropy (Impurity):***

Entropy is a measure of disorder or uncertainty within a dataset regarding the target variable (class labels).

A higher entropy indicates a more diverse dataset with a mix of classes.

A lower entropy indicates a more homogenous dataset with a clear dominant class.

→ eq(1)

· S: The dataset being considered (e.g., all emails).

· pi: The probability of encountering an instance belonging to class I

**Information Gain:**

* Information gain measures the reduction in entropy (uncertainty) achieved by splitting the data based on a particular feature.
* The feature with the highest information gain is chosen as the splitting criterion at each node in the decision tree, as it leads to the most significant reduction in uncertainty and helps create purer child nodes.

· S: The parent node (dataset before splitting).

· A: The feature used for splitting.

· : The number of instances in subset Sw (data points belonging to a specific value of feature A).

· W: The total number of instances in S.

· Entropy(): The entropy of the child node Sw (data points after splitting based on a specific value of feature A).

## BOOSTING CLASSIFIERS

Boosting classifiers are a powerful ensemble machine learning technique for classification tasks. They combine the predictions from multiple weak learners (typically decision trees) into a stronger final model. Boosting classifiers like *XGBoost and LightGBM* provide powerful tools for classification tasks by leveraging the strengths of ensemble learning. Their ability to combine multiple weak learners, handle complex data, and prevent overfitting makes them valuable choices for various machine learning applications.

**Working**

1. **Sequential Learning:** Boosting algorithms build models sequentially. Each new model (referred to as a boosting stage) focuses on learning from the errors of the previous ones.
2. **Weighted Predictions:** During each stage, the weights of training instances are adjusted based on their misclassification in the prior models. Instances that were previously misclassified receive higher weights, forcing the next model to pay more attention to them.
3. **Ensemble Power:** As stages are added, the combined model becomes more robust and accurate by leveraging the collective strengths of the individual weak learners.

# CHAPTER 5 RESULTS AND DISCUSSION

In this chapter, we will discuss the results of our project to quantify the percentage of accuracy , precision , recall , F1Score to choice to best model by hyperparameter tuning. We will also discuss the choosen best model via analysis of results from GridSearchCV.

## OBTAINED RESULTS:

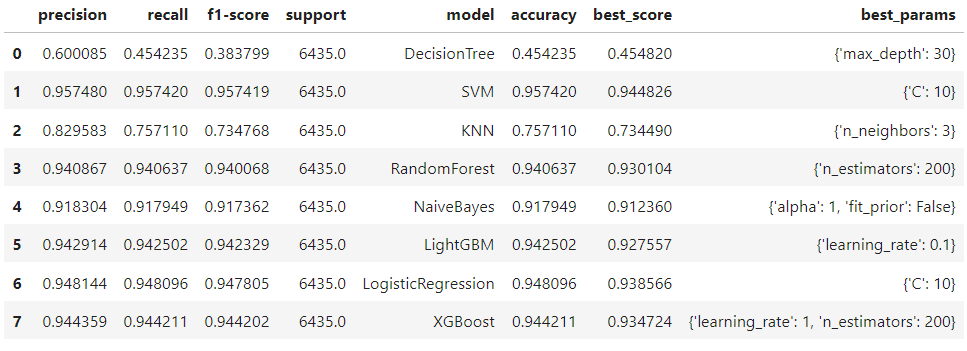
**Table 5.1 CLASSIFIERS AND RESULTS**

|  |  |
| --- | --- |
| **CLASSIFIER** | **RESULTS AND CONFUSION MATRIX** |
| **SUPPORT VECTOR MACHINE**  **(On Train data - Balanced set)** | IMG_256 |
| **SUPPORT VECTOR MACHINE**  **(On Validate data - Balanced set)** | IMG_256 |
| **SUPPORT VECTOR MACHINE**  **(On Test data - Balanced set)** | IMG_256 |
| **SUPPORT VECTOR MACHINE**  **(Classification Report on validate data)** | **precision recall f1-score support**  **0 0.95 0.92 0.93 452**  **1 0.91 0.93 0.92 428**  **2 0.97 0.98 0.98 396**  **3 0.97 0.94 0.95 413**  **4 0.96 0.94 0.95 454**  **5 0.96 1.00 0.98 431**  **accuracy 0.95 2574**  **macro avg 0.95 0.95 0.95 2574**  **weighted avg 0.95 0.95 0.95 2574** |

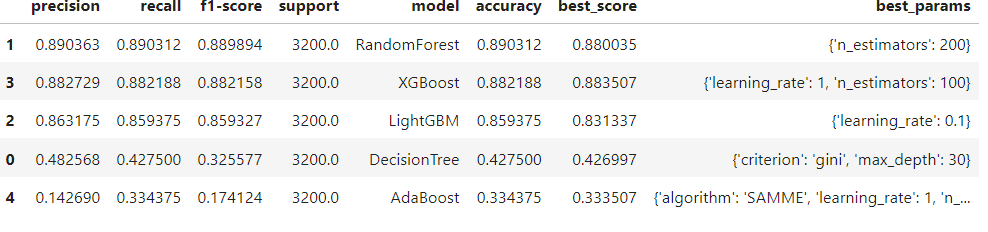
|  |  |
| --- | --- |
| **CLASSIFIER** | **RESULTS AND CONFUSION MATRIX** |
| **LOGISTIC REGRESSION**  **(On Train data - Balanced set)** | IMG_256 |
| **LOGISTIC REGRESSION**  **(On Test data - Balanced set)** | IMG_256 |
| **LOGISTIC REGRESSION**  **(On Validate data - Balanced set)** | IMG_256 |
| **LOGISTIC REGRESSION**  **(Classification Report on validate data)** | **precision recall f1-score support**  **0 0.95 0.92 0.94 452**  **1 0.95 0.89 0.92 428**  **2 0.93 0.99 0.96 396**  **3 0.93 0.95 0.94 413**  **4 0.95 0.92 0.93 454**  **5 0.93 0.99 0.96 431**  **accuracy 0.94 2574**  **macro avg 0.94 0.94 0.94 2574**  **weighted avg 0.94 0.94 0.94 2574** |
| **RANDOM FOREST**  **(On Train data - Balanced set)** | IMG_256 |
| **RANDOM FOREST**  **(On Validate data - Balanced set)** | IMG_256 |
| **RANDOM FOREST**  **(On Test data - Balanced set)** | IMG_256 |
| **RANDOM FOREST**  **(Classification Report on validate data)** | **precision recall f1-score support**  **0 0.94 0.86 0.90 452**  **1 0.90 0.86 0.88 428**  **2 0.92 0.98 0.95 396**  **3 0.92 0.95 0.94 413**  **4 0.95 0.93 0.94 454**  **5 0.94 0.99 0.96 431**  **accuracy 0.93 2574**  **macro avg 0.93 0.93 0.93 2574**  **weighted avg 0.93 0.93 0.93 2574** |
| **XGBoost Classifier**  **(On Train data - ImBalanced set)** | IMG_256 |
| **XGBoost Classifier**  **(On Validate data - ImBalanced set)** | IMG_256 |
| **XGBoost Classifier**  **(On Test data - ImBalanced set)** | IMG_256 |
| **XGBoost Classifier**  **(Classification Report on ImBalanced validate set)** | **precision recall f1-score support**  **0 0.91 0.91 0.91 369**  **1 0.90 0.91 0.91 461**  **2 0.79 0.76 0.77 104**  **3 0.83 0.82 0.82 163**  **4 0.85 0.83 0.84 141**  **5 0.74 0.88 0.80 42**  **accuracy 0.87 1280**  **macro avg 0.84 0.85 0.84 1280**  **weighted avg 0.87 0.87 0.87 1280** |
| **LIGHTGBM Classifier**  **(On Train data - ImBalanced set)** | IMG_256 |
| **LIGHTGBM Classifier**  **(On Validate data - ImBalanced set)** | IMG_256 |
| **LIGHTGBM Classifier**  **(On Test data - ImBalanced set)** | IMG_256 |
| **LIGHTGBM Classifier**  **(Classification Report On Validate data - ImBalanced set)** | **precision recall f1-score support**  **0 0.91 0.88 0.89 369**  **1 0.83 0.92 0.87 461**  **2 0.79 0.69 0.74 104**  **3 0.80 0.80 0.80 163**  **4 0.85 0.72 0.78 141**  **5 0.74 0.76 0.75 42**  **accuracy 0.85 1280**  **macro avg 0.82 0.79 0.81 1280**  **weighted avg 0.85 0.85 0.84 1280** |
| **RANDOM FOREST**  **(On Train data - ImBalanced set)** | IMG_256 |
| **RANDOM FOREST**  **(On Validate data - ImBalanced set)** | IMG_256 |
| **RANDOM FOREST**  **(On Test data - ImBalanced set)** | IMG_256 |
| **RANDOM FOREST**  **(Classification Report on Validate data - ImBalanced set)** | **precision recall f1-score support**  **0 0.89 0.92 0.91 369**  **1 0.90 0.89 0.90 461**  **2 0.76 0.69 0.72 104**  **3 0.84 0.83 0.83 163**  **4 0.83 0.85 0.84 141**  **5 0.77 0.79 0.78 42**  **accuracy 0.87 1280**  **macro avg 0.83 0.83 0.83 1280**  **weighted avg 0.87 0.87 0.87 1280** |

## TABULATION OF THE OBTAINED OUTPUT USING GRIDSEARCHCV

**Table 5.2 Weighted Values ,Best score , parameter for balanced set**



**Table 5.2 Weighted Values ,Best score , parameter for Imbalanced set**



## INFERENCE:

The results of our project show that for balanced dataset SVM achieved the highest accuracy (95.52%), followed closely by Logistic Regression (94.42%). This indicates that these models might be the best choices for this specific task based on this metric.

Random Forest, XGBoost, and LightGBM also delivered high accuracy (above 93%), suggesting they are strong contenders as well .Naive Bayes performed reasonably well (91.77%), but its accuracy was lower than the top models. It might still be a suitable choice if interpretability or simplicity is a priority.

For Imbalanced Dataset Random Forest achieved the highest accuracy (89.03%), followed closely by XGBOOST (88.35%). This indicates that these models might be the best choices for this specific task based on this metric.

# CHAPTER 6 CONCLUSION AND FUTURE WORK

## CONCLUSION

The Project successfully compared the performance of different machine learning models for text classification. By carefully considering the achieved accuracies, other relevant metrics, and the specific characteristics of the dataset (balanced or imbalanced).SVM achieved the highest accuracy (around 95.52%), followed closely by Logistic Regression. Random Forest, XGBoost, and LightGBM also delivered high accuracy, making them strong contenders. Naive Bayes performed reasonably well, while KNN and Decision Tree showed lower accuracy.Due to the imbalanced nature of the data, it's expected that overall accuracy might be lower for all models compared to the balanced case. However, specific inferences require evaluating metrics like precision, recall, and F1-score, with a focus on the minority class performance. Models like SVM, Random Forest, and XGBoost are generally good candidates for imbalanced data.

## FUTURE WORK

Future works could focus on enhancing the Classifiers algorithm's robustness for diverse Textual Data.Exploring hyperparameter tuning for the promising models to potentially improve their performance. Investigating the use of cost-sensitive learning and data balancing techniques for the imbalanced dataset.Testing the chosen model on a separate hold-out test set to assess its generalizability to unseen data.Incorporating techniques like ensemble learning (combining multiple models) for potentially better performance.

# REFERENCES

1. **A NLP Tutorial for Text Classification in Python**. https://medium.com/analytics-vidhya/nlp-tutorial-for-text-classification-in-python-8f19cd17b49e.
2. **Chapter 2 DataScience** https://livebook.manning.com/book/introducing-data-science/chapter-2/216
3. **Sentiment Analysis- Lexicon Models vs Machine Learninge**. https://medium.com/nerd-for-tech/sentiment-analysis-lexicon-models-vs-machine-learning-b6e3af8fe746
4. **Lexicon-based sentiment analysis using Twitter datas**. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8989068/
5. **“Decision Tree Algorithm”** ,https://www.kdnuggets.com/2020/01/decision-tree-algorithmexplained.html#:~:text=Information%20gain%20is%20a%20decrease,tree%20algorithm%20uses%20information%20gain.
6. **“Multi-Class Text Classification Model Comparison and Selections**," 2016 Published Towards Data science [Multi-Class Text Classification Model Comparison and Selection | by Susan Li | Towards Data Science](https://towardsdatascience.com/multi-class-text-classification-model-comparison-and-selection-5eb066197568).