# C:\Users\Prashant\Desktop\download (1).png

# Infosys Springboard Internship 4.0

# 2024

# PROJECT REPORT

# ON

# “Text-Classification on Emotions Detection ”

# BY

# Prashant Donarkar

# donarkarprashant1@gmail.com

# In Partial Full fillment Of

# (Data Visualization Internship)

**Infosys Springboard Internship 4.0 (2024)**

|  |  |  |
| --- | --- | --- |
| SR.NO | TOPIC | PAGE.NO |
| 1 | **Problem Statemen**t | **3** |
| 2 | **Datasets** | **6** |
| 3 | **Data Preprocessing** | **7** |
| 4 | **Modeling Approach** | **10** |
| 5 | **Models Built &**  **Hyper parameter Tuning** | **19** |
| 6 | **Modeling Results** | **21** |
| 7 | **Conclusion** | **24** |

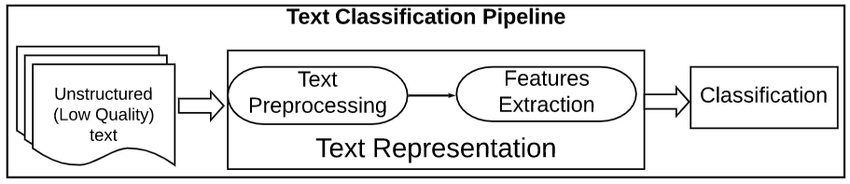
**1: Problem Statement**

* 1. **Text Classification for Emotions Detection**

### 

### **Text Classification:**

Text classification is a fundamental task in natural language processing (NLP) that involves assigning predefined categories or labels to text documents based on their content. This process leverages machine learning algorithms and NLP techniques to analyze and understand the text, enabling various applications such as sentiment analysis, spam detection, topic categorization, and emotions detection.

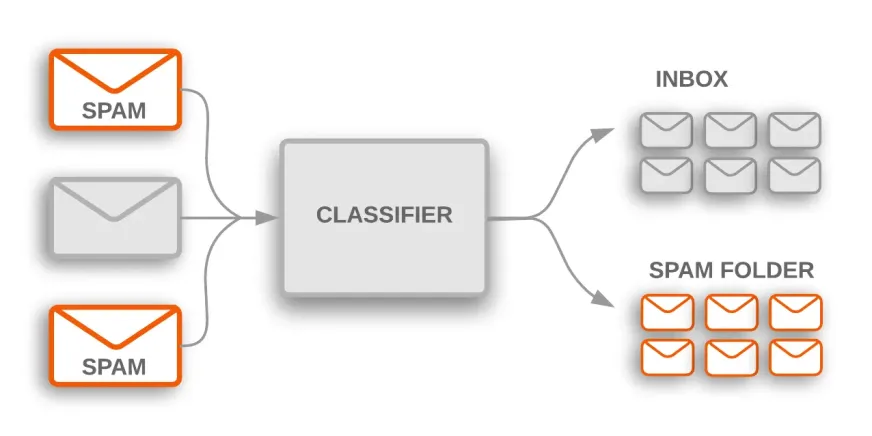


Text classification is a powerful and widely used task in NLP that can be used to automatically categorize or predict a class of unseen text documents, often with the help of supervised machine learning.

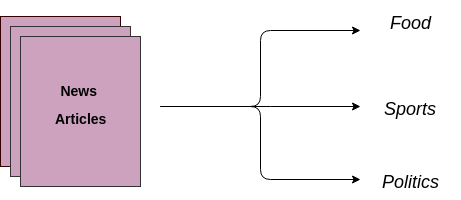
It is not always accurate, but when used correctly, it can add a lot of value to your analytics. There are many different ways and algorithms to go about setting up a text classifier, and no single approach is best

## Text Classification Use-Cases and Applications

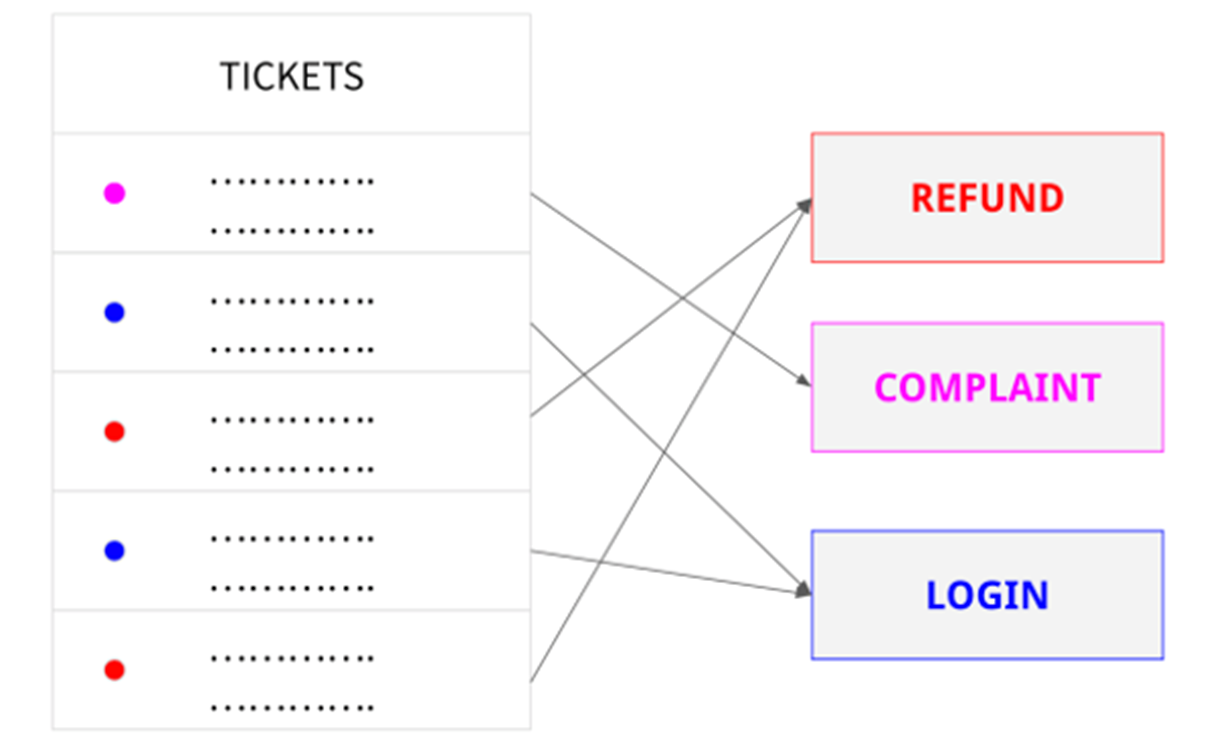
1. **Spam classification:-** A spam filter is a common application that uses text classification to sort emails into spam and non-spam categories.



### **Classifying news articles and blogs:-** A supervised machine learning model is trained on labeled data, which includes both the raw text and the target. Once a model is trained, it is then used in production to obtain a category (label) on the new and unseen data (articles/blogs written in the future).



**3. Categorize customer support requests:-** A company might use text classification to automatically categorize customer support requests by topic or to prioritize and route requests to the appropriate department.



### **4. Hate speech detection:-** With over 1.7 billion daily active users, Facebook inevitably has content created on the site that is against the rules. Hate speech is included in this undesirable content.

### Facebook tackles this issue by requesting a manual review of postings that an AI text classifier has identified as hate speech

### Facebook Text Classifier

### 

### **1.2 Problem Business Use Case**

In the competitive landscape of customer-centric industries such as e-commerce, telecommunications, and social media, understanding customer emotions from textual feedback is paramount. Imagine a global e-commerce company that receives thousands of customer reviews daily. Manually analyzing these reviews to gauge customer satisfaction and identify areas of improvement is not feasible. By implementing an automated text classification model for emotions detection, the company can efficiently process and analyze vast amounts of textual data in real-time.

This project can identify emotions such as happiness, sadness, anger, and frustration in customer reviews, social media comments, and support tickets. For example, detecting a high volume of negative emotions linked to a particular product can trigger a quality review or prompt customer support to proactively address issues. Positive emotions can be highlighted to identify popular products or successful marketing campaigns. Additionally, this insights-driven approach allows for personalized customer interactions, enhancing customer satisfaction and loyalty. Overall, the use of emotions detection in text analysis provides actionable insights that drive strategic business decisions and improve customer experience**.**

**2.About Dataset**

**Overview:** The dataset used for this project, titled "Emotion.csv," is designed to facilitate the training and evaluation of machine learning models for emotions detection in textual data. It comprises two primary columns: **text** and **label**. The **text** column contains the textual data, which are sentences or short paragraphs, and the **label** column contains integer labels representing different emotion categories.

**Columns:**

1. **text**: This column contains the actual text data in string format. Each entry represents a piece of text, such as a sentence, a review, a comment, or a social media post.
2. **label**: This column contains integer values that represent different emotions. Each integer corresponds to a specific emotion category (e.g., happiness, sadness, anger). The labels are encoded as integers for compatibility with machine learning algorithms.

Example:-

|  |  |
| --- | --- |
| text | label |
| i didnt feel humiliated | 0 |
| i can go from feeling so hopeless to so damned hopeful just from being around someone who cares and is awake | 0 |
| im grabbing a minute to post i feel greedy wrong | 3 |
| i am ever feeling nostalgic about the fireplace i will know that it is still on the property | 2 |
| i am feeling grouchy | 3 |

**Emotion Labels:** The **label** column uses integer encoding to represent different emotions. For example:

* 1: Happiness
* 2: Suprise
* 3: Sadness
* 4: Anger

**3. Data Preprocessing**

Data preprocessing is a crucial step in the data analysis and machine learning pipeline. It involves transforming raw data into a clean and usable format to improve the quality and performance of models. The goal is to ensure that the data is consistent, accurate, and ready for analysis. This process includes a variety of techniques to handle missing values, noise, and inconsistencies, and to transform the data into a format suitable for modeling.

Steps followed during data preprocessing :-

### **1. Importing Libraries:-**

To get started with data preprocessing and building machine learning models for the "Text Classification for Emotions Detection" project, we need to import several Python libraries. These libraries provide various tools and functions for data manipulation, preprocessing, feature extraction, model building, and evaluation.

### **2. Importing Dataset:-**

The first step in working with your dataset is to import it into your Jupyter Notebook. For this project, the dataset is named **Emotion.csv** and contains two columns: **text** and **label**. We'll use the **pandas** library to load and inspect the dataset.

**3.Fixing Datatype:-**

Checking that the data types of the columns in your dataset are appropriate for analysis and modeling, you need to check and, if necessary, correct the data types. In the Emotion.csv dataset, the text column should be of string type, and the label column should be of integer type**.**

**4.Checking for null values:-**

To ensure data quality, it's crucial to identify and handle any missing values in your dataset. Missing values can cause issues in data analysis and modeling, so addressing them early in the preprocessing stage is essential.

**5.Dropping Null Values:-**

After identifying the presence of null values, the next step is to remove the rows containing these null values to maintain data integrity. This step ensures that the dataset is clean and ready for further analysis and modeling without the risk of errors or biases caused by incomplete data.

**6. Reviewing the content in the "text" column:-**

Reviewing the content in the text column is a crucial step in understanding the nature and quality of the textual data you're working with. This involves inspecting the content for inconsistencies, patterns, and potential issues that might need to be addressed during preprocessing.

**7.Coverting the content in “text” to Lower case :-**

Converting all text data to lowercase is a common text preprocessing step that helps normalize the data. This step ensures that words like "Happy," "happy," and "HAPPY" are treated as the same word, improving the consistency of the text data for analysis and modeling.

**8. Removing commas from the "text" column:-**

Removing commas from the text data is another preprocessing step that helps to clean and standardize the text. This can be important for reducing noise in the text data and ensuring that punctuation does not interfere with text analysis and modeling.

**9. Removing new lines using regex from the "text" column:-**

Removing new lines from the text data is an essential preprocessing step that helps ensure the text is clean and standardized. New lines can introduce unwanted breaks and inconsistencies in the text data, which can affect text processing and analysis.

**10. Removing links from the "text" column:-**

Removing links from text data is important for ensuring that the text is clean and contains only meaningful content. Links can introduce noise and irrelevant information that can affect text analysis and modeling.

**11. Removing alphanumeric words from the "text" column:-**

Removing alphanumeric words from text data helps clean the text and ensures that it contains only meaningful words. Alphanumeric words often do not carry useful semantic information and can introduce noise into the data.

**12. Removing words containing numbers:-**

Removing words that contain numbers helps to clean the text data and ensures that it contains only meaningful words. Words with numbers often do not carry useful semantic information and can introduce noise into the data.

**13. Removing extra spaces:-**

Removing extra spaces from the text data is a crucial preprocessing step to ensure that the text is clean and standardized. Extra spaces can introduce inconsistencies and affect text processing and analysis.

**14. Removing special characters:-**

Removing special characters from text data is important for cleaning the text and ensuring that it contains only alphanumeric characters and spaces. Special characters can introduce noise and affect text processing and analysis.

**15. Removing stopwords:-**

Stopwords are common words that do not carry much meaning and are often removed from text data during preprocessing. Removing stopwords helps to reduce noise and improve the quality of text analysis and modeling.

**16. Stemming:-**

Stemming is the process of reducing words to their word stem or root form. It helps in reducing variations of words and can improve text analysis by grouping together words that have the same meaning but are written in different forms.

**17. Lemmatization:-**

Lemmatization is the process of reducing words to their base or root form, called a lemma. Unlike stemming, which reduces words to a truncated form, lemmatization ensures that the root word belongs to the language and is meaningful. This can be particularly useful in tasks like text classification where the meaning of words is important.

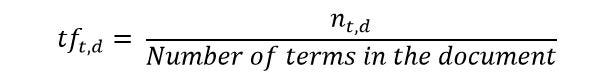
**4.Modelling Approach And Models Used**

After preprocessing, the following steps were performed:-

1. **Text Data Transformation:-**

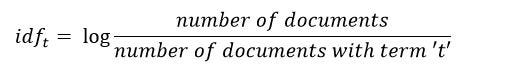
TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus). Converting text data into a TF-IDF matrix is a common preprocessing step in natural language processing (NLP) tasks.

* **Term Frequency**: It is a measure of how frequently a term ‘t’ appears in a document ‘d’.



(Fig: https://medium.com/swlh/feature-transform-of-text-data-nlp-c6ccedbeb3cc)

* **Inverse Document Frequency**: It is a measure of how important a term is. We need the IDF value because computing just the TF alone is not sufficient to understand the importance of words.



(Fig: https://medium.com/swlh/feature-transform-of-text-data-nlp-c6ccedbeb3cc)

By converting text data into a TF-IDF matrix, we prepare the data for further analysis and modeling, allowing you to extract meaningful insights from the text.

**2.Feature Scaling:-**

Feature scaling is a technique used to standardize the range of independent variables or features of data. / Feature scaling, in the context of machine learning, refers to the process of transforming the numerical features of a dataset into a standardized range. It is often necessary in machine learning models to ensure that all features contribute equally to the computation. Standard Scaler is one of the methods used for feature scaling, which scales the data so that it has a mean of 0 and a standard deviation of 1. By applying Standard Scaler to the label column, you ensure that the label data is scaled appropriately, which can improve the performance and interpretability of your machine learning models.

**Importance of Feature Scaling in Machine Learning**

1.Enhancing Model Performance

2.Addressing Skewed Data and Outliers

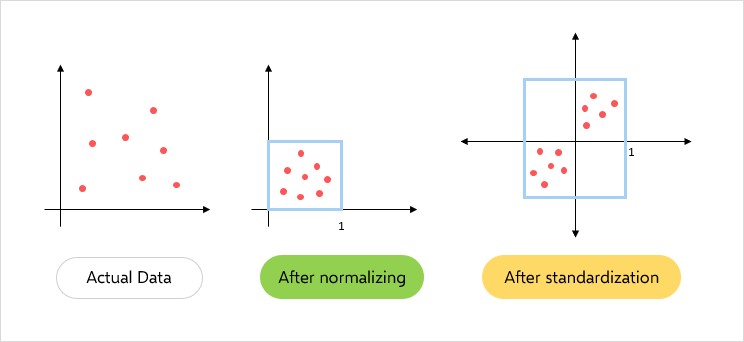
3.Faster Convergence during Traning

4.Imporved Algorithm Behaviour

**Types of Feature Scaling Techniques**

**1.Normalization**

**2.Standarzation**



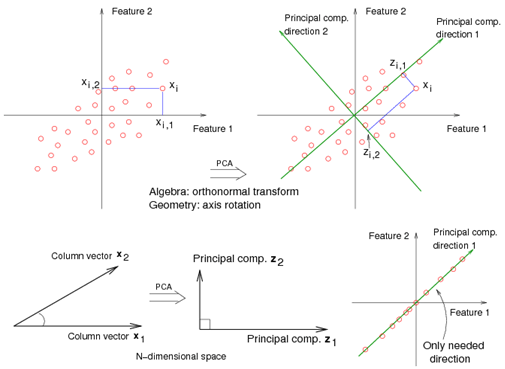
(Fig: https://datasciencedojo.com/blog/feature-scaling/)

**3.Principal Component Analysis (PCA):-**

Principal Component Analysis (PCA) is a technique used to reduce the dimensionality of the data by transforming it into a new coordinate system (i.e., a set of principal components) where the variance of the data is maximized along the axes. PCA is often used in text data analysis to reduce the number of features (words) while retaining most of the variance in the data.

**How does PCA work?**

PCA works by finding the principal components of a dataset. Principal components are the directions in which the data varies the most. The first principal component is the direction with the highest variance, and each subsequent principal component is orthogonal to the previous ones and has the highest variance possible under that constraint. PCA then transforms the data into the new coordinate system defined by the principal components.



(Fig: https://medium.com/@brijesh\_soni/topic-14-principal-component-analysis-or-pca-53dfb15b8b9e)

**4. Converting text to word counts:-**

Converting text data into a matrix of word counts is a common preprocessing step in natural language processing (NLP) tasks. This process involves representing each document as a vector where each element corresponds to the frequency of a particular word in the document. This matrix is often referred to as a "bag of words" representation.

**5**. **Data Splitting:-**

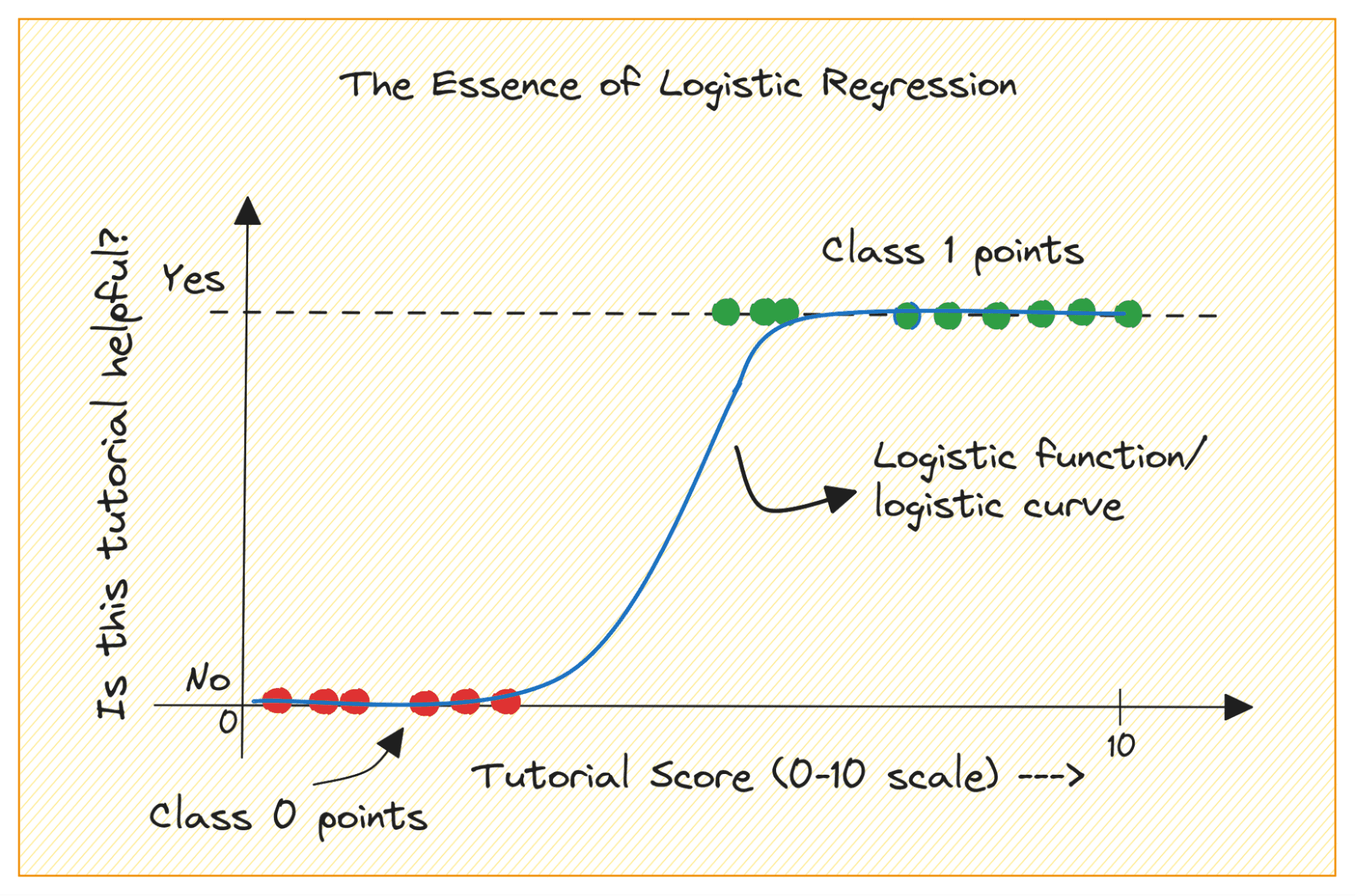
Splitting the dataset into training, testing, and validation sets is an important step in machine learning to evaluate the performance of the model on unseen data and prevent overfitting. Typically, the dataset is divided into three parts: training set (70%), testing set (20%), and validation set (10%).

After all this we build the following models to classify text based on emotions:-

**1.Logistic Regression Model:-**

Logistic Regression is a statistical model used for binary classification tasks, where the output is a binary label (e.g., positive or negative sentiment). In the context of text classification for emotions, logistic regression can be used to predict the emotion label (e.g., happy, sad, angry) based on the text content.

By employing logistic regression for text classification, we can build a model that predicts emotion labels based on the text content, providing insights into the emotional content of the text data.

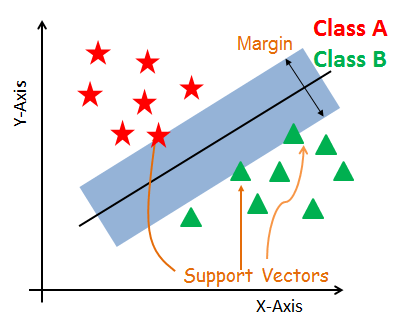


(Fig: https://www.kdnuggets.com/building-predictive-models-logistic-regression-in-python)

**3.Support Vector Machine:-**

Support Vector Machine (SVM) is a powerful supervised machine learning algorithm used for classification tasks. SVM works by finding the hyperplane that best separates the different classes in the feature space. In the context of text classification for emotions, SVM can be used to predict the emotion label based on the text content.

Generally, Support Vector Machines is considered to be a classification approach, it but can be employed in both types of classification and regression problems. It can easily handle multiple continuous and categorical variables. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane(MMH) that best divides the dataset into classes.



(Fig: https://www.datacamp.com/tutorial/svm-classification-scikit-learn-python?utm\_source=google&utm\_medium=paid\_search&utm\_campaignid=19589720824&utm\_adgroupid=157156376311&utm\_device=c&utm\_keyword=&utm\_matchtype=&utm\_network=g&utm\_adpostion=&utm\_creative=698229374827&utm\_targetid=aud-)

**Support Vectors**

Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins. These points are more relevant to the construction of the classifier.

**Hyper plane**

A hyper plane is a decision plane which separates between a set of objects having different class memberships.

**Margin**

A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points.

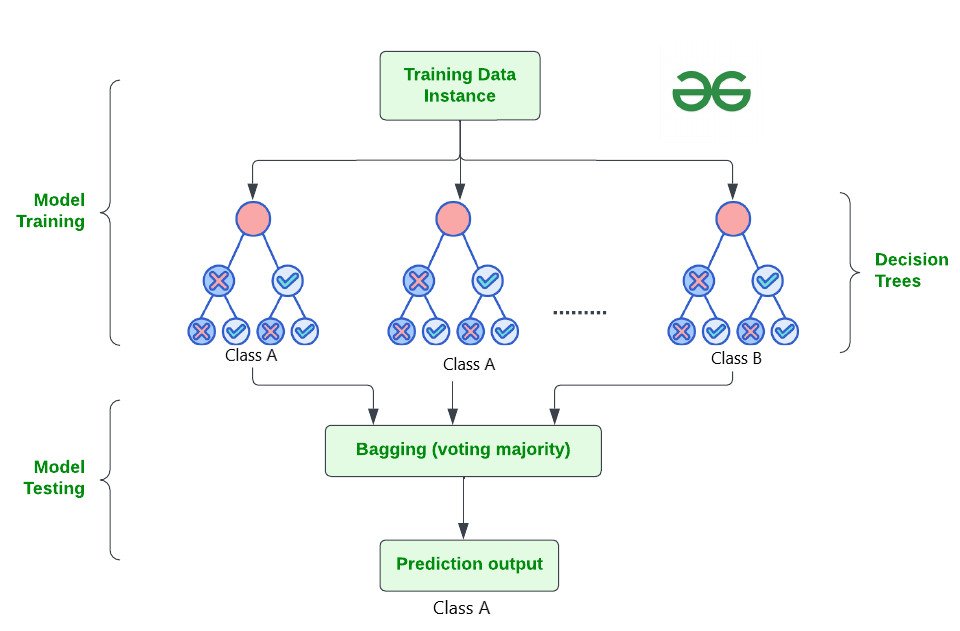
By using SVM for text classification, we can build a model that effectively predicts emotion labels based on the text content, providing valuable insights into the emotional content of the text data.

**4.Random Forest:-**

Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees during training and outputs the class that is the mode of the classes of the individual trees. It is robust to overfitting and tends to generalize well to unseen data. In the context of text classification for emotions, Random Forest can be used to predict the emotion label based on the text content.

* Ensemble Learning: Random Forest combines multiple decision trees to improve classification accuracy.
* Feature Importance: Random Forest provides a feature importance score, which can help in understanding the importance of different words/features in predicting the emotion label.
* Robustness to Overfitting: Random Forest is less prone to overfitting compared to individual decision trees.

By using Random Forest for text classification, we can build a model that effectively predicts emotion labels based on the text content, providing valuable insights into the emotional content of the text data.



(Fig: https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/)

## Working of Random Forest Algorithm

Before understanding the working of the random forest algorithm in machine learning, we must look into the ensemble learning technique. **Ensemble**simplymeans combining multiple models. Thus a collection of models is used to make predictions rather than an individual model.

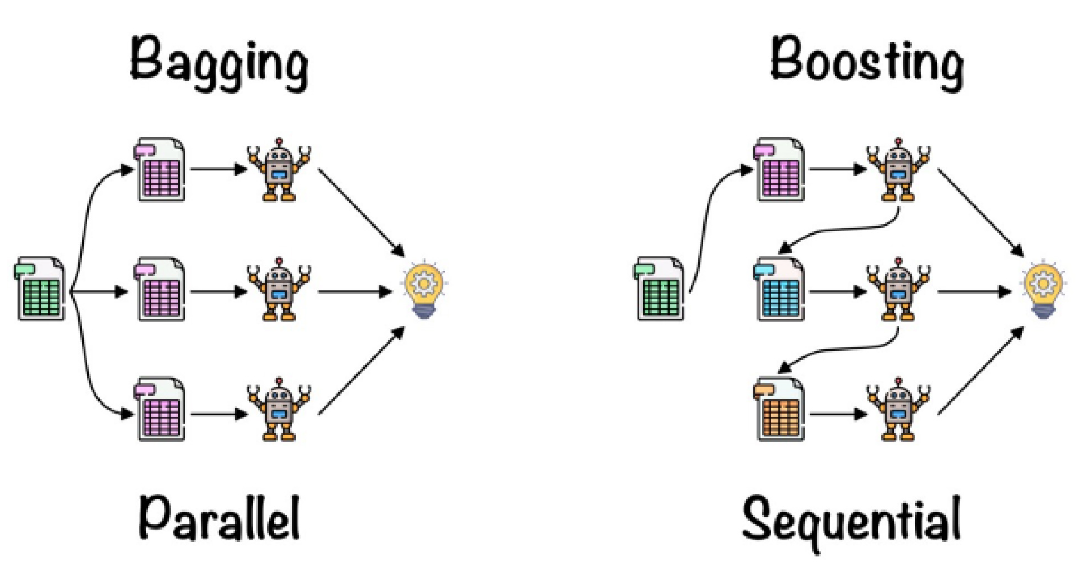
Ensemble uses two types of methods:

#### ****Bagging****

It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example,  Random Forest.

#### ****Boosting****

It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example,  ADA BOOST, XG BOOST.

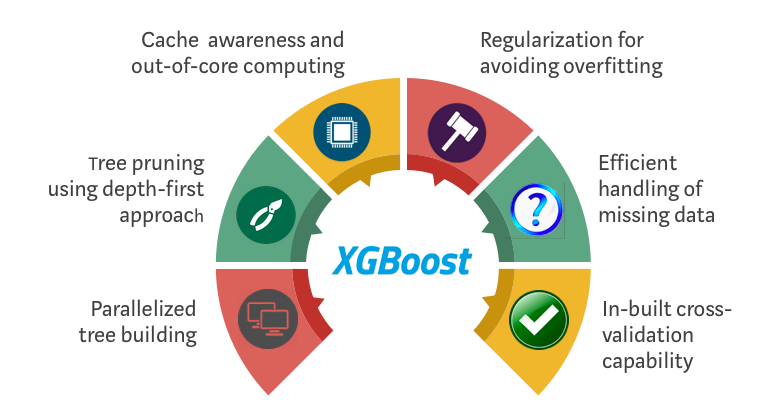


**4.Xgboost :-**

XGBoost (Extreme Gradient Boosting) is an efficient and scalable machine learning algorithm used for regression and classification tasks. It is based on the gradient boosting framework and is known for its speed and performance. In the context of text classification for emotions, XGBoost can be used to predict the emotion label based on the text content.

* **Speed and Performance:** XGBoost is known for its speed and performance, making it suitable for large datasets.
* **Regularization:** XGBoost includes L1 and L2 regularization to prevent overfitting.
* **Handling Imbalanced Datasets:** XGBoost has techniques to handle imbalanced datasets, which are common in text classification tasks.

By using XGBoost for text classification, we can build a model that efficiently predicts emotion labels based on the text content, providing valuable insights into the emotional content of the text data.

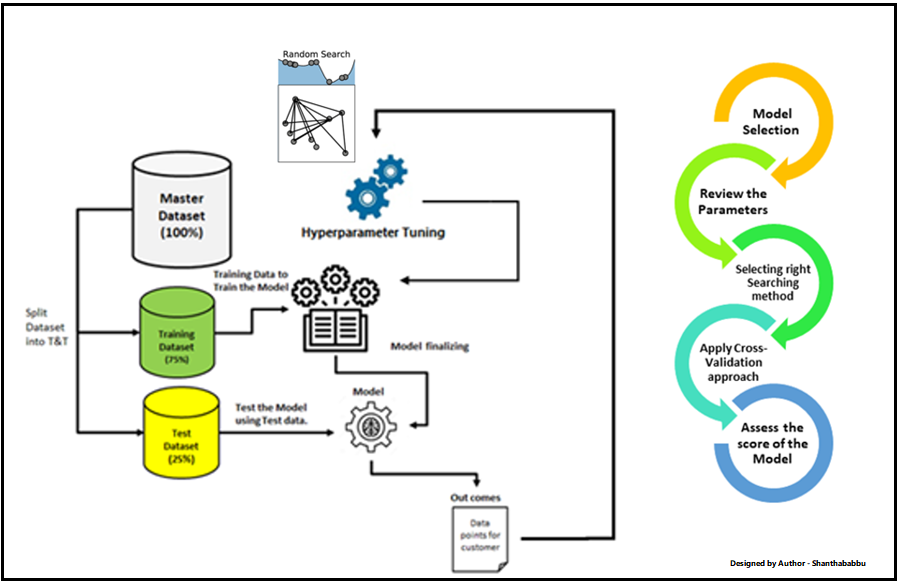


5. **Hyper parameter Tuning**

**Hyper parameter Tuning:-**

Hyper parameter tuning for Random Forest involves finding the optimal values for hyper parameters that control the behavior of the Random Forest algorithm. These hyper parameters include the number of trees in the forest, maximum depth of each tree, minimum number of samples required to split a node, and minimum number of samples required at each leaf node. Tuning these hyper parameters can significantly impact the performance of the Random Forest model.

Hyperparameter tuning is basically referred to as tweaking the parameters of the model, which is basically a prolonged process.Below fig mention the steps to be followed to perform Hyper parameter tuning.



(Fig: https://www.analyticsvidhya.com/blog/2022/02/a-comprehensive-guide-on-hyperparameter-tuning-and-its-techniques/)

**Hyper parameter Tuning Process**

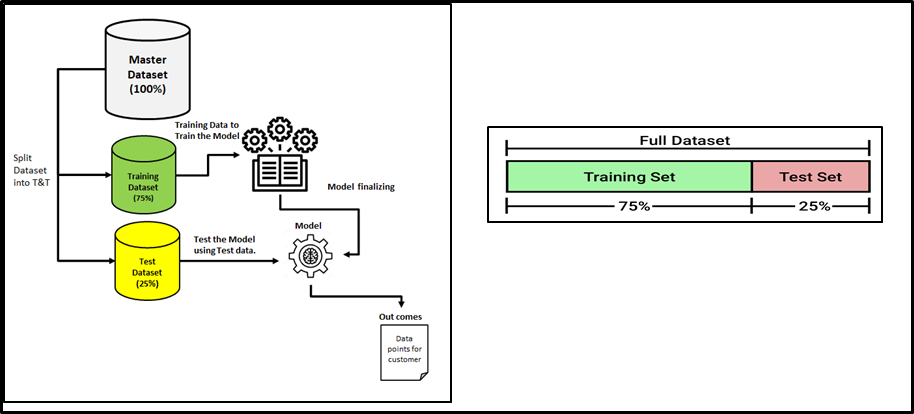
1. **Define Hyper parameter Grid:**
   * Specify the hyper parameters and their respective ranges to be tuned.
2. **Cross-Validation:**
   * Use cross-validation to evaluate the model with different hyperparameter settings.
3. **Grid Search:**
   * Use Grid Search CV to search for the best hyper parameter combination.
4. **Fit the Model:**
   * Fit the model using the best hyper parameters found during the grid search.

* **Improving Model Performance:** Hyper parameter tuning helps in finding the best set of hyper parameters, which can improve the model's performance on unseen data.
* **Preventing Over fitting:** By tuning hyper parameters, we can prevent overfitting and improve the model's generalization ability.
* **Optimizing Resources:** Hyper parameter tuning helps in optimizing computational resources by finding the best model configuration.

By performing hyper parameter tuning for the Random Forest model, we can optimize its performance and select the best model configuration for classifying text based on emotions.

For each of the models (Logistic Regression, SVM, and XG Boost), we performed hyper parameter tuning to optimize their performance. Hyper parameter tuning helps in finding the best set of hyper parameters for a model, which can improve its performance on unseen data. We evaluated the results based on training and testing accuracy.

Hyper parameter Tuning Process



(Fig: https://www.analyticsvidhya.com/blog/2022/02/a-comprehensive-guide-on-hyperparameter-tuning-and-its-techniques/)

**6.Modelling Results**

The modeling process involved training and evaluating several machine learning models for emotion recognition. The performance metrics, including training and testing accuracy, were computed and displayed for each model. These metrics provide insights into how well the models learned the training data and generalized to unseen data.

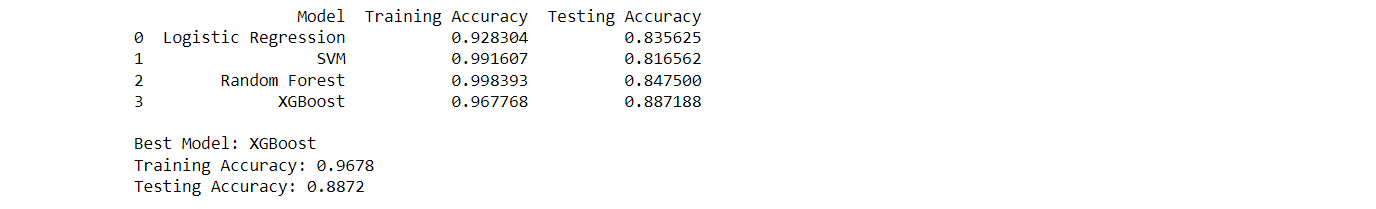
Confusion matrices were generated to visually represent the performance of each model. These matrices illustrate the number of correctly and incorrectly classified instances for each emotion label, allowing for a comprehensive evaluation of the models' strengths and weaknesses.

To facilitate a more granular comparison, the precision, recall, and F1-score were calculated for each model across different emotion labels. These metrics were then plotted using bar graphs, enabling a clear visualization of the models' performances in terms of their ability to correctly identify positive instances (precision), capture all relevant instances (recall), and strike a balance between these two aspects (F1-score).

Additionally, the training and testing accuracy for each model were plotted together, providing a visual comparison of how well the models learned the training data and how they performed on unseen data. This visualization helps identify potential overfitting or underfitting issues and assists in selecting the most robust and generalizable model.

Based on the comprehensive analysis of these performance metrics and visualizations, the best-performing model was identified. This model exhibited the highest overall accuracy, balanced precision and recall scores across emotion labels, and demonstrated its ability to generalize well to unseen data, making it the most suitable choice for deployment in real-world emotion recognition tasks.

Below mentioned are the results for the same :-



**7. Recommendations or Resolutions**

After a comprehensive evaluation of the modeling results, the best-performing model was determined by comparing the training and testing accuracy metrics across all the models under consideration. The model with the highest testing accuracy, while maintaining a reasonable training accuracy, was selected as the optimal choice, as it demonstrated the ability to generalize well to unseen data without overfitting to the training set.

While the identified best-performing model exhibited promising results, several recommendations were provided to further improve the emotion detection capabilities of the system. One potential area of improvement was the exploration of more advanced feature engineering techniques or the incorporation of transfer learning methodologies to leverage pre-trained models on related tasks, potentially enhancing the model's ability to capture complex emotional patterns.

Additionally, suggestions were made to investigate the use of ensemble techniques, which combine the predictions of multiple models, as this approach has been known to improve overall performance and robustness in various machine learning tasks, including emotion recognition.

For future projects, potential enhancements were suggested, such as exploring the integration of multimodal data sources, incorporating not only textual data but also visual or audio cues, as emotions are often expressed through a combination of modalities. Furthermore, the inclusion of contextual information or the development of domain-specific models tailored to particular applications or industries were recommended to improve the models' performance and applicability in real-world scenarios.

Overall, the recommendations and resolutions aimed to provide guidance for iterative improvements, with a focus on leveraging advanced techniques, exploring multimodal data sources, and developing domain-specific models to enhance the accuracy and applicability of emotion detection systems.

**8.Conclusion**

After a thorough evaluation of various machine learning models, including logistic regression, support vector machines (SVM), random forest, and XGBoost, the XGBoost model emerged as the best-performing model for emotion detection. This determination was made by comparing the training and testing accuracy metrics across all models. The XGBoost model exhibited the highest testing accuracy while maintaining a reasonable training accuracy, indicating its ability to generalize well to unseen data without overfitting to the training set.

While the XGBoost model demonstrated promising results, recommendations were made to further enhance its performance through advanced feature engineering, transfer learning, ensemble techniques, and the incorporation of multimodal data sources and contextual information. Overall, the XGBoost model's superior performance, coupled with the proposed recommendations, positions it as a robust and accurate solution for emotion detection tasks.