

**SPRINGBOARD INTERNSHIP 4.0**

**Internship Report**

**“Sentiment Analysis And Text Classification Based On Emotion”**

**Submitted by:**

ARADHANA DUBEY

(Project Intern)

**Under the Guidance of:**

SUDHEER KUMAR

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Topic** | **Page No.** |
| **1.** | **Introduction** | **1** |
| **2.** | **Business Use Case** | **2-4** |
| **3.** | **Sentiment Analysis** | **4-7** |
| **4.** | **Implementation Steps Of Text Classification** | **8** |
| **5.** | **About Dataset** | **9** |
| **6.** | **Data (Text) Pre-processing** | **10-11** |
| **7.** | **Modeling Approach** | **12-13** |
| **8.** | **Model used** | **14-16** |
| **9.** | **Hyperparameter tuning** | **17-20** |
| **10.** | **Conclusion** | **21** |

***Introduction***

***Automatic Ticket Classification for Customer Support***

In today’s digital age, businesses receive an overwhelming volume of customer support tickets daily. Efficiently managing these tickets is crucial for maintaining high customer satisfaction and operational efficiency. Automatic ticket classification has emerged as a powerful solution, leveraging artificial intelligence (AI) and machine learning (ML) to categorize and prioritize tickets, thereby enhancing the customer support experience. Automatic ticket classification involves using algorithms to analyse and categorize support tickets based on their content. This process typically involves natural language processing (NLP) to understand the text within tickets and machine learning models to classify them into predefined categories. These categories might include issues related to billing, technical support, account management, or product information. The future of automatic ticket classification is promising, with ongoing advancements in AI and ML poised to further enhance its capabilities. Developments in deep learning, particularly in areas such as transformers and attention mechanisms, are expected to improve the accuracy and efficiency of ticket classification systems. Additionally, the integration of real-time analytics and feedback loops can enable continuous learning and improvement, ensuring that support systems remain responsive to customer needs

***Business Use Case:***

A company operates a customer support system that receives a large volume of support tickets from customers. These tickets cover various topics, including technical issues, billing inquiries, product feedback, and more. The company faces several challenges:

1. **Manual Ticket Assignment**: Currently, customer support agents manually categorize each ticket based on its content. This process is time-consuming, error-prone, and inefficient.

2. **Scalability**: As the company grows, the number of support tickets increases. Handling this volume manually becomes unsustainable.

3. **Consistency**: Different agents may categorize similar tickets differently, leading to inconsistencies in handling customer queries.

***Solution with Text Classification***

To address these challenges, the company aims to implement an automated system for ticket classification using machine learning:

**Tag Definition**: Define relevant tags or categories (e.g., “website functionality,” “shipping,” “complaint,” “refund request,” etc.).

**Training Data:** Gather a labeled dataset of historical customer support tickets, where each ticket is associated with the appropriate tag.

**Feature Extraction**: Convert the text content of each ticket into numerical vectors using techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or word embedding’s.

**Model Selection**: Choose an appropriate text classification algorithm (e.g., Naïve Bayes, Support Vector Machines, or deep learning models).

**Model Training**: Train the selected model using the labeled dataset. The model learns to recognize patterns in the text data and associate them with specific tags.

**Evaluation:** Assess the model’s performance using metrics like accuracy, precision, recall, and F1-score.

**Deployment**: Deploy the trained model into the company’s customer support system. When a new ticket arrives, the model predicts its category based on the text content.

**Automation**: Automatically route the ticket to the appropriate support team based on the predicted category.

**Continuous Learning**: Periodically retrain the model with new labeled data to adapt to changing customer queries.

***Business Use Case Benefits:***

**Efficiency**: Customer support agents spend less time manually categorizing tickets.

**Consistency**: The model ensures consistent and unbiased categorization.

**Scalability**: The system can handle a growing volume of tickets without increasing manual effort.

**Insights**: Analyse patterns in ticket categories to improve products/services and enhance customer experience.

***Sentiment Analysis:***

Sentiment analysis is becoming very important to study growing opinions faster and faster within social media and other sites, the huge explosion in information in recent years in the sites of communication, air traffic and alternative markets, all this huge amount of information cannot be controlled and analysed used the traditional way, so the scientists and researchers developed a high-efficiency techniques to deal with this data. This requires the SA to process data and know its polarity to determine the right decision. SA involves five steps to process data; those are data collection, text preparation, sentiment detection, sentiment classification, and presentation of output.

***1. Data Collection:***

The collection of data from sources like user groups, Twitter, Facebook, blogs and commercial website such as amazon.com and alibaba.com, etc. This data cannot be analysed using traditional methods like scanning, text analysis, and language processing, which is used for extraction and classification.

***2. Data pre processing:***

Before analysing the data, it’s essential to clean and pre-process it. Remove any noise, irrelevant information, or formatting issues.

Common pre-processing steps include converting text to lowercase, removing special characters, handling misspellings, and eliminating stop words (common words like “the,” “and,” etc.).

***3. Analyse Your Data:***

Apply sentiment analysis techniques to the cleaned data. There are two main approaches:

**Machine Learning-Based Approach:**

Train a machine learning model using labeled data (supervised learning).

The model learns to classify text into positive, negative, or neutral sentiments based on features extracted from the text.

**Lexicon-Based Approach:**

Use predefined lexicons or dictionaries containing words and their associated sentiment scores.

Assign sentiment scores to individual words in the text and aggregate them to determine the overall sentiment.

***4. Data visualization:***

Create visualizations to understand the sentiment distribution.

Bar charts, word clouds, or sentiment heat maps can help you visualize positive, negative, and neutral sentiments.

***Implementation Steps Of Text Classification:***

***1. Data Collection:*** Gather historical support tickets with their corresponding labels (categories and urgency levels).

***2. Pre-processing:*** Clean and pre-process the text data (Lower Case, Remove links, Remove next lines, Remove words containing numbers, Remove extra spaces, Remove special characters, Remove stop words, Stemming, Lemmatization).

***3. Feature Extraction:*** Convert text into numerical features (**TfidfVectorizer**).

***4. Model Selection:*** Choose an appropriate machine learning algorithm (Multinomial Naive Bayes, LogisticRegression, RandomForestClassifier, support vector classifier, KNeighborsClassifier).

***5. Training***: Train the model using labelled data.

***6. Evaluation***: Evaluate the model’s performance using metrics such as accuracy, precision, recall, and F1-score.

***7. Deployment***: Deploy the trained model in the production environment to automatically classify incoming support tickets.

***About Dataset:***

Dataset consist of following column:

1. Label: Integer

2. Text: String

The total length of the dataset is 16001 data. It consist of labels such as 0:sadness , 1:joy , 2:love , 3:anger , 4:fear , 5:surprise.The count of sadness is: 4666,joy is: 5362, love is: 1304,anger is: 2159,fear is: 1937,surprise is : 572.

A few classifications, similar to surprise and love, have less information compared with others, similar to satisfaction and misery. This imbalance should be tended to during model preparation. This is vital to guaranteeing that the classifier functions admirably for all feelings. To deal with this irregularity, strategies include, for example, adding more information or changing the quantity of tests. At first, we dealt with the nostalgic investigation utilizing different datasets, which gave us a better understanding of the text grouping.

***Data (Text) Pre-processing:***

***Lower Case:*** Convert all text to lowercase to ensure consistency and avoid duplication of words that differ only in case.

***Remove Links:*** Eliminate any hyperlinks present in the text as they typically do not contribute to the meaning of the text and can be distracting.

***Remove New Lines (\n):*** Strip out newline characters to ensure that text is continuous and easier to process.

***Words Containing Numbers:*** Exclude words that contain numbers since they often represent identifiers or measurements rather than meaningful words.

***Extra Spaces:*** Trim excess spaces to standardize the text and ensure consistent tokenization.

***Special Characters:*** Remove special characters like punctuation, symbols, or emoji’s, as they may not contribute to the semantics of the text.

***Removal of Stop Words:*** Filter out common words (stop words) like "the," "is," "and," which occur frequently but often do not carry significant meaning.

***Stemming:*** Reduce words to their base or root form, removing suffixes and prefixes. For example, "running" becomes "run." This helps in reducing the dimensionality of the feature space and capturing the essence of the word. Popular stemming algorithms include Porter Stemmer, Snowball Stemmer, and Lancaster Stemmer.

***Lemmatization:*** Similar to stemming, but instead of just removing prefixes and suffixes, lemmatization maps words to their base or dictionary form (lemma). For example, "better" becomes "good." This approach ensures that the transformed words are valid lemmas, which can improve interpretability.

***Modeling Approach:***

**1. Data (Text) Pre-processing:**

Lower Case

Remove links

Remove next lines (\n)

Words containing numbers

Extra spaces

Special characters

Removal of stop words

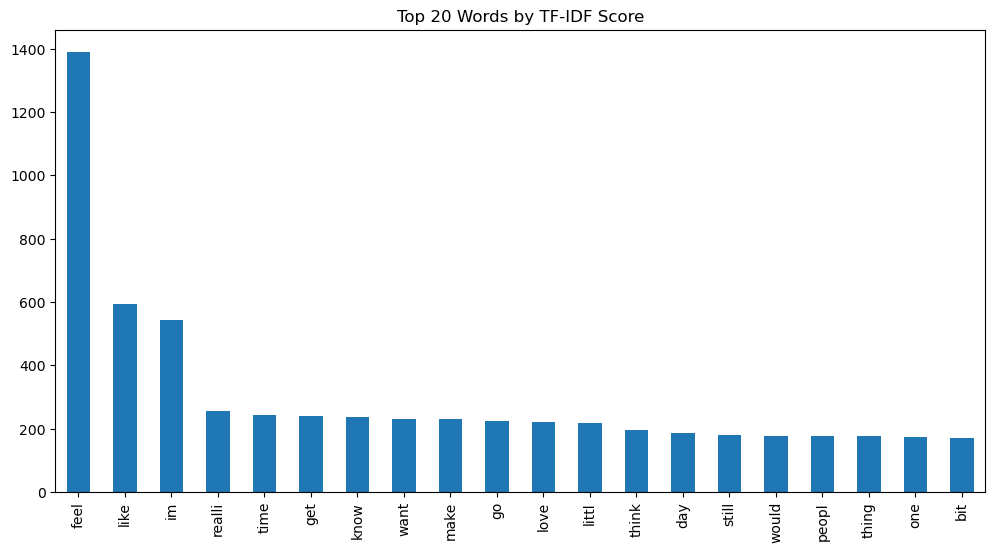
Stemming

Lemmatization

***2. Featuring Engineering:***

• The TfidfVectorizer is used to convert the pre-processed text into numerical features using the Term Frequency-Inverse Document Frequency (TF-IDF) representation.

• X contains the TF-IDF features, and y contains the corresponding labels (target variable).



**3. Model Building:**

1. Divide the dataset in to Train (70%), Test (20%) and Validation (10%) datasets.

2. Build at least 3 classification models

**Step 1:** Build model 1 and generate the classification report (Performance metrics using Confusion Metrics) for both

Training and Test datasets.

**Step 2:**

• Use grid search or binary search for Hyperparameter Tuning.

• Use at least 2 values for each hyperparameters.

• Choose the best model parameters based on grid search and generate the classification report (Performance metrics using Confusion Metrics) for both Training and Test datasets.

Step 3: Repeat step 1 and 2 for Model 2 and Model 3 as well.

Step 4: Now choose the final model based on the classification report (Performance metrics using Confusion Metrics) for both Training, Test and validation datasets.

**4. Data Visualization: Input and Output plots**

***Model used:***

***1. Multinomial Naive Bayes (MNB):***

Multinomial Naive Bayes is a probabilistic classifier based on Bayes’ theorem.

It’s particularly useful for text classification problems in NLP.

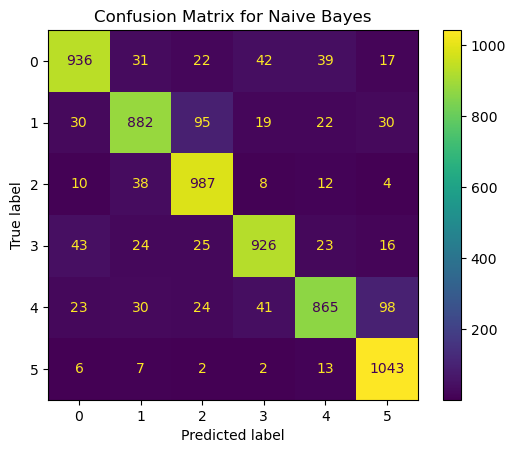
MNB assumes that features (e.g., word counts) are conditionally independent given the class label.

It’s well-suited for data with features representing discrete frequencies or counts (e.g., word occurrences).

**Hyperparameters**:

Alpha is Regularization strength (similar to C in Logistic Regression).

Suggested values: [0.01, 0.1, 1]



***2. Logistic Regression:***

Logistic Regression is a popular supervised learning algorithm used for binary classification (where the target variable has two classes) and multiclass classification (where the target variable has more than two classes).

Despite its name, it is used for classification, not regression.

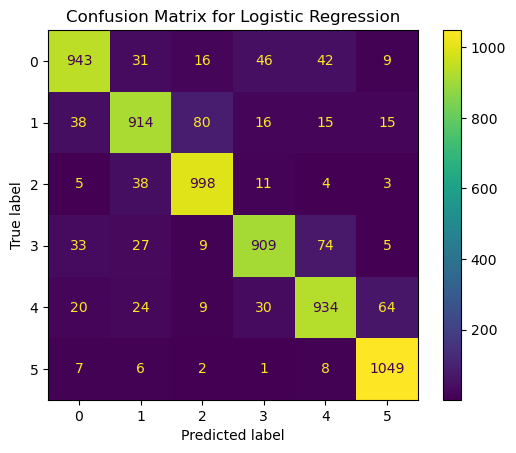
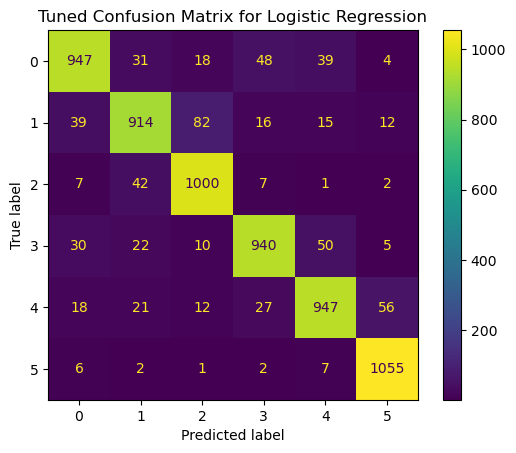
The goal of logistic regression is to predict the probability that an instance belongs to a particular class.

It models the relationship between a set of input features (predictors) and a binary output (0 or 1) using the logistic function (also known as the sigmoid function).

**Hyperparameters**

C: Regularization parameter controlling the inverse of regularization strength.

Suggested values: [0.1, 1, 10]

***3. RandomForestClassifier:***

The RandomForestClassifier is part of the ensemble learning family in skit-learn.

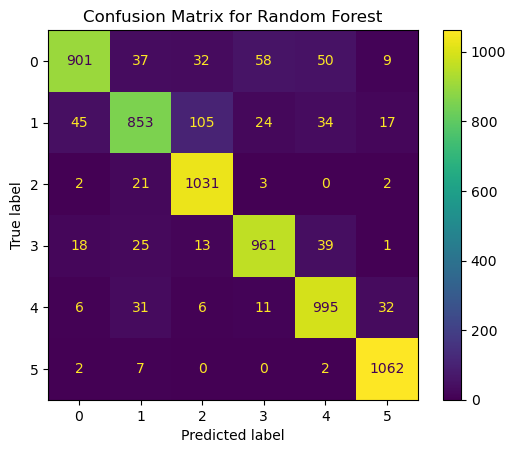
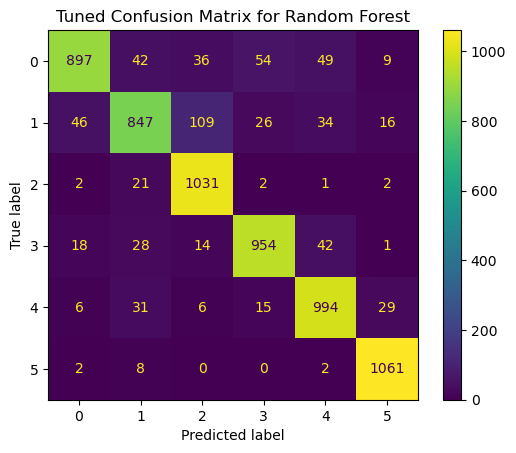
It is a meta-estimator that combines multiple decision tree classifiers to improve predictive accuracy and control overfitting.

***Hyperparameters***

Number of decision trees (controlled indirectly by the number of estimators).

max\_depth: Maximum depth of each tree.

Suggested values: n\_estimators: [50, 100], max\_depth: [10, 20].

***Hyperparameter tuning:***

Hyperparameter tuning refers to the process of selecting the optimal values for a machine learning model’s hyperparameters.

The goal of hyperparameter tuning is to find the best combination of hyperparameter values that maximizes the model’s performance.

***Hyperparameters:***

Hyperparameters are settings or parameters that control the learning process of a machine learning model.

Unlike model parameters (such as weights learned during training), hyperparameters are not learned from the data but are set before training begins.

Examples of hyperparameters include:

Learning rate

Number of hidden layers in a neural network

Regularization strength

Kernel size in a support vector machine (SVM)

Number of neighbors in k-nearest neighbors (KNN)

**Hyperparameter Tuning python**

def build\_and\_evaluate\_model(model, param\_grid):

...

This function handles the training and evaluation of a model with hyperparameter tuning:

**Grid Search for Hyperparameter Tuning python:**

grid\_search = GridSearchCV(model, param\_grid, cv=5, scoring='accuracy')

grid\_search.fit(X\_train, y\_train)

GridSearchCV is used to perform an exhaustive search over specified parameter values (defined in param\_grid) for the given model. It uses 5-fold cross-validation to evaluate each combination of parameters.

***Conclusion:***

In our project, we evaluated three different models to identify the best performing one for our classification task: Logistic Regression, Random Forest, and Naive Bayes. Among these models, Logistic Regression emerged as the best performer with an impressive accuracy of 0.91. However, to ensure a comprehensive evaluation, we considered additional metrics such as precision, recall, F1-score, and accuracy

The Logistic Regression model not only delivered the highest accuracy but also demonstrated a robust performance across all other evaluation metrics. Its high precision and recall values indicate that the model is effective in correctly identifying both the positive and negative classes, minimizing false positives and false negatives.

In contrast, while the Random Forest model captured more complex relationships within the data, it exhibited slightly lower performance metrics and higher computational complexity. The Naive Bayes model, although simple and efficient, did not match the predictive performance of Logistic Regression.

Additionally, Logistic Regression offers the advantage of interpretability, making it easier to understand the influence of each feature on the prediction. This is particularly beneficial for stakeholders who require transparent and explainable models for decision-making.

Considering the balance of accuracy, interpretability, and computational efficiency, we recommend the Logistic Regression model for deployment. Its consistent performance across various metrics and ease of interpretation make it a suitable choice for our classification task, ensuring reliable and actionable insights.