Text Classification

by

Ajay Chougule

Table of content

1.Introduction

Text classification is a machine learning technique used to assign predefined categories to text documents. This process involves several key steps and techniques, which can vary depending on the complexity of the problem and the chosen approach.

Advantage of text class:

1. Automated Processing
2. Consistency and Accuracy
3. Real-time Processing
4. Improved Decision-making
5. Cost-effectiveness

Applications of Text classification:

Spam Filtering: Email providers like Gmail use text classification to filter out spam emails.

Sentiment Analysis: Companies analyze customer feedback and social media posts to gauge public sentiment towards their products.

Topic Categorization: News websites categorize articles into sections like sports, politics, and entertainment.

Customer Support: Chatbots classify user queries to provide relevant responses or escalate to human agents.

2.Problem statement

2. Problem statement and business usecase

Understanding customer feedback is crucial in today's competitive market, as it is often unstructured and difficult to analyze manually. The current manual approach is time-consuming, error-prone, and inefficient, overlooking subtle trends and sentiments. To address these limitations, an automated text classification system is being developed, accurately categorizing customer feedback into predefined categories like positive, negative, neutral, product-related, and service-related. This system will use natural language processing and machine learning techniques to understand the context and sentiment of the feedback, providing actionable insights for businesses.

An automated text classification system can revolutionize businesses by providing real-time insights into customer feedback. This technology can enhance customer experience, optimize operations, and drive strategic decisions, thereby reducing the labor-intensive manual analysis process and ensuring a comprehensive understanding of customer sentiments.

3.Dataset and Prepreocessing

3.About dataset and Prepreocessing

Dataset:

1. Dataset Description:

Source of Data: The dataset comprises customer feedback collected from multiple sources including:

- Social media platforms (e.g., Twitter, Facebook)
- Review sites (e.g., Yelp, Google Reviews)
- Customer service logs (e.g., emails, chat transcripts)

Dataset size

- Number of records:16000

2.Preprocessing:

Data preprocessing is a crucial step in the data analysis and machine learning workflow. It involves transforming raw data into a clean and usable format, preparing it for analysis or model training. The main goal of data preprocessing is to improve the quality of the data, address any inconsistencies, and transform it into a format suitable for analysis.

Here's an overview of the key steps involved in data preprocessing:

1. Remove all the special characters (any letter or a digit)
2. remove all single characters (surrounded by whitespace)
3. Remove single characters from the start
4. Substituting multiple spaces with single space
5. Converting to Lowercase
6. Stemming
7. Lemmatization- splits into list of words [‘The’, ‘quick’, ….]

3. Feature Engineering(Tokenisation, Vectorising Using TF-IDF):

Feature engineering is the process of using domain knowledge to create new features from raw data, which can enhance the performance of machine learning models. It involves transforming existing data into formats that are more suitable for analysis and improving the model's ability to capture relevant patterns.

Tokenization is a crucial preprocessing step in natural language processing (NLP) where text is segmented into smaller units called tokens. These tokens can be words, phrases, or even characters. The primary goal of tokenization is to break down the text into manageable pieces for further analysis and modeling.

Steps of Tokenization:

1. **Text Input:** Start with the raw feedback text.
2. **Lowercasing:** Convert all text to lowercase to ensure consistency.
3. **Splitting:** Divide the text into individual words or tokens. This is usually done by splitting the text at spaces and punctuation marks.
4. **Cleaning:** Remove any unwanted characters or punctuation that were not needed for the analysis.

   Example:

- **Raw Text:** "Great product! Works perfectly."
- **Tokens:** ["great", "product", "works", "perfectly"]

4.Vectorising Using TF-IDF:

Vectorization is the process of converting text data into numerical format so that it can be used by machine learning algorithms. TF-IDF stands for Term Frequency-Inverse Document Frequency. It is a numerical statistic that reflects the importance of a word in a document relative to a collection of documents (corpus). TF-IDF helps in transforming text data into meaningful numerical representations for machine learning models

Steps for Vectorization Using TF-IDF:

1. **Collect Text Data:**
   - Gather the feedback texts you want to analyze.
2. **Tokenization:**
   - Break the text into individual words (tokens).
3. **Compute Term Frequency (TF):**
   - Count how many times each word appears in a document.
   - Example: In the document "great product works perfectly", the word "great" appears once, so its term frequency is 1.
4. **Compute Document Frequency (DF):**
   - Count how many documents contain each word.
   - Example: If the word "great" appears in 10 out of 100 documents, its document frequency is 10.
5. **Compute Inverse Document Frequency (IDF):**

- Calculate the inverse of the document frequency to find how unique a word is.
- Formula: IDF = log(Total Number of Documents / Document Frequency)
- Example: If "great" appears in 10 out of 100 documents, IDF = log(100 / 10) = 1.

6. **Compute TF-IDF Score:**
    - Multiply the term frequency (TF) by the inverse document frequency (IDF) for each word.
    - Example: If the term frequency of "great" is 1 and its IDF is 1, then TF-IDF = 1 * 1 = 1.
7. **Create Vectors:**
    - Convert each document into a vector of TF-IDF scores, where each score represents the importance of a word in that document.
    - Example: If a document has the words "great", "product", "works", "perfectly", its TF-IDF vector might look like [1, 0.5, 0.3, 0.7] depending on the TF-IDF scores of these words.

4.Model Approaching

4.Model approaching:

Since Model approaching refers to the process of selecting, designing, and implementing a machine learning model to solve a specific problem

Following model will be approached:

1. Multinomial Naive Bayes (MNB)
2. A multilayer perceptron (MLP)
3. Support Vector Machine (SVM)
4. Decision tree
5. Random forest

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

5.Model building

5.Model used:

1. Multinomial Naive Bayes (MNB):
   Multinomial Naive Bayes (MNB) is a very popular and efficient machine learning algorithm that is based on Bayes' theorem. It is commonly used for text classification tasks where we need to deal with discrete data like word counts in documents.

2. Multilayer perceptron (MLP):

Multilayer perceptron is a name for a modern feedforward artificial neural network Consisting of fully connected neurons with a nonlinear kind of activation function, organized in at least three layers notable for being able to distinguish data that is not linearly separable.

3. Support Vector Machine (SVM) :

Support Vector Machine (SVM) is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and even outlier detection tasks. SVMs can be used for a variety of tasks, such as text classification, image classification, spam detection, handwriting identification, gene expression analysis, face detection, and anomaly detection.

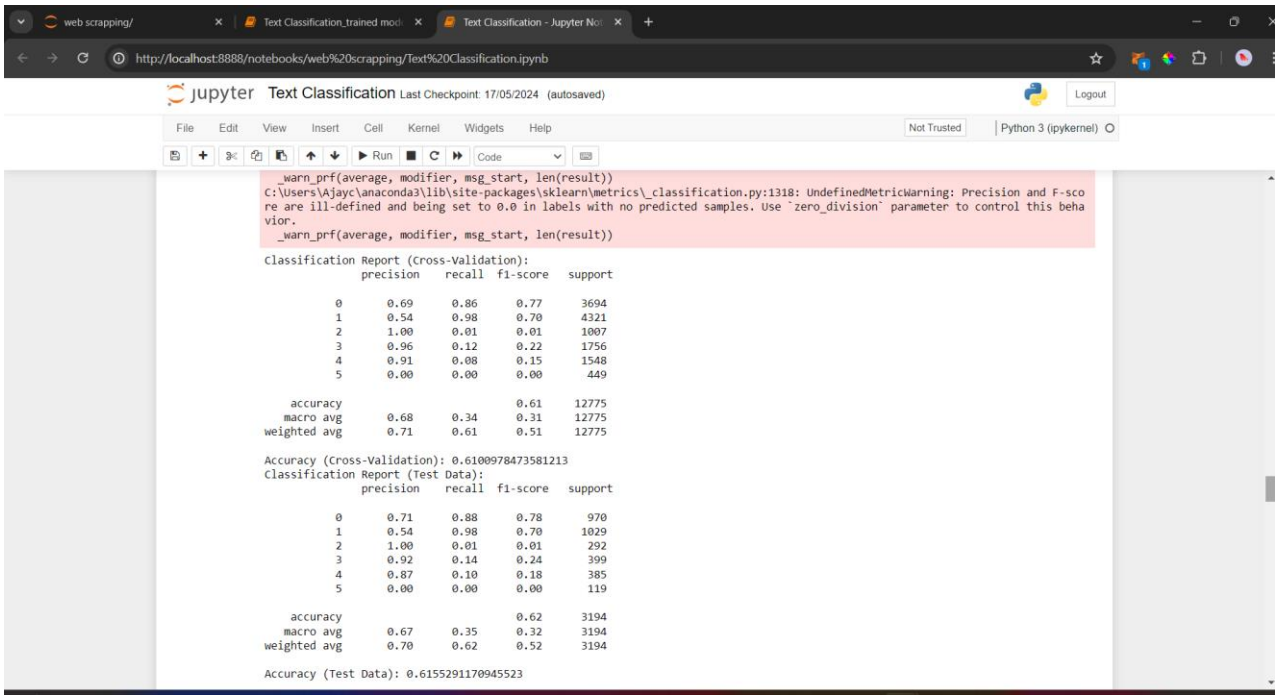
6.Model result

6.Model result:

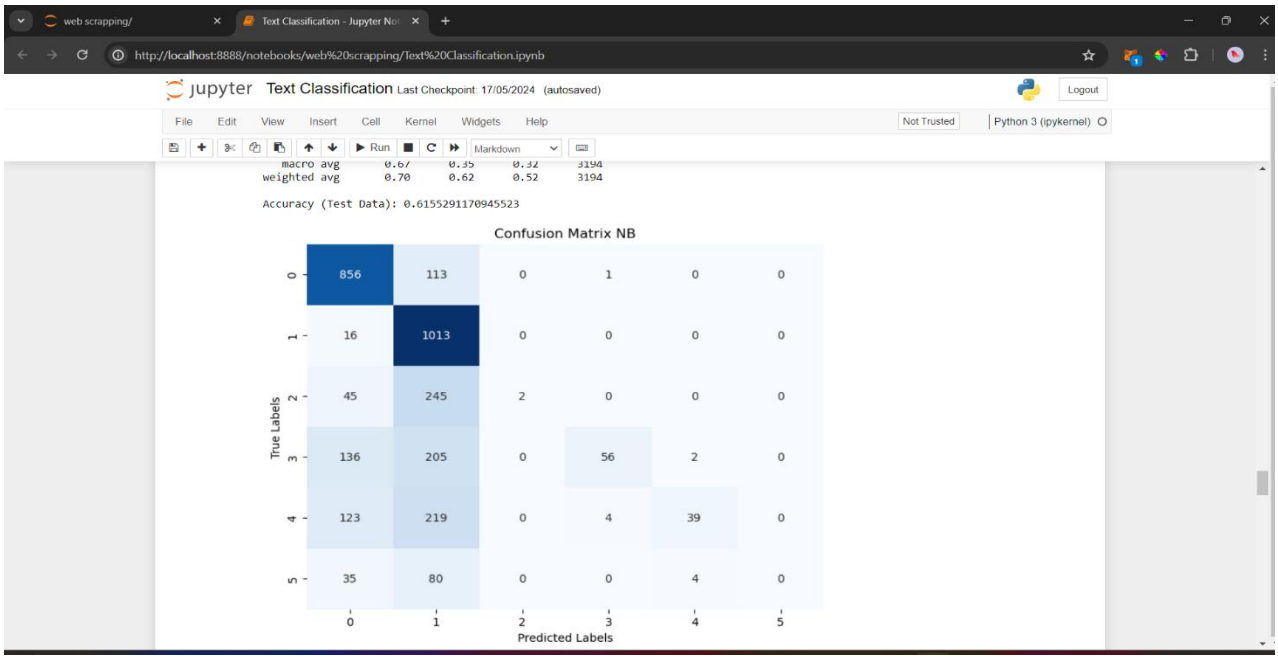
These results help determine how well the model is performing and whether it meets the objectives of the project. Here are the main aspects to consider:

- Accuracy
- Precision
- Recall
- F1-score

1.MNB(multinomial Naive Bayes classifier):

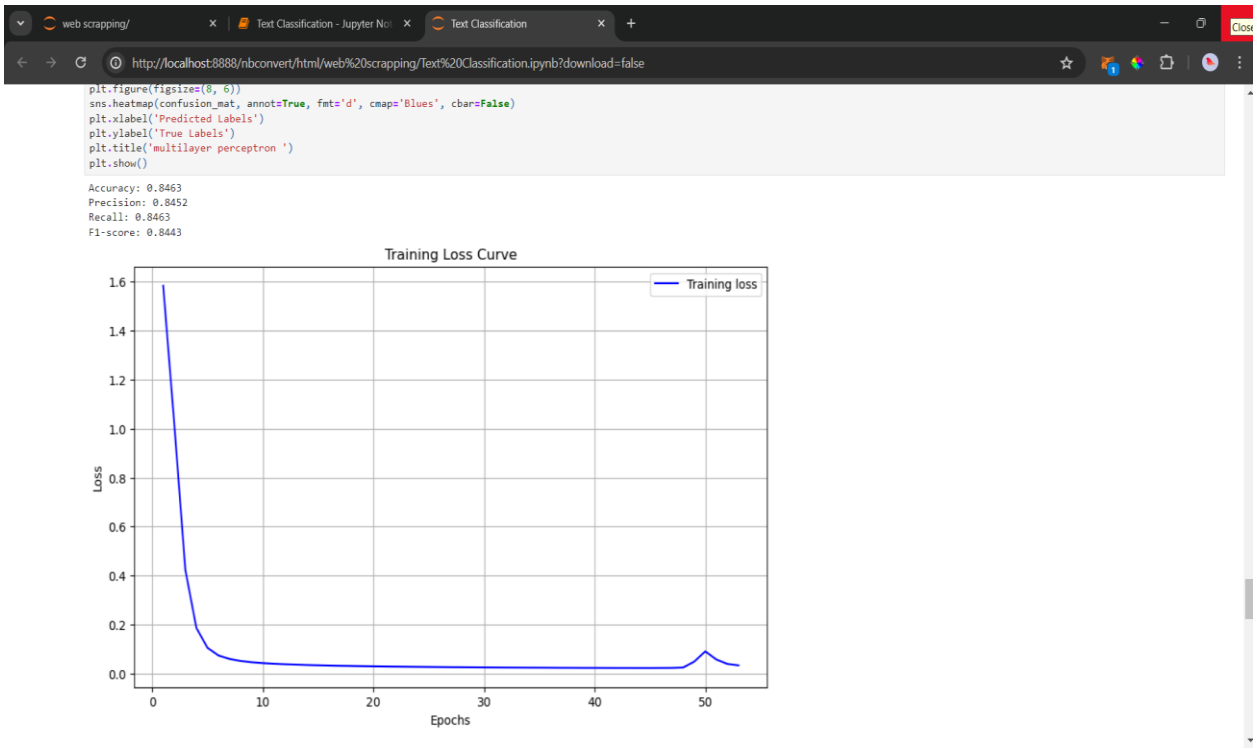


Result

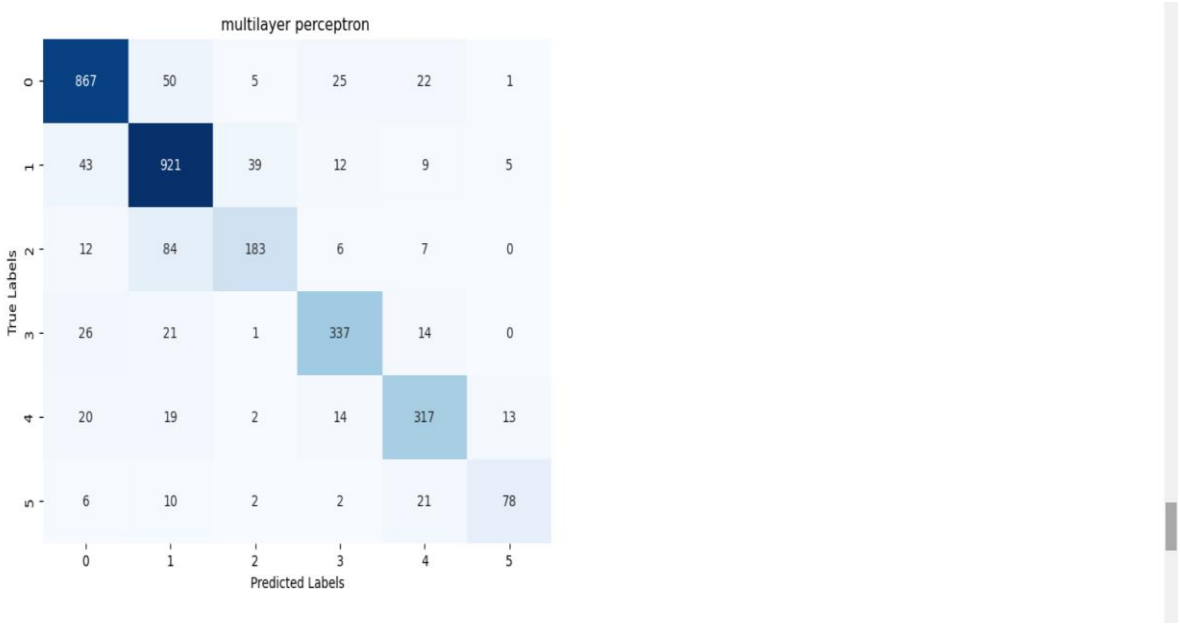


Confusion matrix

2. MLP(Multilayer perceptron):

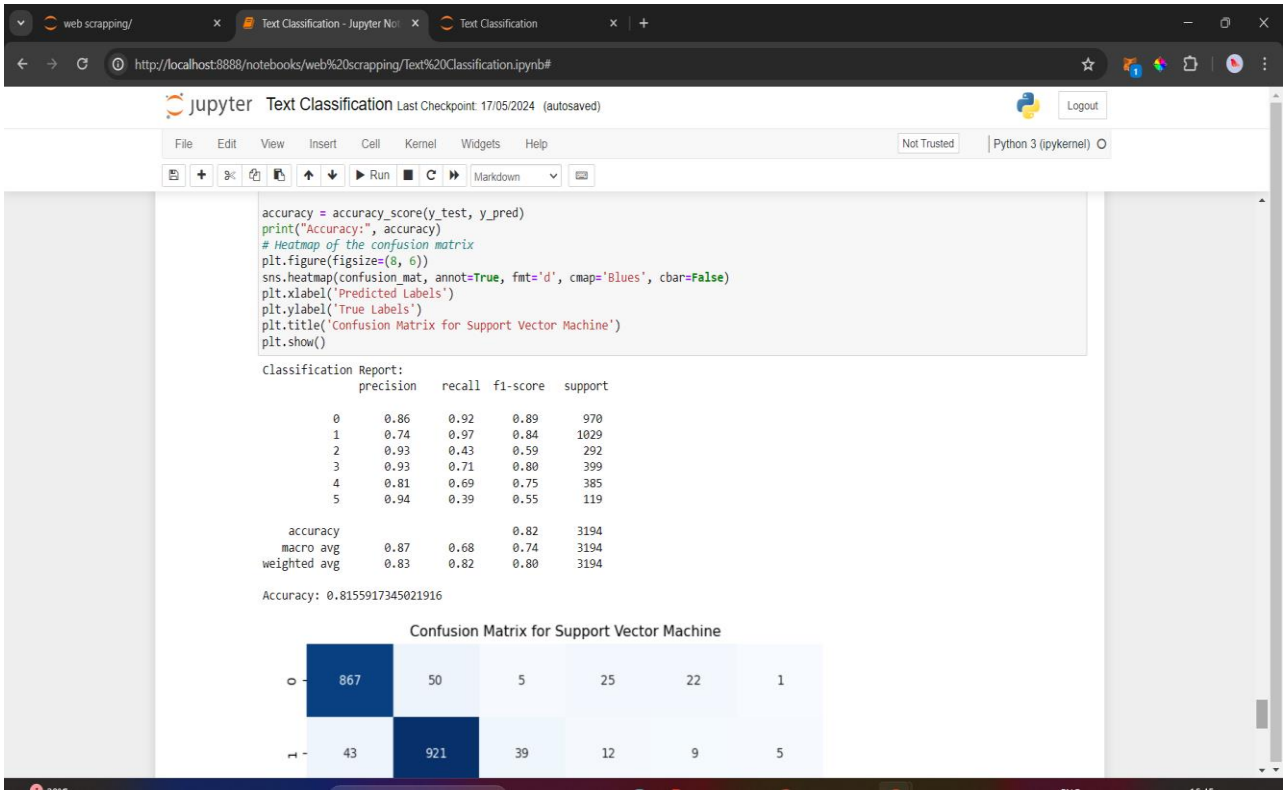


Result

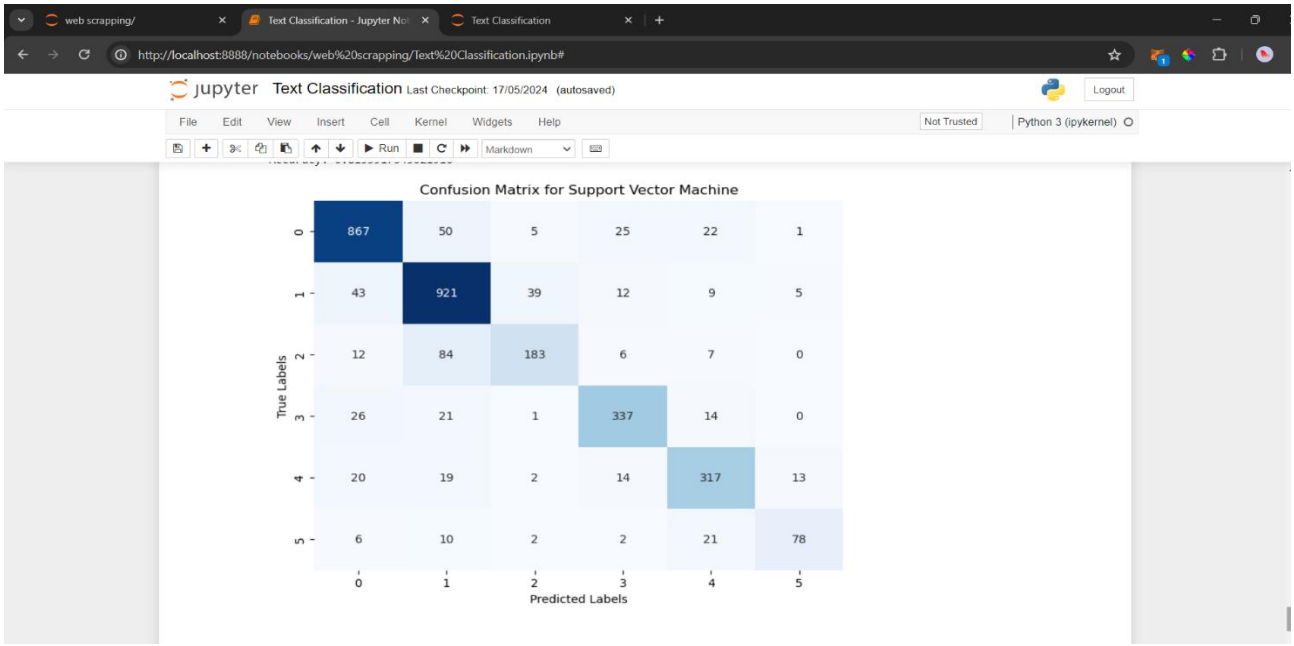


Confusion matrix

SVM(Support Vector Machine):



Result



Confusion matrix

7.Recommendations and resolutions

Recommendation:

Conclusion from above three model Support Vector Machine (SVM) perform the best with the **Accuracy: 0.8155917345021916.**

Resolution:

As multilayer perceptron and **multinomial Naive Bayes classifier does not perform as we aspected instead of we will use decision tree and Random forest.**

- **Decision Trees: Offer a simple, interpretable model that can capture non-linear relationships and handle different data types. They are transparent but can overfit to training data.**
- **Random Forests: Enhance decision trees by using an ensemble approach, reducing overfitting, and improving accuracy and robustness. They handle high-dimensional data well and provide insights into feature importance, making them versatile and powerful for various predictive tasks.**