# Endsem Report


1st Khushi Patel
*Btech Computer Science*
*(Ahmedabad University)*
Ahmedabad, India
khushi.p4@ahduni.edu.in

2nd Devyash Shah
*Btech Computer Science*
*(Ahmedabad University)*
Ahmedabad, India
devyash.s@ahduni.edu.in

3rd Aastha Gaudani
*Btech Computer Science*
*(Ahmedabad University)*
Ahmedabad, India
aastha.g2@ahduni.edu.in

4th Simran Khoja
*Bba Honors*
*(Ahmedabad University)*
Ahmedabad, India
simran.k1@ahduni.edu.in


## I. ABSTRACT


Text classification is indeed a crucial task in natural language processing (NLP) that involves categorizing text data into predefined categories or labels. It is widely used in various applications, such as sentiment analysis, spam detection, topic modelling, and content filtering. manually labelling text data is a challenging and time-consuming task, especially when dealing with large datasets. That's why machine learning-based approaches have gained significant attention in recent years, as they can automate the process of text classification and produce more accurate and efficient results. There are various machine learning algorithms and techniques used for text classification, including Naive Bayes, Support Vector Machines (SVMs), Decision Trees, Random Forests, and Neural Networks. These methods use different features and models to learn the patterns and relationships between text data and labels. Text classification is a vital task in NLP that enables efficient data search and analysis. Machine learning-based approaches have made significant progress in automating this task, and further advancements in this field are expected in the coming years.


## II. INTRODUCTION

This project aims to perform text category classification on a BBC news dataset using machine learning techniques. The dataset consists of articles published by the British Broadcasting Corporation (BBC) across five categories: business, entertainment, politics, sport, and tech. The goal is to develop a model that can accurately predict the category of a given article based on its text content.

Classifying news articles automatically can have significant practical applications, such as helping news organizations understand their readership better or enabling personalized news recommendation systems. The project will use a variety of machine learning algorithms, including traditional models such as logistic regression and support vector machines.

The main challenges in this project include feature extraction from the text data, dealing with class imbalance (i.e., some categories having fewer samples than others), and optimizing the model's performance in terms of accuracy, precision, and recall. To address these challenges, we will experiment with different text representation techniques, such as bag-of-words and word embeddings, and employ various strategies for addressing the class imbalance, such as oversampling or undersampling.

## III. LITERATURE REVIEW

The base paper considered provides an algorithm that includes both the LDA as well as SVM. The algorithm proposed first applies LDA for dimensionality reduction, and applies SVM for classification afterwards. The purpose of using LDA is to separate the samples from different classes so to get such a lower dimensional space. The main function of LDA is to minimize the distance within the class and to maximize the distance between classes considering the assumption that each class follows Guassian probability density function with same covariance. The resulting lower dimensional space is more efficient and discriminative for text representation tha LSI. The Support Vector Machine is applied on the later stage to classify the text. The results observed from the experiments on a benchmark dataset depict the effectiveness of the algorithm that the paper proposes for text classification.

## IV. METHODOLOGY

This project aims to develop a machine learning model that accurately predicts the category of news articles from the BBC news dataset. This will involve several techniques, such as data cleaning and preprocessing, feature extraction, and machine learning algorithms. The first step is cleaning the text data using NLTK libraries. The model will be trained using MNB and evaluated using an accuracy score. The experiment will be repeated using CountVectorizer + MNB and Tfidf + Catboost to compare the performance of both techniques. The aim is to demonstrate the effectiveness of various machine learning techniques in text category classification and identify the best approach for accurately classifying news articles from the BBC dataset.

1) Data Collection: The first step was to collect the BBC News dataset. We used the OS and glob modules to import data from each category folder and read the text files.
2) Data Preprocessing: We then performed preprocessing on the raw text data to clean and normalize the text. We removed punctuations, stopwords, and performed lemmatization to extract the root word of each token.

3) Feature Extraction: The next step was to convert the preprocessed text data into numerical features that machine learning algorithms can use. We used the CountVectorizer from Scikit-learn to generate a bag of words representation of the text data.
4) Multinomial Naive Bayes (MNB): Multinomial Naive Bayes (MNB) is a regularly employed probabilistic technique for text categorization because to its effectiveness, scalability, and simplicity. Modelling the conditional probability distribution of each word in a document given its class label is based on the assumption that given the class label, the probabilities of distinct words are independent of one another. As a result, it is now possible to quickly estimate the model parameters from large datasets and to calculate the joint probability distribution of the words in a text. By learning the statistical patterns of the training data, MNB can classify new texts using the Bayes theorem. MNB works well with high-dimensional, sparse text data and is capable of handling noise and irrelevant information. For optimum performance, its hyperparameters, including the smoothing factor alpha, can be adjusted.
5) Cat Boost Classifier: The decision trees' gradients are boosted by the CatBoost Classifier algorithm. It automatically manages missing data and works well with both categorical and numerical features. One of its key advantages is that it can handle categorical features with high cardinality without the need for one-hot encoding. It also contains a built-in cross-validation strategy to avoid overfitting. Automatic hyperparameter adjusting is possible with CatBoost Classifier thanks to the built-in parameter search tool. Gradient-based and symmetric tree pruning are two additional regularisation techniques utilised by the CatBoost Classifier to lessen overfitting and improve generalisation performance.

## V. Implementation

1) Data Preprocessing: The text data was preprocessed by performing the following steps:

    a. Removing punctuation and special characters
    b. Converting all text to lowercase
    c. Removing stop words
    d. Lemmatizing the text

2) CountVectorizer: CountVectorizer is a method for converting text data into numerical features. It counts the frequency of each word in the text data and creates a sparse matrix of the word counts. The resulting matrix can be used as input to a machine-learning algorithm.

3) MultinomialNaiveBayes: The Multinomial Naive Bayes (MNB) algorithm uses CountVectorizer to convert a set of text documents and their corresponding categories into numerical format. The algorithm then learns the relationship between the features (i.e., words) and the categories by assuming that each characteristic is independent of the others. A model that can predict the categories of novel, unstudied text data is produced via the MNB approach. After assessing the model's performance on a held-out test set, the technique returns the best hyperparameters found using GridSearchCV, the suitable cross-validation score, the accuracy of the predictions, the classification report, and the confusion matrix. Overall, the Multinomial Naive Bayes method provides a realistic approach to text categorization, making it useful for a range of applications.

4) Catboost: The TF-IDF vectorization of preprocessed text input is used to train the CatBoost algorithm. A number of hyperparameters, including iterations, depth, and loss function, characterise the CatBoost classifier. A pool object made from the TF-IDF feature matrix and the related labels is then used to train the model using the training data. On the basis of the test data, the algorithm predicts outcomes and assesses how well the model classified the text data into the appropriate categories. Overall, the CatBoost algorithm offers a strong method of classifying texts by utilising techniques for gradient boosting to increase the model's precision and interpretability.

5) Accuracy: Accuracy is a measure of how well a machine learning model can predict the correct label for a given input. It is defined as the number of correctly predicted labels divided by the total number of predictions. It can be expressed as a percentage.

    The formula for accuracy:

$$Accuracy = \frac{(Numberofcorrectlypredictedlabels)}{(Totalnumberofpredictions)}$$

6) Prediction of new text inputs: To predict the category label for a new input text, the following steps are performed:

    a. Preprocess the new input text using the same preprocessing steps as the training data.
    b. Tfidf assigns weights to each term based on their frequencies in the documents.
    c. Extract the topic distribution features using Tfidf.
    d. Use the trained catboost model to predict the category label for the new input text based on the Tfidf topic distribution features.
    OR
    b. Count Vectorizer counts the frequency of each term in the text data and gives a matrix of numerical features.
    c. Extract the topic distribution features using Count Vectorizer.

d. Use the trained Multinomial Naive Bayes model to predict the category label for the new input text.

7) TF-IDF: TF-IDF stands for Term Frequency-Inverse Document Frequency. It is a numerical statistic that reflects how important a word is to a document in a collection or corpus of documents. TF-IDF is a measure that combines two factors:
Term Frequency (TF): the frequency of a term (word) in a document. It measures how often a word appears in a document.
Inverse Document Frequency (IDF): the inverse of the frequency of the term in the whole collection of documents. It measures how unique or rare a word is across all documents.
The formula for calculating TF-IDF is as follows:
TF-IDF = (Term Frequency / Total number of terms in the document) * log(Total number of documents / Number of documents containing the term).

The higher the TF-IDF score of a term in a document, the more important or relevant it is to that document.
The use of TF-IDF and countVectorizer is done in different models. First we have generated a model using TF-IDF for extraction and then training the model using SVM. In the second model we have used countVectorizer to calculate word frequency and we have used LDA to predict the categories of our model, and we have trained the model with SVM.

## VI. RESULTS

The accuracy of the model implemented using TF-IDF + Catboost was 0.95, whereas the accuracy of the model implemented with CountVectorizer + MNB 0.98.

## VII. CONCLUSION

The difference in the accuracy was because of the dataset comprising of just news articles which are written in a very standardized manner. MNB performs well when the dataset is relatively small as it requires less data to get trained. In the MNB model prediction were based on the probabilities. Hence we conclude that MNB model performed best out of four.

## REFERENCES


[1] Z. Liu, M. Li, Y. Liu and M. Ponraj, "Performance evaluation of Latent Dirichlet Allocation in text mining," 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Shanghai, China, 2011, pp. 2695-2698, doi: 10.1109/FSKD.2011.6020066.

[2] Support Vector Machine (SVM) algorithm - javatpoint. www.javatpoint.com. (n.d.). Retrieved March 11, 2023, from https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

[3] Jain, P. (2021, June 1). Basics of countvectorizer. Medium. Retrieved March 11, 2023, from https://towardsdatascience.com/basics-of-countvectorizer-e26677900f9c

[4] Sklearn.svm.SVC. scikit. (n.d.). Retrieved March 11, 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html