

**A REPORT ON:  
TOPIC CATEGORISATION**

BY

Name(s) of the  
Student(s)

ID.No.

Akshat  
Prakhar Jain  
Aakarsh Goyal  
Saksham Bansal

2019A8PS0492G  
2019A3PS0370G  
2019A3PS0096G  
2019A7PS0056G

AT

**Celebal Technologies**



Practice School-I Station Of:



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE**

**(JUNE 2021)**

**A REPORT ON:  
TOPIC CATEGORISATION**

BY

**Name of the  
Students**

**ID.No.**

**Discipline**

Akshat	2019A8PS0492G	B.E. Electronics and Instrumentation
Prakhar Jain	2019A3PS0370G	B.E. Electrical and Electronics
Aakarsh Goyal	2019A3PS0096G	B.E. Electrical and Electronics
Saksham Bansal	2019A7PS0056G	B.E. Computer Science

Prepared in partial fulfillment of the Practice School - I  
Course No. BITS C221/BITS C231/BITS C241

AT

**Celebal Technologies**



Practice School-I Station Of:



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE**

**(JUNE 2021)**

# ACKNOWLEDGEMENT

First of all, we are grateful to BITS Pilani's Practice School Division for successfully organising this internship programme for us. We also want to express our gratitude to Celebal Technologies, Jaipur, for providing such a wonderful working atmosphere in which we can gain both soft and technical skills that we would need in the near future as we enter corporate or industry life. I'd also like to thank the same for providing us with The Canvas LMS, an interactive and efficient learning management system, without which this virtual PS school would not have been possible.

We are really fortunate that we had the kind association of Mr. Sarthak, as our Industry mentor. His constant encouragement throughout the internship has been of immense importance and has helped us a lot in performing better on our project domain. We would also like to thank Prof. Chitranajan Hota, our off-Campus PS faculty, whose constant support and guidance regarding our project domain and the evaluations conducted, has proven to be very encouraging and motivating.

His exemplary guidance, constant encouragement, and careful monitoring throughout the internship has been of immense importance and has helped us a lot in performing better on our project domain.

We are grateful to the Director, Dr. Sudhirkumar V Barai, and the Vice-Chancellor, Dr. Souvik Bhattacharyya, for giving us this wonderful opportunity.

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI**  
**(RAJASTHAN)**  
**Practice School Division**

**Station:** CELEBAL TECHNOLOGIES Pvt. Ltd

**Centre:** Jaipur

**Duration:**

**Date of Start:** 31 May 2021

**Date of Submission:**

**Title of the Project:** Topic Categorization

ID No.	Name(s)	Discipline(s)/of the student(s)
2019A8PS0492G	Akshat	Electronics and Instrumentation
2019A3PS0096G	Aakarsh Goyal	Electrical and Electronics
2019A7PS0056G	Saksham Bansal	Computer Science
2019A3PS0370G	Prakhar Jain	Electrical and Electronics

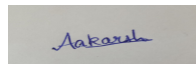
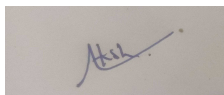
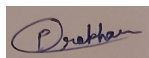
**Name(s) and designation(s) of the expert(s):**

**Name(s) of te PS Faculty:** Prof. Chittaranjan Hota

**Key Words:** Python, Machine Learning, NLP, LDA

**Project Areas:** Machine learning

**Abstract:** Topic Categorization is the title of our project, and it requires us to identify the topic of any paragraph. To determine the various topics covered by product reviews, survey responses, e-mails, and other documents, we will employ an algorithm that works by assuming that each document is made up of a mixture of topics, and then attempting to determine how strong a presence each topic has in a given document. This is accomplished by grouping documents together based on the words they contain and looking for patterns. To do so, we'll use Python data structures to extract data, followed by a machine learning method to determine the topic name.



Signature(s) of Student(s)

Signature of PS Faculty

Date

Date

# Table of Contents

Acknowledgement.....	3
Abstract Sheet.....	4
Introduction.....	6
Machine Learning.....	7
1. Overview.....	6
2. Algorithms and libraries which we will be using.....	8
Implementation.....	9
1. Tokenization.....	9
2. Cleaning of document.....	9
3. Applying the LDA.....	10
4. Scoring.....	11
3. Code.....	12
Conclusion.....	16
References.....	17
Glossary.....	18

## **Introduction:**

Celebal Technologies is a premier software services company in the field of Data Science, Big Data, and Enterprise Cloud. Technologies like Robotics, Artificial Intelligence and Machine Learning algorithms are used to improve business efficiency of our associates. Employees at celebal have strong expertise in diverse industry-standard technologies along with niche emerging areas including Blockchain, Analytics & Visualization, IoT, Chatbot, Data Science- AI & ML.

We got an internship at this company in the field of machine learning, particularly its sub-branch NLP , which is the most basic and useful technique to extract the entities from a text. Later, implementing it with LDA will help us in defining the topic of the article by using probability function.

We 're aiming to determine the heading of a paragraph in this project and to do so, we ' ve constructed a Python code that uses machine learning algorithms to generate a set of words that are closely related to the content of the paragraph. Output also provides statistical data to illustrate how closely the word is related to the paragraph, in addition to the collection of relatable words. The user can obtain a good notion about the topic by looking at the output.

# Machine Learning

## Overview:

Machine learning is a method of data analysis that automates analytical model building. It is a sub-part of artificial intelligence based on the idea that systems can learn from data provided to them, identify patterns and make decisions with the help of minimal human intervention.

In other words, machine learning is the science of making computers act without being explicitly programmed. In the last few years, machine learning has given us so many technologies like self-driving cars, automatic classification of articles and documents, practical speech recognition, effective web search, and a vastly improved understanding of the human genome, etc . Machine learning is so extensive today that you probably use it dozens of times a day without even knowing that you are using it. Many researchers also think that it is the best way to make progress towards human-level AI.

## Algorithms which we will be using in our project of “Topic Categorisation” :

### 1. Natural Language Processing (NLP) :

Natural language processing (NLP) is the ability of a computer program to understand human language as it is spoken and written which is also known as the natural language. It is a component of artificial intelligence (AI).

Text classification is one of the fundamental tasks performed in natural language processing with broad applications such as sentiment analysis, topic labeling, spam detection, and intent detection.

Topic modeling is a branch of unsupervised natural language processing which is used to represent a text document with the help of several topics that can best explain the underlying information in a particular document.

This can be thought of in terms of clustering, but with a difference. Now,

instead of numerical features, we have a collection of words that we want to group together in such a way that each group represents a topic in a document.

## 2. Latent Dirichlet Allocation (LDA):

**Latent:** This refers to everything that we don't know theoretically and are hidden in the data. Here, the themes or topics that document consists of are unknown, but they are believed to be present as the text is generated based on those topics.

**Dirichlet:** In the context of topic modeling, the Dirichlet is the distribution of topics in documents and distribution of words in the topic.

**Allocation:** This means that once we have Dirichlet, we will allocate topics to the documents and words of the document to topics.

While implementing this algorithm we will perform the following steps:

- Tokenization: Split the text into sentences and the sentences into words. Lowercase the words and remove punctuation.
- All stopwords are removed.
- Words are lemmatized – words in third person are changed to first person and verbs in past and future tenses are changed into present.
- Words are stemmed – words are reduced to their root form.

## Libraries used:

1. **nltk (Natural Language Toolkit):** It contains text processing libraries for tokenization, parsing, classification, stemming, tagging, and semantic reasoning.
2. **gensim:** It is designed to process raw, unstructured plain text using unsupervised machine learning algorithms



# IMPLEMENTATION

## 1. Tokenization:

It is the process of breaking text into pieces, called tokens, and ignoring characters like punctuation marks and spaces. There are a couple of different ways we can approach this:

sentence tokenization :- In this, the tokenizer looks for specific characters that fall between sentences, like exclamation points, and newline characters.

word tokenization :- breaking up the text into individual words. This is a critical step for many language processing applications.

We have used word tokenization in our project.

## 2. Cleaning the input text/document:

In this part the paragraph will be converted into the list of words (bag of words) and then the words will be lemmatized, the first letter of each word will be lowercase and our set of words will be cleaned by removing the unnecessary stopwords like punctuations and full stop.

The sample input and output of the above two steps which we have demonstrated in our code:

Sample Input :

Text classification machine learning is one of the most commonly used NLP tasks. In this article, we saw a machine learning simple example of machine learning how text classification machine learning can be performed in Python. We performed a sentimental analysis of movie reviews. We loaded machine learning machine learning machine learning our

trained model and stored it in the model variable. Let's predict the sentiment for the test set using a machine learning machine

Sample output:

After text-cleaning: [['text'], [' classification '], [' machine '], ['learning'], [' one '], [' commonly '], [' used'], [' nlp '], ['task'], [' article '], [' saw '], [' simple '], [' example '], [' performed'], [' python '], [' sentimental'], [' analysis '], [' movie '], [' review '], ['loaded'], ['trained'], [' model'], [' stored'], [' variable '], ['let'], [' predict'], [' sentiment'], ['test'], [' set'], [' using']]

### 3. Applying the Latent Dirichlet Allocation (LDA):

With the help of this algorithm we will try to allocate the articles into different categories with the help of frequency of words which we have already found in the first step.

As we already know that a document or article is a collection of words and this algorithm tries to assign the probability to each word based on its relation with each category.

$$P(\mathbf{W}, \mathbf{Z}, \boldsymbol{\theta}, \boldsymbol{\varphi}; \alpha, \beta) = \prod_{j=1}^M P(\theta_j; \alpha) \prod_{i=1}^K P(\varphi_i; \beta) \prod_{t=1}^N P(Z_{j,t} | \theta_j) P(W_{j,t} | \varphi_{Z_{j,t}})$$
$$\prod_{j=1}^M P(\theta_j; \alpha) \prod_{t=1}^N P(Z_{j,t} | \theta_j)$$

These parts are used to find the category of the article by assigning the probability to each word based on its relation with that particular category.

$$\prod_{i=1}^K P(\varphi_i; \beta) P(W_{j,t} \mid \varphi_{Z_{j,t}})$$

This part is used to find the relation of the word by assigning the probability to each category based on its relation with that particular word.

**4. Scoring the input text based on the similarities with our datasets documents:**

In this part each word from our dataset will be assigned weights and then we will compare the cleaned set of words of our input text with our dataset and try to assign the score based on their similarities with the document present in our dataset.

**5. Scoring the set of 5 weighted words generated in order to check their relevance for being the topic:**

In this part 20 sets of 5 weighted words will be generated from the input text and scored which will help us to judge the relevance of being the topic of the input text.

Code to implement scoring of input text based on the similarities with our datasets documents: [test.py \(github.com\)](https://github.com/test.py)

```
1 import nltk
2 import matplotlib.pyplot as plt
3 import numpy as np
4 from collections import Counter
5 nltk.download('wordnet')
6
7
8 from nltk.stem import WordNetLemmatizer
9 from nltk.corpus import stopwords
10 nltk.download('stopwords')
11
12
13
14 import gensim
15 import string
16 from gensim import corpora
17
18
19
20 nltk.download('punkt')
21
22
23 text1 = "Text classification machine learning is one of the most commonly used NLP tasks. In this article, we saw a machine learning simple example of machine learning how text
24 tokens = nltk.word_tokenize(text1)
25 lowercase_tokens = [t.lower() for t in tokens]
26 #print(lowercase_tokens)
27
28
29 bagofwords_1 = Counter(lowercase_tokens)
30 print("Top 10 most common words", bagofwords_1.most_common(10))
31 print()
32
33
34 alphabets = [t for t in lowercase_tokens if t.isalpha()]
35
36 words = stopwords.words("english")
37 stopwords_removed = [t for t in alphabets if t not in words]
38
39 print("Excluding the stopwords if any", stopwords_removed)
40 print()
41
42
43 lemmatizer = WordNetLemmatizer()
44
45 lem_tokens = [lemmatizer.lemmatize(t) for t in stopwords_removed]
46
47 bag_words = Counter(lem_tokens)
48
49
50 stopwords = set(stopwords.words('english'))
51 exclude = set(string.punctuation)
52 lemma = WordNetLemmatizer()
53
54 def clean(document):
55     stopwordremoval = " ".join([i for i in document.lower().split() if i not in stopwords])
56     punctuationremoval = ''.join(ch for ch in stopwordremoval if ch not in exclude)
57     normalized = " ".join(lemma.lemmatize(word) for word in punctuationremoval.split())
58     return normalized
59
```

```

68 print("After text-cleaning:",final_doc)
69 print()
70
71 texts = [['science','water','space','environment','tree'],
72          ['machine','learning','nlp','classification','task'],
73          ['sports','ball','dholi','skills','kit'],
74          ['politics','referendum','politician','elections','constitution'],
75          ['mud','water','mud','tree','paragraph'],
76          ['money','transaction','bank','finance','note']]
77
78 dictionary = corpora.Dictionary(texts)
79 #print (dictionary)
80
81 DT_matrix = [dictionary.doc2bow(doc) for doc in texts]
82 #print (DT_matrix)
83
84 Lda_object = gensim.models.ldamodel.LdaModel
85
86
87 from gensim import models
88 num_topics = 20
89 lda_model = models.LdaModel(DT_matrix, num_topics=num_topics,
90                             id2word=dictionary,
91                             passes=5, alpha=[0.01]*num_topics,
92                             eta=[0.01]*len(dictionary.keys()))
93
94
95 bow_vector = dictionary.doc2bow(lem_tokens)
96 for index, score in sorted(lda_model[bow_vector], key=lambda tup: -1*tup[1]):
97     print("Score: {}\t Topic: {}".format(score, lda_model.print_topic(index, 5)))

```

## OUTPUT

```

C:\Users\Akshit\PycharmProjects\pythonProject2\venv\Scripts\python.exe C:/Users/Akshit/PycharmProjects/pythonProject2/test.py
[nltk_data] Downloading package wordnet to
[nltk_data]   C:\Users\Akshit\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]   C:\Users\Akshit\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
C:\Users\Akshit\PycharmProjects\pythonProject2\venv\lib\site-packages\gensim\similarities\_init_.py:15: UserWarning: The gensim.similarities.levenshtein submodule is disabled, bec
warnings.warn(msg)
[nltk_data] Downloading package punkt to
[nltk_data]   C:\Users\Akshit\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
Top 10 most common words [('machine', 10), ('learning', 9), ('the', 5), ('.', 4), ('of', 3), ('in', 3), ('we', 3), ('text', 2), ('classification', 2), ('performed', 2)]

Excluding the stopwords if any ['text', 'classification', 'machine', 'learning', 'one', 'commonly', 'used', 'nlp', 'tasks', 'article', 'saw', 'machine', 'learning', 'simple', 'exampl

After text-cleaning: [['text'], ['classification'], ['machine'], ['learning'], ['one'], ['commonly'], ['used'], ['nlp'], ['task'], ['article'], ['saw'], ['simple'], ['example'], ['p

Score: 0.9918103218078613   Topic: 0.192*"machine" + 0.192*"nlp" + 0.192*"classification" + 0.192*"learning" + 0.192*"task"

Process finished with exit code 0
|

```

**Code to implement scoring of set of 5 weighted words generated in order to check their relevance for being the topic ([main.py \(github.com\)](#))**

```
1 import nltk
2 import matplotlib.pyplot as plt
3 import numpy as np
4 from collections import Counter
5 nltk.download('wordnet')
6
7
8 from nltk.stem import WordNetLemmatizer
9 from nltk.corpus import stopwords
10 nltk.download('stopwords')
11
12
13
14 import gensim
15 import string
16 from gensim import corpora
17
18
19
20 nltk.download('punkt')
21
22
23 text1 = "Text classification machine learning is one of the most commonly used NLP tasks. In this article, we saw a machine learning simple example of machine learning how text
24 tokens = nltk.word_tokenize(text1)
25 lowercase_tokens = [t.lower() for t in tokens]
26 #print(lowercase_tokens)
27
28
29 bagofwords_1 = Counter(lowercase_tokens)
30 print("Top 10 most common words" , bagofwords_1.most_common(10))
31 return()
```

```
34 alphabets = [t for t in lowercase_tokens if t.isalpha()]
35
36 words = stopwords.words("english")
37 stopwords_removed = [t for t in alphabets if t not in words]
38
39 print("Excluding the stopwords if any", stopwords_removed)
40 print()
41
42
43 lemmatizer = WordNetLemmatizer()
44
45 lem_tokens = [lemmatizer.lemmatize(t) for t in stopwords_removed]
46
47 bag_words = Counter(lem_tokens)
48
49
50 stopwords = set(stopwords.words('english'))
51 exclude = set(string.punctuation)
52 lemma = WordNetLemmatizer()
53
54 def clean(document):
55     stopwordremoval = " ".join([i for i in document.lower().split() if i not in stopwords])
56     punctuationremoval = ''.join(ch for ch in stopwordremoval if ch not in exclude)
57     normalized = " ".join(lemma.lemmatize(word) for word in punctuationremoval.split())
58     return normalized
59
60
61
62
63 final_doc = [clean(document).split() for document in bag_words]
64
```

```
63 final_doc = [clean(document).split() for document in bag_words]
64
65
66 # print("Before text-cleaning:", bag_words)
67
68 print("After text-cleaning:", final_doc)
69 print()
70
71
72 dictionary2 = corpora.Dictionary(final_doc)
73 #print(dictionary2)
74 DT_matrix2 = [dictionary2.doc2bow(doc) for doc in final_doc]
75 #print(DT_matrix2)
76
77 Lda_object_2 = gensim.models.ldamodel.LdaModel
78
79
80
81 from gensim import models
82 num_topics2 = 20
83 lda_model2 = models.LdaModel(DT_matrix2, num_topics=num_topics2,
84                               id2word=dictionary2,
85                               passes=5, alpha=[0.01]*num_topics2,
86                               eta=[0.01]*len(dictionary2.keys()))
87
88
89 bow_vector = dictionary2.doc2bow(lem_tokens)
90 for index, score in sorted(lda_model2[bow_vector], key=lambda tup: -1*tup[1]):
91     print("Score: {} \t Topic: {}".format(score, lda_model2.print_topic(index, 5)))
```

```
main x
[nltk_data] Package stopwords is already up-to-date!
C:\Users\Akshit\PycharmProjects\pythonProject4\venv\lib\site-packages\gensim\similarities\_init_.py:15: UserWarning: The gensim.similarities.levenshtein submodule is disabled,
warnings.warn(msg)
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\Akshit\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
Top 10 most common words [('machine', 10), ('learning', 9), ('the', 5), ('.', 4), ('of', 3), ('in', 3), ('we', 3), ('text', 2), ('classification', 2), ('performed', 2)]

Excluding the stopwords if any ['text', 'classification', 'machine', 'learning', 'one', 'commonly', 'used', 'nlp', 'tasks', 'article', 'saw', 'machine', 'learning', 'simple', 'exa

After text-cleaning: [['text'], ['classification'], ['machine'], ['learning'], ['one'], ['commonly'], ['used'], ['nlp'], ['task'], ['article'], ['saw'], ['simple'], ['example'],

Score: 0.2150402069091797 Topic: 0.439*"machine" + 0.439*"let" + 0.004*"trained" + 0.004*"review" + 0.004*"stored"
Score: 0.17597763240337372 Topic: 0.777*"learning" + 0.008*"loaded" + 0.008*"trained" + 0.008*"stored" + 0.008*"review"
Score: 0.1369139850139618 Topic: 0.191*"trained" + 0.191*"model" + 0.191*"predict" + 0.191*"performed" + 0.191*"sentimental"
Score: 0.11738255620002747 Topic: 0.191*"commonly" + 0.191*"variable" + 0.191*"review" + 0.191*"classification" + 0.191*"sentiment"
Score: 0.05878903344273567 Topic: 0.439*"text" + 0.439*"python" + 0.004*"model" + 0.004*"loaded" + 0.004*"variable"
Score: 0.039257604628801346 Topic: 0.439*"movie" + 0.439*"example" + 0.004*"trained" + 0.004*"stored" + 0.004*"loaded"
Score: 0.039257604628801346 Topic: 0.439*"set" + 0.439*"one" + 0.004*"review" + 0.004*"trained" + 0.004*"stored"
Score: 0.039257604628801346 Topic: 0.439*"article" + 0.439*"test" + 0.004*"trained" + 0.004*"loaded" + 0.004*"review"
Score: 0.039257604628801346 Topic: 0.439*"loaded" + 0.439*"task" + 0.004*"model" + 0.004*"trained" + 0.004*"review"
Score: 0.039257604628801346 Topic: 0.439*"analysis" + 0.439*"used" + 0.004*"model" + 0.004*"stored" + 0.004*"trained"
Score: 0.019726397469639778 Topic: 0.777*"nlp" + 0.008*"loaded" + 0.008*"model" + 0.008*"variable" + 0.008*"trained"
Score: 0.019726397469639778 Topic: 0.777*"saw" + 0.008*"trained" + 0.008*"loaded" + 0.008*"stored" + 0.008*"review"
Score: 0.019726397469639778 Topic: 0.777*"simple" + 0.008*"loaded" + 0.008*"trained" + 0.008*"stored" + 0.008*"review"
Score: 0.019726397469639778 Topic: 0.777*"stored" + 0.008*"trained" + 0.008*"review" + 0.008*"loaded" + 0.008*"let"
Score: 0.01972639560699463 Topic: 0.777*"using" + 0.008*"trained" + 0.008*"model" + 0.008*"stored" + 0.008*"review"

Process finished with exit code 0
```

Output:

# CONCLUSION

We learned a lot about machine learning algorithms and Python fundamentals in this assignment. We attempted to tackle the problem in the most efficient manner possible. Thanks to topic analysis with the help of which we can execute complex tasks more effectively and obtain valuable insights from the available data which will lead to better business decisions..We hope that our solution will help to solve real-world issues.

We have tried our best to optimize the algorithm and have had some success, but there is still a lot of room for improvement. For example, a more streamlined algorithm may assist us in directly reaching possible paragraph titles rather than providing relatable words. While working on the project, we realized that the algorithms we're using may be applied to a variety of situations. For example, in the output, we're receiving words that are closely related to the paragraph, and we're using it to find headings instead of taking into account a larger number of words. Further, we can correctly order the words and put them into a brief summary.



# REFERENCES

## 1. Introduction to LDA

<https://medium.com/analytics-vidhya/topic-modeling-using-lda-and-gibbs-sampling-explained-49d49b3d1045>

## 2. Introduction to NLP

<https://monkeylearn.com/text-classification/>

## 3. Implementation

<https://www.youtube.com/watch?v=T05t-SqKArY>

<https://towardsdatascience.com/how-to-start-an-nlp-project-f76e3f7d0e61>

### Technologies Used:

- [PHP](#)
- [NLP](#)
- [LDA](#)

# Glossary

1. **Bag of words:** List of words obtained after converting paragraphs into words.
2. **Stopwords:** Punctuations, Full stop, Question mark ,Exclamation mark etc. (~,!,@,#,\$,%,^,&,\* )
3. **Cleaning of Document :** Removing stopwords from the bag of words.
4. **Scoring:** Assigning probabilities of relevance.