Extractive Text Summarisation

Dhruv Kaushik - MT18037 Manshi Goel - MT18039
K.Srivatsava - MT18054 P.Akhil Kumar - MT18130

Abstract

Document extracts consist of nearly 20% of the original document and at the same time can be as informative as the full text of the original document. In this project, we propose a solution to deal with this problem using Supervised Learning techniques. Our first goal was to achieve features that uniquely identify the sentences that make their way to the summary. The features can be categorised as surface, content, and relevance features. Surface features are related to the structure and position of a sentence in the document. Content features are related to the amount of content related information being conveyed by the sentence. Relevance features are based on the relevance of each sentence with respect to other sentences of a document. We used the above features and applied Supervised Learning techniques on them, where each sentence of a document is considered as a data point and the label for a corresponding data point describe whether the data point is present in the document summary. The summaries generated are evaluated using ROUGE. We observed that the relevance features improved the overall performance of our summarisation task.

1 Introduction

There are two kinds of summarisation techniques – Extractive and Abstractive. In Abstractive summarisation, a document is first 'understood' and then based on the feature measures of each sentence, significant amount of information from it is paraphrased to resemble Natural Language. Whereas, in Extractive summarisation, the sentences in a document which convey significant information are extracted and summarised.
There are several features related to a sentence in a document which can be used for summarisation. These are categorised as surface, content and relevance features. Surface features are listed as:

Name	Description
Position	1/sentence no.
Doc_First	Whether the sentence is first line of a document
Para_First	Whether the sentence is first line of a paragraph
Length	The no. of content words in a sentence
Quote	$\frac{\text{no. of non-quoted words}}{\text{no. of words in the sentence}}$

Table 1. Surface Features

Content features measure significance of content words in a sentence with respect to all content words in its document. Every unigram as well as bigram in a sentence is taken as a centroid in order to measure its significance by calculating the tf-idf score. Five most frequent content words are selected as topics and their signatures with associated weights are found by using word2vector algorithm on Google news corpus. Signature terms score for a sentence is calculated by adding the weights of signature terms present in the sentence. Frequency score for every unigram in a sentence is calculated by counting its frequency in the sentence divided by its total frequency. Frequency score for every bigram in a sentence is also calculated in the same manner. Content Features are listed as:

Name	Description
CentroidVar_Uni	Average of tf-idf scores of all unigrams in a sentence
CentroidVar_Bi	Average of tf-idf scores of all bigrams in a sentence
SigTerm	The weighted sum of signature terms in a sentence
FreqWord_Uni	Weighted avg. of frequent unigrams in a sentence
FreqWord_Bi	Weighted avg. of frequent bigrams in a sentence

Table 2. Content Features

Relevance Features describe how each sentence is related to other sentences in the document. For a sentence, its relevance score is calculated with respect to the first sentence of the document, with the first sentence of the paragraph it lies in and collectively with all the sentences summed together. Relevance features can be listed as:

Name	Description
FirstRel_Doc	Similarity of each sentence with the first sentence of the document
FirstRel_Para	Similarity of each sentence with the first sentence of a paragraph it belongs to
PageRankRel	Similarity of each sentence with every other sentence in the document

Table 3. Relevance Features

3 Dataset

A collection of 2223 BBC news articles along with their respective summaries belonging to different genres like Entertainment, Sports, Business, Politics, and Technology. Each sentence in an article is considered as a data point.

https://www.kaggle.com/pariza/bbc-news-summary/

4 Methodology

4.1 Data Preparation

1775 of 2223 documents are used as training data. Remaining 448 documents are used as test data. There are 32615 sentences from 1775 documents which form the data points for our learning models. The models are tested on 448 documents which contained 8635 sentences in total. There are 13 features in total and the class label is binary.

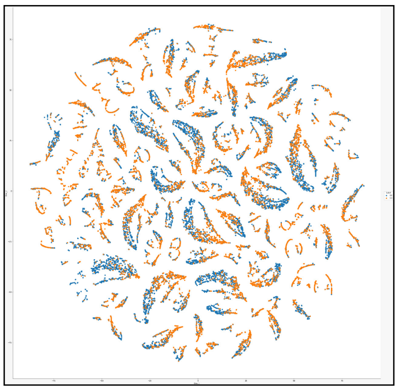


Fig. 1. Training Data Scatter Plot

4.2 Learning Models

4.2.1 Logistic Regression

In Logistic Regression, the cost function is calculated using cross-entropy which is also called as log loss. Mean Square Error cannot be used because if we square the sigmoid function, it will create many local minima. So, we use log loss with two separate cost functions, one for each class label.

$$J(\theta) = \frac{1}{m} \times \sum_{i=1}^{m} Cost(h_\theta(x_i), y_i)$$

$$Cost(h_\theta(x), y) = -log(h_\theta(x)) \quad \text{if y=1}$$

$$Cost(h_\theta(x), y) = -log(1 - h_\theta(x)) \quad \text{if y=0}$$

The reason for applying logarithm is to smooth the functions. The above two equations can be combined to calculate the actual cost function

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} [y_i log(h_\theta(x_i)) + (1 - y_i) log(1 - h_\theta(x_i))]$$

Now, we have to minimise the above cost function.

4.2.2 Gaussian Naive Bayes

Naive Bayes is a conditional probability model. Gaussian Naive Bayes learns the probability distribution of each feature to calculate the objective function.

$$c = argmaxP(c \mid x_1, x_2, x_3, \dots . x_n)$$

By applying Bayesian Rule, we have

$$P(C_k \mid x) = \frac{P(C_k)P(x \mid C_k)}{P(x)}$$

$$posterior = \frac{prior \times likelihood}{evidence}$$

$$P(C_k, x_1, x_2, \dots . x_n) = P(x_1, x_2, \dots . x_n, C_k)$$

$$P(C_k, x_1, \dots . x_n) = P(x_1 \mid x_2, \dots x_n, C_k)P(x_2, \dots . x_n, C_k)$$

Assuming conditional independence between features, we have

$$P(C_k \mid x_1, \dots . x_n) = \frac{1}{Z}P(C_k)\prod_{i=1}^{n} P(x_i \mid C_k)$$

The class which maximises the above equation will be assigned as label to the data point.

$$y = argmaxP(C_k)\prod_{i=1}^{n} P(x_i \mid C_k)$$

4.2.3 K-Nearest Neighbours

This technique uses local neighbourhood to obtain a prediction. The distance function is needed to compare the examples similarity using:

Euclidean Distance $d(x_j, x_k) = \sqrt{\sum_i (x_{j,i} - x_{k,i})^2}$

The hyper parameter in K-NN is 'K'. After deciding upon 'K', the algorithm will find the euclidean distance between the query point and all the training data. Among the smallest 'K' points, the class label is decided based on the majority vote.

5 Results

We first trained a support vector machine on our training data. The observed results are as follows:

Feature Combinations	Accuracy	F1-score
Surface + Content + Relevance	0.587	0.153

Table 4. SVM Results

The results are unsatisfactory for a classifier. We observed the distribution of the data is imbalanced. We tried up-sampling the data using SMOTE algorithm which will generate new synthetic points based on the neighbourhood properties of K-NN. The results are tabulated below:

Feature Combinations	Accuracy	F1-score
Surface + Content + Relevance	0.620	0.004

Table 5. SVM Results after up-sampling

The results are still unsatisfactory.

From the scatter plot plotted in *Fig. 1*, we observed that the data is highly inseparable. So, we decided to go with neighbourhood algorithm like K-NN, and probabilistic classifier model like Naive Bayes. We also implemented Logistic Regression.

The results of Logistic Regression are tabulated below:

Feature Combinations	Accuracy	F1-score
Surface	0.649	0.307
Content	0.710	0.585
Relevance	0.758	0.650
Surface + Content	0.717	0.597
Surface + Relevance	0.763	0.662
Content + Relevance	0.766	0.670
Surface + Content + Relevance	**0.767**	**0.671**

Table 6. Logistic Regression performance for different feature groups

The results are satisfactory despite the data being highly inseparable. The model learnt on our data is plotted.

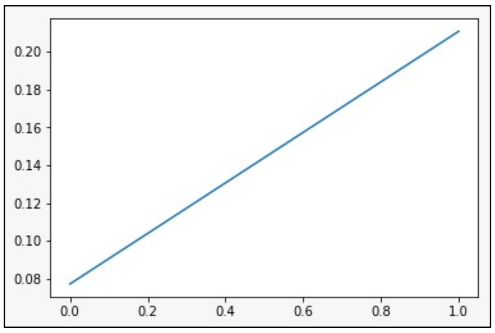


Fig. 2. Logistic Regression Model

From *Fig. 1 & Fig. 2*, we can observe that the data is almost equally distributed on both the sides of the classifier plane.

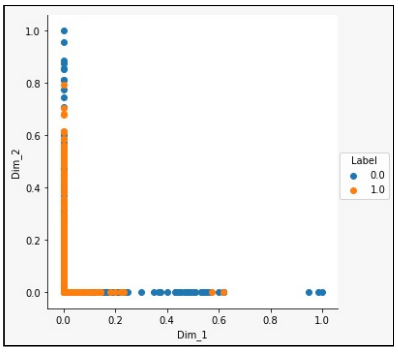


Fig. 3 Logistic Regression - Dimension Altered

But, visualising the data in a different dimension, we found that the data is not equally distributed on both the sides of the plane as shown in *Fig. 3*. Our query point lies on either side of the plane.

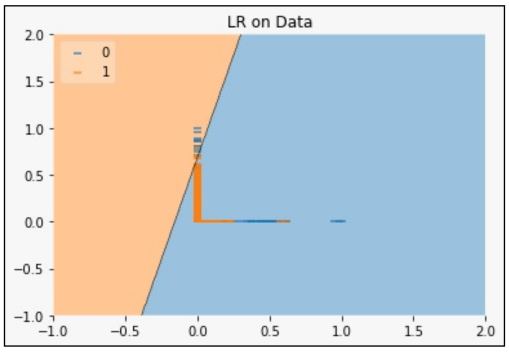


Fig. 4 Decision boundary for Logistic Regression

From *Fig. 4,* we can observe that the blue region is more, which represents the points that are not included in the summary. This is a desired property of our problem *i.e.,* the summary size should be small.

The results for Gaussian Naive Bayes are tabulated below:

Feature Combinations	Accuracy	F1-score
Surface	0.633	0.380
Content	0.695	0.627
Relevance	0.737	0.575
Surface + Content	0.692	0.590
Surface + Relevance	0.698	0.507
Content + Relevance	**0.747**	**0.667**
Surface + Content + Relevance	0.725	0.618

Table 7. Gaussian Naive Bayes performance for different feature groups

The results are better than Logistic Regression. This is because Gaussian Naive Bayes is not a distance based learning technique. It learns the probability distribution of each feature, which works better for our data distribution which is highly inseparable.

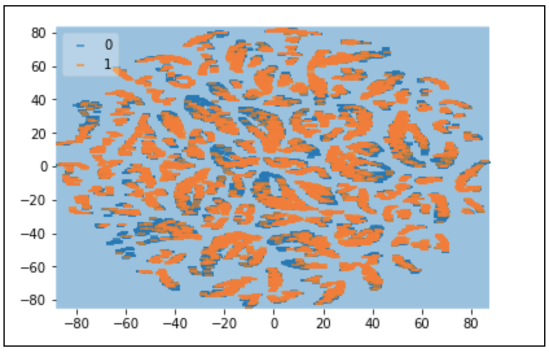


Fig. 5 Decision boundary of Gaussian Naive Bayes

The next model we selected based on our data distribution is a neighbourhood based model which is K-Nearest Neighbourhood.

The results of K-Nearest Neighbourhood are tabulated as:

Feature Combinations	Accuracy	F1-Score
Surface	0.610	0.418
Content	0.659	0.636
Relevance	0.713	0.684
Surface + Content	0.667	0.642
Surface + Relevance	0.725	0.687
Content+Relevance	0.710	0.683
Surface+Content+Relevance	**0.717**	**0.687**

Table 8. K-NN performance for different feature groups

The best results are found in K-NN model. This is because for given query point, it will only observe its neighbourhood before making a decision. So, the property of the data point is preserved. For choosing 'K', we have done 5-Fold cross validation and the results are as below:

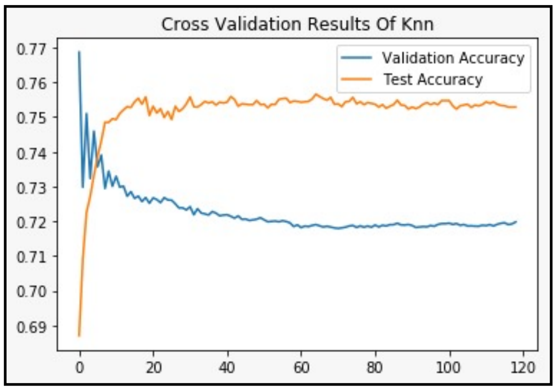


Fig. 6 Graph plotted for tuning hyper parameter of K-NN

From the above results, we came to a conclusion that by setting 'K' value around 60, we can get satisfactory results.

The decision boundary for K-NN is shown in *Fig. 7*

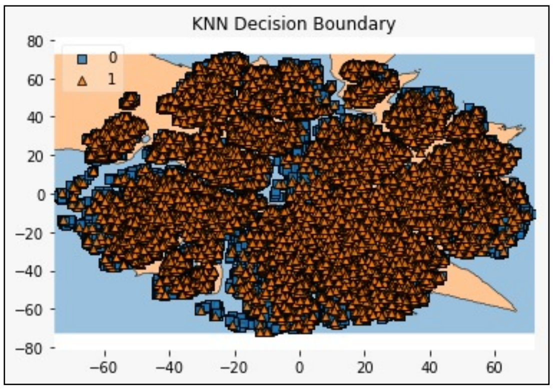
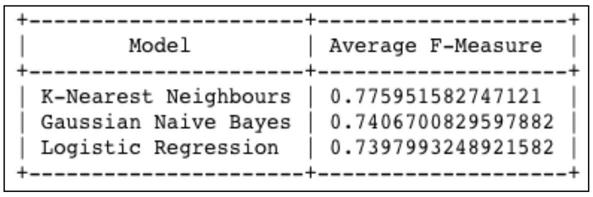


Fig. 7 Decision boundary for K-NN

ROUGE(*Recall-Oriented Understudy for Gisting Evaluation*) is an automatic evaluation package which is used for evaluating summarisation performance based. It compares the machine generated summaries with the reference summaries. F1-Scores are measured to evaluate the classification performance.



```
+----------------------+--------------------+
|        Model         | Average F-Measure  |
+----------------------+--------------------+
| K-Nearest Neighbours | 0.775951582747121  |
| Gaussian Naive Bayes | 0.7406700829597882 |
| Logistic Regression  | 0.7397993248921582 |
+----------------------+--------------------+
```

Table 9. ROGUE Evaluation results for our learning models

For every learning model, we calculate ROUGE F1-Scores for all the generated summaries and take their average as a model's performance measure. The average ROUGE F1-Scores for all learning models are tabulated in *Table 9.*

6 Conclusion

We experimented for different feature combinations and for different training models. We got best performance for K-NN learning model. The feature combination for which our learning models usually give best result is when all the features are considered collectively. The summaries created for relevance features individually have higher accuracy and a better F1-Score than surface or content features taken individually. The ROUGE-L score for machine generated summary for K-NN is 0.77595, which shows that the machine generated summaries are very close to the reference summaries.

8 Future Work

We intend to include more categories of features like Event features. We also plan to find the performance of our model over sensitive datasets like Terms & Conditions and User Agreement documents.

9 References

1. LexPageRank: Prestige in Multi-Document Text Summarisation published by Güneş Erkan and Dragomir R. Radev In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pages 365-371.

2. LexRank: Graph-based Lexical Centrality as salience in Text Summarisation by Güneş Erkan and Dragomir R. Radev published in Journal of Artificial Intelligence Research, Volume 22 Issue 1, July 2004, Pages 457-479

3. The Automated Acquisition of Topic Signatures for Text Summarisation by Chin-Yew Lin and Eduard Hovy published in COLING '00 Proceedings of the 18th conference on Computational linguistics - Volume 1 Pages 495-501.

4. A Compositional Context Sensitive Multi-document Summariser: Exploring the Factors That Influence Summarisation by Ani Nenkova, Kathleen McKeown published in SIGIR '06 Proceedings of the 29th annual international ACM SIGIR conference on Research and Development in Information Retrieval Pages 573-580.

5 A Trainable Document Summariser by Julian Kupiec, Jan Pedersen and Francine Chen published in: SIGIR '95 Proceedings of the 18th annual international ACM SIGIR conference.

6. Extractive Summarisation Using Supervised and Semi-supervised Learning by Kam-Fai Wong, Mingli Wu, Wenjie Li published in COLING '08 Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1 Pages 985-992.

7. The effects and limitations of automated text condensing on reading comprehension performance by A. H. Morris, G. M. Kasper, and D. A. Adams published in Information Systems Research, pages 17–35, March 1992.