Automatic News Headlines Categorization Using Classification

Raghad Bogery1, Nada AlKabour2, Nora AlBabtain3, Yara AlHashim4, Dr. Nida Aslam5

College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Kingdom of Saudi Arabia

[12150007480@iau.edu.sa,](mailto:12150007480@iau.edu.sa,) [22150000081@iau.edu.sa,](mailto:22150000081@iau.edu.sa,) [32150003496@iau.edu.sa,](mailto:32150003496@iau.edu.sa,) [42150002739@iau.edu.sa](mailto:42150002739@iau.edu.sa%20) , [5naslam@iau.edu.sa](mailto:5naslam@iau.edu.sa)

# Abstract

One of the major problems with online news sets is the categorization of the vast number news and articles. In order to solve this problem, a technique called Natural Language Processing is widely used to solve problems related to text classification and clustering. However, based on the discussion in the literature review most of papers in NLP uses huge documents to train the prediction model, thus it is hard to classify a short text without using semantics. Therefore, this paper attempts to use semantics and ensemble learning to improve the short text classification. The proposed methodology starts with preprocessing stage then applying feature engineering using word2vec with TF-IDF vectorizer. Afterwards, the classification model was developed with different classifier KNN, SVM, Naïve Bayes and Gradient boosting. The best classifier out of the used classifiers is Naïve Bayes which gives an accuracy of 90.12% and recall 90.14%.

**Keywords**: NLP, Feature Engineering, Word Embedding, Text Classification, Ensemble Learning.

# Introduction

Due to the focus on mobility and the internet in the recent years and to reduce the paper waste, many news companies went online and changed the traditional way of printing newspapers and articles. Because of that, there’s a huge number of different articles in the news website databases. However, categorizing each old or new news article in its respective category manually is very difficult and time consuming. Hence, this project proposes an automatic news headline categorization program that is based on machine learning techniques.

NLP is mainly used to automatically categorize documents and speech by words count or frequency without considering the meaning behind the words. This method is useful for document or huge chunk of text categorizing. However, news headlines and descriptions are usually short, so such methods might not accurately categorize the articles in their respective category. Because of that, many researchers started researching semantic classification instead of relying on single word meanings to achieve better accurate results.

To achieve a good classification or clustering results, it is important to consider four methodologies that helps in the semantic analysis which are the background knowledge, word representation and feature vectorization technique, topic modeling, the similarity measure that might be used to assess the clustering algorithm. Table 1 summarizes these methodologies with a brief explanation and examples [1] [2].

|  |  |  |
| --- | --- | --- |
| Method | Definition | Examples |
| Background knowledge | Background knowledge can help learning model to better understand the relationships, the context and the meaning of words. | Ontologies, WordNet, semantic networks, treasures, and taxonomies. |
| Topic Modeling | Models the topics of different documents. | Lantent Dirchlet Allocation algorithm (LDA), Lantent Semantic Indexing technique (LSI). |
| Word Representation | The words in NLP are usually represented in the vector space model, each vector represent a word against its occurrence in corpus. Vectors with single words are called Bag of Words (BOW). | BOW, term frequency, Tf-IDF, Word2Vec, GloVe. |
| Similarity Measures | Used to see wither two words have the same or the opposite meanings in the vector space. | Cosine similarity, Euclidean distance, Jacob similarity. |

Table : Methodologies to enhance clustering algorithm

# Review of Related Literatures

This section discusses the literature reviews of the most important studies related to our topic. These topics are feature engineering, such as word embedding and text summarization, clustering using different methods, and finally classification.

## **2.1** Feature Engineering

One of the important keys in NLP is applying feature engineering for the text dataset first to preserve the context of text and second to reduce the vector’s dimensionality of the text. The word embeddings and semantic network like WordNet can help in preserving the context which will be discussed in the following sections, while text summarization can be used to reduce the dimensionality of text. In the following sections, word embedding, and text summarization will be discussed furthermore.

### Word embedding

As the BOW, term frequency, TF-IDF; all represents the words in vector space model for the learning algorithm. However, they don’t preserve the context and the relationships of the words in the documents. Word embedding can solve this problem as it models the semantics and relationships of words in a corpus using a vector with low dimensional as compared to the dimensional size of vocabularies in a corpus [1]. There are two main approaches for words embedding that are known for their efficiency and accuracy and they are Word2Vec and GloVe.

In 2013 Mikolov et. al [2], proposed for word2vec which has two architectures to learn and represent the words in the vector space. The first architecture is the continuous bag of words (CBOW) and second the skip-gram architecture. They performed word analogy to test their model, the word analogy was based on the semantic and syntactic questions that were produced by the authors. The authors observed a higher accuracy for both semantics and syntactic questions when both dimensional size and the number of training vocabulary increased. The methods that were proposed have the advantage of low computation time, but both consider the context locally in a document without making the advantages of the occurrence in different documents.

To overcome the previously mentioned problem, Jeffery et. al [3] proposed GloVe Model which stands for Global Vectors. The word’s vector in the GloVe model is represented by not only considering the word co-occurrence probability in one document, but also considers the ratio co-occurrence probability across the documents. They tested their model on different tasks and they conducted a comparison between CBOW and GloVe model and other baselines. They used the same testing approach as Mikolov et. al. The performance of the model of the task analogy was increasing with the number of dimensions. In their comparison, they showed that the proposed model outperforms both architectures of word2vec in word analogy in semantic and syntactic questions.

### Text summarization

Chi et al. [6] proposed a summarization model named Sentence Selection with Semantic Representation (SSSR). Through learning semantic sentence representation and implementing appropriate selection methods. SSSR also has two main parts which are sentence selection strategy and the sentence representation learning. In the sentence selection strategy is to select a sentence that can rebuild the original document with the minimum falsification. While in the semantic representation of sentences before implementing the selection strategy. Sentences can be represented using two representation the weighted mean of words embedding (SSSR-w) and deep coding (SSSR-d). The word embeddings were based on word2vec model, each word embedding is weighted based on the TF-IDF. Their experiment was conducted on DUC2006 and DUC2007 datasets they used Recall-Oriented Understudy for Gisting Evaluation (ROUGE) to evaluate the text summary results. Both models good results compared to other baselines using the F-measure metric, though SSSR-d has outperformed SSSR-w.

## **2.2 Clustering**

Text clustering is considered as a challenging task in the following sections, three methodologies will be discussed for text clustering, dependency graph clustering, word embeddings clustering, and WordNet and lexical chains clustering.

### Clustering using dependency graph

Asmaa K., et al. [5] proposed a way for reducing the problem that occurs from clustering using the traditional methods and in fact increase the clustering accuracy by using a method called dependency graph. A dependency graph represents one document, where each node is associated with a word and can be used as meta-data for the document. While semantic relations between words can be captured by using edges that are between their corresponding nodes, every edge has term weight based on TF-IDF. Dependency graph will affect the clustering result despite the clustering algorithm that is being used, and to display this K-means clustering algorithm was used to cluster the dataset that was used in this paper. Where the number of clusters was 20 and so the value of K is 20. The number of correctly cluster documents was 188 out of 200 when using the dependency graph, while it decreases to 173 without the dependency graph.

### Clustering using word embeddings

In contrast, Juneja et. al [1], used word embeddings to improve text clustering results. They compared between the word embeddings algorithms that were mentioned in section 1.1. GloVe, CBOW, and skip-gram all of them have high dimensional space, however, GloVe has a higher dimensional space and time complexity. They used GloVe for text clustering since we’re more concerned about accuracy rather than the time complexity. Their proposed methodology used T-SNE algorithm to reduce the dimensionality of the GloVe model, this will help to better understand and visualize the results of their work. In addition, it also helps in reducing the curse of dimensionality where the irrelevant words mask the relevant words. They also saved the words embedding in files to reduce the time complexity of the GloVe model and used k-means as the clustering algorithm. They also saved the words embedding in files to reduce the time complexity of the GloVe model and used k-means as the clustering algorithm. They tested their methodology on two datasets, one dataset showed an increase of error rate with the decrease of the number of dimensions, and it is due to the data loss. While the second dataset doesn’t have a specific number of K and thus the result was acceptable.

### Clustering using WordNet and lexical chains

Using word embedding isn’t the only way to improve the semantic analysis of a corpus. Tingting Wei, et. al [6] used WordNet and lexical chains and a modified Word Sense Disambiguation (WSD) to propose a method that meaningfully cluster texts while reducing the text dimensions. They modified the WSD similarity measure by combining two methods to create a more accurate similarity measure which is Wu–Palmer measure based on the least common subsume (LCS) and Banerjee and Pedersen’s measure based on mutual words in the word’s definitions. After performing WSD using the modified similarity measure, and extracting core semantics using lexical chains, Tingting Wei, et. al performed clustering using Bisecting K-means by assigning K as the number of classes was previously known. They compared it against other methods without using lexical chains by the same clustering method as shown in Table 2.

|  |  |
| --- | --- |
| Method Name | Method Explanation |
| Base | WSD is not performed while performing all basic preprocessing such as removing stop words. |
| Disambiguated Concepts | WSD is performed as well as performing all basic preprocessing. |
| Disambiguated Core Semantic | WSD is performed using lexical analysis as well as performing all basic preprocessing. |

Table : Experiment Results

They have shown that disambiguated core semantic method that uses lexical chains produce the highest F1-measure and purity on three groups. These results prove that the proposed method not only produces purer clusters, but also decreases the computational cost by decreasing the text dimensions using lexical chains.

## **2.3** Classification

Another task in NLP is text classification, in the following section different algorithms were used to for text classification.

### Classification using Different

Vishwanath et al [7] proposed an improved term graph model and conducted a comparison between the KNN, term graph algorithm model and Naïve Bayes. The term graph model was used to preserve the semantics of the words in the datasets by using a weight in a directed graph for frequently co-occurring words. In this model, documents are treated as transactions and uses frequent item set mining algorithms. On the other hand, The KNN uses the vector space model which is based on TF-IDF and similarity measures to see wither one document belong to a certain class or not. The Naïve Bayes assign a probability for terms that belong to a certain class, the document, in the end, is classified to one class by the sum each term probability for a certain class. They trained KNN, term graph model and Naïve Bayes on the dataset then they compared between the three model results. Among the three algorithms, the KNN outperformed both the term graph model and Naïve Bayes; term graph model has higher and closer accuracy to KNN, while Naïve Bayes has the worst accuracy between all the models. The proposed term graph model showed an improved result compared to the other baseline term graph model.

### Classification using WordNet

In Marouance et al. [8] aimed to produce a model to predict suicidal thoughts by collecting data from Twitter using Twittr4J and used Weka as a data mining tool. This paper also implements its own algorithm that calculates the semantic similarity between the collected data depending on a semantic analysis resource using WordNet. They manually constructed a vocabulary related to with suicide and then collected data from Twitter. After that they applied IB1, J48, CART, SMO and Naïve Bayes algorithms to perform the classification. Then they improved their results using semantic analysis based on WordNet. The precision of the algorithms is shown in Table 3 based on the precision of the algorithms that were used.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithm | IB1 | J48 | CART | SMO | Naïve Bayes |
| Precision (tweets with risk of suicide) | 71% | 81.2% | 83.1% | 89.5% | 87.5% |
| Precision (tweets without risk of suicide) | 63% | 75.4% | 66.7% | 70% | 61% |

Table : Results from suicide prediction models

Finally, in word embedding GloVe and word2vec both have their own strength and weakness, GloVe is designed for preserving the context in a large corpus with multiple documents while the word2vec is designed for preserving context in one document with multiple records. SSSR-w and SSSR-d both used in word representation in the SSSR model, while SSSR-d tends to have a better performance over SSSR-w. On the other hand, all the proposed papers from clustering section preserve the context of words by using either a dependency graph, a semantic graph or a word embddings. While in the classification, they used a either TF-IDF or wordNet to represent the words and to preserve its semantics. As most of the traditional classification algorithms doesn’t work with the word embeddings technique, we can conclude with this gap which is how to combine between the word embeddings and the classical classification algorithms while maintain a high accuracy.

# Description of the Proposed Techniques

For the semantic analysis will be specified in detail, showing why this technique should be used and how it fit our project’s objectives. Our methodology comprises of 5 stages which are preprocessing, feature engineering using word2vec, classification using K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Multinomial Naïve Bayes classifiers and finally improving the model using boosting classifier.

## 3.1 Preprocessing Techniques

The preprocessing stage is very important, and the technique should be specified as it is important for the semantic analysis project. As our dataset contains two columns that might help in future news headlines prediction, those two columns were combined and cleaned. As it is important for our model to maintain the semantics, thus returning each word to its root is important ex. play and playing is the same word. Therefore, there was a trade-off between using stemming and lemmatization to improve our model. Stemming is a technique that removes suffix, it is simple and uses less computational time however it might lead to over-stemming errors due to its simplicity. On the other hand, Lemmatization uses the relationship between words and depends on WordNet to return the root of the word. This means that Lemmatization ensures less error but actually take more computational time. Since over-stemming might give wrong results for our classifier, we preferred the lemmatization over the stemming.

## 3.2 Feature Engineering Using Word2Vec

As mentioned previously, Word2Vec is a word embedding technique that uses a shallow neural network to represent the words in the vector space based on their context. There are two approaches to this technique, continuous bag of words (CBOW) and the skip-gram, as explained previously. This feature engineering method was used because it preserves the words semantics while lowering the dimensionality by dropping words that appear less than ﻿min\_count, which is a hypermeter of the Word2Vec model. It also has 2 more important hypermeters, the dimensionality size of the word vector and the maximum distance between the current and predicted word within a sentence. In this study, the words were mapped to its produced vectors into a dictionary which was pipelined with the classification model using a TF-IDF vectorizer. Both Word2Vec approaches were tested with the classifiers.

## 3.3 Classification KNN, SVM and Naïve Bayes classifiers

The following section discusses the different classifiers that were used which are the K-Nearest Neighbor, Support Vector Machine, and Naïve Bayes classifiers.

### K-Nearest Neighbor (KNN) Classifier

K-Nearest Neighbor (KNN) is a supervised learning algorithm that classifies data based on the training data using a similarity measure. It is one of the simplest supervised learning algorithms. This classifier was chosen because, despite its simplicity, it performs well. Moreover, in contrast to eager lerners such as Naïve Bayesian, it is a lazy learner that stores the training data for future predictions and doesn’t generate rules from them, so it doesn’t require prior knowledge. KNN works by searching for the nearest similar K neighbor points in the training data and count their majority voting to predict the unknown class. In other words, it simply matches the unknows class attributes with the training data attributes and looks for the closest match. Because of that, its training time is short since it simply stores the training data. On the other hand, the testing time in KNN is usually far longer than the training time because it needs to compute the K neighbor voting for every test data [[1](#SNe13)].

From the previous description, KNN relies heavily on its training data, any noise can influence the prediction. Furthermore, a huge amount of training data will take time to test. Finally, the K value is also very important since it defines how many neighbors the algorithm consider while classifying. Usually, K is an odd number to avoid evenly split voting as in Figure 1.

-

-

-

-

-

-

-

-

-

-

**+**

+

**+**

+

**+**

+

**+**

+

**+**

+

1-nearest-neighbor outcome is **+**

1-nearest-neighbor outcome is **+**

2-nearest-neighbor outcome is unknows

2-nearest-neighbor outcome is unknows

5-nearest-neighbor outcome is **-**

5-nearest-neighbor outcome is **-**

Figure : KNN Example

As mentioned, KNN computes how similar the neighbor points using a similarity measure (distance measures). This study will use the Euclidian distance measure. The Euclidian is defined in equation (1) [[1](#SNe13)].

(1)

Where X= (x1, x2,…, xi) are the set of attributes for the first data and Y= (y1, y2,..., yi). for the second data. The result of *d(X,Y)* coordinate is plotted and compared to its neighbors.

### Support Vector Machine (SVM) Classifier

The Support Vector Machine classifiers are one of the best machine learning techniques that generate a great result. Support Vector Machine classifiers were first introduced by Corinna Cortes and Vladivir Vapnik in 1995. It is a learning algorithem that works for both classification and regression problems. This classifier was picked because of its high performance even though it has a huge computational time. The basic idea or SVM is that two classes are supported by a function and the goal of this classifier is to find the optimal separating hyperplane that gives the maximum separation margin between the hyperplane and the nearest points of both classes as shown in Figure 2. For the set of training data that are shown in (2) [12]:

(2)

A hyperplane can be found to separates the two class. A hyperplane is shown in equation (3):

(3)

It can be said that the hyperplane is separating the classes efficiently if the distance between the nearest point and the hyperplane is maximum.

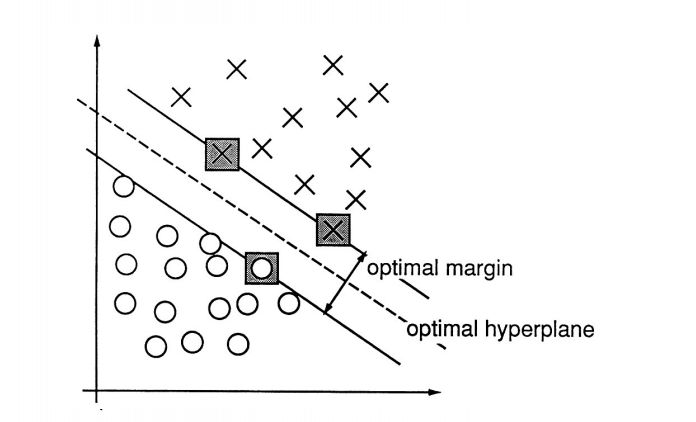


Figure : Example of SVM [12]

There are some parameters that affect the result of the SVM classifier. The first one is the Regularization (C) parameter, lower value of C turns a high error rate that is given for the training set and the hyperplane margin will be large which means a smaller decision function. On the other hand, a higher value of C turns low error rate that is given for the training set and the hyperplane margin will be small. The second parameter is the gamma, it defines how far the effect of a single training point reach. If the gamma is large that means the point that is close will be used for calculation while a small value of gamma means that points that are far will be used for calculation.

### Naïve Bayes Classifier

Multinomial naïve bays classifier or known as term frequency or raw term frequency *tf*, is an approach for characterizing text based on a number of times a term *t* appears in document *d* as shown in equation 4.

(4)

(4)

A multinomial naïve bay is known to be simple to implement, but very efficient since it is based on the assumption that the features are mutually independent, which is why it is one of our choices in the set of classifiers to implement. In practice, usually the term frequency *tf* is normalized by dividing it over the document length (or the sum of the number of terms in the document) as seen in equation 5.

(5)

(5)

Using the term frequency, we can estimate the maximum-likelihood from the training data to find the class-conditional probabilities, where equation 6 shows the calculations needed to find this estimation. Table 4 explains the equation 6 in detail.

(6)

(6)

|  |  |
| --- | --- |
| Symbol | Explanation |
|  | A word from a particular sample in the feature vector |
|  | The total sum of the term frequencies of a specific word from the document in the training sample *d* that belongs to the class . |
|  | Smoothing parameter. |
|  | The total sum of all the term frequencies *N* in the training dataset *d* that belong to the class . |
|  | The vocabulary size that is in the training set. |

Table : Explanation of the Symbols in Equation 6

Then we can use the product of the likelihoods of individual words in the document to give us the class conditional probability of encountering a word, as shown in equation 7.

(7)

(7)

## 3.4 Improving the Model Using Ensemble Learning

Ensemble learning is used to increase the classifier accuracy and reduce the variance and bias, it follows different approaches including bagging, boosting, stacking and voting. The basic idea behind the ensemble learning is using multiple classifiers to improve the model’s prediction. In this project gradient boosting classifier will be used to improve the model.

### Model Improved by Gradient Boosting Classifier

The Gradient boosting classifier use many weak classifiers the number of classifiers indicates how many times the model will be trained. In each training phase the misclassified instances will be given higher weight to reclassify them correctly. This can be an advantage for the imbalanced classes in our dataset and will decrease the number of misclassified classes in each training iteration.

# Empirical Studies

This section gives a brief description of the dataset characteristics. Also, describes the experimental setup that have been done to produce the models using the different selected classifiers K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Multinomial Naïve Bayes classifiers and finally improving the model using boosting classifier. As well as the followed optimization strategy.

## 4.1 Description of dataset

The learning of any model relies on the nature and condition of the data used. This project shall use the [***News Category Dataset***](https://www.kaggle.com/rmisra/news-category-dataset) obtained from Kaggle website. This dataset is about collected news headlines from the year 2012 to 2018 obtained from HuffPost. It contains 202,372 records and 6 attributes, namely category, headline, authors, link, short description, and date. The target is the category of the headlines, containing 41 classes. After cleaning the data and dropping records with empty cells, the dataset contains 200,746 records. This study will use the 3 classes containing the most records, as shown in Table 5. The distribution of the dataset’s classes is shown in Figure 3.

|  |  |
| --- | --- |
| Class | Records |
| ﻿ ﻿travel | ﻿ ﻿9887 |
| ﻿ ﻿style & beauty | ﻿ ﻿9649 |
| ﻿ ﻿parenting | ﻿ ﻿8677 |

Table : Filtered Dataset Classes

A screenshot of a cell phone

Description automatically generated

Figure : Classes Distribution in the Dataset

## 4.2 Experimental Setup

### Dataset Cleaning

To clean this dataset, we started by removing the stop words, emojis and numbers. Next the missing values in the dataset were removed. The words were lemmatized using through stem.wordnet from nltk library. After that, we applied word tokenization.

### Feature Extraction using Word2Vec

After cleaning, tokenizing, and combining the data, the Word2Vecotrization was performed on the combined features using both CBOW and ﻿Skip-Gram Word2Vec methods. Both models were initialized with different measures as shown in Table 6. Based on the similarities between words using the cosine similarity measure, the CBOW3 model was chosen for the classification. For example, Figure 4 shows the cosine similarity between the words ‘health’ and ‘care’. Finally, the CBOW3 model’s words were mapped to its vector as a dictionary to be used for classification.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Model name | Window distance | Minimum word count | Vector dimension |
| CBOW | CBOW1 | 3 | 50 | 100 |
| CBOW2 | 5 | 100 | 70 |
| CBOW3 | 7 | 250 | 50 |
| Skip Gram | SkipGram1 | 3 | 50 | 100 |
| SkipGram2 | 5 | 100 | 70 |
| SkipGram3 | 7 | 250 | 50 |

Table : Word2Vec Models

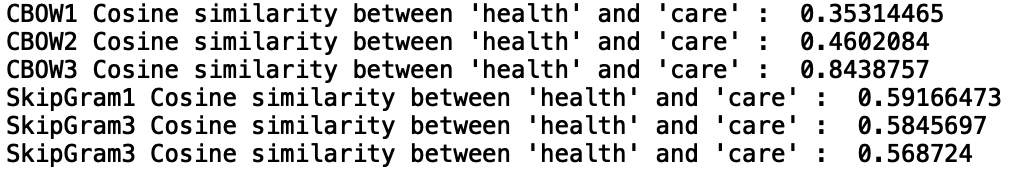


Figure : Cosine Similarity Between the Words Example

### Experimental Setup of K-Nearest Neighbor (KNN) Classifier

The KNN classifier of scikit-learn python package was used to perform this study. Firstly, a pipeline was used to combine the KNN classifier with the ﻿Tf-idf Vectorizer Word2Vec CBOW3 dictionary. Secondly, the dataset was split using the stratified technique to 70% for training data and 30% for testing data. Thirdly, a brute force grid search method of 5-fold was used on the training data to find the best odd K values between [7 - 15], resulting in 5 \* 5 = 25 fits. Fourthly, the folds results were plotted as shown in Figure 5. Lastly, the KNN classifier was validated using the unseen testing data using the best K parameter as found by the grid search previously.

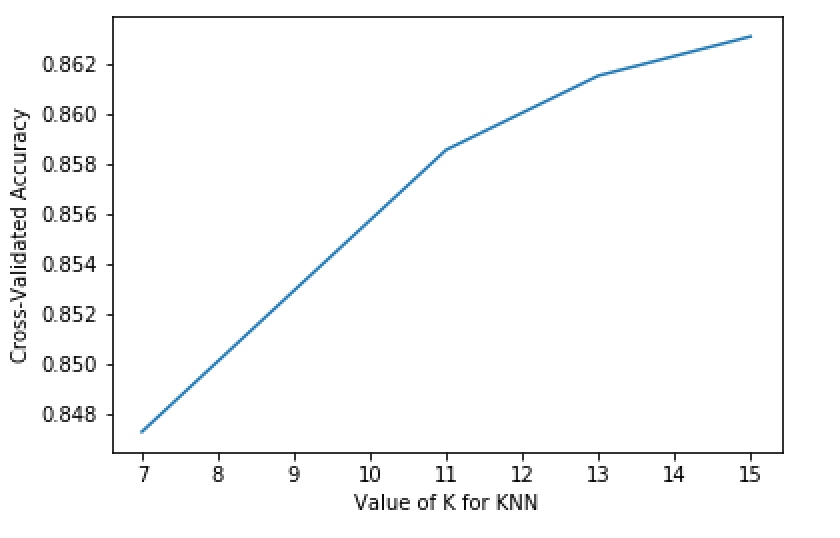


Figure : 5-fold Grid Search Results

### Experimental Setup of Support Vector Machine (SVM) Classifier

For this experiment, a scikit-learn python package was used to get the SVM classifier. The first step is to create a pipeline that combines the SVM classifier with the ﻿Tf-idf Vectorizer Word2Vec CBOW3 dictionary. The second step is to select a number of C and gamma parameters to be tested in the next step. The third step is to perform a 3-fold cross validation grid search on a data that was split to 70% testing a 30% training. Finally, we printed the results of the best fit parameter for the SVM, the recall score, and the confusion matrix.

### Experimental Setup of Naïve Bayes Classifier

The multinomial Naive Bayes Classification can be achieved by using the sklearn library. This library provides many forms of Naïve Bayes such as multinomial, Gaussian, as well as Bernoulli in an easy way by providing predefined functions for each. However, since our main goal here is to preform multinomial Naïve Bayes on our dataset, we will ignore the rest. The way to use it is fairly simple and will be explained in the following steps:

* Make sure that you have already split your data into testing and training
* Make an object from the multinomialNB() class
* Pipeline the object by the following code:
* MultiNB = Pipeline([('vect', TfidfVectorizer()), ('clf', MultinomialNB()) ])
* Then use It to train the data by fitting it to the object
* Define a predicted value to compare with: predicted = MultiNB.predict(X\_test)
* And finally find its accuracy by finding the mean the values where the predicted = the test.

### Experimental Setup of Gradient Boosting Classifier

The Gradient Boosting Classifier was used from sklearn library which have different parameters that will help in tuning the boosting classifier. Those parameters are the base classifier, the number of estimators, the learning rate, the minimum sampling leaf (the number of samples to consider in the leaf) and split (the number of samples to consider when splitting the tree), the maximum depth for the classifying tree and finally the criteria of measuring the error in each iteration. All these parameters will also be tuned through grid search technique to select the best parameters that gives the optimal accuracy for the classifier. The base classifier was set on the default classifier (tree classifier) since the SVM and KNN doesn’t work with classifier as a base classifier. Different parameters were tested to get the obtained results, which are listed in the following:

* Number of estimators: 50,100,200,300,400,500,600.
* Learning rate: 0.25, 0.1, 0.01,0.001.
* Maximum depth: 3,4,5,6,7,8.
* Minimum split sampling: 2,3,4,5,6,7,8.
* Minimum split leaf: 0.1,0.2,0.3,0.4.

Not all these parameters were included in the grid search, instead the best parameter from each was selected iteratively to reduce the computation time. The grid search was finally used with cross validation = 5 with the best selected parameters.

## 4.3 Optimization Strategy

To optimize the results in each classification model, pipeline and grid search were used. In the pipeline, different parameters were selected, the grid search uses this pipeline and create a combination from these parameters to train the model with cross validation value and then select the best results from these parameters.

# Result and Discussion

This section describes the final results produced from our model. Based on the grid search, the best K parameter for the KNN classifier is 15 with 84.57% training accuracy. The testing accuracy of the 15-KNN model is 84.73%. Moreover, the confusion matrix is shown in Table 7. the highest false classification is in the third class ‘travel’ with 289 classified as ‘style & beauty’ and 177 as ‘parenting’. The 15-KNN model recall score is 84.6%.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Actual  Predicted | parenting | style & beauty | ﻿travel | Total Predicted |
| parenting | 2110 | 173 | 271 | 2554 |
| ﻿ ﻿style & beauty | 204 | ﻿2545 | ﻿178 | 2927 |
| ﻿ travel | ﻿289 | 177 | ﻿2517 | ﻿2983 |
| Actual Total | 2602 | 2895 | 2966 |  |

Table : KNN Confusion Matrix

While SVM best parameters are for gamma is 15 with C equals to 0.01 based on the grid search results, which gave a 90% training accuracy and 89.19 % testing accuracy. Additionally, the result of the confusion matrix is shown in Table 8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Actual  Predicted | parenting | style & beauty | ﻿travel | Total Predicted |
| parenting | 2341 | 81 | 202 | 2624 |
| ﻿ ﻿style & beauty | 157 | ﻿2509 | ﻿181 | 2847 |
| ﻿ travel | ﻿144 | 81 | ﻿2768 | ﻿2993 |
| Actual Total | 2642 | 2671 | 3151 |  |

Table : SVM Confusion Matrix

From the grid search the best parameters found was 600 for the number of estimators, 0.25 for learning rate, 7 for the tree maximum depth, 6 for minimum split sampling and 0.1 for minimum split leaf. This classifier increased the accuracy and the recall it also minimized the misclassified classes, see the confusion matrix in Table 9.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Actual  Predicted | parenting | style & beauty | ﻿travel | Total Predicted |
| parenting | 2281 | 158 | 192 | 2631 |
| ﻿ ﻿style & beauty | 135 | 2575 | 132 | 2842 |
| ﻿ travel | 187 | 162 | 2642 | 2991 |
| Actual Total | 2603 | 2895 | 2966 |  |

Table : Boosting Confusion Matrix

Finally, Naïve bays showed a high accuracy with 90.16% in training while the testing got a bit lower with 90.12% while the recall is 90.14%. Table 10 shows each classifier result with recall, and accuracy metric. As seen from the table most of the classifiers have low variance and low bias which indicates a good sign. Also, most of the classifiers has a high recall which matters in the text classification problem. The Gradient Boosting classifier has a better result than the KNN, the SVM outperform the two of them. While, the naïve Bayes gives the best accuracy and recall between them all.

|  |  |  |  |
| --- | --- | --- | --- |
| Classifier | Training Accuracy | Testing Accuracy | Recall |
| KNN | 84.57% | 84.73% | 84.6% |
| SVM | 90% | 89.19% | 89.94% |
| Naïve Bayes | 90.16% | 90.12% | 90.14% |
| Gradient Boosting | 87.66% | 88.58% | 88.55% |

Table : Results Comparison

# Conclusion

In this paper, we produced a text classification model that maintains the semantics of text to gain more accuracy and recall score. The semantics of the text was preserved by using word2vec word embeddings with TF-IDF vectorizer. We conducted a comparison between different classifiers the KNN, SVM, Naïve Bayes and the Gradient Boosting classifiers. The training was done with parameter tuning and optimization to give a better result. The best classifier out of these was Naïve Bayes which has higher accuracy and recall compared to other classifiers. Also, compared to the other reviewed studies with the same classifier our classifier has better accuracy. In the future, a possible enhancement to our work is to apply the classification on more than 3 targets in the dataset and improve the model using methodologies like the neural network.

# Acknowledgement

All thanks for our instructor and reviewer Dr. Nida Aslam for her continuous assistance and support. We appreciate her efforts to inspire us, encourage us and help us in writing this paper and the time she spends to bring it to the lights. The thanks continue to all team members who invest their time to give this paper a value and a meaning.

# References

|  |  |
| --- | --- |
| [1] | N. Maitri, P. Harshadkumar and D. Vipul, "A survey on semantic document clustering," in *Electrical, Computer and Communication Technologies*, India, 2015. |
| [2] | R. Sinoara, J. Antunes and S. J. Rezende, "Text mining and semantics: a systematic mapping study," *Journal of the Brazilian Computer Society,* vol. 23, no. 9, pp. 1-20, 2017. |
| [3] | P. Juneja, D. Tanay, H. Jain, T. Deshmukh, S. Somani and B. Tripathy, "Context Aware Clustering Using Glove and K-Means," *International Journal of Software Engineering and its Applications,* vol. 8, no. 4, pp. 21-38, 2017. |
| [4] | T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *Proceedings of Workshop at ICLR,* 2013. |
| [5] | J. Pennington, R. Socher and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *Conference on Empirical Methods in Natural Language Processing* , Doha, Qatar, 2014. |
| [6] | Chi Zhang,Lei Zhang Chong-jun Wang, Jun-Yuan Xie, "TextSummarizationBasedonSentenceSelectionwithSemanticRepresentatio," in *2014 IEEE 26th International Conference on Tools with Artificial Intelligence*, Nanjing , 2014. |
| [7] | A. Khazaal and S. Kamaruddin, "Graph based text representation for document clustering," *Journal of Theoretical and Applied Information Technology,* vol. 76, no. 1, p. 13, June 2015. |
| [8] | Tingting Wei, Yonghe Lu, Huiyou Chang , Qiang Zhou , Xianyu Bao, "A semantic approach for text clustering using WordNet and lexical chains," *Expert Systems with Applications,* p. 2264–2275, 2014. |
| [9] | V. Bijalwan, P. Kumari, J. Pascual and V. B. Semwal, "Machine learning approach for text and document mining," 2014. |
| [10] | A. B.-H. M. E. Marouane Birjalia, "Machine Learning and Semantic Sentiment Analysis based," in *The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks*, El Jadida, 2017. |
| [11] | E. S.Neelamegam, "Classification algorithm in Data mining: An Overview," *International Journal of P2P Network Trends and Technology (IJPTT) -,* vol. 3, no. 5, October 2013. |
| [12] | Corinna Cortes,Vladivir Vapnik, "Support-Vector Networks," in *Kluwer Academic Publishers*, Boston., 1995. |
| [13] | S. Raschka, "Naive Bayes and Text Classification I - Introduction and Theory," *eprint arXiv,* no. 4, p. 20, October 2014. |