Journal of Engineering Science and Technology
Vol. 16, No. 5 (2021) 3880 - 3906

PRODUCTS DATASET ANALYSIS USING DATA MINING TECHNIQUES

HANAN QASSIM JALEEL[1,*], JANE JALEEL STEPHAN[2], SINAN A. NAJI[2]

[1] Baghdad College of Medical Sciences, Baghdad, Iraq
[2] University of Information Technology and Communications, Baghdad, Iraq
*Corresponding Author: hanan.qassim.galeel@bcms.edu.iq

Abstract

Data mining is the process of getting new information and knowledge from datasets. Data pre-processing encompasses cleaning data, data normalization, feature extraction, feature selection. These processes can be applied to the dataset as data preparation steps. Clustering is the process of gathering a set of objects (or texts) such that the textual words in one group are very similar comparing to the words in other groups. Classification can process a large data variety and it is the most used method in data mining. This paper analyses a product text keywords dataset, by aggregating and clustering the more relevant and associated keyword records according to the TF-IDF and cosine similarity measurements. The resulting clusters are labelled and divided to train and test datasets, and the proposed system uses the Naïve Bayes classifier to classify the tested dataset records to their appropriate classes. The performance of the classifier is measured using measurements like accuracy, recall, and precision, and the proposed technique yields a good accuracy rate.

Keywords: Bernoulli model, Classification, Clustering, Cosine similarity, Dataset pre-processing, Naïve Bayes classifier, Similarity measures.

1. Introduction

Data pre-processing are the processes that makes the handling of data simpler and more straightforward for data mining algorithms [1]. Data quality has a great impact on the models of date mining. It is believed that the data and attributes determine the obtained knowledge upper limit, and data mining algorithms are resembling the upper limit [2, 3]. To prepare the data to be suitable for the input specification of the data mining model, different pre-processing methods have been assumed. These pre-processing methods can enhance the relevancy of predicting the right class target, and simplify the model optimization [2, 3].

Clustering is the most unsupervised mining methodologies that manages creating and discovering groups in a collection of unlabelled data objects [3]. It can be used to divide a set of data objects into various groups such that the objects in one cluster enjoys higher similarity among them and are not similar to data objects in another cluster [4]. Clustering analysis has been widely used in different applications including data mining, machine learning, pattern recognition and the analysis of consumer purchases [5].

Clustering needs accurate determination of the proximity between two objects, either they are similar or not. A various similarity and distance measurements have been applied in clustering algorithms like Cosine and Jaccard similarities [4, 5]. Meantime, similarity can be imagined as distance also. Different distance measurements such as Euclidean distance and relative entropy are implemented to evaluate the distance in clustering techniques [6].

The similarity measures among text or documents are a fundamental issue in text mining. It can be utilized in various applications such as information retrieval, subject detection, clustering and classification of texts and documents. The basic objective of similarity measures is to quantify similarity among a pair of documents or texts [6, 7].

The similarity metrics are mapped either to the scope -1,1 or to 0,1. The 0 states minimum similarity, while 1 illustrates perfect similarity. Selecting a suitable similarity measurement is critical for analysing clusters [7]. A number of similarity measures is needed to discover the dense regions and find the assignment of new text (documents) to the clusters. In order to choose the best measure, it needs to understand the efficiency of these similarity measures [7, 8].

Classification is able to process a wide variation of data in contrast to regression. Classification is the task of discovering a model that differentiates classes and data. It is a type of analysing data in order to discover models that are expressing well the classes of data. Several classification techniques have been invented in different fields such as machine learning, image processing, fraud detection, medical diagnosis, and manufacturing, etc. [7, 9]. Classification is a process of two steps, including a learning phase where the model of classification is built, and a testing or classification phase where the classifier model predicts and estimates the class label for the new data [4, 9].

The learning process is called a training step, in which the classification technique constructs the classifier according to a set of database records [4]. The classifier model can estimate the class label for the tested data in the testing phase. Classification is a supervised learning technique since it needs a set of training data [10]. The classifier constructs the knowledge from the trained data in order to classify a new example in predetermined classes according to this knowledge [11].

There are several classification algorithms such as Decision Tree, Naïve Bayes, Neural Networks, Support Vector Machine (SVM) and Associative Classifier, etc. [9, 10]. The Naïve Bayes classifier is a simple and popular algorithm used in supervised learning technique. It is a fast-learning classifier and can handle any number of attributes and classes [10, 11]. Despite its simplicity, the Naïve Bayes classifier works well in various problems. In addition, Naïve Bayes classifier is powerful and robust against little noise, such that the classifier results are not affected [11, 12]. The Naïve Bayes model is a probabilistic method which depends on facilitating the conditional independence assumption in features. Given in a problem, a training dataset includes the feature values, then the Naïve Bayes classifier estimates the class of the new example based on the previous probabilities of features that appear in that example [12, 13].

The arrangement of the current paper is as follows: Section 2 illustrates the previous related work. Section 3 summarizes the research contributions. Section 4 illustrates the clustering concept and text clustering. Section 5 introduces the similarity measure metrics with discussing the concepts of TF-IDF and cosine similarity. Section 6 provides the text classification with Naïve Bayes classifier model, then it explains Bernoulli Model and the performance measurements of the classifier. Section 7 explains the proposed system design. Section 8 depicts the proposed system results, and finally Section 9 concludes the paper.

2. Related Work

In the work of Popat et al. [14], a hierarchical clustering technique for documents have been presented. They introduced a methodology that uses an additive approach and calculates the group likeness among documents, which produces more improved results comparing to other conventional approaches. Mahesh et al. [15], proposed a method that implements Naïve Bayesian technique to classify research documents depending on their contents and the result of their analysis. They perform a comparative study and implementation of weighted Bayesian classification method with Naïve Bayes classification.

In journal published by An-Anazi et al. [16], the authors introduced various methodologies to aggregate projects documents by utilizing three clustering methods: k-means, k-means fast and k-medoids. The proposed technique uses various similarity measurements such as cosine similarity, Jaccard and Correlation coefficients. The best performance results are obtained by using k-means and k-medoids methods integrated with cosine similarity. Hasan et al. [17], proposed a technique to classify text documents depending on their contents using soft cosine measurement. The proposed classification approach takes the similarity of text features instead of finding their compatibility.

Ogbuabor and Ugwoke [18], analysed different clustering methods using healthcare dataset which can give the optimal clusters. The proposed technique compares the performances of two clustering methods (k-means and DBSCAN) by using silhouette score. The results of experiments signify that the two algorithms have heavy intra-cluster coherence and inter-cluster splitting.

Ahmed et al. [19] presented a classification technique of SMS to determine Spam and Ham messages by using Apriori algorithm and Naïve Bayes classification. The proposed system uses Naïve Bayes method and implements

minimum support and confidence determined by the user. The proposed technique gets a notable improvement with high accuracy score.

Lydia et al. [20], provided a technique to achieve documents clustering by using TF-IDF and Euclidean distance measure. The proposed system aims to accomplish the term weights of the documents by calculating the positive values and Euclidean distance. The proposed technique also gives continuity for Non-Negative Matrix Factorization (NMF) and k-means method of clustering.

3. Research Contribution

The research contributions can be summarized as follow

- The proposed technique suggests a new method for pre-processing and handling unlabelled dataset text records in order to get a labelled dataset record. The pre-processing steps involves tokenization, stop word removal, stemming and filling missing values.
- The proposed system uses a novel clustering technique that uses the concept of K-means and agglomerative hierarchical clustering techniques, that includes three phases of aggregating text records using cosine similarity with TF-IDF method, in order to obtain 23 classes of product records.
- The proposed technique implements Bernoulli model of Naïve Bayes classifier in order to classify the testing dataset records into their classes and calculate the performance measurements of the classifier. The proposed system classifies and predict the testing set records with a good accuracy value of 0.86979.

4. Text Clustering

Clustering is a text mining technique, that collects similar text words or documents in order to formulate a consistent cluster [3], while different text words or documents have been separated to other clusters. Though, the determination of two documents to be similar or dissimilar is not obvious and depends on the settings of the problem [4, 21]. Almost text clustering techniques depend on the vector space model (VSM). Every text record or document d is expressed as a frequencies vector of the remaining words as the following [4]:

$$d = (tf_1 \dots \dots tf_n) \quad (1)$$

Frequently, the vectors of dataset records or documents are standardized to unit length in order to compare documents that have various lengths [9]. The vector space model contains a high dimensionality because it contains thousands of terms, even the pre-processing steps have been applied on text documents [9,14]. Conventional clustering methods may be classified into two primary categories like hierarchical and partitional as follows [3, 4]

4.1. Partitional clustering methods

Partitional methods generate non-overlapped sections of text documents that improve a clustering criterion. Given a number of K clusters, the first and primary portion is built, next the clustering is revised repeatedly by transporting documents among clusters. There are many common partitional techniques such as K-means, Bisecting K-means, K-medoids, etc. [8, 10].

K-means

The popular K-means technique is depended on division and breakdown, most common method in the field of data mining. The foundation of K-means algorithm utilizes the parameter k, splits n data points into k of clusters, in order to get higher cluster similarity, and somewhat lower similarity among clusters, as well as decrease the whole distance among data values within each cluster centroid. The centroid of every cluster represents the mean value [7, 8]. The similarity evaluation is performed by calculating the mean of data points. The Euclidean distance can be used to measure the similarity among data points. The adjacent the distance, the larger the similarity, and vice versa [18].

K-means algorithm dispense all data points to k clusters randomly. Its procedure can be summarized as follows Find the means of every cluster, which represents the cluster centroid; re-dispense and spread the data points to the nearest cluster depending on its distance to the centroid of the cluster; upgrade and modify the cluster mean and evaluate the mean values of points in every cluster; evaluate the criteria function, till it converges [7, 20].

The k-means method criteria function E assumes the square error criteria, defined as follows [9, 20]:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i| \quad (2)$$

where E is the whole square error of all data points in the cluster, P is the data point m_i is the mean of cluster C_i.

The objective of this criteria is to make the produced cluster to be as independent and distinct as possible. The benefits of K-means technique are that it represents one of the most significant partitioning methods and it is most popularly utilized because of its plainness, speed, flexibility, high achievement and simplicity to be understanded and fulfilled. K-means has numerous disadvantages such as it can be applied to numerical data only. It requires to determine k number of clusters beforehand. Its choses random centroids, and finally it is sensible to outliers and sensible data [7, 14, 20].

4.2. Hierarchical clustering techniques

Hierarchical clustering methods generate a clustering hierarchy known as a dendrogram. These methods can be classified as divisive or top down, and agglomerative or bottom-up algorithms.

4.2.1. Divisive clustering

Divisive techniques begin with a single cluster of documents, and at every repetition, divide the most suitable cluster till a stopping criterion like the required number of clusters k is accomplished. In this method, the cluster in every step that has the larger diameter partition, i.e., the cluster that involves the most remote couple of documents. Because of utilizing document similarity rather than distance as closeness measurement, the intended cluster to be partitioned is the cluster that involves the minimal similar couple of documents. In this cluster, the document that has the lower average similarity comparing to others is eliminated to compose a new individual cluster. The technique continues by repeatedly allocating the

documents inside the cluster being partitioned to the new one if they contain larger similarity to the new cluster documents [4, 10, 12].

4.2.2. Agglomerative hierarchical clustering

These algorithms begin with every document in a separated cluster, and at every repetition, combine the similar and analogous clusters till the stopping criteria is reached. They are fundamentally classified to single link, complete link and average link based on the approach that determines the similarity of the inter-cluster [14, 16].

- **Single-Link**: The single-link approach determines the similarity of pair of clusters C_A and C_B to be the similarity of the most both analogous documents [8, 18].
- **Complete-Link**: This approach denotes the similarity of pair of clusters C_A and C_B to be the similarity of both lower similar clusters [8, 18].
- **Average-Link**: This approach determines the similarity of pair of clusters C_A and C_B to be the average of the pair documents similarity from every cluster [8, 18].

5. Similarity Measurements

The notion of similarity measure is the most essential approach in clustering since it decides the efficient similarity calculation among two items. This similarity will affect the cluster representation, since some items may be closer to each other depending on one distance and far away depending on the other [3, 8].

5.1. TF-IDF

TF-IDF represents an abbreviation for (Term Frequency-Inverse Document Frequency).TF-IDF value is a weight that is utilized in text classification, clustering, and retrieval of information. The weight determines how fundamental the word in a document is. The weight raises according to the time numbers a certain word appears in a document [2, 20].

5.1.1. Term frequency (TF)

Term frequency (TF) illustrates the appearance of word that is common in a document [2]. The document length is various, there are probabilities that a word may occur more times in longer documents than in shorter ones. The term or word frequency is depicted in the following equation [2, 22]:

$$TF(t) = \frac{Number\ of\ times\ the\ term\ (t) appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document} \quad (3)$$

5.1.2. Inverse document frequency (IDF)

Inverse Document Frequency (IDF) illustrates the significance of the word. When evaluating the frequency of a term, the terms accept equal significance. Some terms occur more periodically but have little significance [2, 16]. Hence there is a need to evaluate the often periodically terms frequency and recognize the infrequent terms [22]. The inverse document frequency can be evaluated using the following equation:

$$IDF(t) = \log \frac{total\ number\ of\ document}{Number\ of\ documents\ with\ term\ (t) in\ them} \quad (4)$$

5.2. Cosine similarity

The Cosine similarity is a similarity measure among a pair of vectors that use the inner product in order to compute the cosine of angle between them. Cosine similarity helps to discover how relevant the documents are [2, 8]. The cosine similarity among two vectors d_1 and d_2 is expressed as follows:

$$\text{Cos}\,\vartheta = \frac{d1.d1}{||d1||\,||d2||} \tag{5}$$

$$d_1.d_2 = [tf(t_1,d_1) * tf(t_1,d_2)] + [tf(t_2,d_1) * tf(t_2,d_2)] + \cdots + [tf(t_n,d_1) * tf(t_n,d_n)] \tag{6}$$

$$||d_1|| = \sqrt{tf(t_1,d_1)^2 + tf(t_2,d_1)^2 + \cdots + tf(t_n,d_1)^2} \tag{7}$$

$$||d_2|| = \sqrt{ttf(t_1,d_2)^2 + tf(t_2,d_2)^2 + \cdots + tf(t_n,d_2)^2} \tag{8}$$

The values of cosine similarity are ranged between -1 to 1 [3]. The cosine similarity value will be close to 1 if the two documents are similar and their vectors have similar orientation from origin, hence, they create an approximately small angle [8, 22]. The cosine value will be close to -1 if the two vectors have different orientation from origin and they make a wide angle, which means that the two documents ae not similar to each other [7, 22].

6. Text Classification

Text classification has been widely considered in various communities like machine learning, data mining and information retrieval [4]. It is utilized in many applications in different fields [9]. Text classification intends to assign predetermined classes to documents. The classification problem is explained as follows: Consider a training dataset $D = \{d_1, d_2, d_3 \ldots d_n\}$ is a set of documents and every document d_i has a label l; from the set of class labels $L = \{l_1, l_2, l_3 \ldots l_k\}$[23]. The problem is to find a classifier model that can assign the correct class to the new tested document d. Various classification algorithms are summarized as in sub-section 6.

6.1. K-nearest neighbour (KNN)

KNN is a state-based learning technique that depends on similarity or distance function that is applied on couple of observance, such as cosine similarity and Euclidean distance measurements. This algorithm is utilized for various applications as it is efficient, easier to apply and non-parametric. Nevertheless, it has a long classification time, and it is hard to find out the ideal value of k. The better option of k relies on the data. Greater k-values decrease the noise effect on the data classification, but makes the classes borders less independent. A better k value may be chosen by different heuristic methods. To defeat this obstacle, the conventional KNN algorithm can be modified with various values of K for various classes, instead of steady value for the class. This algorithm is called Weighted Adjusted K-Nearest Neighbour (WAKNN) technique that depends on KNN paradigm [4, 6, 13].

6.2. Decision tree

A decision tree is a classification algorithm that is represented as a recursive partitioning of the examples space. The decision tree contains nodes that compose

a tree with a root, which doesn't have arriving edges. The other nodes involve definitely one arriving edge. A node with departing edges represents an arrival node. The remaining nodes are known as leaves, and they represent decision nodes. Every internal node divides the examples space into multiple sub-spaces depending on a particular function of the input feature values. Decision tree classifier is easy to interpret and understand. It is constructed by greedy algorithms, directed by several heuristic that evaluates the 'impurity'. Unrelated features may impact sorely the decision tree construction. Small data diversity may implicate that various looking tree are produced [11, 23].

6.3. Support vector machine (SVM)

SVM is a distinctive classification algorithm that are generally known to be more precise. It is depended on the structural risk minimization foundation from computation theory. The concept is to find out an assumption to ensure the lower rate error. The SVM requires two training datasets positive and negative which are unusual for other classifiers. These two training datasets are required by the SVM to explore and find out the decision surface that better divides and splits up the negative and positive data in the space of n-dimension, which is known as hyperplane. Document representatives that are the nearest and relative to the decision surface are known as the support vector. The SVM classifier performance stays unchanged when the documents which are not belong to the support vectors are eliminated from the training dataset [13, 23].

SVM tries to discover the linear splitting hyperplane that enlarges the margin, which represents the ideal splitting hyperplane, and magnifies the margin among pair of datasets. To evaluate the margin, two hyperplanes are established, each one on a side of the splitting hyperplane. A better splitting hyperplane is fulfilled which has the larger distance to the adjacent data points of the two classes. Whenever the margin becomes larger, the classifier generalization error becomes lower [15, 17].

6.4. Naïve Bayes classifier

Probabilistic machine learning classifiers have obtained a lot of acceptance newly and have performed awfully well. These models make presumptions and inferences regarding how data or words in document are generated, and they suppose a model according to these assumptions. To estimate the probabilistic model parameters, it was used a training sample set [23].

The Naïve Bayes models the documents distribution in every class considering that the different words (terms) distribution is independent from other terms [15]. It is a probabilistic model used in classification depending on the presumption of conditional independence between model attributes [19, 24]. Given a training dataset which includes values of attributes and corresponding class values, the Naïve Bayes classifier can anticipate the class of the test instance depending on the previously remarked feature probabilities that appear in the instance. [25, 26].

Naïve Bayes classifier has two models that are used widely for document classification. These models find the posterior class probability depending on the words (terms) distribution in the documents [26, 27]. These models are Bernoulli and Multinomial Models.

6.4.1. Bernoulli document model

The Bernoulli model may surpass Multinomial model when using small vocabulary size [26]. In the Bernoulli model, if the model has a vocabulary V includes a set of words $|V|$, then the dimension t_{th} of the document feature vector resembles to word w_t. $\boldsymbol{b}_i$ represents the vector of the document i_{th} D_i, and the element t_{th} of $\boldsymbol{b}_i$, which is written as b_{it}, is equal to 0 or 1 and denotes the occurrence or absence of the word w_t in the document i_{th} [28].

Consider $P(w_t|C)$ is the word w_t probability appearing in a text document in class C; the probability of w_t not appearing in the document is given by $(1\text{-}P(w_t|C))$ [27]. According to the assumption of Naïve Bayes, which is the appearance probability of every word is independent from the appearance of other words in the document, then the document probability (or likelihood) $P(D_i|C)$ can be written in terms of likelihoods of individual words $P(w_t|C)$ as follows:

$$P(D_i|C) \sim P(\boldsymbol{b_i}|C) = \prod_{t=1}^{|V|} [\, b_{it} P(C) + (1 - b_{it})\big(1 - P(C)\big)] \tag{9}$$

This product scans and examines every word in the vocabulary [27]. If it is present, then $b_{it} = 1$ and the probability is $P(w_t|C)$; if it is not present, then bit equals 0 and the needed probability is $1 - P(w_t|C)$. This approach is considered a method for producing feature vectors for document in class C [28].

The likelihood values represent the word probabilities in the class of the document $P(w_t|C)$; in addition, the model uses the prior probabilities which are denoted by $P(C)$ [24, 28]. The model can estimate these parameters in the training phase from the training dataset of documents which are labelled with class C. $n_k(w_t)$ will be the documents number of class $C = k$ where w_t is noticed; and N_k will be the documents total number in that class [29]. The model can evaluate the values of word likelihoods as follows:

$$P(w_t|C = k) = \frac{n_k(w_t)}{N_k} \tag{10}$$

The documents frequency of class $C = k$ which includes word w_t. If the training dataset have N documents, then the class $C=k$ prior probability can be evaluated as the document's frequency of class $C=k$ [28, 29].

$$P(C = k) = \frac{N_k}{N} \tag{11}$$

To classify any unlabelled test document denoted by D_j, the Bernoulli model can evaluate the class posterior probability as the following [28]

$$\begin{aligned} P\big(C\big|D_j\big) &= P(C|\boldsymbol{b}j) \\ &\propto P(\boldsymbol{b}j|C)P(C) \\ &\propto P(C)\prod_{t=1}^{|V|}[b_{jt}P(C) + \big(1 - b_{jt}\big)\big(1 - P(W_t|C)\big)] \end{aligned} \tag{12}$$

6.5. Performance measures for classifiers

The performance and accuracy of the classifier can be estimated by calculating the correctly perceived class instances number true positive (TP), the correct number of the perceived instances that are not in the class true negative (TN), the instances that are falsely appointed to the class false positive (FP), and the instances that are

not perceived as class instances false negative (FN). These four values comprise a confusion matrix for binary classifier [10, 30].

A confusion matrix includes information regarding the actual and predicted classes performed by a classifier model [10, 31]. The model performance is estimated by using the confusion matrix values [32]. The matrix for binary classifier can be depicted in Fig. 1.

		Predicted class	
		Negative	Positive
Actual Class	Negative	TN	FP
	Positive	FN	TP

Fig.1. Confusion matrix for binary classifier.

There are several measures that can be identified for the two-class confusion matrix as follows:

Accuracy (Acc): Refers to the rate of predictions total number that are identified correctly [30]. It is calculated using the formula:

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} \tag{13}$$

Recall (True Positive rate TP): Refers to the rate of the positive examples that are correctly classified [31]. It is calculated using the formula:

$$TP = \frac{TP}{FN+TP} \tag{14}$$

True Negative Rate (TN): Refers to the negative examples rate that can be correctly classified [32]. It can be calculated using the formula:

$$TN = \frac{TN}{TN+FP} \tag{15}$$

False Positive Rate (FP): Refers to the negative examples rate thar are falsely identified as positive [10]. It can be evaluated using the formula:

$$FP = \frac{FP}{TN+FP} \tag{16}$$

False Negative Rate (FN): Refers to the positive examples rate that are falsely identified as negative [31], and can be evaluated using the following formula:

$$FN = \frac{FN}{FN+TP} \tag{17}$$

Precision(P): Refers to the predicted positive examples rate that are correctly classified [31], and can be evaluated using the formula:

$$Precision = \frac{TP}{FP+TP} \tag{18}$$

7. Proposed Technique

The proposed system uses an unlabelled text dataset with text records, the system pre-process using several pre-processing steps, then aggregates the similar records with similar keywords into distinct clusters (classes) using the methods of TF-IDF and

cosine similarity measurement. The proposed system then labels the classes with class numbers and assigns class numbers as labels to the records within each class.

Thereafter, the proposed system divides the records in each class into training and testing records, in order to get training set and testing set of the dataset and uses the training set records to train the Naïve Bayes classifier. After the training phase is completed, the classifier can classify the testing set records to their predicted classes with high accuracy rate. Figure 2 illustrates the general block diagram of the system. The overall proposed system work can be illustrated in Algorithm 1.

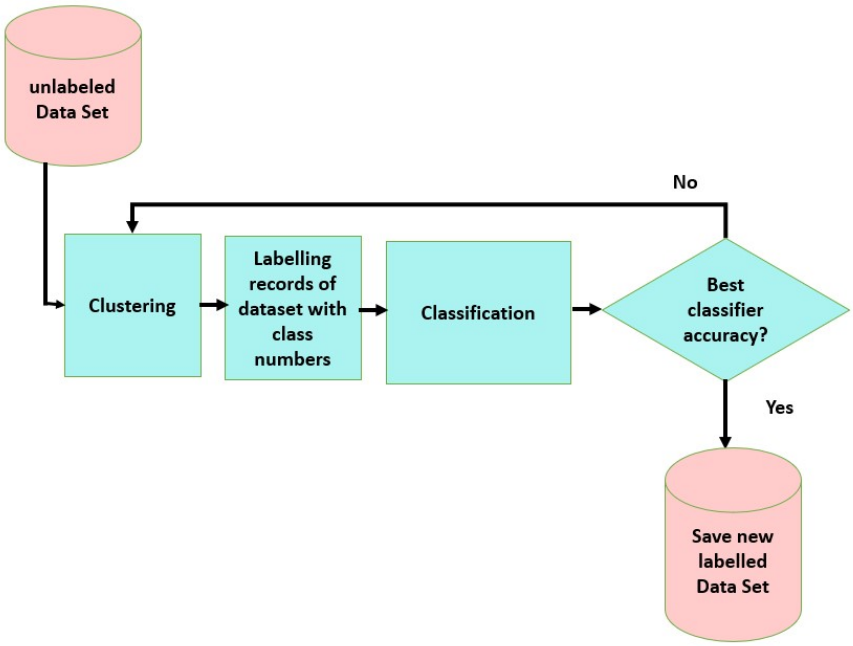

Fig. 2. The general block diagram of the system.

Algorithm 1: The proposed system work.

Input: Unlabelled dataset.
Output: Labelled dataset with classes, the accuracy, precision, recall of the classifier.
Begin
Step 1: Use the (category) attribute from the dataset which has the product keywords.
Step 2: Apply (tokenization) to convert the keywords in each record to tokens.
Step 3: Remove the stop words that appear in the product tokens such as (the, at, a, in, by… etc.).
Step 4: Get the stem of the keywords in each record using Snowball stemmer. The proposed system gets a list of all tokens(keywords) and considers them as features of the dataset.
Step 5: Implement three passes of clustering the dataset records into clusters using cosine similarity. The proposed system gets 23 clusters with similar records.

Step 6: Label each record in every class with its class number, to obtain a new labelled record with class numbers.
Step 7: Divides the records of each class into 70% training dataset records and 30% testing dataset records.
Step 8: Uses the Bernoulli Model of Naïve Bayes Classifier for training by using the training dataset records to train and prepare the classifier model.
Step 9: Classify the records of the testing dataset to their predicted classes using Bernoulli Naïve Bayes.
Step 10: Compare the predictions outcomes of the actual and predicted classes, to calculate the classifier accuracy, precision, and recall.
Step 11: Return the labelled dataset records with classes, as well as the calculated accuracy, precision and Recall of the classifier.
END

7.1. Analysis of the dataset

In this phase, the proposed system analyses the dataset by getting all dataset attributes and selects the attribute (categories) which contains the product keywords that customers have been searched the web using them. Then, the proposed system groups identical records with identical keywords in groups using the SQL command (Group By).

The system utilizes a dataset named GrammarAndProductReviews.csv from the website www.kaggale.com, which contains 65,535 records about products and their brands, manufacturers, as well as customer reviews about the products. The dataset includes the (categories) attribute which contains products keywords. The dataset involves the products manufactures, the city of the user who searches for the product, the URL advertisement of the product, the customer reviews, and many other attributes. The proposed system uses the product keywords (categories) as a selected feature to work on.

The proposed system converts the dataset CSV file format into a table in Microsoft Access database and uses the Access database with SQL statements to manipulate and work on the dataset.

The proposed technique pre-processes un unlabelled dataset with 65,535 records, with products keywords in each record, in order to cluster the similar dataset records with similar product keywords into distinct clusters. Each record in the dataset contains several keywords that the customer searches for them. There are many identical records with identical keywords which mean that the customers may search for the same products and enter the same keywords searching for these products. As a first step, the proposed system aggregates these identical records into groups using the command (Group By) which is an SQL command to group similar identical records with identical product keywords in the dataset. The proposed system gets 557 groups of identical records in the dataset.

7.2. Pre-processing the dataset

In this phase, the proposed system pre-processes the dataset text records using several pre-processing steps. Figure 3 depicts the diagram of pre-processing and clustering the dataset. These steps can be summarized as follows:

- **Tokenization**: The proposed system converts the product keywords in the (category) attribute in each record to tokens such that each keyword represents a single token.
- **Stop word** removal: The proposed system removes the stop words that may occur in the queries of product keywords in each record such as (a, the, in, at, by,… etc.) using the stored stop word table in the database.
- **Stemming**: The stemming process by using Snowball Stemmer is applied on the keywords (or tokens) of each record, in order to return them into their stems or roots, by removing the prefixes and suffixes of each token, such as the token (suppliers) will be returned to the stem (supply), the token (cleaners) and (cleaning) will have the stem (clean) and so on. After pre-processing the dataset records, the proposed system obtains a list of all product keywords after applying the stemming process on them and considers these stemmed keywords as features of the dataset. Figure 3 depicts the diagram of pre-processing and clustering the dataset.

Fig. 3. The diagram of pre-processing and clustering dataset.

7.3. Clustering the dataset into clusters

In order to cluster the records of the dataset into distinct clusters, the proposed system applies 3 passes of calculating cosine similarity measurement on the dataset records. This process can be explained as follows:

7.3.1. First pass of clustering

In the first pass of cosine similarity, the proposed technique calculates TF-IDF which represents the weight for each token (feature) in features list within each group of identical records. Then the system evaluates the cosine similarity among the records of these groups, in order to get the maximum similarity value for each group, and determine which group has this maximum similarity. For example, group 1, which includes 8606 identical records, has a maximum similarity equals

0.37083 with group numbered 207, and this similarity value is the maximum similarity among group 1 and all other groups in the dataset, and so on. In this way, the proposed system gets a list of all maximum cosine similarities among each group and all other groups in the dataset.

The proposed system generates 10 periods (intervals) of similarities, such that the first interval will be ranged from 0.0 to 10.0, the second interval will be from 10.0 to 20.0, until the tenth interval which represents the range from 90.0 to 100.0. The proposed system converts the calculated max similarities to percentage values and assign the max similarities to their appropriate intervals, such as assigning the similarities of 8.097 and 5.034 to interval 1 because the maximum similarities are within the range of 0.0 to 10.0 and assigning the higher similarities of 33.6 and 35.7 to the interval 4 since it has the range bounded between 30.0 and 40.0, and so on.

The proposed system then calculates the number of assigned groups with their similarities in each interval and find the group numbers. For example, interval 2 with the similarity values ranged from 20.0 to 30.0 will have 21 groups, all of these groups of records have higher similarity values which are belong to this range, and the numbers of these groups are as follow: 58, 123, 151, 152, 168, 174. etc. Consequently, the proposed system gets a list of groups that belongs to each interval according to their maximum similarity values. Thereafter, the proposed system calculates the mean of the number of groups in all intervals, and uses the mean as a threshold for comparison, in order to get the primary clusters in the dataset.

The proposed system compares the number of groups in each interval with the threshold (mean), if the number of groups in the interval is less than mean, then the system considers these groups as primary clusters and add them to the list of primary clusters, otherwise, if the number of groups is greater than the mean, then these groups are not assigned to primary clusters. The system proceeds to compare the number of groups with the threshold in each interval until it finds all primary groups and considers them as primary clusters, which are the outcome from the first pass of clustering. The first pass of clustering using cosine similarity can be explained in steps in Algorithm 2.

Algorithm 2: The first pass of clustering.

Input: Dataset.
Output: Primary clusters.
Begin
Step 1: Evaluates TF-IDF (weights) for each token within each group of records.
Step 2: Calculates the cosine similarity among the keywords in the group records to obtain the maximum similarity value for each group, and the group that has the maximum similarity value.
Step 3: Obtains a list of all the maximum cosine similarities among all group's keywords in the dataset.
Step 4: Generates 10 intervals of cosine similarities. The first interval ranged from 0.0 to 10.0, the second interval ranged from 10.0 to 20.0, until the tenth interval bounded from 90.0 to 100.0.
Step 5: Converts the calculated maximum similarities to percentage values and assign the maximum similarities to their suitable intervals.

Step 6: Calculates the number of groups according to their similarities in every interval and find the group numbers. The proposed system gets a list of groups that belongs to each interval according to max similarity values.
Step 7: Calculates the mean value of the number of groups in all intervals. Uses the mean as a threshold for comparison purpose, to obtain the primary clusters.
Step 8: Let interval counter = 1, no_of_intervals=10
Step 9: Get the number of groups in the interval_counter
Step 10: Compares the number of groups in the interval with the threshold, if the number of groups in the interval is less than mean, then these groups will be primary clusters, otherwise, these groups are not as primary clusters.
Step 11: interval_counter = interval_counter + 1
Step 12: If interval_counter <=10 then go to step 8, else go to step 13
Step 13: Returns the primary clusters list.
End

7.3.2. Second pass of clustering

The proposed system implements a second pass of calculating cosine similarity measure values among groups in order to get a reduced list of primary clusters, which represents a list of primary clusters after decreasing them. In the second pass, the proposed technique calculates cosine similarity values among primary clusters in order to get the maximum similarity values for clusters.

First of all, the proposed system calculates the TF-IDF (weights) for each token in all records in the primary clusters. Then, the system evaluates the cosine similarity values among these primary groups (clusters) to get the higher similarity value for each cluster. The system continues to find the maximum similarities for each cluster until it gets a list of all maximum similarities values for clusters.

The proposed system finds the mean of the maximum similarities list of clusters as a threshold value for comparison. Subsequently, the proposed technique compares the similarity value of each cluster with the threshold, if it is less than mean, then the system considers this group (cluster) a new primary cluster, else, the system considers it to not be a new primary cluster. The system repeats the comparison process for all the max similarity values of all primary clusters in order to get a new list of reduced primary clusters. This list of primary clusters after pass 2 is reduced and decreased from 52 primary clusters (in pass 1) to 23 primary clusters (in pass 2). Consequently, the system obtains a list of 23 reduced primary clusters as an outcome of pars 2. The steps of second pass of clustering can be explained in Algorithm 3.

Algorithm 3: The second pass of clustering.

Input: Primary clusters.
Output: Reduced list of primary clusters.
Begin
Step 1: Calculates the TF-IDF weights of each keyword in the records of the primary clusters.
Step 2: Evaluates the cosine similarity values among the keywords of the records of these primary clusters to obtain the higher cosine similarity of each cluster.

Step 3: Calculates the mean of maximum similarity list of clusters. the mean represents the threshold for comparison.
Step 4: Compares every cluster similarity with the mean(threshold), if it is less than mean, then the proposed system adds this cluster to the list of new primary clusters, otherwise, it the cluster is not a new primary cluster.
Step 5: Iterates the comparison task for all the maximum similarity values of all the primary clusters, to obtain a new reduced primary cluster.
Step 6: Returns the new list of reduced primary clusters.
End

7.3.3. Third pass of clustering

In this pass, the proposed technique calculates the cosine similarity among the new reduced primary clusters obtained from pass 2, and all the dataset groups of identical records which are 557 groups. Thereafter, the system assigns the dataset groups to new primary clusters according to their similarity, and it merges the group records to the corresponding similar primary cluster.

Firstly, the proposed system calculates the TF-IDF for all tokens in the dataset groups and primary clusters, then it finds the cosine similarity among dataset groups and primary clusters. Then, the system finds the dataset group which has the maximum similarity value with each cluster. For example, group 87 has similarity value of 0.95 with cluster 3, thereafter, the proposed system assigns group numbered 87 to cluster 3 and merge the records of the group into cluster 3 since the group has greater similarity value with the records of product keywords in cluster 3, and so on.

The proposed system assigns all dataset groups to their similar primary clusters and merges the records of all groups to their corresponding primary clusters. The system gets the final 23 primary clusters with similar records of product keywords, and it labels the clusters or classes with class numbers such as class 1, class 2 … class 23. The proposed technique labels each record in each class with its class number, in order to get labelled records in the dataset classes and stores the new labelled classes with their records in a new dataset. The steps of the third pass of clustering can be mentioned in Algorithm 4.

Algorithm 4: Third pass of clustering.

Input: Reduced primary clusters.
Output: New labelled dataset.
Begin
Step1: Calculates the TF-IDF (weights) for all keywords in the primary clusters and dataset groups.
Step2: Calculates the cosine similarity values of keywords among primary clusters and dataset groups.
Step3: Detect the dataset group that has the max similarity value within each cluster.
Step4: Allocate all dataset groups to their appropriate primary clusters and aggregates them
Step5: Gets the final 23 primary clusters with similar records.
Step6: Labels each record with its class number.
Step7: Returns the new labelled dataset.
End

7.4. Naïve Bayes classification

The proposed technique uses the Bernoulli model of Naïve Bayes classifier in order to train the classifier using the records of training set classes, and to classify any new record in the testing set to the correct class. Figure 4 illustrates the Naïve Bayes classifier diagram.

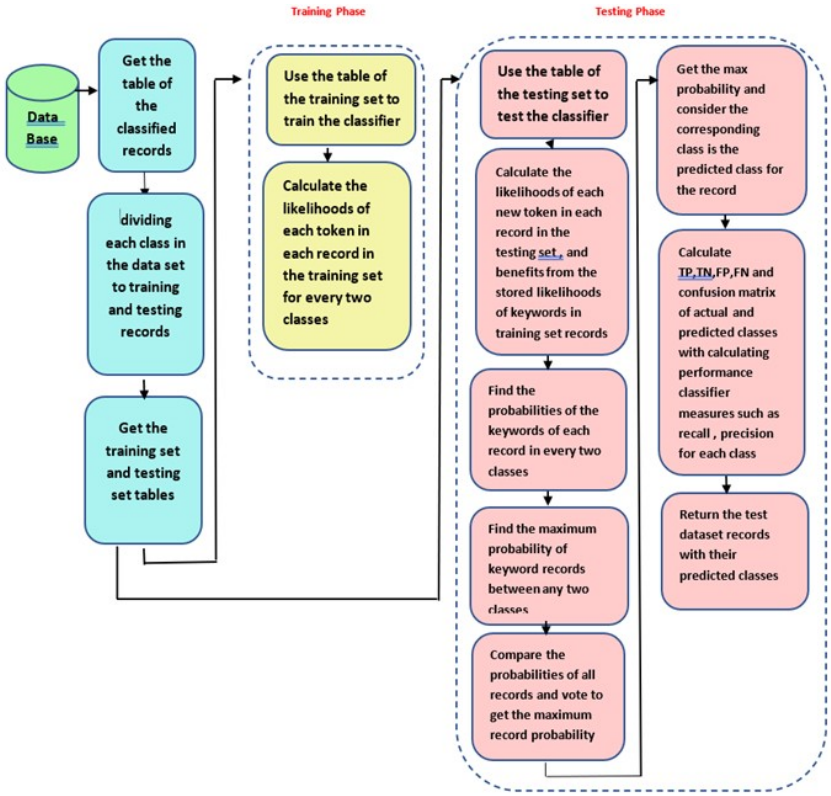

Fig. 4. The Naïve Bayes classifier diagram.

The proposed system uses the table of the new classes with their labelled records and divides the records in each class to training and testing records according to a training percentage value entered by the user. If the percentage equals 70% then the proposed system selects 70% of class records for training and 30% of class records for testing. This process is repeated for each class to select the training and testing records.

The system aggregates the training and testing records for all classes in separated tables, one represents the training set and the other represents the testing set. Then, the system shuffles the order of record in the training and testing sets in order to permute them. Performing the shuffling operation can change the accuracy of the classifier each time the proposed system is executed.

7.5. Training phase of classifier

In the training phase, the proposed system utilizes the table of training set records in order to train the classifier. The proposed technique uses the list of keywords

(features) that represents the stemmed keywords of all records in the dataset and finds the occurrence (or appearance) of each keyword (features) in all the records of every two classes. Since the Naïve Bayes classifier is a binary classifier, which means that it calculates the probabilities between two classes only, then the proposed system takes every two classes (such as class 1 and class 2 for example) and find the occurrence (repetition) of every keyword (feature) in these two classes according to Bernoulli model.

The proposed system finds the likelihoods (or probabilities) of each keyword in the two classes class 1 and class 2 and store the calculated occurrences and likelihoods of keywords in each two classes in a table in the dataset. These calculated likelihoods are stored in separated tables, each table for two classes in the dataset. The steps of training the classifier can be explained in Algorithm 5.

Algorithm 5: Bernoulli model of NB training algorithm.

Input: The training set.
Output: The keywords occurrences and likelihoods for every two classes in the dataset.
Begin
Step1: Finds out the occurrences (frequencies) of each keyword (feature) in all records of every two classes.
Step2: Calculates the probabilities(likelihoods) of each keyword (feature) of the two classes.
Step3: Stores the calculated occurrences and likelihoods of features of every two classes in a separated table.
Step4: Returns the occurrences and likelihoods of keywords in every two classes in the dataset.
End

7.5.1. Testing phase of the classifier

The proposed system uses the testing set records table in order to classify the records of the testing set by the classifier. The proposed system takes every two classes in the dataset (for example, class 1 and class 2), and finds the probabilities of records in these classes. The proposed technique utilizes the stored calculated likelihoods of keywords (features) and finds the probabilities of every record in each two classes in the dataset. The proposed system evaluates the probability of the record by multiplying the likelihoods of all the keywords in the record, then multiplying the result by the prior probability of the class, in order to get the probability of the record in the first class. Thereafter, the system calculates the probability of the same record with its keywords in the second class.

The probabilities of the record within the two classes may be different according to the appearance and absence of the keywords in the class records. For example, the probability of record 17 in class 1 is 0.0381754 and the probability of record 17 in class 2 is 0.8020227. The proposed system compares the two probabilities of record 17 in the two classes and gets the maximum probability value 0.8020227 with its corresponding class, which is class 2.

The proposed system proceeds to find the maximum probabilities of the same record between each two classes in the dataset, it gets the maximum probability values, compares them in order to obtain the greater probability among all 23

classes in the dataset. Subsequently, the proposed system assigns record 17 to the class with the higher probability in the dataset, which is class 7, and classifies record 17 as it belongs to class 7. The proposed system repeats this operation for all records in the testing set in order to classify them into their predicted classes according to the higher-class probability. The steps of testing the classifier can be illustrated in Algorithm 6.

Algorithm 6: Bernoulli model of NB testing algorithm.

Input: Testing dataset & keywords likelihoods.
Output: Labelled testing dataset records & performance measurements
Begin
Step1: record_no=1, maximum_test_records=19657
Step2: Get the next testing record corresponding to record_no
Step3: Find the probability of the record in every two classes in the dataset by using calculated likelihoods of keywords in the tested record of test dataset
Step4: Evaluates the record probability by multiplying the calculated likelihoods of all the keywords by the prior probability of the first class. In the same way, evaluates the probability of the record with all its keywords in the second class.
Step5: Compares the two probabilities of the record in the two classes, to get the maximum (higher) probability of the record with its corresponding class.
Step6: Repeat the operation to find out the max probabilities of the record among every two classes in the dataset, it obtains the max probability values, compare them to get the greater probability among all classes in the dataset.
Step7: Assigns the record to the class with the higher probability in the dataset.
Step 8: Record_no=Record_no + 1
Step 9: If Record_no <= max_test_records, go to step 2, else go to step 8
Step8: Constructs the confusion matrix for 23 classes, calculates the values of TP TN, FP, FN, to calculate the performance classifier measurements: accuracy, precision, and recall.
Step9: Returns the classified testing dataset records into their predicted classes with the performance classifier measurements.
End

8. Results and Discussion

This section explains the results obtained from clustering the dataset into clusters or classes with their three passes of cosine similarity measures, as well as the performance results of Naïve Bayer classifier.

8.1. Clustering dataset results

The clustering results can be depicted in the tables that represent the three passes of cosine similarity measurements among dataset groups and clusters. For the first pass of cosine similarity measurements, Table 1 depicts a data sample of the calculated TF-IDF (weights) for each keyword in all 557 groups in the dataset. This table presents the TF-IDF values for each token(feature) from the dataset feature list within every group that have similar records. The dataset has 557 groups where

the TF-IDF values should be computed for every keyword in them. The table illustrates a sample of the TF-IDF results for five keywords w1, w2, w3, w4 and w5 in 10 groups. The zeros in the table indicates that the TF-IDF values for these words in these groups in the sample are zeros, which means that the keywords (dataset features) are not exist in this group.

Table 1. The calculated TF-IDF (Weights) for each keyword in all groups in the dataset (First Pass of Clustering).

Group No.	TF-IDF[w1]	TF-IDF[w2]	TF-IDF[w3]	TF-IDF[w4]	TF-IDF[w5]
1	0.36947	0.33630	0.16756	0.09944	0.3642
2	0	0	0	0	0
3	0.24939	0	0.22621	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0.55420	0.11210	0.16756	0	0.4856

Table 2 illustrates a sample of the evaluated cosine similarity values among 557 groups and the maximum similarity values, with the group numbers that has the maximum similarity values. This table illustrates a data sample of the calculated cosine similarity among the groups 1,2,3,4 and the groups 2,3,4,5 of the dataset. For example, the textual group 1 has a cosine similarity value of its keywords equal to 0.01 with the keywords of group 2, and it has a cosine similarity value equals 0.009 of its keywords with the keywords of group 3, and so on. The table illustrates the group that has a maximum cosine similarity with other groups. For example, group 1 has a maximum similarity equals 0.37083 with group numbered 207, and this similarity value is the maximum similarity among group 1 and all other groups in the dataset, and so on. Hence, the proposed system gets a list of all maximum cosine similarities among each group and all other groups in the dataset.

Table 2. The evaluated Cosine similarity values among groups and the maximum similarity values (First Pass of Clustering).

Group No.	Cosine similarity among groups (sample of data) Group No.				Max similarity	Group location
	2	**3**	**4**	**5**		
1	0.01	0.009	0.084	0.0	0.37083	207
2	0.009	0.01	0.011	0.254	0.31617	247
3	0.084	0.011	0.01	0.0	0.50715	343
4	0.0	0.254	0.0	0.353	0.43946	52

Table 3 explains the 10 generated intervals of similarity values and the number of groups that are belongs to each interval according to their similarity values. This table depicts the 10 intervals and the group numbers that lies in each interval according to the cosine similarity values, with the number of groups in each interval. The proposed system generates 10 periods (intervals) of cosine similarities such that the first interval will be ranged from 0.0 to 10.0, the second interval will be from 10.0 to 20.0, until the tenth interval expresses the range from 90.0 to 100.0.

The proposed system transforms the calculated max similarities to percentage values and assign the max similarities to their suitable intervals. For instance, assigning the similarities of 8.097 and 5.034 to interval 1 because the maximum similarities are within the range of 0.0 to 10.0, and assigning the similarities of 33.6 and 35.7 to the interval 4 since it has the range bounded between 30.0 and 40.0 and so on. The proposed system then calculates the number of assigned groups with their similarities in each interval and finds the group numbers. The table presents the number of groups in each interval according to their cosine similarity as well as the number of groups in each interval. For example, interval 2 with the similarity values ranged from 20.0 to 30.0 will have 21 groups, all of these groups of records have higher similarity values which are belong to this range, and the numbers of these groups are as follow: 56, 123, 151, 152, 168, 100, etc. Consequently, the proposed system gets a list of groups that belongs to each interval according to their maximum similarity values. Thereafter, the proposed system calculates the mean value of the number of groups in all intervals, and uses the mean as a threshold for comparison, in order to get the primary clusters in the dataset.

Table 3. The 10 generated intervals of similarity values with the groups number in each Interval (First Pass of Clustering).

Interval	From	To	Group numbers	Groups count
1	0	10	369	1
2	10	20	56,123, 151,152,168,100	21
3	20	30	11,13,27,36,57,58,65,75,78,88,97,103, 109,111,112,130,148	65
4	30	40	1,2,5,6,7,8,9,10,14,17,18,19,21,26,32,3 3,37,43,44,48,50,52	146
5	40	50	4,12,16,20,22,23,24,25,29,35,38,39,45, 47,49,51,53,54,59,6	149
6	50	60	3,28,34,41,55,60,72,77,82,90,93,95,96, 108,115,117,118,11	86
7	60	70	15,30,40,42,46,61,64,68,71,79,91,99,1 41,153,159,176,197	57
8	70	80	31,62,73,104,121,122,134,169195,220, 238,244,256,272,283	25
9	80	90	139,290,325,500,507	5
10	90	100	0	0

Table 4 displays the primary clusters that have group numbers less than the threshold (mean). The table illustrates a data sample of the primary clusters with their keywords. For example, the primary clusters in this sample are: 370, 57, 124 and 152, which are presented with their keywords. The proposed system compares the number of groups in each interval with the threshold (mean), if the number of groups in the interval is less than mean, then the system considers these groups as primary clusters, and add them to the list of primary clusters, otherwise, these groups are not assigned to primary clusters. The system proceeds to compare the number of groups with the threshold in each interval until it finds all primary groups and considers them as primary clusters.

Table 4. The primary clusters with group numbers less than the threshold (First Pass of Clustering).

Group No.	Group Keywords
370	Food, Packaged Foods, Dairy & Dairy Substitutes, Desserts, Ice Cream Bars & Novelties, Food & Beverage, Frozen Foods, Ice Cream, Ice Cream & Novelties, Grocery
57	Personal Care, Deodorants & Antiperspirants, Deodorants & Antiperspirant, Health & Beauty
124	Movies, Music & Books, Movies, Sci-Fi & Fantasy, Movies & TV Shows, Insta watch Movies By VUDU, Insta watch Movies, Movies & TV, Shop Insta watch, Kids & Family, Insta watch, Extended Editions
152	Personal Care, Skin Care, Face Care, Mask, Beauty, Facial Masks, Facial Treatments, Natural Beauty, Natural Skin Care, Featured Brands, Health & Beauty, Shea Moisture, Gift Finder, Gifts for Her, Beauty Gifts, Bath & Body, Body Wash & Cleansers, Face, Masks, Face Masks, Mask
153	Personal Care, Makeup, Lipstick, Lip Gloss, & Lip Balm, Lip Gloss, Beauty, Lips, Beauty & Personal Care, Skin Care, Lip Care, Lip Balms & Treatments, Health, Lip Glosses

8.2. Naïve Bayes classifier results

Table 5 illustrates a data sample of the training phase of Bernoulli model of Naïve Bayes classifier. It illuminates the stemmed keywords (features) from dataset records, and it takes each two classes (like class 0 and class 1), finds the occurrence of the keywords in the records of each class, and evaluates the likelihoods of the keywords in the two classes. The table takes a sample of the two classes 0 and 1, with the list of stemmed features Wi, such as Food, Package, Dairy, Dessert, Ice etc. and calculates their frequencies X(W) and Y(W), where X and Y represents classes 0 and 1, respectively. The table also presents the likelihoods in the two classes 0 and 1. For example, P(W|X) represents the likelihoods of the words W in class X, while P(W|Y) represents the likelihoods of the words in class Y. Because the Naïve Bayes classifier is a binary classifier, which means that it calculates the probabilities between two classes only, then the proposed system takes every two classes (such as class 0 and class 1 for instance) and find the occurrence (frequency) and likelihood of every keyword (feature) in these two classes according to Bernoulli model. Notice that zeros in the table indicates that the keywords are not present in the class, and consequently, their likelihoods also will be zeros. For example, the frequency X(W) of the keyword W=Food in class X=0 is equal to 205, its likelihood in class X=0 equals to 0.0192. On the other hand, the frequency Y(W) of the keyword W=Food in class Y=1 and its likelihood is equal to 0, which indicates that the keywords are not present in the class 1, and consequently, their frequencies and likelihoods will be zeros.

Table 5. The training phase results of Bernoulli model of Naïve Bayes Classifier.

Id	Class[x,y]	Words vector	X(W)	P(W|X)	Y(W)	P(W|Y)
0	[0, 1]	FOOD	205	0.0192	0	0
1	[0, 1]	PACKAG	119	0.01115	0	0
2	[0, 1]	DAIRI	11	0.00103	0	0
3	[0, 1]	DESSERT	16	0.0014	0	0
4	[0, 1]	ICE	10	9.37E-04	0	0

Table 6 depicts the performance measurements of accuracy, precision and recall of each class from the confusion matrix. In addition, the table depicts the accuracy, recall and precision of the Naïve Bayes classifier as a whole. The confusion matrix can be built separately for each class to find the values of TP (true positive), FP (false positive), TN (true Negative) and FN (false negative) and calculates the performance measures of the classifier. The table presents the class number, with its associated TP, TN, FP, FN, precision, recall and accuracy for each class. For example, in class no. 1, the TP=38, TN=3702, FP=7885, FN=151, accuracy=0.31759, precision=0.2010582 and recall=0.00479

Table 6. The performance measurements of accuracy, precision and recall of each class.

Class No.	TP	FP	FN	TN	Accuracy	Precision	Recall
1	38	7885	151	3702	0.31759	0.2010582	0.00479
2	852	0	3720	2888	0.50134	0.18635	1
3	14	0	91	3726	0.97624	0.13333	1
4	8	0	59	3732	0.98446	0.11940	1
5	7	0	16	3733	0.99574	0.30434	1
6	5	0	13	3735	0.99653	0.27777	1
7	32	0	154	3708	0.96045	0.17204	1
8	80	0	290	3660	0.92803	0.21621	1
9	28	0	86	3712	0.97752	0.24561	1
10	9	0	48	3731	0.98732	0.15789	1
11	15	0	56	3725	0.98524	0.21126	1
12	1124	0	4824	2616	0.43671	0.188971	1
13	43	0	185	3697	0.95286	0.18859	1
14	151	0	619	3589	0.85799	0.19610	1
15	11	0	68	3729	0.98214	0.13924	1
16	13	0	70	3727	0.98162	0.15662	1
17	62	0	247	3678	0.93804	0.20064	1
18	3	8020	0	3737	0.31802	1	3.74E-04
19	343	0	1347	3397	0.73520	0.20295	1
20	7	0	15	3733	0.99600	0.31818	1
21	850	0	3639	2890	0.50684	0.18935	1
22	45	0	200	3695	0.94923858	0.18367347	1
23	0	0	7	3740	0	0	0
Av.	0	0	0	0	0.86979	0.2256373	0.7941405

First of all, the proposed system evaluates the confusion matrix with calculating the accuracy, precision and recall values of each class, then the proposed system calculates these measurements for the whole classifier. The proposed technique gets a precision value of 0.2256373, recall value of 0.7941405 and accuracy of 0.86979 for the Naïve Bayer classifier, which is considered as a high accuracy rate.

9. Conclusion

This paper presents a new proposed technique to cluster an unlabelled text dataset into clusters or classes and classifies new records to the appropriate classes using the Bernoulli model of Naïve Bayes classifier.

- The major contribution of the proposed technique is pre-processing, clustering, and labelling an unlabelled dataset to get a new labelled dataset of classes of textual records.

- The proposed system implements a proposed novel clustering that uses the concept of K-Means algorithm by finding primary clusters as initial central clusters, then aggregates the groups to the primary clusters after measuring the cosine similarity values among the dataset groups and primary clusters. The proposed system implements three passes of calculating cosine similarity measurements among dataset groups in order to find primary clusters that include similar records with high similarity values.
- Moreover, the proposed system uses the concept of agglomerative hierarchical clustering which aggregates the dataset groups to their closer nearest groups after calculating the cosine similarity among them, in order to combine them to get more larger clusters. The system gets 23 primary clusters after combining and aggregating the similar groups to them according to their similarity values.
- The proposed system uses the Bernoulli model of Naïve Bayes classification for training using the training set records, in order to classify the testing, set records to their predicted classes. Bernoulli model of Naïve Bayes classifier is one of the straightforward, efficient and effective probabilistic methods used in text classification
- To evaluate the performance of the classifier, the proposed technique evaluates the TP, TN, FP, FN, and confusion matrix for each class, in order to find the precision, recall, and accuracy for classes. In addition, the proposed method evaluates these measurements for the Bernoulli NB classifier as a whole system and gets a classifier accuracy of 0. 86979 which is a high accuracy score.

Nomenclatures

$(1\text{-}P(w_t\|C))$	Probability of w_t not appearing in the document
$\boldsymbol{b}_i$	Represents the vector of the document i th D_i
b_{it}	Occurrence or absence of the word w_t in the document i th
C	Class
d	Document
D	Dataset
d_i	Document vector i
L	Set of class labels
N	Number of documents in the training set
N_k	The documents total number in the class
$n_k(w_t)$	The documents number of class $C=k$ where w_t is noticed
$P(C)$	The prior probability of class C
$P(C\|D_j)$	Class C posterior probability of unlabelled document D_i
$P(D_i\|C)$	The document probability (or likelihood)
$P(w_t\|C)$	Word w_t probability appearing in a text document in class C
$P(w_t\|C=k)$	The probability of word w_t in class k
t	Term in the document
t *th*	Dimension t of the document feature vector

tf	Term frequency of a word
V	Vocabulary of words
w_t	Word

Greek Symbols

ϑ	Angle between two document vectors

Abbreviations

Acc	Accuracy
AV	Average
CSV	Comma separated values
FN	False negative
FP	False positive
IDF	Inverse document frequency
NMF	Non-negative matrix factorization
P	Precision
SQL	Structured query language
SVM	Support vector machine
TF	Term frequency
TN	True negative
TP	True positive
URL	Uniform resource locator
VSM	Vector space model

References

1. Canchen, L. (2019). Preprocessing methods and pipelines of data mining: An overview. *arXiv Archive: 1906.08510v1*, 1-7.
2. Sailaja, D.; Kishore, M.V.; Jyothi, B.; and Prasad, N.R.G.K. (2015). An overview of pre-processing text clustering methods. *International Journal of Computer Science and Information Technologies*, 6 (3), 3119-3124.
3. Sitikhu, P.; Pahi, K.; Thapa, P.; and Shakya, S. (2019). A comparison of semantic similarity methods for maximum human interpretability, *arXiv Archive: 1910.09129v2*, 1-4.
4. Allahyari, M.; Pouriyeh, S.; Assefi, M.; Safaei, S.; Tripple, E.D.; Gutierrez, J.B.; and Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv Archive: 1707.02919v2*, 1-13.
5. Sarti, S.; Darnall, N.; and Testa, F. (2018). Market segmentation of consumers based on their actual sustainability and health-related purchases. *Journal of Cleaner Production*, 192, 270-280
6. Etaiwi, W.; and Naymat, G. (2017). The impact of applying different preprocessing steps on review spam detection. *The Proceeding of the 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks*, Lund, Sweden.
7. Thangarasu, M.; and Inbarani, H.H. (2015). Analysis of k-means with multi view point similarity and cosine similarity measures for clustering the document. *International Journal of Applied Engineering Research*, 10(9), 6672-6675.

8. Afzali, M.; and Kumar, S. (2017). Comparative analysis of various similarity measures for finding similarity of two documents. *International Journal of Database Theory and Application*, 10(2), 23-30.
9. Alapati, Y.K.; and Sindhu, K. (2016). Combining clustering with classification: A technique to improve classification accuracy. *International Journal of Computer Science Engineering*, 5(6), 336-338.
10. Srivastava, S. (2014). Weka: A tool for data preprocessing, classification, ensemble, clustering and association rule mining. *International Journal of Computer Applications*, 88(10), 26-29.
11. Rajeswari, R.P.; Juliet, K.; and Aradhana. (2017). Text classification for student data set using Naive Bayes classifier and KNN classifier. *International Journal of Computer Trends and Technology*, 43(1), 8-12.
12. Lydia, E.L.; Kumar, K.V.; Reddy, P.A.; and Ramya, D. (2018). Text mining with Hadoop: Document clustering with TF_IDF and measuring distance using Euclidean. *Journal of Advances Research in Dynamical and Control Systems*, 10(14-Special Issue), 1784-1792.
13. Korde, V.; and Mahender, C.N. (2012). Text classification and classifiers: A Survey. *International Journal of Artificial Intelligence and Applications*, 3(2), 85-99.
14. Popat, S.K.; Deshmukh, P.B.; and. Metre, V.A. (2017). Hierarchical document clustering based on cosine similarity measure. *Proceeding of the First International Conference on Intelligent Systems and Information Management*. Aurangabad, India.
15. Mashesh, K.M.; Saroja, D.S.; Desai, P.G.; and Chiplunkar, N. (2015). Text mining approach to classify technical research documents using Naïve Bayes. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(7), 386-391.
16. Al-Anazi, S.; AlMahmoud, H.; and Al-Turaiki, I. (2016). Finding similar documents using different clustering techniques. *Proceedia Science Computer*, 82(2016), 28-34.
17. Hasan, M.Z.; Hossain, S.; Rizvee, M.A.; and Rana, M.S. (2019). Content based document classification using soft cosine measure. *International Journal of Advanced Computer Science and Applications*, 10(4), 522-528.
18. Ogbuabor, G.; and Ugwoke, F.N. (2018). Clustering algorithm for a healthcare dataset using silhouette score value. *International Journal of Computer Science and Information Technology*, 10(2), 27-37.
19. Ahmed, I.; Guan, D.; and Chung, T.C. (2014). SMS classification based on Naïve Bayes Classifier and Apriori Algorithm frequent itemset. *International Journal of Machine Learning and Computing*, 4(2), 183-187.
20. Lydia, E.L.; Govindaswamy, P.; Lakshmanaprabu, SK.; and Ramya, D. (2018). Document clustering based on text mining k-means algorithm using Euclidean distance similarity. *Journal of Advances Research in Dynamical and Control Systems*, 10(02-Special Issue), 208-214.
21. Alasadi, S.A.; and Bhaya, W.S. (2017). Review of data preprocessing techniques in data mining. *Journal of Engineering and Applied Sciences* 12(16), 4102-4107.
22. Bisandu, D.B.; Prasad, R.; and Liman, M.M. (2019). Data clustering using efficient similarity measures. *Journal Statistic and Management Systems*, 22(5), 901-922.

23. Kowsari, K.; Meimandi, K.J.; Heidarysafa, M.; Mendu, S.; Barnes, L.; and Brown, D. (2019). Text classification algorithms: A survey. *Information*, 10(4), 1-68.
24. Xu, S. (2018). Bayesian Naïve Bayes classifiers to text classification. *Journal of Information Science*, 44(1), 48-59.
25. Xu, S.; Li, Y.; and Wan, Z. (2017). Bayesian multinomial Naïve Bayes classifier to text classification. *Proceeding of the First International Conference on Future Information Technology*. Seoul, South Korea.
26. Raschka, S. (2014). Naïve Bayes and text classification: Introduction and Theory. *arXiv Archive: 1410.5329v4*, 1-20.
27. Cong-Cuong, L.; Prasad, P.W. C.; Alsadoon, A.; Pham, L.; and Elchouemi, A. (2019). Text classification: Naïve Bayes classifier with sentiment lexicon, *International Journal of Computer Science*, 46(2), 141-148.
28. Shimodaira, H. (2015). Text classification using Naive bayes. *Informatics 2B-Learning and Data Note 7*, 1-12
29. Singh, G.; Kumar, B.; Gaur, L.; and Tyagi, A. (2019). Comparison between multinomial and Bernoulli Naïve Bayes for text classification. *Proceeding of the International Conference on Automation, Computational and Technology management.* London, United Kingdom.
30. Patil, T.R.; and Sherekar, S.S. (2013). Performance analysis of Naive Bayes and J48 classification algorithm for data classification. *International Journal of Computer Science and Applications*, 6(2), 256-261.
31. Hossin, M.; and Sulaiman, M.N. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining and Knowledge Management Process*, 5(2), 1-11.
32. Stapor, K. (2017). Evaluation of classifiers: Current methods and future research directions. *Proceeding of the Federated Conference on Computer Science and Information Systems*. Prague, Czech Republic.