CHAPTER 4

FEATURE ENGINEERING

All machine learning classifiers and clustering algorithms use features as input to calibrate and produce output (Sheriff, 2020). Features are defined as individual measurable characteristics or properties of a review and are represented in a 2-dimensional form with rows representing reviews and columns representing the features extracted from it. The features used, initially are in its crudest form, which has to be optimized before it can be used as input. Feature engineering is a task that is used for this purpose.

Today, machine learning technology is growing powerful and hence is used by e-commerce companies to design tools that gain a competitive edge (Behgounia and Zohuri, 2020). In order to design a successful and accurate machine learning algorithm, it is mandatory to put together coherent and appropriate features, which by itself is a challenging task. The collection, cleaning and engineering features is the most cumbersome comprehension of the machine learning process. As quality of features is very important and has a strong influence on the classifier's efficiency, a quality feature set construction is the valuable substantial.

Feature engineering is defined as a task of constructing an optimal feature vector, whose quality has a huge impact on improving the quality of features that are used to train the classifier (Ayuya, 2020). The purpose of using feature engineering is to achieve the following two goals.

1. To prepare the input review dataset in a manner that it is compatible with and best fits the machine learning classifier.

2. To upgrade the machine learning classifier's accomplishment.

Phase I of the research methodology is focused on feature engineering that produces a feature vector with only the significant, relevant and unique features extracted from online reviews. As already mentioned, the spam reviews resemble ham reviews and it is a challenge to find features that can find the dissimilarities (or differences) that exist

between them and can be used to design a successful classifier to find ham and spam reviews. To achieve this goal, several processes are used as listed below

(i) Feature Extraction – Process that produces a reduced representation of online review dataset.

(i) Feature Selection – Process that is employed to select a set of important features from the above collection of large features extracted, in order to reduce the size of feature set. The size reduced feature vector reduces processing time required, thus is fast and more cost efficient. It also helps to improve the performance of the classifier that detects ham and spam reviews.

(ii) Feature Fusion – Process that combines different types of optimal feature sets into a single super feature vector.

In the problem of review spam detection, the three main components, namely, review, reviewer and product, can provide clues for detecting spams. Existing solutions focus on one over the other. In this research work, features obtained from all the three components are considered and the method of extraction and converting it into a quality feature vector is discussed.

4.1. FEATURE EXTRACTION

As mentioned in Chapter 1 (Introduction), three types of features, that can help to identify ham and spam reviews, can be extracted from online reviews. They are,

- the content of the review (Review Centric Features)
- the reviewer who wrote the review (Reviewer Centric Features)
- The product being reviewed (Product Centric Features)

The importance of these features on ham and spam review detection has been reviewed and studied by several researchers (Hussain *et al.*, 2019; Ullah, 2017; Singh, 2015; Rungta, 2015). As of now referenced, the objective of the proposed feature engineering methodology is to envelop procedures that completely use all accessible data in regards to reviews, reviewers and product, to precisely identify spam action in online reviews.As

reported by several authors (Jindal and Liu, 2007; 2008; Fei *et al.*, 2013; Mukherjee *et al.*, 2013; Hammand, 2013; Crawford *et al.*, 2015), usage of multiple features to train a classifier produces better classification result when compared to single type of features. In accordance to this, this research work, extracts multiple features from reviews, reviewers and products. Each review is processed using a series of feature extraction methods which results with a group of feature vectors, each having n set of features belonging to one review, reviewer and product. The study uses the following rating labels (Equation 4.1) during feature extraction.

$$\text{User Rating} = \begin{cases} 1 \text{ and } 2 & \text{Bad} \\ 3 & \text{Average} \\ 4 \text{ and } 5 & \text{Good} \end{cases} \quad (4.1)$$

4.2. REVIEW CENTRIC FEATURES

Review centric features are built up with the help of knowledge that can be extracted from the content of the reviews (Chiny *et al.*, 2021). Seven pigeonholes of review centric features, namely, textual, meta, content similarity, N-grams, rating, Burst/Peak Patterns and Sentiment Score are extracted from the review content. A total of 36 features were extracted and the list of various review centric features (RF1-RF36) extracted in each group is given in Table 4.1.

4.2.1. Textual Features

The first set of review centric features extracted is textual features, which are derived from the review content or text. These features can provide a good idea on the language and sentiments of a review. Several statistical details regarding the text content of the review can be estimated and used to detect spam reviews. In this research work nine such features are extracted and described in this section. The first three features are countable features, namely, number of words, average number of words per sentence and number of remarkable words used. The methods of estimating the rest of the six features are described below.

The percentage of capital words used in a Review is the proportion of number of words in uppercase in the review to the total number of words in the review (Equation 4.2).

$$\text{Percentage of Capital Words} = \frac{\text{No. of Capital Words}}{\text{No. of Words}} \text{x} 100 \qquad (4.2)$$

TABLE 4.1

REVIEW CENTRIC FEATURES

• **Textual Features**
RF1. Number of Words RF2. Average Number of Words Per Sentence RF3. Number of Unique Words Used RF4. % of Capital Words Used in a Review RF5. Frequency of Brand Names Used RF6. Ratio of Number of Characters to Number of Words RF7. Number of Digits RF8. % of Positive Opinion Words in a Review RF9. % of Negative Opinion Words in a Review
• **Meta Data**
RF10. Review Length RF11. Date of Posting the Review RF12. Time of Posting the Review RF13. Reviewer Id RF14. Review Id RF15. Product Id RF16. Number of Feedbacks RF17. Number of Reviews / Day
• **Content Similarity**
Bag of Words RF18. All Words RF19. Unique Words RF20. Polarity-Oriented Words RF21. Term Frequency (TF) RF22. BoW + TF

POS Tagging RF23. Number of Nouns RF24. Number of Adjectives RF25. Number of Propositions RF26. Number of Determines RF27. Number of Verbs RF28. Number of Adverbs RF29. Number of Pronouns RF30. Number of Connector of Words RF31. Number of first person pronouns
• **N-grams** RF32. Spam Hit Score RF33. Ham Hit Score RF34. Spam Miss Score RF35. Ham Miss Score
• **Rating** RF36. Rate Deviation Score of a Review
• **Others** RF37. Sentiment Score RF38. Burst/Peak Patterns

Example : BATTERY life of this mobile is BEST. There are two words in uppercase and therefore this feature value is 2/7*100 = 28.57%.

When the reviewer give a statement of opinion relating to only its brand or product name, then it is considered as a spam feature. The frequency of brand names used is the ratio of the number of times the product is reviewed divided by total number of reviews (Equation 4.2).

$$\text{Frequency of Brand Names} = \frac{\text{No. of reviewes using brand names}}{\text{Total Reviews}} x100 \quad (4.2)$$

The ratio of number of characters (N_c) to number of words indicates the distribution of characters with respect to words in a review of a reviewer for a particular product (Equation 4.3) or average length of each word.

Average Word Length = N_c / Number of Words (4.3)

The percentage of digits feature is the ratio of number of digits in the review to the total number of words in the review (Equation 4.4).

$$\% \text{ of Digits} = \frac{\text{No. of Digits}}{\text{No. of Words}} \text{x100} \quad (4.4)$$

The amount of positive opinion words in a review divided by the number of words in the review yields the rate of positive opinion words in the review. (Equation 4.5).

$$\% \text{ of +ve Opinion Words/Review} = \frac{\text{No. of positive Opinion Words}}{\text{No. of Words}} \text{x100} \quad (4.5)$$

Example: Let the review text be "The mobile I got as gift is one of the **best** one in the market, has **excellent** features and **good** color, shape". The number of positive words is three, while the total number of words is 22, thus, the calculated feature value for this review is 13.64%.

Percentage of negative opinion words in a review is estimated as the amount of words in a review with negative opinions divided by the overall number of words in the review (Equation 4.6).

$$\% \text{ of –ve Opinion Words/Review} = \frac{\text{No. of Negative Opinion Words}}{\text{No. of Words}} \text{x100} \quad (4.6)$$

Example : Let the review text be "My experience with this hotel is **dreadful**. The service was very **bad** !", The % of –ve opinion words for this review is 2/12 = 16.67%.

4.2.2. MetaData Features

MetaData features are features about the reviews, as opposed to features about the text content of the review (Rastogi *et al.*, 2020). Examples of metadata include date, time, rating, reviewer id, review id and product id. Example of metadata features include review ID, product ID, Reviewer ID, Rating, Review characters, Review Words, Date and Time. These features can be used to detect anomalous features and the reviewer id which is associated with it. Once the spammer or reviewer is found, then all reviews related to this

reviewer can be labeled as spam.In this research work, the following metadata are extracted and used during ham and spam review detection.

4.2.3. Content Similarity

A fool-proof method for identifying spams in internet reviews is to look for similarities between evaluations published by the same person. As mentioned earlier, spammers write bounteous reviews in an ephemeral period and hence will not have time to coin new words and have a tendency to make more copies of the same content across the reviews. Even experienced spammers, while altering the content, tend to use the same vocabulary. Thus, using content similarity between reviews of an author can identify spam reviews effectively.

In this research work, content similarity is identified using the cosine similarity between the reviews. The method of estimating this feature consists of two steps, as listed below.

Step 1 : Create Feature Vector

Step 2 : Calculate Similarity

Feature vector is created using three versions of Bag-of-Word (BoW) methods, Term Frequency (TF) and a combined BoW and TF Method. All the three methods are word level features that can be used during ham and spam review detection.

(i) Bag of Words

The conventional BoW method for extracting features takes the review content as input and construct a vector space model having all unique words in them. In BoW representation, the review passage is presented as an unorganised jumble of words, with no regard for word order or grammar. A BoW is a collection of independent words that is not in any particular order, where a text is represented by a *term-frequency* vector and each term of the vocabulary (e.g. English-language vocabulary) defines one dimension of the vector space. The result of Bag of Words model is the Vector Space model. The BoW approach records only the word frequency in each document and projects each document on a fixed dimensionality vector, where each component of the vector represents the value

of one attribute of the document. In other words, the number of times the word 'w_i' occurs in the document 'd_j' is recorded. This quantity of the BoW approach is called Term Frequency, TF(w_i, d_j).

The performance of spam detection depends on the correct selection of a representative BoW. But before obtaining BoW, the reviews need to be preprocessed. The preprocessing is done using four processes, namely, stop word removal, stemming punctuation character removal and lower case conversion. Stop words are words that do not have any useful information associated to it. Examples include prepositions, pronouns and articles. It has proved by Liau and Tan (2014), that the removal of stop words increasing the performance of text processing. Removal of punctuations and conversion to lower case makes the text homogenized. Stemming is the process that transforms the words in a review to its root. Removal of derivational affixes makes different variants of a word to be mapped to a single word. Stop word removal was done using SMART stop word list (Salton, 1989) and stemming was performed using Porter Stemmer algorithm (Porter, 1980). Both these are standard algorithms used in natural language processing and information retrieval applications.

An example of BoW features is shown in Figure 4.1, where each event of a word inside a review is addressed by the recurrence of event of that word in the review content. Thus, the generated BoW feature vector is a 7-dimensional vector of integers.

Review Text : RedMi 10 is a decent smartphone						
Redmi	**decent**	**smartphone**	**Greatest**	**flaw**	**limited**	**memory**
1	1	1	0	0	0	0
RedMi smartphone's Greatest FLAW ! Limited memory						
1	0	1	1	1	1	1

Figure 4.1 : BoW Feature Example

To further improve the quality of BoW features, three methods of extracting BoWs were used. They are all features, unique features and polarity oriented unique words. All features indicates all unique BoWs as shown in above figure. Unique features are words

that can be associated uniquely to one of the predefined class labels (Toral *et al.*, 2018). In this research work, this would mean, words unique to ham and spam classes. The third method polarity oriented unique words, separates the BoWs into four categories by combining them with sentiment polarity. They are BoWs uniquely identified to positive spam reviews, positive ham reviews, negative spam reviews and negative ham reviews. It was proved by Martinez-Torres and Toral (2019) that the third category is more efficient in finding highly discriminating and useful features, while separating ham and spam reviews and hence is used during the construction of the review-centric feature vector.

(ii) Term Frequency

The TF of the words in a review is used to denote the importance of each work in a review. It is estimated as a weight associated to the distribution of each work in the reviews (Karbasi and Boughanem, 2006). The advantages of using this feature during ham/spam classification has been proved by Ott *et al.*(2011) and Jindal and Lu (2008). This method is similar to BoW but differs by considering the frequency with which each word occurs along with its presence / absence in a review. Term frequency is estimated using Equation (4.7).

$$TF_{w,r} = \frac{n_{w,r}}{\text{Number of words in r}} \tag{4.7}$$

where n is the number of times a word ‘w’ occurs in review ‘r’. This makes sure that each review and word have its own TF. Consider the review “RedMi10 is a decent smart phone”, which preprocessing will be “Redmi10 decent smart phone”. Thus, n = 3. As each word appears only once, the TF of all the three words is 1/3. As another example, consider the review “Review : We had a great time at this hotel great stay !”, which after preprocessing will be “great time hotel great stay”. Thus, n = 5. Here, the word ‘great’ appears twice and hence the TF is 2/5, the rest of the words (time, hotel, stay) appears only once and hence the TF = 1/5.

(iii) Combined BoW and TF

The third word level feature vector constructed combines BoW and TF. This, first uses the polarity based BoW to scrape together a words, for which the TF is estimated.

This feature combines the advantage of polarity oriented features with TF, thus producing an optimal set of words that can differentiate ham and spam reviews efficiently.

4.2.4. POS Tagging

POS stands for "part of speech." Tagging is a task that assigns a part of speech to each word in a review based on the analogy and divides the phrase into word groups or lexical categories. (Pennebaker *et al.*, 2007). Li *et al.* (2014) reach a goal of acquiring results in a superior way by along with involving these features for ham and spam detection. POS tagging uses a tag set (Example: Table 4.2), which is the collection or set of tags used to tag the words, to describe how a word is used in the review.

TABLE 4.2
POS TAGSET (PARTIAL LIST)

S.No	TAG DESCRIPTION	TAG	S.No	TAG DESCRIPTION	TAG
1	Noun	<NN>	14	Quantifier	<QF>
2	Noun Location	<NST>	15	Cardinal	<QC>
3	Proper Noun	<NNP>	16	Ordinal	<QO>
4	Pronoun	<PRP>	17	Common Noun	<CL>
5	Demonstrative	<DEM>	18	Intensifier	<INTF>
6	Verb Finite	<VM>	19	Interjection	<INJ>
7	Verb Aux	<VAUX>	20	Negation	<NEG>
8	Adjective	<JJ>	21	Quotative	<UT>
9	Adverb	<RB>	22	Symbol	<SYM>
10	Postposition	<PSP>	23	Compounds	<C>
11	Particles	<RP>	24	Reduplicative	<RDP>
12	Conjunction	<CC>	25	Echo	<ECH>
13	Question Words	<VQ>	26	Unknown	<UNK>

The POS tagging is performed on the tokenized review. In this research work, nine types of POS tags, as listed below, are used for constructing a feature vector, to discriminate ham and spam reviews. Table4.3 shows example reviews along with its POS tags along with its meaning and count.

TABLE 4.3

POS TAGGING - EXAMPLE

Word	POS Tag	Word Class
Review Text : RedMi 10 is a decent smart phone		
RedMi	NP	Noun
10	CD	Number
Is	VBZ	Verb
A	DAT	Determiner
Decent	JJ	Adjective
Smartphone	NN	Noun
Noun – 2; Number – 1, Verb – 2, Determiner – 1, Adjective - 1		
RedMi smart phone's Greatest FLAW! Limited memory		
RedMi	NP	Noun
Smartphone s	NP	Noun
Greatest	JJ	Adjective
Flaw	NP	Noun
Limited	JJ	Adjective
Memory	NN	Noun
Noun – 4, Adjective – 2		

4.2.5. N-grams

The n-grams algorithm has established facts on advantageous testimonials in text categorization (Cavnar and Trenkle, 1994; Hornik *et al.*, 2013). The n-grams algorithm can provide several advantages and some of them are listed below.

(i) The n-grams have more context

(ii) Have high distinctive capacity

(iii) As the number of unique n-grams is finite for a given language, the feature analysis is a finite and feasible process.

The n-grams area bordered arrangement of 'n' items in a particular order. Text, voice, organic material, and so on can be grouped, and the survey words are used as an arrangement in this study. Words or characters can be used to create N-grams. The number n is a characteristic value in n-grams, and it refers to the volume. In this way, an n-gram of size 1 is referred to as a 1-gram or unigram, an n-gram of size 2 is referred to as a 2-gram or bigram, an n-gram of size 3 is referred to as a 3-gram or trigram, and so on. The value of 'n,' and thus the word 'n-gram,' allude to larger sizes. Bigrams (that is, n = 2) are used in this investigation.

Usage of n-grams as features allows the analysis of both content and context of a word in a review. The idea behind the usage of n-grams is that people who have genuine experience will use similar words in the review, because the characteristics or features of a product or service do not change frequently. However, spammers are not compelled to use the same word set. The process of generating n-gram features is discussed below.

The review text (both training and testing) is first converted to lowercase and punctuations are removed. Then, for all reviews in the training set, n-grams are constructed. The n-grams considered as uni- and bi-grams. For example, the bigram for the review "RedMi 10 is a decent smart phone" would be {redmi 10, 10 is, is a, a decent, decent smart phone}. Similarly, the bigram for the review "RedMi smart phone's Greatest FLAW ! Limited memory" is {redmi smart phone's, smart phone's greatest, greatest flaw, flaw limited, limited memory). The results are stored in two separate dictionaries, Spam n-gram dictionary (S_nD) and ham n-gram dictionary (H_nD).

For each review in testing set, similar n-grams are constructed, which are then used to estimate four scores (Table 4.4). The absence (marked as 0) or presence (indicated as 1) of these items is used to calculate the scores (indicated as 1)of an n-gram in the spam or ham set of words. Thus, the examination and determination of n-gram reckons will indicate

- how much a review is similar or dissimilar to spam reviews and
- how much a review is similar or dissimilar to ham reviews

TABLE 4.4

SIMILARITY SCORE USING N-GRAMS

No.	Score	Remarks
1	Spam Hit Score (SHS)	No. of n-grams in test review found in (S_nD)
2	Ham Hit Score(HHS)	No. of n-grams in test review found in (H_nD)
3	Spam Miss Score(SMS)	No. of n-grams in test review not found in (S_nD)
4	Ham Miss Score(HMS)	No. of n-grams in test review not found in (H_nD)

These scores are then used as features during ham/spam reviews detection. Calculate total score to estimate how similar the RR_j is to ham or spam set. If this total score is high, then it is a spam else the RR_j is considered genuine. The procedure is summarized in Figure 4.2.

For each review RR_i in training set

Step 1 : Convert text to lower case characters and remove punctuation symbols

Step 3 : Construct uni- and bi-grams of RR_i

Step 4 : Create dictionaries

For each review RR_j in testing set

Step 5 : Construct uni- and bi- grams of RR_j

Step 6 : Estimate SHS, HHS, SMS, HMS scores

Figure 4.2 : N-Grams as Features

4.2.6. Rate Deviation Score of a Review

One important feature which acts as an indicator to quality of an online product or service is its overall rating. Overall rating is measured by considering respective ratings of all reviews that are associated to it. Spammers manipulate ratings either by increasing or decreasing it. Several researchers, thus, focused on detecting this rate manipulation.

Manipulated rate scenario can be detected by sudden or abnormal deviations in the rating. The rate deviation score is used to identify suspicious high rate deviations from its mean and the method used to estimate this score is described below.

Let RR_i be a review, P_i be the product being reviewed by user 'A' and 'S' be the review score. The Review Deviation score (R_d) is estimated as the deviation of the review score from its mean score (R_{mean}) (Equation 4.8).

$$R_d(RR_i) = |S - R_{mean}| \quad (4.8)$$

The R_d value thus obtained, will negatively impact spam review detection system performance, if the ratings (written by the same reviewer) is included. In order to obtain an accurate R_d, the ratings as of now impacted by different evaluations contributed by a similar reviewer have to be avoided. Thus, the R_{mean} is estimated by excluding all ratings from the same reviewer. The Overall Rating Deviation Score ($OR_D(r)$) is measured by normalizing according to a rating scale, like for example, 5-star scale where $N_{scale} = 5$ (Equation 4.9).

$$OR_D(RR_i) = R_d / N_{scale} \quad (4.9)$$

Apart from the above described review features, two more features, namely, sentiment score and burst pattern were also extracted.

4.2.7. Sentiment Score

Spam reviews have a high percentage of words that show positive sentiments when compared to true positive sentiments. In a similar fashion, spam negative reviews have more negative words than ham negative reviews (Ligthart *et al.,* 2021). In order to handle such situations, sentiment score or sentiment strength, which represents the sentiment polarity of the review and produces a value in the range -1 to 1 (`-1 1]), is used.

The method starts by extracting features/angle things from each sentence in the review. The following stage tracks down the relating conclusion words present in the sentence. It was noticed that the extremity of the assessment word diminishes with the separation from the feature word. The following stage, computes the quantity of

invalidation words to switch extremity because of negative words present. Then, finally, the aggregation (or total score) of a review (Equation 4.10) is obtained to calculate the sentiment score.

$$\text{Total Score } (RR_i) - \frac{\sum(-1)c_n o(w_i)}{\text{dis tan ce}(w_i, f)} \quad (4.10)$$

Where RR_i is the review, f is aspect/feature in a sentence, $o(w_i)$ is the word w_j's supposition extremity (a positive word receives a sentiment polarity score of +1, while a negative word receives a sentiment polarity score of -1), and c_n denotes the number of nullification words in a single component.(If no refuting word is used, c_n rises to 0). While there is only one invalidation word, the sentiment's extreme is shifted by multiplying by -1. The distance between feature f and word w_j is dist(w_i, f).

4.2.8. Burst/Peak Patterns

Bursts are patterns of spamming activity and serves as an efficient method to identify spam reviews. In a review platform, spammers negate the effect of genuine reviews using falsified text that moves the product on the road to a desired tendency (positive or negative), determined by their mission of appointment. They do this by bombarding the review platform with a series of reviews in a short span of time. As a result, reviewing activity increases, resulting in activity peaks or bursts during specific time periods. These high activity time periods are analyzed and used to detect spam reviews. The burst or peak detection is performed using a time series analysis on the amount of reviews written for a single product. Each review along with the review content also stores information regarding the time and date it was created. Let R be the set of reviews and Let P be the product related to N reviews(Equation 4.11). Thus,

$$R(P) = \{r_1, ..., r_N\} \quad (4.11)$$

let T be the time stamp that has a set of dates on which reviews are created, grouped according to the product (Equation 4.12).

$$T(P) = \{d_1, ..., d_N\} \quad (4.12)$$

Here, r_i is the review that was written on date d_i and $1 \leq i \leq N$ and $d_1 \leq d_i \leq d_N$. The duration of time or length, measured as number of day, is estimated as the time between the first and last review (Equation 4.13).

$$D_{Len} = d_n - d_1 \quad (4.13)$$

In this research, the time interval is set to 7 days. A time window of a date Dt_w is defined as a process that divides D_{Len} into balanced downtime I_k(Equation 4.14), where k is the number of downtime for a product's time series and is estimated as

$$k = \text{number of intervals} = len / dt_w \quad (4.14)$$

Assuming that there are I_i reviews posted in time interval i, the average number of reviews / interval can be measured using Equation (4.15).

$$Average(I_i) = N/k, 1 <= I <= k \quad (4.15)$$

When the total number of reviews in a time interval is higher than the number of reviews in its neighbouring intervals, then, it is a candidate for spam analysis. To detect such a scenario, the dt_w is slid on the time series and patterns, relying on the following pattern, are found.

$I_i > I_{i+1}$ for $i = 1$

$I_{i-1} < I_i > I_{i+1}$ for $1 < i < n$

$I_i > I_{i-1}$ for $i = n$

All the intervals that match the above conditions are suspicious to have high spam review activity and all the interregnum whose consummate of reviews is not higher than Average(I_i) are considered genuine and therefore, are ignored. After identifying high spam activity time intervals, all reviews written in this period is considered for further investigation, as it might be the result of a genuine user. For this purpose, the BoW with polarity * + TF content similarity method, discussed above, is used. A score indicating the level of similarity is added to the reviews extracted from this bursty time interval. All those reviews, whose similarity score is high are considered spam.

4.3. REVIEWER CENTRIC FEATURES

Reviewer driven features investigate the entirety of the reviews composed by a specific individual who wrote the review along with information extracted about the reviewer. The substance of a review that expresses an opinion on a product can reveal the extent of spamminess in reviews sent by a spammer. (Dominic and Lijo, 2017). The reviewer-based features can be grouped in the direction through to three sections, namely, behavioral features, rating features and countable features. In this research work, 13 reviewer related features are extracted (Table 4.5) and the method used to extract them are discussed in this section.

TABLE 4.5

REVIEWER CENTRIC FEATURES

RevF1.	Reviewer activities
RevF2.	Maximum Number of Reviews
RevF3.	Review length
RevF4.	Reviewer deviation
RevF5.	Burst Review Ratio (BRR)
RevF6.	Ratio of Verified Purchases
RevF7.	Reviewer Burstiness
RevF8.	Extreme ratings
RevF9.	Reviewer average proliferation
RevF10.	Reviewer spamicity
RevF11.	% of positive reviews
RevF12.	% of negative reviews

4.3.1. Reviewer Activities

Reviewer activities, or the number of reviews posted by a reviewer for a single product, is an obvious indicator of ham and spam review. In general, a buyer will review an item they have bought just a single time, so as to give out their experience with the purchase and/or the product quality. The spam reviews are written with the aim of influencing a person's purchase decision and therefore, spammers write as many fake reviews as possible, so as to dominate either positive or negative opinions. Thus, this

feature is a promising indicator to identify ham and spam reviews. The number of reviews written for a single product by a reviewer is obtained using reviewer ID and date fields.

4.3.2. Maximum Number of Reviews

The number of reviews posted in a due course is a strong indicator in identifying abnormal reviewing pattern and can be used to detect spam reviews. The maximum number of reviews posted by a reviewer on a particular date is computed and is standardized by the greatest worth in the dataset. It was noticed that approximately 75% of spam reviewer compose in excess of five reviews each day. Thus, estimation of this feature can help to detect spam reviews as genuine review writers normally write only one review per day.

4.3.3. Review Length

Spammers, in general, do not write long reviews, as they write several reviews at a time, they will not have time to write a long review (Elmogy *et al.*, 2021). According to Fayazi *et al.* (2015), on average, the spammers write reviews which use less than 400 characters. Similarly, Crawford *et al.* (2016) found out that the spam review length in terms of words is 135. This research work uses these values as thresholds (T1=400 and T2=135) and labels all reviews whose character count is less than T1or word count is less than 135, as spam. Equation (4.16) presents the method used to detect spam/ham review.

$$\begin{cases} \text{Spam} & \text{if } CL(RR_i) < T1 \text{ or } WL(RR_i) < T2 \\ \text{Ham} & \text{Otherwise} \end{cases} \quad (4.16)$$

Where RR_i is the review, CL indicates the count heads of characters and WL denotes the number of words in RR_i. Consider the examples shown in Figure 4.3, all of which fails Equation (4.16) and hence are classified as spam reviews.

Figure 4.3 : Review Length Example

4.3.4. Reviewer Deviation

Deviation from product rating refers to two situations, where the reviewer,

Situation 1 : lays upon a fine rating and addresses bad review

Situation 2 : lays upon a bad rating and addresses a fine review

Both the situations indicates that the content of a review that doesn't coordinate with the rating. This conflict is a classic indication to spam review (Lim *et al.*, 2010; Savage *et al.*, 2015). To estimate this, the sentiment class for each review was identified using the tool provided in http://www.alchemyapi.com/api. Some examples of reviews along with their sentiment classes are shown in Figure 4.4.

ID	Review	Sentiment Class	Rating	Result
R1	This RedMi 10 is a **decent** smart phone	Neutral	3 (Average)	Ham
R2	I **loved** all features in this phone, camera, battery, screen and storage. **Awesome!**	Positive	5 (Good)	Ham
R3	I **liked** this phone and I think it is at a **great** price.	Positive	1 (Bad)	Spam
R4	RedMi's Greatest **FLAW** ! Limited memory	Negative	5 (Good)	Spam

Figure 4.4 : Sentiment Classes of Reviews - Examples

Here, Review R1 has a neutral sentiment class and an average rating of 3 and hence can be classified as ham review. On the other hand, Review R3 with a positive class has a negative rating of 1 and hence it is classified as spam review.

4.3.5. Burst Review Ratio (BRR)

BRR is defined as the proportion of number of reviews written by a reviewer that occur in bursts to the total number of reviews, the reviewer wrote. As the genuine reviews always occur in random, if the BRR is high, then that reviewer is a spammer and reviews

originated from this person can be marked as spam. Equation (4.17) is used to estimate BRR.

$$BRR(A) = B^*(A) / V^*(A) \quad (4.17)$$

where B*(A) is the number of reviews from reviewer A that appeared in review burst and V*(A) is the total number of reviews composed by reviewer A.

4.3.6. Ratio of Verified Purchases (RVP)

In a review platform, if a review has a mark "Verified Purchase", then it would mean that the reviewer has purchased the item and hence, the review written by the purchases can be considered genuine. Ratio of verified purchases is processed as the proportion of number of verified purchases of a reviewer to the total number of reviews written by him/her (Equation 4.18).

$$RVP(A) = N_{VP} / V^*(A) \quad (4.18)$$

Here N_{VP} is the number of verified purchases within V*(A).

4.3.7. Reviewer Burstiness

As mentioned earlier, bursts are patterns of spamming activity and are used to influence buyers. The method of extracting this feature is similar to that of the review burstiness, however, this feature is more oriented towards the reviewer characteristics and not the review text. The method of extracting this feature is discussed below.

As before, dt_w denotes the time window, which is the time period tested for burst activity of a reviewer and is set to 7 days. Moreover, the reviewers who have posted more than five reviews are only considered. Equation (4.19) presents the method of estimating the burstiness (B) of a reviewer (A).

$$B(A) = \begin{cases} 0 & LR(A) - FR(A) > dt_w \\ 1 & \text{Otherwise} \end{cases} \quad (4.19)$$

where, LR is the last review creation date and FR is the first review creation date. On the off chance that a reviewer's general posting movement isn't restricted under a time sp and

t_w, then the reviewer is considered as a spammer and all reviews written by this spammer are spam reviews.

4.3.8. Extreme Ratings

Extreme rating is estimated as the ratio of ratios having extreme ratings to the total number of reviews and is calculated for each reviewer and each product. Equation (4.20) presents the formula used to estimate the extreme ratings.

$$ER(A) = \frac{\left|\{r : r \in R_P(A)\right\|}{|R(A)|} \tag{4.20}$$

Here, A is the reviewer, r is the review written by A and P is the product being reviewed by A.

4.3.9. Reviewer Average Proliferation

This feature considers the reviewers historical activities regarding reviews for other products. Let H be the historical data regarding the number of reviews posted by a reviewer r for a set of discreet products, n. Then, the average proliferation is estimated using Equation (4.21)

$$AvgP(r) = size(H_{A,r}) / n \tag{4.21}$$

Thus, the above equation first ascertains the quantity of reviews(r), written by the reviewer (A) historically for a product. Then, average proliferation is estimated as the proportion between this entirety and all out number of reviewed products (n). A high AvgP indicates unnecessary hyper-activity and untrustworthiness.

4.3.10. Reviewer Spamicity

This feature is used to determine the overall degree of spam of a review in relation to the review, author behaviour of a product, creator's dependability and notoriety dependent on their historical enterprise data. Let r represent the review written by the reviewer (A) for product P. The spamicity (S(r)) is then estimated as(Equation 4.22)

$$S(r) = RD(r)+CS(A)+NR(A)+B(A)+RB(A)+ER(A)+AvgP(A) \quad (4.22)$$

Here, RD is a reviewer's rating deviation score r, CS(a) is a reviewer's content similitude score a, and A reviewer's NR(A) is the number of reviews he or she has written. A, B(A) is a reviewer A's score based on the number of written reviews they've written in a short period of time, and RB(A) is a reviewer A's Reviewing burstiness score based on their overall behaviour. AvgP(A) is the typical number of reviews per product of reviewer A, and ER(A) is the outrageous rating score of reviewer A.

To improve the effectiveness of this feature, a weight, based on its importance, is multiplied. The weights were allotted according to the following basic facts.

- Content similarity is assigned a high weight of 1.5, as it plays an important role during spam review identification.
- Bursty activeness, Reviewer Burstiness and Rating Deviation comes alongside in their effectiveness to find spam reviews and were assigned as weight of 1.
- Finally, number of reviews, extreme rating and average proliferation were assigned a weight of 0.5.

The modified weighted spamicity WS(r) is given in Equation (4.23).

$$WS(r) = RD(r) + 1.5 * CS(A) + 0.5 * NR(A) + B(A) + RB(A) + 0.5 * ER(A) + 0.5 * AvgP(A) \quad (4.23)$$

The method of extracting this feature, also includes a reciprocal value of 1 to all the mentioned reviews, whose reviewers have reproduced the text across all of their reviews. This complementing act on content duplication is performed using Equation (4.24) for each review r, written by the reviewer A.

$$\text{If } (NR(A) > 1 \;\&\&\; CS(A) > 0.99) \text{ then } WS(r) = WS(r) + 1.0 \quad (4.24)$$

A reviewer is considered as a spammer, if the weighted spamicity score is greater than a threshold, T, which is set to 4 in this research work.

4.3.11. Percentage of Positive Reviews

At any point of time, around 80% of the reviews written by 85% of spammers are positive reviews. Thus, an immense proportion of positive reviews indicate spamming. It is calculated as the number of positive reviews written by a reviewer on a product P to the overall quantity of reviews written by the reviewer.

4.3.12. Percentage of Negative Reviews

In a similar fashion, too many negative reviews also indicate spamming. It is calculated as the quantity of negative reviews composed by a reviewer on an item P to the complete quantity of reviews written by the reviewer.

4.4. PRODUCT CENTRIC FEATURES

Product centric features are used to obtain details regarding the product being reviewed. The importance of product centric features on ham/spam review detection is described in Ioannis (2017). The product centric features can provide knowledge regarding the public popularity of the product. In this work two product related features, namely, rank-in-sale and average rating are extracted.

4.4.1. Rank in Sale

Rank-in-sale feature denotes the sales rank of a product and is generally provided by the e-commerce website like Amazon in an hourly basis. An example taken from Amazon website is shown in Figure 4.5.

Figure 4.5 : Amazon Sales Rank Example

This feature is considered as a helpful feature during spam review detection because, too few selling or very high selling product are generally targeted by the spammers.

For example, consider a product with ID 'A00006ALPR'. Let the e-commerce site based rank be #2 and let the reviewer-based rating be 2.0. According to this, the e-commerce site-based rank in sale is #2, which indicates that the product under analysis is among the best selling products. On the other hand, the rating provided by the review is 2, which means that the product has very poor or bad rating. Thus, the review can be labelled as spam.

4.4.2. Average Rating

When a product receive extreme rating (that is maximum or minimum), then it is said to be a spam review. As its rating deviates from average rating received by the product. It is estimated using the formula in Equation (4.25).

$$AvgR = \sum_{i=0}^{n} (RR_i) / n \qquad (4.25)$$

In the above equation, the i^{th} review appraisal is RR_i, and the total number of reviews is n. Assuming that a review's individual rating differs from the average rating, it

is supposed to be spam. The Rating Deviation (RD) is defined as the difference between the RR_i from its average rating (AvgR) and is estimated using Equation (4.26).

$$RD = RR_i - AvgR \quad (4.26)$$

Consider for instance a circumstance where the normal rating of 8 reviews is 3.0. In the event that a Review Ri is higher (or lower) than this normal rating with the distinction surpasses one, then, at that point Ri is marked as spam review.

4.5. CREATION OF OPTIMAL FEATURE VECTOR

In this research work, in order to accurately identify spam reviews from a huge collection of online reviews, 38 review centric features, 12 reviewer centric features and 2 product centric features were extracted. A single feature, even if it is a strong indicator to spam, might not be definitive in its nature. For example, an unusual rating, which normally increases the rate deviation score, may be attributed to specific experience of a purchaser, who might be either too satisfied or heavily dissatisfied with the purchase, despite majority of opinion being on the opposite rating. Similarly, an increased quantity of positive (or negative) words in the passage might also be the reason of extremely happy or extremely angry genuine customer. Therefore, it was decided to use a feature vector that is combined with multiple features to provide a more reliable and accurate classification. When a review has high scores in several features indicates that the review under consideration is bound to be a spam rather than analyzing the score of one or two spam conditions.

A highly accurate online ham/spam review detection system can be built while using a huge sized feature vector as it will provide high discriminative capacity (Liu *et al.*, 2017). However, these huge feature vectors also imposes several disadvantages like time complexity (reduced speed), increased number of redundant data and increased number of irrelevant data whose contribution during classification is null. Moreover, presence of irrelevant and redundant data also results with overfitting of training data. These issues can be solved using feature selection algorithms. While using feature selection algorithms on the large feature vector, its size is reduced, which in turn reduces the time complexity,

improves the classification algorithm's generalization ability and provides a fast and cost effective system.

The act of obtaining a small subset of features, F*, from a large feature set, F, that satisfies Equation (4.27) is known as the feature selection issue

$$P(C \mid F^* = C_{F^*}) \cong P(C \mid F = C) \quad (4.27)$$

Where F* is called the optimal feature vector obtained from the feature selection algorithm, $P(C \mid F^* = C_{F^*})$ presents the probability distribution of various reckons habituated of C in F* and $P(C \mid F = C)$ is the primordial probability distribution of C in F.

However, the usage of feature selection algorithms can reduce several quantity of fragment in features and choosing the most effective in connection with them is a difficult and challenging problem (Schmidt *et al.*, 2019) and therefore the feature selection problem is NP-Hard (Too *et al.*, 2019). The steps associated with a feature selection algorithm are displayed in Figure 4.6.

The first step generates subset that has a set of optimal features, which are analysed in the second step using two types of algorithms, namely, filter and wrapper based methods (Hossain *et al.*, 2013). The filter and wrapper both attempt to create a feature vector having optimal features and differs in the manner of evaluating the subsets generated in the first step. The filter based approaches selects the feature subsets separately without any interaction with the learning algorithm and depends on a few proportions of the training data like information, dependency, consistency and distance. Oppositely, the wrapper based approaches use the learning algorithm to select the feature subsets.

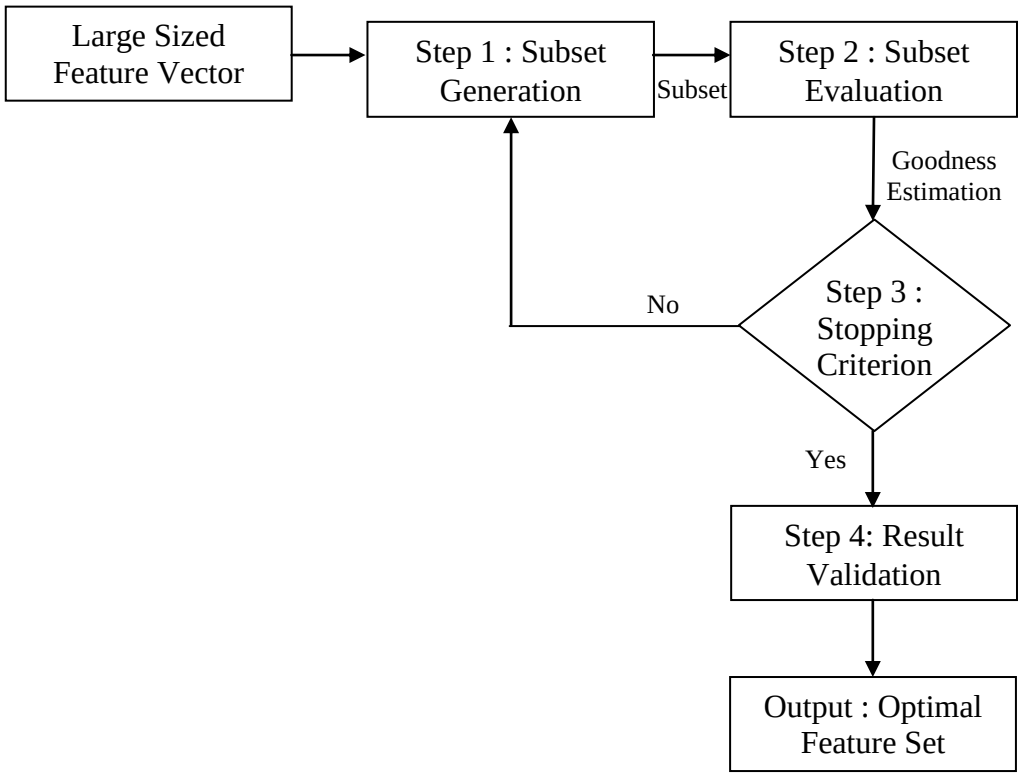

Figure 4.6 : Feature Selection Algorithm

During the decision of whether to include a feature to the optimal set, a subset evaluation method (Guyon and Elisseeff, 2003) is used. This technique evaluates features based on how well they distinguish between different target labels' features. In this step, the inappropriate and extraneous aspects are eliminated and chosen solely these aspects that satisfy some goodness measure. Several algorithms are available to perform this task, but not all of them are suitable for very large feature set, as is the case of this research. Thus, different methods that can better handle the redundant and irrelevant features are needed.

The stopping criteria is used to make the decision on when the optimal feature subset evaluation has to be stopped, with regard to fabricate the final optimized set. The stopping criterion is generally implemented as a threshold checking. When the checking fails, a new subset is generated and the steps are iterated, else, it outputs the result. In general, the threshold is related to either range of features selected or quantity of iterations. Thus, the feature selection mechanism is stopped in response to the following two questions and is related to the first and second step.

- Has a predetermined quantity of features are handpicked?
- Has a predetermined quantity of iterations accomplished?

In addition, the stopping criterion also has the answer to the following questions while using the evaluation function.

- Is the addition or deletion of a feature degrade the performance of the subset?
- Has a subset produces high value to some evaluation function?

Finally, the validation step is used to find whether the result of subset evaluation is valid or not. This is done by comparing the result of classification while using the large sized feature set with the result of classification while using the feature set obtained after using feature selection algorithm.

4.6. PROPOSED FEATURE SELECTION ALGORITHM

All the feature selection algorithms use two concepts from information theory, namely, relevancy and redundancy. Relevancy is defined as a measure of dependency between random features and has a direct effect at the enforcement of the detection system. Relevant features are highly correlated to the predefined target labels (that is, ham and spam). Redundancy is defined as the repetitive nature of a feature and represents a scenario where a feature takes the role of another features. They are highly uncorrelated to each other and degrade the performance of the detection system. Thus, the feature selection algorithm must be able to remove as many irrelevant and redundant features. However, existing algorithms impose high time complexity and incur performance degradation, especially with large sized feature sets (Kumar and Minz, 2013). To avoid this, a two-step filter based algorithm, as introduced in Chapter 3 (Methodology), is proposed. This algorithm performs three steps (Figure 4.7), namely,

1. Candidate feature vector sets using filter based algorithm
2. Feature fusion to obtain super feature vector
3. Optimal feature vector construction

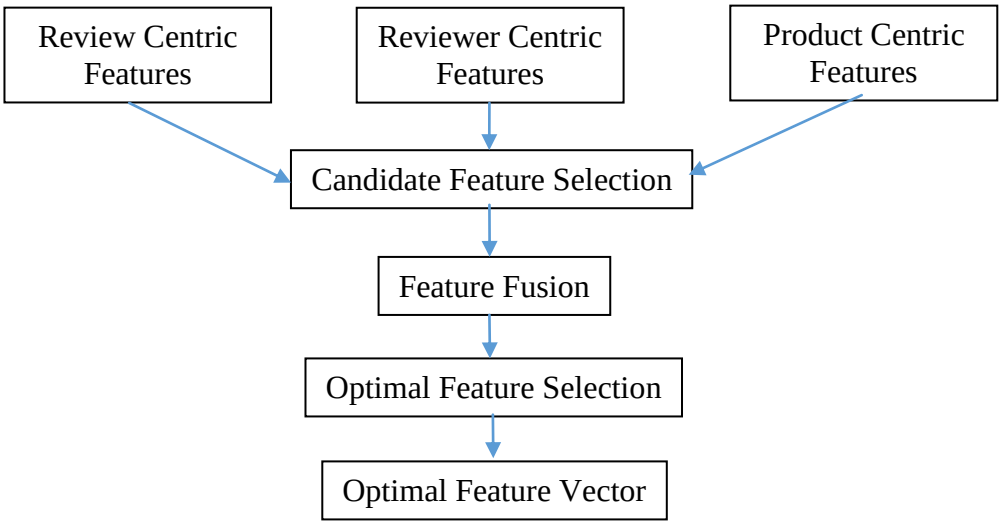

Figure 4.7 : Steps in Proposed Hybrid Feature Selection Algorithm

The first step uses an enhanced filter based algorithm based on Maximum Relevant Minimum Redundant (MRMR) algorithm to produces a set of candidate feature sets. This step is mainly used as a reduction step to minimize the dimension of the large sized feature sets extracted and also is incorporated to reduce the complexity of the third step. The filter based algorithms are simple and computationally inexpensive and hence are very fast. They do not need any iterative procedure and has to be executed only once. These advantages make sure that the results from the first step is fast and removes a huge bunch of redundant and irrelevant details, as MRMR is focused on these two issues.

The resultant sized reduced feature sets are then fused to form a single feature vector in the second step. This step is included in the proposed spam identification system mainly to solve the incompatibility issues raised by multiple modalities of features and unknown relationship between the feature sets. The third step then uses an Ant Colony Optimization (ACO)-based algorithm enhanced through the adaption of Genetic Algorithm (GA) to obtain the final optimal feature vector. Each of these steps is described in detail in the following section.

4.6.1. Candidate Feature Selection

Initially, feature selection algorithms dedicated itself to identify only relevant features, only in the past few years, the focus has shifted towards redundancy checking

also (Yu and Liu, 2004). To accommodate this, the candidate feature set is created using an enhanced MRMR algorithm. This algorithm was selected because it can remove both irrelevant (using irrelevancy analysis) and redundant data (using redundancy analysis) within a single procedure. The usage of MRMR algorithm in the proposed ham/spam review detection system provides the following advantages.

This design, when compared with the conventional manner of FS, brings in the following advantages.

(i) Do not use time consuming search procedure and produces a feature set that works similar to the original feature vector

(ii) Reduced time complexity as the algorithm first identifies and removes irrelevant features first, thus, reducing the number of computations required during redundancy analysis.

The algorithm is improved through the use of two versions of MRMR, which differs in the way feature selection is performed. The two methods used for this are Information gain or IG (to get informative features) and Mutual Information or MI (to get features with maximum discrimination capability).The algorithm further examines the relationship that exists between features and classes using two metrics. The first metric examines the relationship between two features (Feature to Feature relationship or FFR) and the second metric examines the relationship between a feature and target classes (Feature to Class Relationship or FCR). The algorithm proposed in this step combines the results of these two versions of MRMR and aim to reduce the discrepancies that exist between features apart from removing irrelevant and redundant features. Thus, the candidate feature selection algorithm consists of two steps, namely, analysis of features and selection of optimal candidate features.

The analysis of features is done according to the relevancy of features and is grouped into three categories, namely, strongly relevant features; weakly relevant features and irrelevant features John *et al.* (1994). Let F be the feature vector having the extracted review centric or reviewer centric or product centric features, with n number of features. That is, $F = \{f_1, \ldots f_n\}$. A feature, f_i, is said to strongly relevant or weakly relevant or

irrelevant if conditions in Equations (4.28) – (4.30) respectively are satisfied and uses the correlation between features.

$$P(C \mid f_i, S_i) \neq P(C \mid S_i) \quad (4.28)$$

$$P(C \mid f_i, S_i) = P(C \mid S_i)\ \exists\ S_i' \subset S_i, \text{ such that } P(C \mid f_i, S_i') \neq P(C, S') \quad (4.29)$$

$$\forall\ S_i' \subset S_i,\ P(C \mid f_i, S_i') = P(C, S') \quad (4.30)$$

All strongly relevant features have to be added to the candidate feature set, as removal of them will result in performance degradation of ham/spam identification system. All features identified as weakly relevant, may not be present in the final candidate set. But, under certain situations, they may become important. All features identified as irrelevant can be removed, as its presence has a negative influence on the performance of the detection system. All relevant features should be selected, all irrelevant features should be removed, and a subset of weakly relevant features should be selected in an ideal feature selection method.

The next step of the analysis algorithm is thus to decide how to catch sight of the portion of weakly relevant features. Sahy *et al.* (2018) portrays as all redundant features in the relevant and weakly relevant feature categories can be removed and the rest can be added to the optimal. Thus, it is required to define redundancy of features among relevant and weakly relevant features. Thus the feature vector is grouped into four categories, namely, strongly relevant, weakly relevant and redundant features, weakly relevant and non-redundant features and irrelevant features. The Candidate Feature Vector (CFV) is formed using Equation (4.31).

CFV = Strongly Relevant + Weakly Relevant with Non-Redundant Features (4.31)

The two versions of MRMR proposed with an aim to create a feature set as desired by Equation (4.31).

(i) MRMR-IG Based Feature Selection Algorithm

The conventional MRMR method is a filter based algorithm proposed by Ding and Peng (2003). The algorithm selects features which are dissimilar to each other and combines minimum redundancy criteria with maximum relevancy criteria (using correlation analysis). This approach is therefore called as the minimum redundancy – maximum relevance (MRMR) approach. This section uses information gain to perform the correlation analysis.

Correlations between two features are measured using either linear correlation coefficient or non-linear correlation coefficient. Conventionally, there are two ways the correlation can be measured between two random features. They are, linear correlation coefficient and non-linear correlation coefficient (Mitra *et al.*, 2002). In this research work, non-linear correlation coefficient is used. The non-linear correlation coefficient is based on entropy, which is defined as the criterion of the perplexity of an incidental feature X and is defined as in Equation (4.32).

$$H(X) = -\sum P(X_i) \log_2(P(x_i)) \quad (4.32)$$

and the entropy of X after observing values of another feature Y is given in Equation (4.33).

$$H(X \mid Y) = \sum_i P(y_i) \sum_i P(x_i \mid y_i) \log_2(P(x_i \mid y_i)) \quad (4.33)$$

In the above equation, $P(x_i)$ denotes the aforementioned chances for all values of X and $P(x_i|y_i)$ symbolizes the subsequent probabilities of X given the values of Y. More information of X provided by Y can be measured by analysing the amount by which the entropy of X decreases and this information is termed as "Information Gain (IG)" (Quinlan, 1993) (Equation 4.34).

$$IG(X \mid Y) = H(X) - H(X \mid Y) \quad (4.34)$$

IG measure confers that a feature Y is highly related mutually to feature X compared to feature Z if condition in Equation (4.35) is satisfied.

$$IG(X \mid Y) > IG(Z, Y) \quad (4.35)$$

IG is a symmetric measure, where symmetry is defined as a property that measures correlation amidst features and makes sure that the order of two features (X, Y) or (Y, X) has no bearing on the measure's worth. Immune globulin is frequently normalised with their equivalent entropy since it favours features with additional values. For this reason, a Symmetrical Uncertainty (SU) measure is chosen (Press et al., 1988) (Equation 4.36).

$$SU(X,Y) = 2\left(\frac{IG(X \mid Y)}{H(X) + H(Y)}\right) \quad (4.36)$$

The SU atones for IG's distorts toward features with highest number of values and restricts the values to the range [0, 1]. Here, a value 1 illustrates that they are correlated and a value 0 indicates that they are independent. Moreover, as mentioned previously, it treats the features symmetrically and can handle both nominal and discrete values. As the online reviews dataset has several continuous features, they need to be properly discretized in order to use entropy-based measured. For this purpose, the method proposed by Usama and Keki (1993) is used. The steps used by the proposed MRMR-IG version are given in Figure 4.8.

- Relevancy Analysis (Output F*)
  - Estimate FCR of each feature
  - Arrange features in decreasing order of FCR
  - Select top K features as features (that is, features with high FCR) that are relevant to class C
- Redundancy Analysis
  - Identify predominant features and non-predominant features
  - All non-predominant features related to that class are removed
  - For each predominant features
    - Apply approximate Markov Blanket on F*
    - Select predominant features having high FFR with its class and which is not repeated itself

Figure 4.8 : Steps Involved in MRMR-IG

The FCR and FFR measures are estimated based on the measure of correlation, SU. Let F_i be a feature. The first step performs relevancy analysis. The feature, F_i is relevant, if F_i and class C are highly correlated. The feature F_i is considered relevant if $SU_{ic} > T$, where T is the relevance threshold, which is set to a value 1 after several empirical experiments.

The second step, redundancy analysis, is next performed to identify redundant features from the results of first step. For this, details of FFR are utilized. According to the definition of redundancy (described previously), a feature is considered as redundant feature, assuming that it is weakly relevant and has a Markov blanket not beyond the prevailing collection of features. The markov blanket was proposed by Koller and Sahami (1996), who found that optimal feature set can be constructed using backward elimination method and termed this method as Markov blanket filter and is described below.

The method begins by removing a feature fi from the feature set F, supposing that there is a Markov blanket for this feature in the prevailing collection of F. This ensures that when removing features, the best feature subset excludes the previously eliminated features. Even if other functions are deleted, the redundant functions previously deleted are still redundant.

To find the FFR correlation value between two features, f_i and f_j, again the SU_{ij} is estimated. When $SU_{j,c} \geq SU_{i,c}$, then an evaluation method that determines whether feature F_j can be an almost accurate Markov blanket for feature F_i (this will help to retain information regarding the class) is used. The threshold T is used to determine whether the FFR and SU_{ij} is strong or not. Given two relevant features F_i and F_j ($i \neq j$), the feature F_j is considered as a similar Markov blanket of F_i, if and only if both Equations (4.37) and (4.38) are satisfied.

$$SU_{j,c} \geq SU_{i,c} \quad (4.37)$$

$$SU_{i,j} \geq SU_{i,c} \quad (4.38)$$

The above two equations makes sure that in case of uncertainty redundancy between features Fi and Fj, Fi is chosen over Fj, if Fi has more class information than Fj. When addressing uncertainty redundancy, the essential objective is to look for significant features in an approximate Markov blanket. This method utilizes the backward elimination

method to make sure only the redundant features are removed. However, this method has a small issue which can be best explained using the following example. Let Fj be the only thing that forms the approximate Markov blanket of Fi and allows Fk to form the characteristic of Fj approximates the Markov blanket. After removing Fi according to Fj, and then removing Fj according to Fk, there is no approximate Markov blanket of Fi in the current set. , Deleting articles can only avoid this situation if you can find an approximate Markov blanket composed of dominant articles. In the current set, if the corresponding function does not have an approximate Markov mat, it is called the dominant element.

Since the objective is to bonanza a small part of non-redundant functions, and those functions that constitute the near proximate Markov blanket of the function Fi may be more closely related to Fj, the MB candidates for Fi are created by collecting all Fi-related functions. The algorithm is shown in Figure 4.9. This sequential heuristic method is more effective than the method of performing a broad combinatorial search on a subset of the function set.

Initialize : G = F

Iterate

- For each feature F_i?F, let M_i be the set of k features F_j? F - $\{F_i\}$ for which the correlations between F_i and F_j are the highest.
- Compute $\Delta(F_i|M_i)$ for each i
- Choose the i that minimizes $\Delta(F_i|M_i)$ and define F= F - $\{F_i\}$

Figure 4.9 : Markov Blanket Filter

The proposed MRMR based on IG is thus formed using the above described details and the steps involved are presented in Figure 4.10.

```
for i = 1 to N do
        Estimate SUic for Fi
        if SUic > T
                S = S + Fi
        end if
end for
Arrange S in increasing order of SUic value

Fj = S(1)
do
{       Fi = getNextElement(S, Fj)
        if Fi ≠ NULL
                do
                {       if SUi,j >= SUic
                                S = S - Fi
                        end if
                        Fi = getNextElement(S, Fi)
                }until(Fi == NULL)
                Fj = getNextElement(S, Fj)
}until (Fj == NULL)
F*= S
```

Figure 4.10 : The MRMR-IG Algorithm

The algorithms uses the training dataset (Tr) as input along with the relevancy threshold, T and results with the optical candidate feature set, F*. In this method, N is the total number of features, C is the set of target class labels, S is the set of candidate subsets created during the process of candidate feature selection.

The algorithm works in two steps. The first step constructs a list of candidate feature subsets using three processes. The first process measures SU for each feature. The second process selects relevant features from first process based on threshold T and adds them to S. The third process, then arranges this list in descending order.

The second step, considers each feature set from the arranged list and identifies the predominant features in it. A feature Fj that has already been declared as predominant are used to remove divergent features for which Fj contours a near proximate Markov blanket. All the features with highest FCR do not have resembling Markov blanket and hence are considered as predominant features.

For all the remaining features, if F_j creates a near proximate Markov blanket with respect to F_i, F_i is removed from S. This is repeated for all features. Then, the iterative steps are repeated for the next element of S. The iteration is stopped when the set of predominant features is empty. Finally, S is returned as the selected optimal feature subset, F'.

(ii) MRMR-MI Based Feature Selection Algorithm

The method of select optimal candidate set is the same of MRMR based on IG. The difference in the method used to estimate the correlation. Here, the mutual information is used. MI is used to measure the degree of affinity in the midst of two discrete random features X and Y (Hoque *et al.*, 2014) and is estimated using Equation (4.39).

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log\left(\frac{p(x,y)}{p_1(x) p_2(y)} \right) \qquad (4.39)$$

where p(x, y) is the joint PDF (Probability Distribution Function) of X and Y and the marginal PDF of X and Y respectively are denoted by $p_1(x)$ and $p_2(y)$. I(X, Y) can also be given as Equation (4.40).

$$I(X, Y) = H(X) - H(X \mid Y) \qquad (4.40)$$

The boundary entropy is H (X), the conditional entropy is H (X | Y), and the common entropy is H (X; Y). If H (X) is a measure of the uncertainty of a random variable, then H (X | Y) measures Y without mentioning X. This is the level of uncertainty in X when we know Y, confirming the intuitive value of mutual information, or knowing how much knowledge one variable provides about another. Mutual information measurement is utilised in the MRMR-MI method to determine the information increment between objects as well as between objects and class properties. Greedily, every time an object is selected from a set of features, what is left is the object that provides the most information about the class attribute with the least redundancy.

For simplicity, the following notations are used while describing the algorithm. Let fi be the i^{th} instance random feature in a feature set F. Also assume that fi is described

using k observations, that is, fi = {fi1, fi2, ..., fik}. Further, I(Fi, Fj) represents the MI of i^{th} and j^{th} feature, where i, j = 1,2, .., d, and d denotes the amplitude (which is the same as the number of features) of the feature set. The working of the MRMR-based on MI is described below.

The method starts with the estimation of FCR using mutual information. All features having high FCR are then selected and added to S and removed from F. In the second step, the average FFR is calculated for each of the opted features. Then, all the features having high FCR and low FFR are selected. These algorithms also use the concepts of strongly, weakly and irrelevant features and are modified to consider MI and are respectively given in Equations (4.41–4.43).

$$I(C \mid fi, Si) \neq I(C|Si) \tag{4.41}$$

$$I(C \mid f_i, S_i) = I(C \mid S_i)\ \exists\ S_i' \subset S_i, \text{ such that } P(C \mid f_i, S_i') \neq P(C, S') \tag{4.42}$$

$$\forall\ S_i' \subset S_i, I(C \mid f_i, S_i') = I(C, S') \tag{4.43}$$

A feature F_i will be added to relevant features list of class C_i, if its relevance (f_i, C_i) is large. Similarly, high FFR valued feature are added to the optimal list. Consider two features f_i and f_j. The steps involved in the proposed MRMR based on MI is presented in Figure 4.11, which takes the original feature set f as input and produces the optimal feature set, F*.

Step 1 : Estimate MI of all features in F

Step 2 : Select f_i with maximum MI

Step 3 : Add f_i to F_{new} and Remove f_i from F

Step 4 : Repeat

Estimate sum of F-F-R(f_i, f_{new})

AF = Average F-F-R(f_{new})

Estimate C-F-R(f_{new},C) for all features in F

Step 5 : Select next f_{new} that has maximum AF but minimum C-F-R

Step 6 : Go to Step 3 until all f_i are processed

Step 7 : F* = F_{new}

Figure 4.11 : Steps in MRMR-MI

(iii) Combining Results

An important steps while combining two algorithms is the fusion or combining the results from them (Rajab, 2017). In this research work, a scoring algorithm is used. This method starts with a normalization procedure, which is incorporated to make the two optimal candidate feature sets comparable (Equations 4.44 and 4.45), where MRMR-IG_{Max} and MRMR-MI_{Max} denotes the maximum values. Next, a score vector that has details regarding both MRMR-IG and MRMR-MI, is constructed using Equation (4.46).

$$\overline{\text{MRMR\_IG}}_{f_i} = \frac{\text{MRMR\_IG}_{f_i}}{\text{MRMR\_IG}_{\text{Max}}} \tag{4.44}$$

$$\overline{\text{MRMR\_MI}}_{f_i} = \frac{\text{MRMR\_MI}_{f_i}}{\text{MRMR\_MI}_{\text{Max}}} \tag{4.45}$$

$$SV_{f_i} = \left(\frac{\overline{\text{MRMR\_IG}}_{f_i}}{\text{MRMR\_MI}_{f_i}} \right) \tag{4.46}$$

The magnitude of the score vector is measured using Equation (4.47). This result of this equation is used as scalar metric of the vector and is measured as square root of sum of squares of its coordinates and is used as a scalar metric of the vector.

$$\left|SV_{f_i}\right| = \sqrt{\left(\overline{MRIG_{f_i}}\right)^2 + \left(\overline{MRMI_{f_i}}\right)^2} \quad (4.47)$$

All features have high magnitude values have high rank and therefore will be added to the optimal candidate feature set. The steps involved are summarized in Figure 4.12, where F_{IG} and F_{MI} are the feature sets obtained from MRMR based on IG and MI respectively.

- NF_{IG} = Normalize F_{IG} and NF_{MI}= Normalized F_{MI}
- Construct scalar vector using NF_{IG} and NF_{MI}
- Estimate magnitude of the scalar vector
- Arrange features in descending order of magnitude
- Select top K features as optimal candidate features

Figure 4.12 : Combining Features

4.6.2. Feature Fusion

The second step of the proposed feature selection algorithm is the integration of the candidate feature vectors obtained from the above step. For this purpose, a feature-level fusion algorithm is proposed, where the optimal feature sets from the three groups (review, reviewer and product) are combined to form single super vector. This method has been successfully used by several researchers (Liu and Wechsler, 2001; Yang *et al.*, 2003; Yang *et al.*, 2013) in the field of content based image retrieval and this work uses the same for combining the optimal features extracted from online reviews. The procedure is presented in Figure 4.13.

Input : Candidate Feature Sets (F_i); Output : Optimal feature vector, F.

Step 1 : Let TF = {}

Step 2 : Normalize each feature set using zero mean and unit-variance method. Let the normalized feature sets be denoted respectively as f_1^*.

Step 3 : TF = f_1; D_{TF} = Dimension of f_1
Repeat

if D_{TF} = = D_{fi} then $\theta = 1$ and TF = $\begin{pmatrix} TF \\ f_i \end{pmatrix}$

else if $D_{TF} > D_{fi}$ then $\theta = \frac{D_{TF}^{\ 2}}{D_{f_i}^{\ 2}}$ and TF = $\begin{pmatrix} \theta\, TF \\ f_i \end{pmatrix}$

elseif $D_{TF} < D_{fi}$ then $\theta = \frac{D_{f_i}^{\ 2}}{D_{TF}^{\ 2}}$ and TF = $\begin{pmatrix} f_i \\ \theta TF \end{pmatrix}$

TF = F; D_{TF} == Dimension of TF

Until all feature sets are analyzed;

Step 4 : Output TF

Figure 4.13 : Feature Integration Algorithm

4.6.3. Optimal Feature Selection

This step uses an enhanced ACO-based feature selection on the above obtained super feature vector to form the final optimal feature vector. This section presents the conventional ACO-based feature selection algorithm first, followed by the suggested method that combines this algorithm with GA.

(i) ACO-Based Feature Selection Algorithm

The ACO algorithm (Dorigo *et al.*, 1999) is a branch of swarm intelligence which uses collective intelligent groups of simple agents to optimize the task of feature selection. It is a subject of interest that accordance with the natural and artificial systems that are made up of a group of insects (examples include schools of fish, colony of ants, bees and flock of birds), where each individual insect can perform simple venture on its personal and the colony's cooperative works is

the major purpose identifying the shrewd conduct it indicates (Liu *et al.*, 2004; http://www.scholarpedia.org/article/Swarm_intelligence).

In simple terms, swarm intelligence algorithms are stochastic investigation strategies that mimic the natural biological evolution and/or the social conduct of the species. Among the various swarm intelligence methods, the ACO algorithm is simulated by using the social conduct of the ants. Ants, as such, do not have sight, but have the capability to search the shortest route from their nest to meals supply the usage of chemical substances known as pheromone, which is left behind when they move. This pheromone can be smelled by other ants and which gives them the ability to follow the same path that has been successively passed.

An ACO algorithm, while used for feature selection, is required to be designated as a visual representation as graph, whose nodes denote the feature and edges bounded by the nodes representing the of proximate feature determination. The optimal feature investigation of subset is then described as a task where an ant move through the graph in a direction that requires a narrowest of node visits a stopping criterion fascinated. Let k be an ant and i and j be two features. Let F be the original full feature set and F' be the final optimal feature subset. Equation (4.48) presents the probabilistic transition rule, which describes the probability of k at feature i choosing j at time t.

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{\left(\tau_{ij}(t)\right)^{\alpha}\left(\eta_{ij}\right)^{\beta}}{\sum_{l \in J_i^k}\left(\tau_{il}(t)\right)^{\alpha}\left(\eta_{il}\right)^{\beta}} & j \in J_i^k \\ 0 & \text{Otherwise} \end{cases} \qquad (4.48)$$

Here, η_{ij} represents the desirability of an ant (at node i) to select feature j as the next node to visit. J_i^k is the group of neighbouring nodes at node i that have not been visited but with the aid of k. The two parameters, $\alpha > 0$ and $\beta > 0$ are used to determine the relative importance of pheromone value and heuristic information. After empirical review, the values of and were set to 1 and 0.1, respectively, based on the results of the experiment. The quantity of pheromone on the edge is represented by $\tau_{ij}(t)$in the equation

(i, j).The steps involved during ACO-based feature selection are given in Figure 4.14, which performs feature selection using two tasks.

Task 1 : Application of ant colony optimization (denoted by straight boxes)

Task 2 : Evaluation using SVM classifier (denoted by dashed boxes).

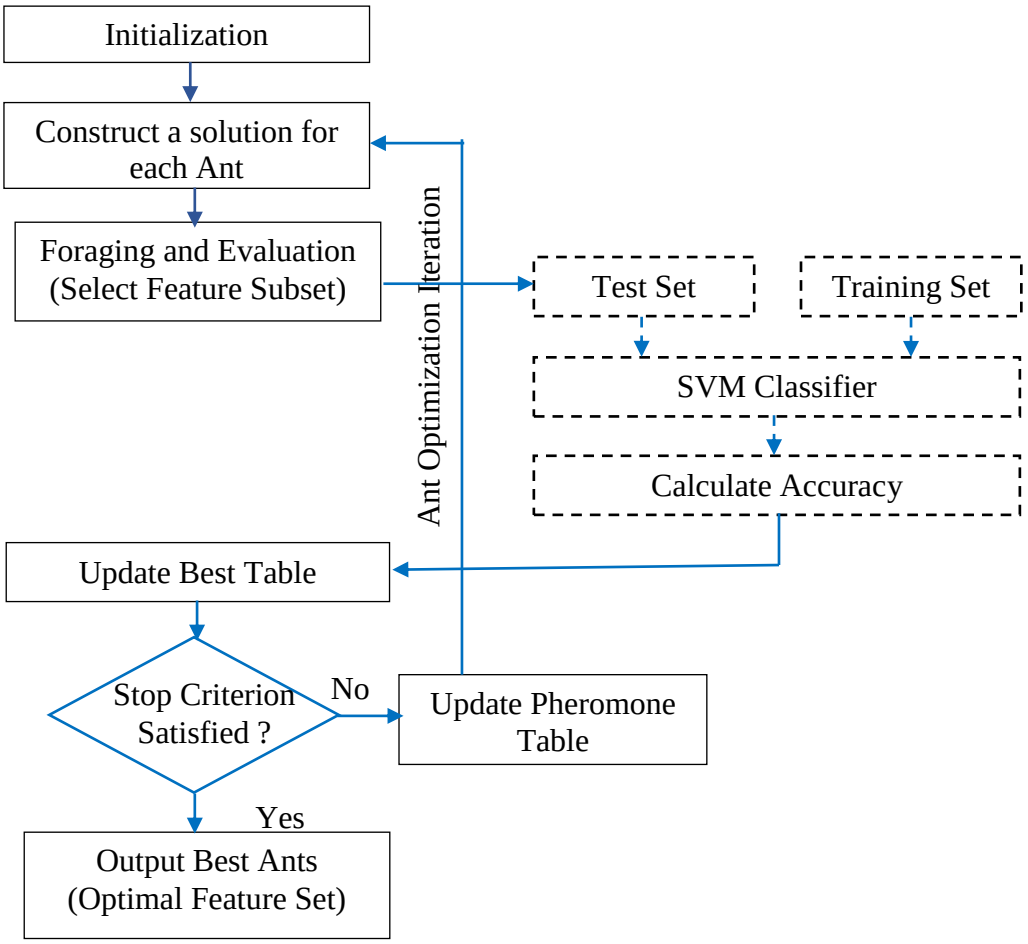

Figure 4.14 : ACO-Based Feature Selection

The algorithm begins with ACO initialization step, which generates a set of ants (say, m) and set them randomly on the graph. This means that the each of the m ants start with one feature selected randomly. In this work, the number of ants, that is, m is set to the number of attributes or features extracted. This enables each ant to start the construction of path at different feature. Thus, in this work, every node depicts a feature and the edge amidst two nodes is the paths between the nodes. During initialization, the initial intensity of pheromone trails associated with any feature is also set to one. In this step, the

maximum number of allowed iterations, T, is set to 200. Thus, the stopping or convergence criterion is 'if Number of iterations are equal to T, then stop'. At this junction, the proposed algorithm produces the best feature set as optimal set.

After initialization, the next step is focused on finding the next probable node to visit. At this point, the ants are replaced by bees which begin to traverse the edges probabilistically. Each bee should visit all features during a traversal and construct solutions completely. The constructed solutions, then are fed to the second task, that is, performance evaluation task. This task, initially trains a Support Vector Machine (SVM) classifier using the training set of the original feature set. This trained model is then used to test the feature set solutions produced by the ACO task and the accuracy of classification is estimated. If a solution is not able to increase the classification accuracy for two consecutive steps, the algorithm completes its work and exits.

All the resulting subsets are then evaluated and the one with optimal performance is considered as the best or optimal feature subset. During the best feature set selection, the evaluated subsets are arranged in descending order of their accuracy and the first set is selected (this is the set with maximum accuracy) and which exceeds an accuracy threshold. If this condition is not reached, the pheromone is updated, a new set of ants are generated by removing the previous ants and generating new ants. The whole process is then repeated with this new set, until either the stopping criterion is met or till an optimal feature set is obtained. The pheromone on each edge is refurbished using Equation (4.49).

$$\tau_{ij}(t+1) = (1-\rho).\tau_{ij}(t) + \sum_{k=1}^{m} \Delta_{ij}^{k}(t) \qquad (4.49)$$

Where, $\Delta_{ij}^{k}(t) = \begin{cases} \gamma(S^{j}) / |S^{k}| & if\ (i, j) \in S^{k} \\ 0 & Otherwise \end{cases}$. Here, ρ, which can take up a value between 0 and 1, is a deteriorated constant and is regulated to fabricate the vaporization of pheromone. In the above equation, S_k is the feature fragment bring into being by ant k. The updation of the pheromone takes into consideration both the ant's feature subset's goodness measure (γ) and the subset size. This method allows all ants to update the pheromone.

(ii) GA-Based Feature Selection Algorithm

Another popular algorithm used to select optimal features is the usage of genetic algorithm (Abualigah and Dulaimi, 2021). The steps involved in the GA-based feature selection are presented in Figure 4.15.

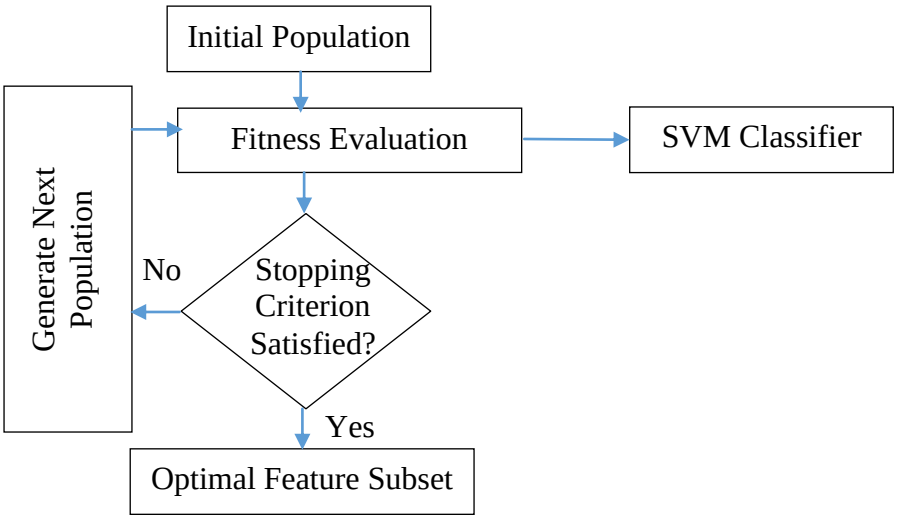

Figure 4.15 : GA-Based Feature Fusion and Selection

The first step, initialization takes care of the representation of hypotheses, where each feature subset is an individual in a population and is encoded (or represented) as n-bit binary vectors. A bit with cost 1 suggests that the corresponding characteristic is chosen and value zero ability that the characteristic is now not selected. The fitness function used is given in Equation (4.50).

$$F = w * c(x) + (1 - w) * (1/s(x)) \quad (4.50)$$

In the above equation, x is the feature subset selected, c(x) denotes the classification accuracy, s(x) denotes the size of x. Any feature selection algorithm works towards increasing the classification accuracy while minimizing the optimal feature set size. To achieve this tradeoff, a weight factor w, associated with the classification accuracy and feature subset size, is used. The fitness of x will (i) increase with increase in classification accuracy and (ii) decrease with increase in size of x. When a higher value of w is used, it indicates that the classification accuracy is given more priority. Alternatively, a low w value indicates that more penalties are given on the size of x. Thus, achieving a

mid-point, a trade-off between accuracy and optimized feature set size can be obtained. The value of w is always between 0 and 1, and this work it is set to 0.75 after empirical evaluation. In this work, the induction algorithm used is the Support Vector Machine (SVM) classifier.

The proposed GA based feature selection and fusion algorithm uses the three genetic operators, selection, crossover and mutation, for generating a new population. This work uses the Roulette wheel selection operator, which selects an individual feature subset probabilistically from a population for later breeding. Equation (4.51) presents the probability of selecting an individual n_i.

$$P(n_i) = \frac{F(n_i)}{\sum_{i=1}^{p} F(n_i)} \tag{4.51}$$

where $F(n_i)$ denotes n_i's fitness value. If the resulting probability is contiguous proportional to its intrinsic applicability and inversely proportional to the applicability of other competing hypotheses in the current population, the individual n_i will be chosen.

Mutation and crossover are the two habitual practice operators in genetic algorithms, which delineate solitaries as binary strings. Mutations work on a line, usually changing a random bit. Crossover interleaving works with two main strings to create two offsprings. In this work, the crossover operator used is the single-point operator, where a point is randomly selected, so that the first i bits are bought from one parent, whilst the last bits are contributed with the aid of any other or 2nd parent. During feature fusion and selection, each individual has a probability to mutate and is denoted as pm. In this research work, n bits to be flipped are randomly selected during each mutation stage.

(iii) Enhanced ACO-Based Feature Selection Algorithm

The conventional ACO-based feature selection algorithm described in the previous section has the advantage that it requires no heuristics to navigate during the search for optimal feature subset during the search iteration. The graph, that represents the features, is traversed by the ants to discover the optimal feature combinations. However, the issue is

the prior knowledge of the features is required. This is solved by combining the ACO-based algorithm with Genetic Algorithm.

The proposed enhanced ACO algorithm combines ACO and GA algorithms in a manner that it complements each another during the search for optimal feature vector. The algorithms have several common factors as listed below.

- Both explores and searches the feature space for optimal features.
- The performance of each feature subset is determined using an evaluation function and is normally measured using a classification output.
- The best feature subset is obtained can be used to improve the classification algorithm

The algorithm commences by procreating a set of ants (for ACO) and a population (for GA). Both algorithms give rise to feature subsets, which are stored together. In the next step, these sets are evaluated and the subset that produces the maximum evaluation measure is considered as the best subset. The algorithm stops when an optimal feature set is established or when the algorithm has been sailed through a predefined number of times. If both the above stopping conditions are not met, then all the ants rejuvenate the pheromone and the best ant deposits additional pheromones on nodes of the best solution. In this method, the best solution can be obtained either through ACO or through GA. Hence, the cross over and mutation operations of GA can be used by ACO algorithms. This change in ACO allows the ants to explore around the optimal solution in the subsequent iterations. Once the updation of pheromones is completed, the process starts the iterative procedure again. The process is presented in Figure 4.16.

4.7. CHAPTER SUMMARY

This chapter presented the two tasks in Phase I of the research methodology. The first task was feature extraction and a total of 52 features were extracted from the online reviews, reviewer and product being reviewed. The problem of curse of dimensionality was handled using a feature selection algorithm. This chapter presented the methods used to extract the features along with the feature selection algorithm. The optimal features constructed by the algorithms in this chapter, is then used to identify the spam and ham

reviews. For this an enhanced ensemble classifier is proposed. The working of this classifier is presented in the next chapter, Design of Enhanced SVM Classification Model.

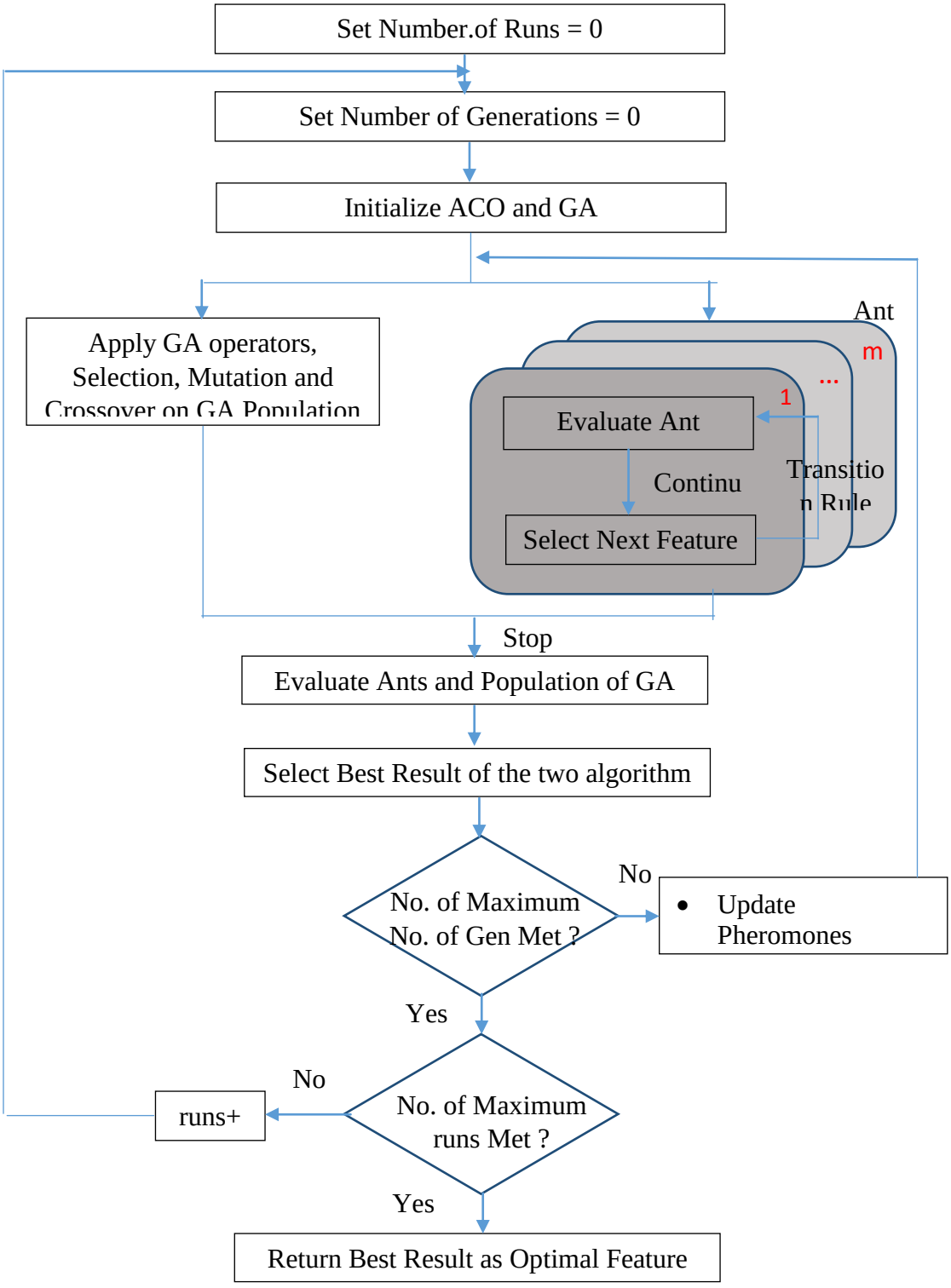

Figure 4.16 : Enhanced ACO-Based Feature Selection Algorithm