

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA  
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH



UNIVERSITY Dr. TAHAR MOULAY SAIDA  
FACULTY : TECHNOLOGY  
DEPARTEMENT : COMPUTER SCIENCE



# MASTER THESIS

Spécialité : Modélisation Informatique Des Connaissances Et  
Du Raisonnement (MICR)

---

## Tweet sentiment analysis based on Grey Wolves Optimizer (GWO)

---

Presented by:

**REZIG Abderrahmane Abedmounaim**

Supervised by:

**DR. MEKKOUI Kheiredine**

Academic Year: 2020-2021

## Abstract

Suicide has been an intractable public health problem despite advances in diagnosing and treating major mental disorders. A growing area is the development of suicide screening technologies through accessing and analyzing social media data. Previous studies have shown that youth are likely to disclose suicidal thoughts and suicidal risk factors online and on social media.

Further, online expression of distress and suicidality may not be disclosed to physicians. It is unclear to what extent such online expressions are comparable to suicidal risk as elicited by physicians. However, some studies show a correlation of suicidal thoughts expressed online with psychometrically assessed suicidal risk. Here we develop a machine learning approach based on Twitter data that predicts individual level future suicidal risk based on online social media data before any mention of suicidal thought.

The main objective of this work is to use sentiment analysis and natural language processing techniques to contribute to the detection of suicidal ideations, our work aims to provide an effective means of detecting

Our approach is based on a Uni-Gram, Bi-Gram, and Tri-Gram models and sentimental characteristics as well as a bio-inspired optimization.

**Keywords:** Social networks, Sentiment analysis, Data mining, Natural language processing, Meta-heuristics, Bio-inspiration.

## RÉSUMÉ

Le suicide est un problème de santé publique insoluble malgré les progrès réalisés dans le diagnostic et le traitement des troubles mentaux majeurs. Un domaine en pleine croissance est le développement de technologies de dépistage du suicide par l'accès et l'analyse des données des médias sociaux. Des études antérieures ont montré que les jeunes sont susceptibles de divulguer des pensées suicidaires et des facteurs de risque suicidaires en ligne et sur les médias sociaux.

De plus, l'expression en ligne de détresse et de suicidalité peut ne pas être divulguée aux médecins. Il n'est pas clair dans quelle mesure ces expressions en ligne sont comparables au risque suicidaire tel qu'il est obtenu par les médecins. Cependant, certaines études montrent une corrélation entre les pensées suicidaires exprimées en ligne et le risque suicidaire évalué de manière psychométrique. Ici, nous développons une approche d'apprentissage automatique basée sur les données de Twitter qui prédit le risque suicidaire futur au niveau individuel sur la base des données des médias sociaux en ligne avant toute mention de pensée suicidaire.

L'objectif principal de ce travail est d'utiliser l'analyse des sentiments et les techniques de traitement du langage naturel pour contribuer à la détection des idées suicidaires, notre travail vise à fournir un moyen efficace de détecter

Notre approche est basée sur un modèle Uni-Gram, Bi-Gram et Tri-Gram et des caractéristiques sentimentales ainsi qu'une optimisation bio-inspirée.

**Mots-clés:** Réseaux sociaux, Analyse des sentiments, Data mining, Traitement automatique du langage naturel, Méta-heuristiques, Bio-inspiration.

## ملخص

كان وما زال الانتحار مشكلة صحية عامة مستعصية على الحل على الرغم من التقدم في تشخيص وعلاج الاضطرابات النفسية الكبرى. ومن المجالات المتنامية تطوير تقنيات فحص الانتحار من خلال الوصول إلى بيانات وسائل التواصل الاجتماعي وتحليلها. وقد أظهرت الدراسات السابقة أن الشباب من المرجح أن يكشفوا عن الأفكار الانتحارية وعوامل الخطر الانتحارية عبر الإنترنت وعلى وسائل التواصل الاجتماعي.

علاوة على ذلك، قد لا يتم الكشف عن التعبير عبر الإنترنت عن الضيق والانتحار للأطباء. ومن غير الواضح إلى أي مدى يمكن مقارنة هذه التعبيرات عبر الإنترنت بالمخاطر الانتحارية التي يثيرها الأطباء. ومع ذلك، تظهر بعض الدراسات وجود ارتباط بين الأفكار الانتحارية التي تم التعبير عنها عبر الإنترنت والمخاطر الانتحارية التي تم تقييمها نفسياً. هنا نطور نهج التعلم الآلي استناداً إلى بيانات تويتر التي تنبأ بالمخاطر الانتحارية المستقبلية على المستوى الفردي استناداً إلى بيانات وسائل التواصل الاجتماعي عبر الإنترنت قبل أي ذكر للفكر الانتحاري.

الهدف الرئيسي من هذا العمل هو استخدام تحليل المشاعر وتقنيات معالجة اللغة الطبيعية للمساهمة في الكشف عن الأفكار الانتحارية، ويهدف عملنا إلى توفير وسيلة فعالة للكشف عنها. تعتمد طريقتنا على نموذج التمثيل الأحادي، الثنائي، والثلاثي للتغريدات، مصحوباً بخصائص عاطفية ومحسنة بتقنية تحسين مستوحاة من الطبيعة.

**الكلمات المفتاحية:** الشبكات الاجتماعية، تحليل المشاعر، تنقيب البيانات، المعالجة التلقائية للغات الطبيعية، الإرشادية العليا، الإلهام الحيوي

# Acknowledgements

First of all, I dedicate this work to the almighty God who gave me strength, power of the mind, protection, and skills and for giving us a healthy life “alhamdulillah.”

I dedicate my dissertation work to my family, professors, and my friends. I am incredibly grateful to my wonderful parents, who have been my source of inspiration and gave me strength when I thought of giving up, who continually provide moral, spiritual, emotional, and financial support. Without their love and support, this project wouldn't have been made possible—May God blessing in their lives.

My beloved grandmother and maternal aunt always encouraged me to go on every adventure, especially this one.

To my sisters, for their extremal love.

I would also like to express my special thanks of gratitude to my professor “**Dr.Kouidri Si-ham**”, for her able guidance and support in completing my project work .

Finally, my thanks and appreciation also go to my best friend “**Attaoui Zakaria**”, who have willingly helped me out with his abilities.

# Contents

<b>List of Abbreviations</b>	<b>VI</b>
<b>List of Figures</b>	<b>IX</b>
<b>List of Algorithms</b>	<b>XI</b>
<b>List of Tables</b>	<b>XII</b>
<b>General Introduction</b>	<b>1</b>
Background and problem . . . . .	1
Objective . . . . .	1
Manuscript Organization . . . . .	2
<b>1 Data Mining</b>	<b>3</b>
1.1 Introduction . . . . .	3
1.2 Data, information and knowledge . . . . .	3
1.3 Definitions of Data Mining . . . . .	4
1.4 Data Mining tasks . . . . .	5
1.4.1 Prediction . . . . .	5
1.4.2 Regression . . . . .	5
1.4.3 Clustering . . . . .	6
1.4.4 Classification . . . . .	6
1.4.5 Association . . . . .	6
1.4.6 Summarization . . . . .	6
1.4.7 Time - Series Analysis (Trend analysis) . . . . .	7
1.4.8 Data visualization . . . . .	7
1.5 Knowledge extraction process . . . . .	7
1.6 Data mining techniques . . . . .	9
1.6.1 Support vector machines (SVM) . . . . .	9
1.6.2 Neural network (back propagation) . . . . .	10
1.6.3 Decision tree . . . . .	10
1.6.4 K-nearest Neighbors . . . . .	11
1.6.5 Associative rule analysis . . . . .	12
1.6.6 Other techniques . . . . .	13
1.7 Areas where data mining is widely used . . . . .	13
1.8 Conclusion . . . . .	14
<b>2 State of the art on bio-inspired methods applied to tweet sentiment analysis</b>	<b>15</b>
2.1 Introduction . . . . .	15
2.2 Optimization . . . . .	15
2.2.1 Optimization problem . . . . .	15
2.2.2 Categories of Optimization . . . . .	16

2.2.3	Three Issues in Optimization . . . . .	17
2.2.4	P, NP, NP-Hard and NP-Complete Problems . . . . .	18
2.2.5	Optimization methods . . . . .	19
2.3	Heuristics and metaheuristics . . . . .	20
2.4	Bio-inspired Techniques . . . . .	20
2.4.1	Genetic Algorithms . . . . .	21
2.4.2	Particle Swarm Optimization . . . . .	22
2.4.3	Ant Algorithms . . . . .	23
2.4.4	Firefly Algorithm . . . . .	24
2.4.5	Artificial Bee Colony Algorithm (ABC) . . . . .	25
2.4.6	Other Algorithms . . . . .	26
2.5	Conclusion . . . . .	26
<b>3</b>	<b>Proposed Approach</b>	<b>28</b>
3.1	Grey Wolf Optimizer . . . . .	28
3.1.1	Introduction . . . . .	28
3.1.2	Inspiration . . . . .	28
3.1.3	Mathematical models of GWO . . . . .	28
3.1.4	Exploration and exploitation in GWO . . . . .	31
3.2	Support Vector Machine . . . . .	33
3.2.1	Support Vector Machine (SVM) . . . . .	33
3.2.2	Convex Duality via Lagrange Multipliers . . . . .	34
3.2.3	Kernels . . . . .	35
3.3	Proposed Approach . . . . .	35
3.4	Conclusion . . . . .	37
<b>4</b>	<b>Experiments and results</b>	<b>38</b>
4.1	Introduction . . . . .	38
4.2	Data set . . . . .	38
4.2.1	Description of the Data Set . . . . .	38
4.3	Development tools and environment . . . . .	39
4.3.1	Programming language . . . . .	39
4.3.2	why Python? . . . . .	39
4.3.3	Tools and Libraries . . . . .	40
4.3.4	Execution environment (Hardware) . . . . .	41
4.4	Implementation . . . . .	41
4.4.1	Data preprocessing . . . . .	41
4.4.2	Features extraction . . . . .	44
4.4.3	Representation of the tweet: . . . . .	44
4.4.4	Classification . . . . .	45
4.4.5	Performance Parameters (Evaluation) . . . . .	45
4.5	Experiments and results . . . . .	47
4.5.1	Classification . . . . .	47
4.5.2	Optimization (Bio-inspired model) . . . . .	54
4.6	Interpretation of results and comparative study . . . . .	57
4.7	Application (Web version) . . . . .	57
4.8	Conclusion . . . . .	61
	<b>General conclusion</b>	<b>62</b>

# List of Abbreviations

**ABC** Artificial Bee Colony.

**ACO** Ant Colony Optimization.

**AI** Artificial Intelligence.

**API** Application Programming Interface.

**AS** Ant System.

**AUC** Area Under The Curve.

**BON** Bag Of A Noun.

**BOW** Bag Of Word.

**BSD** Berkeley Software Distribution.

**CART** Classification And Regression Trees.

**CBNNC** Center-Based Nearest Neighbor Classifier.

**CFKNNC** Coarse To Fine K Nearest Neighbor Classifier.

**DHP** Direct Hashing Pruning.

**DIKW** Data Information Knowledge Wisdom.

**DM** Data Mining.

**DT** Decision Tree.

**EA** Evolutionary Algorithm.

**EC** Evolutionary Computation.

**ES** Evolutionary Strategy.

**FA** Firefly Algorithm.

**FN** False Negative.

**FP** False Positive.

**GA** Genetic Algorithm.

**GP** Genetic Programming.



**GWO** Grey Wolf Optimizer.

**HSVM** Heterogeneous Support Vector Machine.

**ID3** Iterative Dichotomiser 3.

**IDF** Inverse Document Frequency.

**KDD** Knowledge Discovery.

**KNN** K-Nearest Neighbors.

**LHS** Left-Hand-Side.

**LR** Logistic Regression.

**MCC** Matthews Correlation Coefficient.

**MIT** Massachusetts Institute Of Technology.

**ML** Machine Learning.

**MNB** Multinomial Naive Bayes.

**NB** Naive Bayes.

**NFL** Nearest Feature Line.

**NFS** Nearest Feature Space.

**NLP** Natural Language Processing.

**NLTK** Natural Language Toolkit.

**NN** Neaural Network.

**NNLC** Nearest Neighbor Line Classifier.

**NNRW** Neural Networks With Random Weights.

**NP** Nondeterministic Polynomial.

**P** Polynomial.

**PCAR** Predictability-Based Collective Class Association Rule.

**PSO** Particle Swarm Optimization.

**RHS** Right-Hand-Side.

**ROC** Receiver Operating Characteristic.

**RT** Retweet.

**SI** Swarm Intelligence.

**SQL** Structured Query Language..

**SVC** Support Vector Classifier.

**SVM** Support Vector Machine.

**TF** Term Frequency.

**TF-IDF** Term Frequency–Inverse Document Frequency.

**TN** True Negative.

**TNR** True Negative Rate.

**TP** True Positive.

**TPR** True Positive Rate.

**WHO** World Health Organization.

# List of Figures

1.1	The pyramid DIKW [3] . . . . .	4
1.2	Data Mining tasks [7] . . . . .	5
1.3	A typical knowledge discovery process [15] . . . . .	8
1.4	Steps of back propagation learning [19] . . . . .	10
1.5	Steps of KNN algorithm [22] . . . . .	12
2.1	Six categories of optimization algorithms [26] . . . . .	16
2.2	Link between classes P, NP, NP-complete and NP-difficult [28] . . . . .	19
2.3	Hierarchy of traditional optimization methods [29] . . . . .	19
2.4	Taxonomy and nomenclature of various bio inspired optimization algorithms grouped by the area of inspiration [33] . . . . .	21
3.1	Social hierarchy of grey wolves [46] . . . . .	29
3.2	How the mathematical equations allow position updating around a pivot point. Equation 3.1 mathematically models the position updating of a grey wolf ( $X(t)$ ) around a prey ( $X_p$ ). Depending on the distance between the wolf and the prey ( $D$ ), a wolf can be relocated in a circle (in a 2D space), sphere (in a 3D space), or a hypersphere (in an N-D space) around the prey ( $X_p$ ) using Equation 3.1. [47]	29
3.3	How alpha, beta, delta and omega are defined in GWO [48] . . . . .	30
3.4	Impact of A on exploration and exploitation. Note that the value of A when running GWO five times is given in this figure. It is evident that the parameter A fluctuates adaptively from the first to the last iterations, while the range is always in the interval of [-2, 2].Considering Equation 3.1, a wolf moves towards the prey when $-1 < A < 1$ . [46] . . . . .	32
3.5	(a and b) Hyper plane and support vectors [52] . . . . .	34
3.6	Flowchart of optimized SVM with GWO algorithm . . . . .	37
4.1	Tweets preprocessing phases . . . . .	42
4.2	Turning raw text into a bag of words representatio [67] . . . . .	44
4.3	Confusion matrix for our classification problem . . . . .	46
4.4	Comparison of four classification algorithms using lemmatization and TF-IDF representation . . . . .	48
4.5	Training and prediction time for the four classification algorithms using TF-IDF representation . . . . .	49
4.6	Comparison of four classification algorithms using lemmatization and BOW representation . . . . .	52
4.7	Training and prediction time for the four classification algorithms using BOW representation . . . . .	53
4.8	Convergence curve of the GWO-SVM model using Lemmatization . . . . .	55
4.9	Convergence curve of the GWO-SVM model using Stemming . . . . .	56
4.10	Word Cloud & mean of f1-score and execution time of classifiers . . . . .	58
4.11	Classification results & GWO-SVM convergences . . . . .	59

4.12 The prediction section . . . . .	60
---------------------------------------	----

# List of Algorithms

1	Pseudo code of genetic algorithms. . . . .	22
2	Pseudo code of the firefly algorithm (FA) [1] . . . . .	25
3	Pseudo code of the GWO algorithm . . . . .	32

# List of Tables

3.1	Popular kernel functions . . . . .	35
4.1	Example of the Dataset . . . . .	39
4.2	Mapping of emoticons and emojis . . . . .	43
4.3	Comparison of four classification algorithms for the suicidal ideation detection task using lemmatization and TF-IDF representation . . . . .	50
4.4	Comparison of four classification algorithms for the suicidal ideation detection task using stemming and TF-IDF representation . . . . .	50
4.5	Comparison of four classification algorithms for the suicidal ideation detection task using lemmatization and BOW representation . . . . .	51
4.6	Comparison of four classification algorithms for the suicidal ideation detection task using stemming and BOW representation . . . . .	51
4.7	Comparison of four classification algorithms for the suicidal ideation detection task using stemming and BOW representation . . . . .	54
4.8	Comparison of classification results(execution time and f1-score 'macro') for the suicidal ideation detection task using stemming and lemmatization by proposed model GWO-SVM . . . . .	54

# General Introduction

## Background and problem

The process of performing analyzes of the amount of unstructured textual data using software that can recognize concepts, patterns, and themes is known as text mining. This grants rise to new information and change the unstructured text into structured data to later deploy in the interpretation. To identify facts, relationships, and statements that generally would not have been found in the bunch of large textual data. In today's world, many data are created daily, whether through any source: media, blogs, or social networks such as Twitter, Facebook, Tumblr, Instagram, etc. This massive amount of data, comprising of feelings, can be linked to some crucial information. It is very difficult and almost impossible to manually follow such a range of data and extract meaningful information. These facts are taken out and transformed into structured data for investigation and visualization, integrated into structured data in databases or warehouses, and refined using machine learning (ML) systems. Nowadays, Twitter is one of the most admired and used online social networking sites. Twitter authorizes its users to do many activities - create profiles, post status, and share messages with others. Content on Twitter can be published via a web interface, even an SMS or mobile broadcast. It is a free streaming social media site that allows its registered users to interact with others, using 140-character statuses. About 23% of online users use Twitter. It has become a medium where all users can express their thoughts, behaviors, feelings, or intentions, ranging from positive to neutral to negative. Most people express suicidality in their tweets, and one of the prime reasons for death is suicide. According to a WHO survey, one death occurs every 20 s or 3000 per data. It is projected that by 2020, about 1.53 million people will die from suicide. Mental illness is not the only cause that induces suicide, but several other reasons, such as depression, bipolar disorder, anxiety, or schizophrenia, can also stimulate it.

“ Often, people who intend to harm themselves will make an explicit statement to their social media circle or to a specific individual in a text — they say they want to die or kill themselves, talk about shooting or cutting themselves, or wonder if things would be better if they didn't wake up. Other people may send a goodbye message that sounds permanent.”<sup>1</sup>

– Dr. Schabbing

Many suicide deaths are preventable, and therefore, it is crucial to understand how people communicate their depression and thoughts to prevent such deaths. With the help of sentimental analysis, we can help classify and understand a user's feelings about a topic of interest.

## Objective

This thesis's main objective is to design a model for individuals that run a higher risk of committing suicide by designing and implementing an automated machine learning classifier. This

---

<sup>1</sup>Dr. Schabbing, director of Psychiatric Emergency Services for OhioHealth.

---

system will comprise four machine learning classifier such as Naive Bayes, Logistic Regression, Decision Tree, and SVM (support vector machine).

In this proposal, we use a dataset of around a thousand Twitter datasets. We use a bio-inspired method based on gray wolves to improve the classification quality for identifying the risk of committing suicide. We categorize tweets into two classes: suicidal ideation, ordinary language.

## Manuscript Organization

This brief is organized into four chapters that can be summarized as follows:

**Chapter 01:** is dedicated to presenting the field of data mining and information research, and this by introducing the fundamentals.

**Chapter 02:** provides an update on meta-heuristics and bio-inspired methods for problem-solving.

**Chapter 03:** we develop a state-of-the-art analysis of feelings and the techniques of automatic processing of languages dedicated to the classification of opinions, focusing on the automatic detection of suicide risk in social media. We are exploring related work in this area and also presenting some proposed systems on this subject.

**Chapter 04:** describes the corpus and details its methods and the experiments we conducted. Finally, we present the results obtained, accompanied by an analysis and a comparative study that values our contribution. This brief concludes with a general conclusion and some perspectives



# Chapter 1

## Data Mining

### 1.1 Introduction

The development of Information Technology has generated many databases and massive data in various areas. The research in databases and information technology has given rise to an approach to store and manipulate this precious data for further decision making. Data mining is a process of extraction of useful information and patterns from huge data. It is also called as “knowledge discovery process” or data /pattern analysis. In the 1960s, statisticians used the terms “Data Fishing” or “Data Dredging.” That was to refer to what they considered the bad practice of analyzing data. The term “Data Mining” appeared around 1990 in the database community. The notion of Data mining has emerged in many environments, from academia to commercial or medical activities. Although this is an area of research that does not have such a long history, Data mining is one of the fastest-growing computer industry areas.

The aim of this chapter is to introduce and explain the fundamental aspects of data mining. These are related to: what is “Data mining”, why to use it, how to ‘exploit’ the data? We also discuss: Solvable problems data mining, process and modeling model issues,the main data mining applications, methodology and terminology used in data mining.

### 1.2 Data, information and knowledge

Data, information and knowledge all of them are very important and they are closely connected to each other. There are some main differences between these three words which make them completely disparate from each other. So, to eliminate the confusion between them. We refer to these definitions.

- **Data:** are recorded (captured and stored) symbols and signal readings [2].
  - Symbols include words (text and/or verbal), numbers, diagrams, and images (still & video), which are the building blocks of communication.
  - Signals include sensor and/or sensory readings of light, sound, smell, taste, and touch

As symbols, ‘Data is the storage of intrinsic meaning, a mere representation. The main purpose of data is to record activities or situations, to attempt to capture the true picture or real event. Therefore, all data are historical, unless used for illustration purposes, such as forecasting.

- **Information:** is a message that contains relevant meaning, implication, or input for decision and/or action. Information comes from both current (communication) and historical (processed data or ‘reconstructed picture’) sources. In essence, the purpose of information is to aid in making decisions and/or solving problems or realizing an opportunity [2].

- **knowledge:** is the (1) cognition or recognition (know-what), (2) capacity to act (know-how), and (3) understanding (know-why) that resides or is contained within the mind or in the brain. The purpose of knowledge is to better our lives. In the context of business, the purpose of knowledge is to create or increase value for the enterprise and all its stakeholders. In short, the ultimate purpose of knowledge is for value creation [2].

The Figure 1.1 shows a traditional data-information-knowledge-wisdom pyramid.

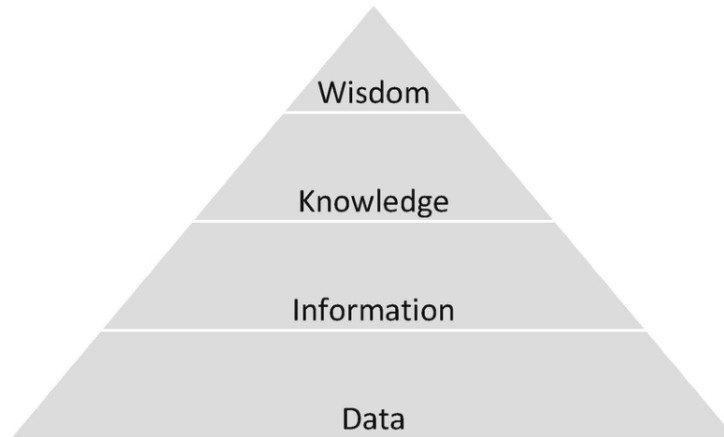


Figure 1.1: The pyramid DIKW [3]

### 1.3 Definitions of Data Mining

There are many definitions in the literature, we retain the three most important:

**definition 01:** Data mining or knowledge discovery in databases, as it is also known, is the nontrivial extraction of implicit, previously unknown and potentially useful information from the data. This encompasses a number of technical approaches, such as clustering, data summarization, classification, finding dependency networks, analyzing changes, and detecting anomalies.[4]

**definition 02:** Discovering relations that connect variables in a database is the subject of data mining. The data mining system self-learns from the previous history of the investigated system, formulating and testing hypothesis about rules which systems obey. When concise and valuable knowledge about the system of interest is discovered, it can and should be interpreted into some decision support system, which helps the manager to make wise and informed business decision.[4]

**definition 03:** Data mining is the process of discovering meaningful, new correlation patterns and trends by sifting through large amount of data stored in repositories, using pattern recognition techniques as well as statistical and mathematical techniques.[4]

If we try to analyze the previous definitions, we can note the following important aspects:

- I. DM is a non-trivial process: it is more than a simple analysis because it operates on extensive datasets.
- II. Knowledge is previously unknown: this knowledge, which can take the form of rules or models, is new and needs to be discovered.
- III. The results must be validated: the results must be evaluated and verified by an expert to avoid trivial or irrelevant results.

- IV. The results must be useful: that is to say, understandable, applicable, and must accompany and help in the decision-making process.

## 1.4 Data Mining tasks

According to their purpose of application, data mining methods are divided into two groups: prediction methods and data description methods. Description tasks aim to find interpretable patterns and associations describing the data, while prediction tasks aim to predict certain events or unknown values in the relevant sphere of interest. The main methodological difference is that the prediction requires that a specific variable (class) be included in the primary data. The solution can be numerical or categorical. Respectively, the DM methods for prediction are divided into regression and classification. A complete list of DM's tasks is not yet well established. However, and in general, a multitude of information sources [5] [6] distinguish the following tasks: Prediction, classification, Regression, Clustering, Association, Summarization, Data visualization, and Time - Series Analysis (Trend analysis).

In Figure 1.2 represents the two major categories of DM tasks: Descriptive and Predictive. In the following points, we present the main tasks that the DM is called to accomplish.

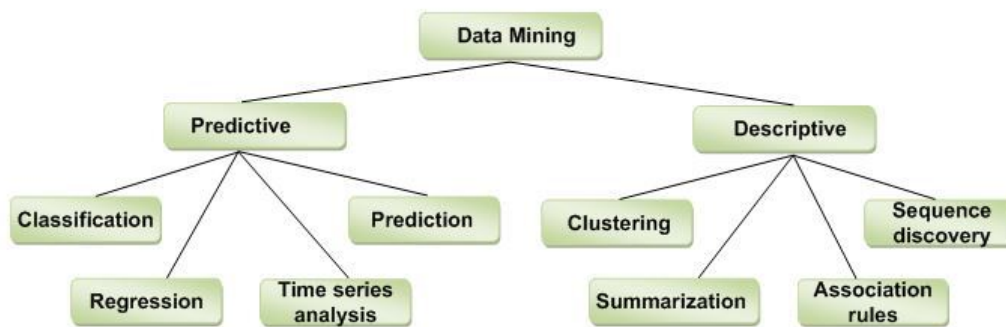


Figure 1.2: Data Mining tasks [7]

### 1.4.1 Prediction

Prediction task predicts the possible values of missing or future data. Prediction involves developing a model based on the available data and this model is used in predicting future values of a new data set of interest [8]. For example, a model can predict the income of an employee based on education, experience and other demographic factors like place of stay, gender etc. Also prediction analysis is used in different areas including medical diagnosis, fraud detection etc.

### 1.4.2 Regression

Regression is finding function with minimal error to model data. It is statistical methodology that is most often used for numeric prediction. Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer causal relationships between the independent and dependent variables. However this can lead to illusions or false relationships.[8]

### 1.4.3 Clustering

Clustering is identifying similar groups from unstructured data. Clustering is the task of grouping a set of objects in a such a way that object in same group are more similar to each other than to those in other groups.[9] Once the clusters are decided, the objects are labeled their corresponding clusters and standard features of the objects in cluster are summarized to form a class description. For example, a bank may cluster its customer into several groups based on the similarities of their income, age, sex, residence, etc. and the command characteristics of the customers in a group can be used to describe that group of customers. This will the bank to understand its customers better and thus provide customized services.

### 1.4.4 Classification

Classification is learning rules that can be applied to new data and will typically include the following steps: pre-processing of data, designing modeling, learning/feature selection, and validation /evaluation. Classification predicts categorical, continuous-valued functions. For example, we can make a classification model to categorize bank applications as either safe or risky. Classification is the derivation of a model which determines the class of an object based on its attributes. A set of objects is given as a training set in which every object is represented by vector of attributes and class. By analyzing the relationship between attributes and the class of the objects in the training set, a classification model can be constructed. Such classification models can classify future objects and develop a better understanding of the objects' classes in the database [10]. For example, from the set of loan borrowers (Name, Age, and Income) who serve as a training set, a classification model can be built, which concludes bank loan application as either safe or risky. (If age = Youth then Loan decision = risky).

### 1.4.5 Association

Association is looking for a relationship between variables or objects. It aims to extract exciting associations, correlations, or causal structures among the objects, i.e., the appearance of another set of objects. The association rules can be useful for marketing, commodity management, advertising, etc. Association rule learning is a popular and well-researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness and based on the concept of strict rules presented in , introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule Onions, potatoes burger found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, he or she is likely also to buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements. In addition to the above example from the market basket analysis, association rules are employed today in many application areas, including Web usage mining, intrusion detection, Continuous production, and bioinformatics.[9]

### 1.4.6 Summarization

Summarization is the generalization or abstraction of data. A set of relevant data is abstracted and summarized, resulting a smaller set which gives a general overview of data [11]. For example, the long distance calls of customer can be summarized into total minutes, total calls, total spending, etc. instead of detailed calls. Similarly, the calls can be summarized into local calls, STD calls<sup>1</sup>, ISD calls<sup>2</sup>, etc.

---

<sup>1</sup>(Subscriber's Trunk Dialling)

<sup>2</sup>(International Subscriber Dialling)

### 1.4.7 Time - Series Analysis (Trend analysis)

A lot of data available now are time-series data that are accumulated over time. For example, a company's sales, a customer's credit card transactions, and stock prices are all-time series data. Such data can be viewed as objects with an attribute time, and the objects are the snapshots of entities with values that change over time. It is interesting to find the patterns and regularities in the data evolutions along the dimension of time.[12]

Trend analysis discovers exciting patterns in the evolutionary history of the objects. One topic in trend analysis identifies patterns in an object's evolution, such as up, down, peak, valley, etc. A model or function is constructed to simulate the object's behaviors, which can predict future behaviors. For example, we can estimate this year's profit of a company from its last year's profit and the estimated annual increasing rate.

Another topic in trend analysis is matching the objects' changing trends, such as increasing streaks, decreasing streaks, etc. By comparing two or more objects' historical changing curves or tracks, similar and dissimilar trends can be discovered, which will help us understand the objects' behaviors. For example, a company's sales and profit figures can be analyzed to find the disagreeing trends and search for the reasons behind such disagreements.

### 1.4.8 Data visualization

Data visualization is one of the most potent forms of descriptive data mining. This refers to the task in which the discovered models are to be visualized and displayed. These representations can include: rules, tables, graphs, decision trees, etc. Although visualization is not always easy, the right picture can say a thousand words, because humans are incredibly accustomed to extracting meaning from visual scenes. For example, visualization can provide a visual representation of the location and distribution of a company's major customers on a city map. Or a province or even a country.[12]

## 1.5 Knowledge extraction process

The term KDD (Knowledge Discovery Data) first appeared in the late 1980s [13] to emphasize that knowledge is the product of a data-driven discovery process, and this is a common to different research areas focused on data analysis and knowledge extraction from different viewpoints, such as databases, statistics, mathematics, logic or artificial intelligence KDD is defined as the process

KDD is defined as the non-trivial process of identifying valid, new, potentially useful and ultimately understandable patterns in data. [14] Here, the data are a set of facts, and the 'pattern' is an expression in a language describing a subset of the data or a model applicable to the subset. Extracting a 'pattern' also refers to the fit of a model to the data. The model discovered should be valid on the new data with some degree of certainty. We also want the models to be new (at least for the system and preferably for the user) and potentially useful, that is, they lead to benefits for the user or the task. Finally, the models must be understandable. The term process implies that KDD has many steps: preparing data, finding patterns, evaluating and refining knowledge, and repeated in several iterations.

According to what we mentioned earlier, KDD refers to the overall process of discovering useful knowledge from data. It involves the evaluation and possibly the interpretation of the

models to decide on what can be qualified as to knowledge. It also includes the choice of coding schemes, preprocessing, sampling, and projections of the data before the data mining stage.

The steps involved in the KDD process present in the Figure 1.3.

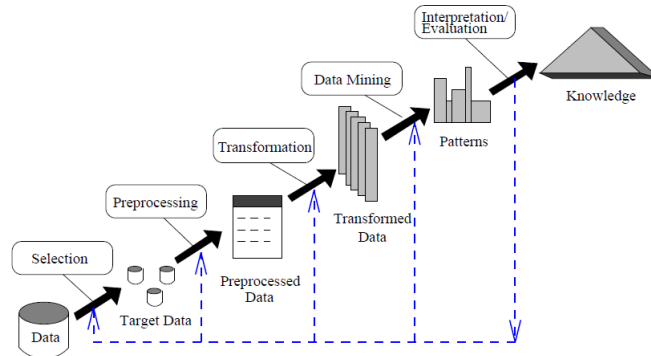


Figure 1.3: A typical knowledge discovery process [15]

The Data mining step refers to applying algorithms to extract data models without the additional steps of the KDD process.

The KDD process is interactive and iterative (with many decisions made by the user), involving five main steps, which we describe above from a practical standpoint:

- **Data Cleaning:** Data cleaning is defined as removal of noisy and irrelevant data from collection.
  - Cleaning in case of Missing values.
  - Cleaning noisy data, where noise is a random or variance error.
  - Cleaning with Data discrepancy detection and Data transformation tools.
- **Data Integration:** Data integration is defined as heterogeneous data from multiple sources combined in a common source(DataWarehouse).
  - Data integration using Data Migration tools.
  - Data integration using Data Synchronization tools.
  - Data integration using ETL(Extract-Load-Transformation) process.
- **Data Selection:** Data selection is defined as the process where data relevant to the analysis is decided and retrieved from the data collection.
  - Data selection using Neural network.
  - Data selection using Decision Trees.
  - Data selection using Naive bayes.
  - Data selection using Clustering, Regression, etc.

**Data Transformation:** Data Transformation is defined as the process of transforming data into appropriate form required by mining procedure. Data Transformation is a two step process:

- **Data Mapping:** Assigning elements from source base to destination to capture transformations.

- Code generation: Creation of the actual transformation program.

**Data Mining:** Data mining is defined as clever techniques that are applied to extract patterns potentially useful.

- Transforms task relevant data into **patterns**.
- Decides purpose of model using **classification** or **characterization**.

**Pattern Evaluation:** Pattern Evaluation is defined as as identifying strictly increasing patterns representing knowledge based on given measures.

- Find **interestingness score** of each pattern.
- Uses **summarization** and **Visualization** to make data understandable by user.

**Knowledge representation:** Knowledge representation is defined as technique which utilizes visualization tools to represent data mining results.

- Generate **reports**.
- Generate **tables**.
- Generate **discriminant rules, classification rules, characterization rules**, etc.

## 1.6 Data mining techniques

Solving a problem through data mining generally requires using a large number of different techniques and algorithms that are more or less easy to understand and use. There are many MD techniques from different scientific disciplines such as statistics, artificial intelligence, or mathematics. In this section, we present the most common data mining techniques.

### 1.6.1 Support vector machines (SVM)

Support Vector Machines, a promising method for the classification of both linear and non-linear data. It used a nonlinear mapping to transform the original training data into a higher dimension. SVM can be used for prediction as well as classification. SVM aims to find the best classification function to distinguish between members of the two classes in the training data. the best such function founded by maximizing the margin between the two classes [16], To ensure that the maximum margin hyperplanes are found, an SVM classifier attempts to maximize the following function concerning  $w$  and  $b$ :

$$l_p = \frac{1}{2} \|\vec{w}\|^2 - \sum_{i=1}^k \alpha_i y_i (\vec{w} \cdot \vec{x}_i + b) + \sum_{i=1}^t \alpha_i \quad (1.1)$$

where  $t$  is the number of training examples, and  $\alpha_i$ ,  $i = 1, \dots, t$ , are non-negative numbers such that the derivatives of  $l_p$  with respect to  $\alpha_i$  are zero.  $\alpha_i$  are the Lagrange multipliers and  $l_p$  is called the Lagrangian. In this equation, the vectors  $\vec{w}$  and constant  $b$  define the hyperplane. SVM has been successfully applied to various classification tasks that use numerical data. However, the topic of training SVM with heterogeneous data has not been fully examined. Shili and Qinghua 2015 introduced design a novel heterogeneous support vector machine (HSVM) algorithm to classify heterogeneous data. HSVM maps nominal attributes into a real space by minimizing generalization error. The main advantages of HSVM are listed as follows. 1) HSVM effectively improved the performance of SVM in dealing with nominal data or heterogeneous data. 2) HSVM improved the interpretability of decisions, and 3) HSVM was effective in learning with imbalanced data[17].

<sup>2</sup>Preprocessing of databases consists of **Data cleaning** and **Data Integration**.

### 1.6.2 Neural network (back propagation)

Neural network is a set of connected input/output units in which each connection has a weight associated with it. During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples [18]. Neural network consist of three layers input, hidden and output layer. NN include their high tolerance of noisy data as well as their ability to classify patterns on which they have not been trained. They can be used when you may have little knowledge of the relationships between attributes and classes. They are well-suited for continuous-valued inputs and outputs, unlike most decision tree algorithms. There are many kinds of neural network algorithms. The most popular is back propagation, which learns by iteratively processing a data set of training tuples, comparing the network's prediction for each tuple with the actual known target value. The weights are modified to minimize the mean squared error between the network's prediction and the actual target value. These modifications are made in the 'backwards' direction [18]. Figure 1.4 displays steps of back propagation learning algorithm.

The Figure 1.4 The figure illustrates the Schematic diagram of the backpropagation training algorithm and typical neuron mode.

Weipeng in 2018 discussed the evolution of feed-forward neural networks with random weights

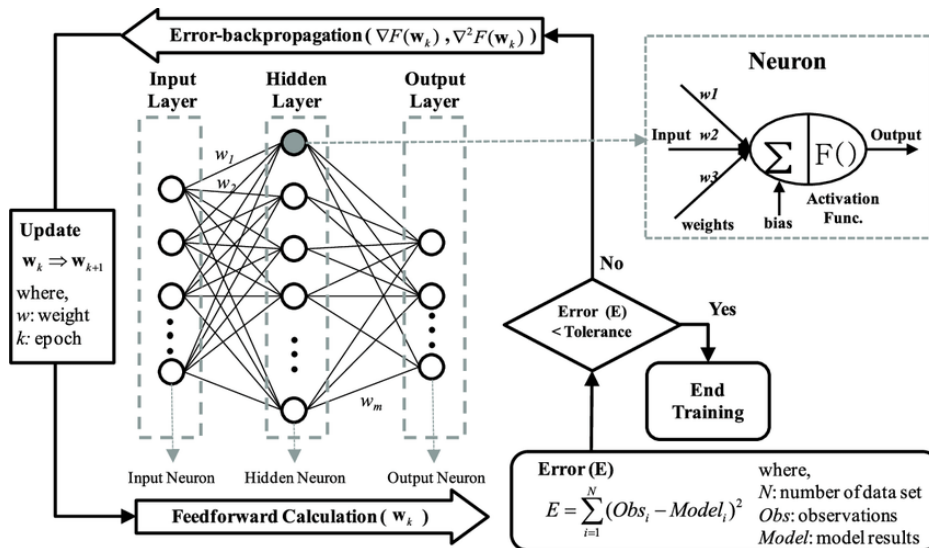


Figure 1.4: Steps of back propagation learning [19]

(NNRW), especially its deep learning applications. In NNRW, the weights and the hidden layer threshold are randomly selected, and the weights of the output layer are obtained analytically. NNRW achieved much faster-learning speed than BP-based methods[20].

### 1.6.3 Decision tree

Decision tree induction is the learning of decision trees from class-labeled training tuples. A decision tree is a flow chart-like tree structure, where each internal node (non-leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label.

A researcher in machine learning developed a decision tree algorithms known as ID3, C4.5, and CART [21]. These algorithms adopted a greedy approach in which decision trees are constructed in a top-down, divide-and-conquer manner. Most algorithms for decision tree induction also follow such a top-down approach, which starts with a training set of tuples and their associated



class labels for more details in. According to class, this algorithm used a heuristic procedure for selecting the attribute that “best” discriminates the given tuples. This procedure employs an attribute selection measure, such as information gain, Gain-ratio, and Gini index.

### Information Gain

Information gain allowing multiway splits tree. ID3 uses information gain as its attribute selection measure. This measure is based on pioneering work by Claude Shannon on information theory, which studied the value or “information content” of messages. The expected information needed to classify a tuple in  $D$  is given by

$$Info(D) = \sum_{i=1}^k p_i \log_2(p_i) \quad (1.2)$$

Where  $p_i$  is the probability that an arbitrary tuple in  $D$  belongs to class  $C_i$  and is estimated by  $\frac{|C_i, D|}{|D|}$ . A log function to the base 2 is used, because the information is encoded in bits.  $Info(D)$  is just the average amount of information needed to identify the class label of a tuple in  $D$ .  $Info(D)$  is also known as the entropy of  $D$ . to calculate the expected information after partitioning use,

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{D} * Info(D_j) \quad (1.3)$$

The term acts as the weight of the  $j$ th partition.  $info_A(D)$  is the expected information required to classify a tuple from  $D$  based on the partitioning by  $A$  with distance  $v$ . The smaller the expected information (still) required, the greater the purity of the partitions. Information gain is defined as the difference between the original information requirements and the new requirement (i.e., obtained after partitioning on  $A$ ). That is,

$$Gain(A) = Info(D) - info_A(D) \quad (1.4)$$

### Gain ratio

The information gain measure is biased toward tests with many outcomes. C4.5, a successor of ID3, uses an extension to information gain known as gain ratio, which attempts to overcome this bias. It applies a kind of normalization to information gain using a “split information” value defined analogously with  $Info(D)$  as

$$SplitInfo_A(D) = - \sum_{j=1}^v \frac{|D_j|}{D} * \log_2\left(\frac{|D_j|}{D}\right) \quad (1.5)$$

This value represents the potential information generated by splitting the training data set,  $D$ , into  $v$  partitions, corresponding to the  $v$  outcomes of a test on attribute  $A$ . the gain ratio is defined as

$$GainRatio(A) = \frac{|Gain(A)|}{SplitInfo(A)} \quad (1.6)$$

The attribute with the maximum gain ratio is selected as the splitting attribute

### 1.6.4 K-nearest Neighbors

Nearest-neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it. The purpose of the  $k$  Nearest Neighbours (kNN) algorithm is to use a database in which the data points are separated into several separate classes to predict the classification of a new sample point.

The KNN algorithm work as follow:

The training tuples are described by  $n$  attributes. Each tuple represents a point in an  $n$ -dimensional space.  $k$  is the nearest neighbors of the unknown tuples. This is shown in the following Figure 1.5 in the first starting with  $k = 1$ , we use a test set to estimate the error rate of the classifier. This process can be repeated each time by incrementing  $k$  to allow for one more neighbors. The  $k$  value that gives the minimum error rate may be selected. Second compute distance between two points or tuples  $x_1, x_2$  by Distance Metrics, such as Euclidean Distance :

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^n (x_{1i} - x_{2i})^2} \quad (1.7)$$

The steps of KNN algorithm are presented in the Figure 1.5.

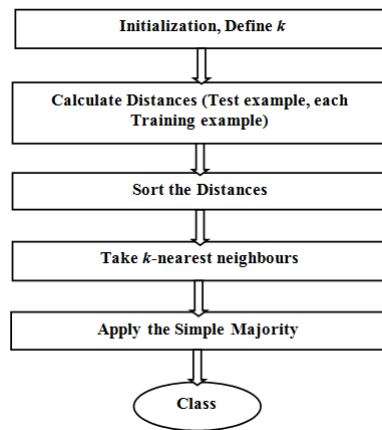


Figure 1.5: Steps of KNN algorithm [22]

To improve the performance of conventional kNN method, different methods have been proposed for gaining a higher accuracy. Yong et al, proposed a classifier called a coarse to fine K nearest neighbor classifier (CFKNNC). This classified much more accurately than various improvements such as the nearest feature line (NFL) classifier, the nearest feature space (NFS) classifier, nearest neighbor line classifier (NNLC) and center-based nearest neighbor classifier (CBNNC). And also, kNN integrated with other methods such as fuzzy set theory in , and with neural that achieved high performance.

### 1.6.5 Associative rule analysis

Associative classification is rule-based involving candidate rules as criteria of classification that provide both highly accurate and easily interpretable results to decision makers. The important phase of associative classification is rule evaluation consisting of rule ranking and pruning in which bad rules are removed to improve performance. Existing association rule mining algorithms relied on frequency-based rule evaluation methods such as support and confidence, Support determines how often a rule is applicable to agiven data set, while confidence determines how frequently items in  $Y$  (right-hand-side or RHS of rule) appear in transactions that contain  $X$  (left-hand-side or LHS). support and confidence calculated as follow :

$$Support(X \Rightarrow Y) = \frac{\sigma(X, Y)}{N} \quad (1.8)$$

$$Confidence(X \Rightarrow Y) = \frac{\sigma(X, Y)}{\sigma(X)} \quad (1.9)$$

where,  $\sigma$  is summation notation, and  $N$  represents the total number of all transactions.

Kiburm and Kichun in 2017, proposed PCAR (Predictability-based collective class association rule) algorithms that applied a new ranking rule called prediction power, the PCAR calculated the predictive power of candidate rules that removed not only redundant rules, but also rules with low predictive power. To identify interesting rule, you need to define measures and constraints such as a matrix-based visualization method [23] to present the measure computation, the distribution of interesting itemsets and the intermediate results of rule mining.

Many algorithms introduced to generate association rule such as Eclat, DHP (Direct Hashing Pruning) and others, but the most popular algorithm called apriori algorithm [23] with their improvements aprioriTID, AprioriHybrid and STEM which extended to implemented directly in SQL.

The association rule updated by using fuzzy concept lattice, When a new attribute added into the fuzzy concept lattice, it was not necessary to calculate all the frequent nodes and association rules. Therefore, the amount of calculation reduced. Another updating to association rule by using Niching Genetic Algorithms which provide a good coverage of the dataset. **Finally**, Associative classification uses association mining techniques that search for frequently occurring patterns in large databases. The patterns may generate rules, which can be analyzed for use in classification.

### 1.6.6 Other techniques

As we can see, data mining is a discipline that has seen the emergence of many works with several techniques and algorithms. We have briefly presented some of the most common and well-known techniques. However, and to close this section, it is essential to point out that other techniques and several variations of the techniques presented are widely used. Bayesian classification, K-means, Rule-based classification, etc.

## 1.7 Areas where data mining is widely used

Data mining is widely used in various fields. Following are the areas in which data mining is widely used:

- I) **Future Healthcare:** DM uses data and analytics to identify best practices that improve care and reduce costs.
- II) **Manufacturing Engineering:** can be used in system-level designing to extract the relationships between product architecture, product portfolio, and customer needs data
- III) **Financial Banking:** Data mining can donate to solving business problems in banking and finance by finding patterns, causalities, and correlations in business information
- IV) **Online Shopping:** Data mining is used to identify customers loyalty by analysing the data of customer's purchasing activities such as the data of frequency of purchase in a period.
- V) **Bioinformatics:** The mining of biological data aids to extract useful knowledge from massive datasets gathered in biology.
- VI) **Weather forecasting analysis:** The weather forecasting system uses an enormous amount of historical data for prediction. As there is a processing of enormous data, one must have to use a suitable data mining technique.
- VII) **Stock Market Analysis:** There is a vast amount of data to be analysed in the stock market. So, the data mining technique is used to model those data to do the analysis.

## 1.8 Conclusion

In this chapter, we have presented the foremost basics and concepts of Data mining. Through the different sections of this chapter, we conclude that the purpose of data mining is responsible for extracting hidden information and knowledge from these databases. The next chapter discusses meta-heuristics and bio-inspired methods.

## Chapter 2

# State of the art on bio-inspired methods applied to tweet sentiment analysis

### 2.1 Introduction

Data mining is not just about aggregating or classifying data. Optimizing data mining itself is an area of significant importance in artificial intelligence. Optimization is the process of finding the optimal solution. The three search mechanisms are analytical, enumeration, and heuristic search techniques. Analytical research is based on calculation. The search algorithms can be guided by the function's gradient, leading to a local minimum solution. Random search and enumeration are unguided search methods that enumerate the search space and exhaustively find the optimal solution. Heuristic search is a guided search that, in most cases, produces high-quality solutions. Many optimization problems are difficult to solve with an exact method, so approximate methods are used. There are two categories under which we can classify an approximate method: heuristics and metaheuristics.

Over the past decade, solving complex optimization problems with metaheuristics has received considerable attention among practitioners and researchers. As a result, many metaheuristic algorithms have been developed in recent years, and a large majority of these algorithms are often inspired by nature. Today, bio-inspired methods are becoming more and more popular. This popularity and success stems mainly from the fact that these algorithms were developed by mimicking nature's most efficient processes, including biological systems and physical and chemical processes.

In this chapter, we will talk about metaheuristics and bio-inspired algorithms in general. From their basic concepts to their applications.

### 2.2 Optimization

#### 2.2.1 Optimization problem

Optimization is the process of adjusting inputs or characteristics of a device, mathematical function or experiment to find output or minimum or maximum results. The input is made up of variables. The process or function is known as cost function, objective function or fitness function, and the result is cost or fitness. If the process is an experiment, the variables are physical inputs from the experience [24].

Optimization problems are those in which a set of unknown parameters must be found and determined in such a way that an  $f(x_i)$  function with  $i = 1, 2, \dots, n$  is minimized (or maximized),

and all constraints are met (satisfied). These problems have the following mathematical form:

$$\text{Minimize } f_i(x), \quad (i = 1, 2, 3, \dots, M) \quad (2.1)$$

$$\text{subject to the constraints } h_j = 0, \quad j = 1, 2, \dots, J \quad (2.2)$$

$$h_k(x) \leq 0, \quad k = 1, 2, \dots, K \quad (2.3)$$

$$x_i^l \leq x_i \leq x_i^u \quad (2.4)$$

- $\{x\} = \{x_1, x_2, x_3, \dots, x_n\}$  is the vector of decision variables in which  $x_i$  for  $i = 1, 2, \dots, n$  represents a decision variable (unknown parameter), these decision variables can have continuous or discrete values .
- $n$  is the total number of decision variables.

The purpose of the optimization process is to find the values of the decision variables that result in a minimum (or maximum) of the  $f(x_i)$  function in the equation Equation 2.1 called objective function, and which is used as a measure of decision effectiveness. Equality and inequality in equations Equation 2.2 and Equation 2.3 are called equality and inequality constraints, respectively, representing the limits imposed on the optimization problem that must be met (respected). Continuous decision variables have a range of variations indicated in the equation. Equation 2.4 Furthermore, it can take any value in this range [25].

### 2.2.2 Categories of Optimization

Figure 2.1 divides optimization algorithms into six categories. None of these six views or their branches are necessarily mutually exclusive. For instance, a dynamic optimization problem could be either constrained or unconstrained. Besides, some of the variables may be discrete and others continuous.

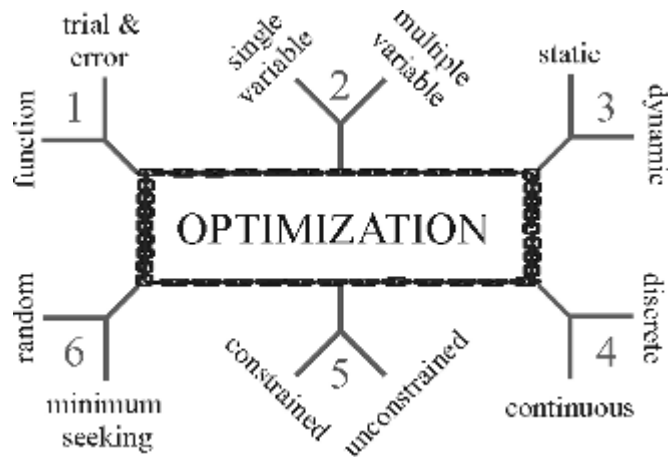


Figure 2.1: Six categories of optimization algorithms [26]

- I. Trial-and-error optimization refers to the process of adjusting variables that affect the output without knowing much about the process that produces the output. A simple example is adjusting the rabbit ears on a TV to get the best picture and audio reception. An antenna engineer can only guess at why certain contortions of the rabbit ears result in a better picture than other contortions. Experimentalists prefer this approach. Many great discoveries, like the discovery and refinement of penicillin as an antibiotic, resulted from the trial-and-error approach to optimization. In contrast, a mathematical formula describes the objective function in function optimization. Various mathematical manipulations of the function lead to the optimal solution. Theoreticians love this theoretical approach.

- II. If there is only one variable, the optimization is one-dimensional. A problem having more than one variable requires multidimensional optimization. Optimization becomes increasingly difficult as the number of dimensions increases. Many multidimensional optimization approaches generalize to a series of one-dimensional approaches.
  
- III. Dynamic optimization means that the output is a function of time, while static means that the output is independent of time. When living in the suburbs of Boston, there were several ways to drive back and forth to work. What was the best route? From a distance point of view, the problem is static, and the solution can be found using a map or the odometer of a car. In practice, this problem is not simple because of the myriad of variations in the routes. The shortest route isn't necessarily the fastest route. Finding the fastest route is a dynamic problem whose solution depends on the time of day, the weather, accidents, and so on. The static problem is difficult to solve for the best solution, but the added dimension of time increases the challenge of solving the dynamic problem.
  
- IV. Optimization can also be distinguished by either discrete or continuous variables. Discrete variables have only a finite number of possible values, whereas continuous variables have an infinite number of possible values. If we are deciding in what order to attack a series of tasks on a list, discrete optimization is employed. Discrete variable optimization is also known as combinatorial optimization, because the optimum solution consists of a certain combination of variables from the finite pool of all possible variables. However, if we are trying to find the minimum value of  $f(x)$  on a number line, it is more appropriate to view the problem as continuous.
  
- V. Variables often have limits or constraints. Constrained optimization incorporates variable equalities and inequalities into the cost function. Unconstrained optimization allows the variables to take any value. A constrained variable often converts into an unconstrained variable through a transformation of variables. Most numerical optimization routines work best with unconstrained variables. Consider the simple constrained example of minimizing  $f(x)$  over the interval  $-1 \leq x \leq 1$ . The variable converts  $x$  into an unconstrained variable  $u$  by letting  $x = \sin(u)$  and minimizing  $f(\sin(u))$  for any value of  $u$ . When constrained optimization formulates variables in terms of linear equations and linear constraints, it is called a linear program. When the cost equations or constraints are nonlinear, the problem becomes a nonlinear programming problem.
  
- VI. Some algorithms try to minimize the cost by starting from an initial set of variable values. These minimum seekers easily get stuck in local minima but tend to be fast. They are the traditional optimization algorithms and are generally based on calculus methods. Moving from one variable set to another is based on some determinant sequence of steps. On the other hand, random methods use some probabilistic calculations to find variable sets. They tend to be slower but have greater success at finding the global minimum.

### 2.2.3 Three Issues in Optimization

There are three main issues in simulation-driven optimization and modelling, and they are: the efficiency of an algorithm and the efficiency and accuracy of a numerical simulator, and assign the suitable algorithms to the right problem. Despite their importance, there is no satisfactory rule or guidelines for such issues. Obviously, we try to use the most efficient algorithms available. However, an algorithm's actual efficiency may depend on many factors such as the inner working of an algorithm, the information needed (such as objective functions and their derivatives) and implementation details.

### Efficiency of an Algorithm

An efficient optimizer is very important to ensure the optimal solutions are reachable. An optimizer's essence is a search or optimization algorithm implemented correctly to carry out the desired search (though not necessarily efficiently). It can be integrated and linked with other modelling components. There are many optimization algorithms in the literature. No single algorithm is suitable for all problems, as dictated by the No Free Lunch Theorems (Wolpert and Macready, 1997).

### The Right Algorithms?

From the optimization point of view, choosing the right optimizer or algorithm for a given problem is crucially important. The algorithm chosen for an optimization task will largely depend on the type of the problem, the nature of an algorithm, the desired quality of solutions, the available computing resource, time limit, availability of the algorithm implementation, and the expertise of the decision-makers.

Obviously, the choice is also affected by the desired solution quality and available computing resource. As in most applications, computing resources are limited, and we have to obtain good solutions (not necessarily the best) in a reasonable and practical time. Therefore, we have to balance the resource and solution quality. We cannot achieve guaranteed quality solutions, though we strive to obtain the quality solutions as best as we possibly can. If time is the primary constraint, we can use some greedy methods or hill-climbing with a few random restarts (We will talk about them in section 2.4).

### Efficiency of a Numerical Solver

A solver's efficiency is even more complicated, depending on the actual numerical methods used and the complexity of the problem of interest. There are many empirical observations for choosing the right algorithms for the right problems, but there are no agreed guidelines. There are no universally efficient algorithms for all types of problems. Therefore, the choice may depend on many factors and sometimes subjective to researchers and decision-makers personal preferences.

#### 2.2.4 P, NP, NP-Hard and NP-Complete Problems

There exist several problems, and these problems can be classified according to their hardness on various categories. These problems can be classified into four complexity classes: P, NP, NP-Complete and NP-Hard. We rely on the definitions proposed in [27]:

- **P (Polynomial) problems:** refer to problems where an algorithm would take a polynomial amount of time to solve, or where Big-O is a polynomial (i.e.  $O(1)$ ,  $O(2)$ ,  $O(n^2)$ , etc). These are problems that would be considered 'easy' to solve, and thus do not generally have immense run times.
- **NP (Non-deterministic Polynomial) Problems:** were a little harder for me to understand, but I think this is what they are. In terms of solving a NP problem, the run-time would not be polynomial. It would be something like  $O(n!)$  or something much larger. However, this class of problems can be given a specific solution, and checking the solution would have a polynomial run-time.
- **NP-Hard Problems:** A problem is classified as NP-Hard when an algorithm for solving it can be translated to solve any NP problem. Then we can say, this problem is at least as hard as any NP problem, but it could be much harder or more complex.



- **NP-Complete Problems:** are problems that live in both the NP and NP-Hard classes. This means that NP-Complete problems can be verified in polynomial time and that any NP problem can be reduced to this problem in polynomial time.

The relationship between all these classes is shown in Figure 2.2.

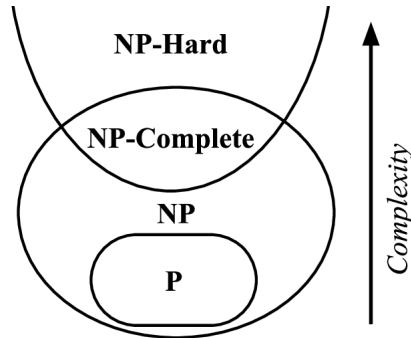


Figure 2.2: Link between classes P, NP, NP-complete and NP-difficult [28]

### 2.2.5 Optimization methods

An optimization problem can be solved either by an exact method or by an approximate method (Figure 2.3 ). This depends on the complexity of the problem.

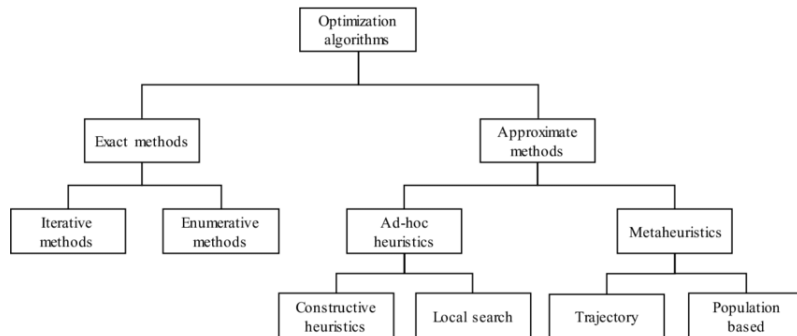


Figure 2.3: Hierarchy of traditional optimization methods [29]

#### Exact methods

The exact methods provide optimal solutions and ensure their optimality. This type of technique is often applied to small instances. For NP-complete problems, the exact algorithms are non-polynomial time algorithms (unless P=NP). This family includes a wide range of methods, including the Branch-and-X family (where the Xs represent the different variants), dynamic programming, stress programming, etc.

#### Methods approached

Approximate (or heuristic) methods generate high-quality solutions within a reasonable time frame for practical use, but there is no guarantee of finding an optimal overall solution. According to [30], in the class of methods approached, two subclasses of algorithms can be distinguished: approximation algorithms and heuristic algorithms.

- Heuristics find ‘good’ solutions on large problem instances. They provide acceptable performance at acceptable costs in a wide range of problems. In general, heuristics do not have a guarantee of approximation on the solutions obtained. Heuristics can be divided into two families: specific heuristics and metaheuristics.
- Unlike heuristics, which generally find reasonably ‘good’ solutions within a reasonable time frame, approximation algorithms provide provable solution quality and provable execution limits.

## 2.3 Heuristics and metaheuristics

Many real optimization problems are difficult to solve by accurate optimization methods due to high dimensionality, multimodality, epistasis (parameter interaction) and non-differentiation. Therefore, approximate algorithms are an alternative approach to these problems. Approximate algorithms can be broken down into heuristics and metaheuristics.

The words ‘meta’ and ‘heuristic’ both have their origins in ancient Greek: ‘meta’ means beyond (or higher level), and ‘heuristic’ refers to the art of discovering new strategies. Heuristics refers to experience-based techniques for problem-solving and learning. It gives a satisfactory solution in a reasonable calculation time, which may not be optimal. The specific heuristic depends on the problem and is designed solely for the solution of a particular problem. Examples of this method include using an empirical rule, an informed guess, intuitive judgment or even common sense. Many algorithms, either accurate algorithms or approximation algorithms, are heuristics [27].

The term metaheuristics were coined by **Glover** in 1986 [30] to refer to a set of methodologies conceptually ranked above heuristics to guide the conception of heuristics.

A metaheuristic is a higher-level procedure or heuristic designed to find, generate or select a lower-level procedure or heuristics (partial search algorithm) that can provide a good enough solution to an optimization problem. By looking for a wide range of feasible solutions, metaheuristics can often find good solutions with less computational effort than computational-based methods or simple heuristics [27].

There are two types of metaheuristics, those based on a unique solution and those population-based:

- The metaheuristics of the first type are based on a single solution at all times and include local research-based meta-devices such as simulated recruit, taboo research, iterated local research [31], guided local research [32], model research or random research etc.
- In the second type, a number of solutions are updated iteratively until the termination condition is met. Population-based metaheuristics are generally classified as evolutionary algorithms and swarm-based algorithms.

Intensification and diversification are two key components of any metaheuristic algorithm [25]. Intensification (also known as exploitation) uses local information in the search process to generate better solutions. Diversification (also known as exploration) aims to explore the research space in greater depth and generate diverse solutions.

Single-solution metaheuristics are considered more exploitation-oriented, while population-based metaheuristics are more exploration-oriented [27].

## 2.4 Bio-inspired Techniques

Metaheuristic algorithms are often nature-inspired, and they are now among the most widely used algorithms for optimization. Nature-inspired algorithms have many advantages over conventional algorithms. Metaheuristic algorithms lie in the fact that it receives its sole inspiration

from nature. They can describe and resolve complex relationships from intrinsically elementary initial conditions and rules with little or no knowledge of the search space Nature is the perfect example for optimization because if we closely examine each and every features or phenomenon in nature it always find the optimal strategy, still addressing complex interaction among organisms ranging from microorganism to fully-fledged human beings, balancing the ecosystem, maintaining diversity, adaptation, physical phenomenon like river formation, forest fire ,cloud, rain .etc. Here will introduce an overview of evolutionary algorithms, the Swarm Intelligence family’s new and emerging algorithms, and algorithms inspired from the natural ecosystem. **Evolutionary computation** (EC) is a paradigm in the artificial intelligence realm that aims

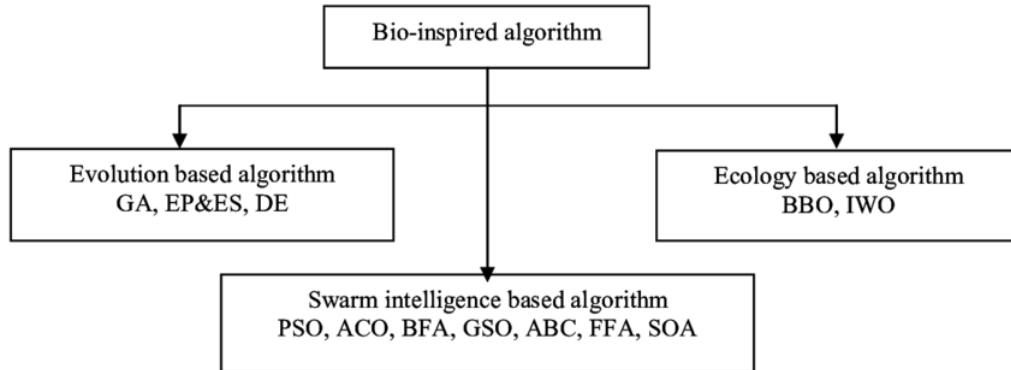


Figure 2.4: Taxonomy and nomenclature of various bio inspired optimization algorithms grouped by the area of inspiration [33]

at benefiting from collective phenomena in adaptive populations of problem solvers utilizing the iterative progress comprising growth, development, reproduction, selection, and survival as seen in a population. EAs are the most well-known, classical, and established algorithms among nature-inspired algorithms based on the biological evolution in nature that is responsible for the design of all living beings on earth and the strategies they use to interact with each other. EAs employ this powerful design philosophy to find solutions to complex problems. EAs are non-deterministic algorithms or cost-based optimization algorithms.

A family of successful EAs comprises genetic algorithm (GA), genetic programming (GP), Differential Evolution, evolutionary strategy (ES), and most recent Paddy Field Algorithm. The members of the EA family share a great number of features in common. They are all population-based stochastic search algorithms performing with best-to-survive criteria. Each algorithm commences by creating an initial population of feasible solutions and evolves iteratively from generation to generation towards the best solution. In successive iterations of the algorithm, fitness-based selection takes place within the population of solutions. Better solutions are preferentially selected for survival into the next generation of solutions.

### 2.4.1 Genetic Algorithms

The genetic algorithm (GA), developed by **John Holland** and his collaborators in the 1960s and 1970s , is a model or abstraction of biological evolution based on Charles Darwin’s theory of natural selection. Holland was probably the first to use the crossover and recombination, mutation, and selection in the study of adaptive and artificial systems. These genetic operators form the essential part of the genetic algorithm as a problem-solving strategy. Since then, many variants of genetic algorithms have been developed and applied to a wide range of optimization problems, from graph coloring to pattern recognition from discrete systems (such as the traveling salesman problem) to continuous systems (e.g., the efficient design of airfoil in aerospace engineering), and from financial markets to multiobjective engineering optimization [1]. There are many advantages of genetic algorithms over traditional optimization algorithms. Two of the

most notable are. The ability to deal with complex problems and parallelism. Genetic algorithms can deal with various types of optimization, whether the objective (fitness) function is stationary or nonstationary (changes with time), linear or nonlinear, continuous or discontinuous, or with random noise. Because multiple offsprings in a population act like independent agents, the population (or any subgroup) can explore the search space in many directions simultaneously. This feature makes it ideal for parallelizing the algorithms for implementation. Different parameters and even different groups of encoded strings can be manipulated at the same time [34]. However, genetic algorithms also have some disadvantages. The formulation of a fitness function, the use of population size, the choice of important parameters such as the rate of mutation and crossover, and the selection criteria of the new population should be carried out carefully. Any inappropriate choice will make it difficult for the algorithm to converge or it will simply produce meaningless results. Despite these drawbacks, genetic algorithms remain one of the most widely used optimization algorithms in modern nonlinear optimization. The steps of GA can be represented schematically as the pseudo-code of genetic algorithms shown in algorithm 1.

```

1 Encode the solutions into chromosomes (strings)
2 Define fitness F
3 Generate the initial population
4 Initialize the probabilities of crossover ( $p_c$ ) and mutation ( $p_m$ )
5 while  $t < Max\ number\ of\ generations$  do
6   | Generate new solution by crossover and mutation
7   | Crossover with a crossover probability  $p_c$ 
8   | Mutate with a mutation probability  $p_m$ 
9   | Accept the new solutions if their fitness increase
10  | Select the current best for the next generation (elitism)
11  | Update  $t = t + 1$ 
12 end
13 Decode the results and visualization

```

**Algorithm 1:** Pseudo code of genetic algorithms.

Iqbal, Farkhund, and Hashmi [35] propose an integrated framework that bridges the gap between lexicon-based and machine learning approaches to achieve better accuracy and scalability. A novel genetic algorithm (GA)-based feature reduction technique is proposed to solve the scalability issue that arises as the feature-set grows. Using this hybrid approach can reduce the feature-set size by up to 42

### 2.4.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a computational intelligence oriented, stochastic, population-based global optimization technique proposed by Kennedy and Eberhart in 1995[36]. It is inspired by the social behavior of bird flocking searching for food. PSO has been extensively applied to many engineering optimization areas due to its unique searching mechanism, simple concept, computational efficiency, and easy implementation. In PSO, the term ‘particles’ refers to population members which are mass-less and volume-less (or with an arbitrarily small mass or volume) and are subject to velocities and accelerations towards a better mode of behavior. Each particle in the swarm represents a solution in a high-dimensional space with four vectors, its current position, best position found so far, the best position found by its neighborhood so far and its velocity and adjusts its position in the search space based on the best position reached by itself (pbest) and on the best position reached by its neighborhood (gbest) during the search process.

**Steps in PSO algorithm can be briefed as below:**

- Initialize the swarm by assigning a random position in the problem space to each particle.

- Evaluate the fitness function for each particle.
- For each individual particle, compare the particle's fitness value with its pbest . If the current value is better than the pbest value, then set this value as the pbest and the current particle's position,  $X_i$ , as  $p_i$ .
- Identify the particle that has the best fitness value. The value of its fitness function is identified as gbest and its position as pg.
- Update the velocities and positions of all the particles using (1) and (2).
- Repeat steps 2–5 until a stopping criterion is met (e.g., maximum number of iterations or a sufficiently good fitness value).

**Advantages over Genetic Algorithm:**

- PSO is easier to implement and there are fewer parameters to adjust.
- PSO has a more effective memory capability than GA.
- PSO is more efficient in maintaining the diversity of the swarm, since all the particles use the information related to the most successful particle in order to improve themselves, whereas in Genetic algorithm, the worse solutions are discarded and only the new ones are saved, i.e. in GA the population evolve around a subset of the best individuals.

There are many similarities between the PSO and EAs. Both of them initialize solutions and update generations, while the PSO has no evolution operators, as does the latter. In a PSO, particles try to reach the optimum by following the current global optimum instead of using evolutionary operators, such as mutation and crossover.

It is claimed that the PSO, in addition to continuous functions, has been showing stability and convergence in a multidimensional complex space also [37].

Li et al. [38] developed a global optimization-based approach called PSOGO-Senti to improve SA by optimizing the feature selection process. Moreover, Support Vector Machine (SVM) was also applied for classification. The proposed algorithm was compared with the Genetic Algorithm (GA) and grid search algorithm. The outcome indicated the efficacy of the algorithm for multi-polarity SA. Moreover, five-polarity sentiment classification gave the most considerable improvements.

Basari et al. [39] presented an opinion mining scheme using hybridization of SVM and PSO. SVM technique analyzes and recognizes the patterns in Twitter data utilized to classify the data as a positive class or negative class. In hybridized SVM-PSO, PSO is used to improve the parameters of SVM. The accuracy level of SVM-PSO still needs improvement.

**2.4.3 Ant Algorithms**

ACO is among the most successful swarm-based algorithms proposed by **Dorigo & Di Caro** in [40]. It is a metaheuristic inspired by the foraging behavior of ants in the wild and the phenomena known as stigmergy, a term introduced by **Grasse** in 1959. Stigmergy refers to the indirect communication amongst a self-organizing emergent system via individuals modifying their local environment. The most interesting aspect of the collaborative behavior of several ant species is their ability to find the shortest paths between the ants' nest and the food sources by tracing pheromone trails. Then, ants choose the path to follow by a probabilistic decision biased

by the amount of pheromone: the more substantial the pheromone trail, the higher its desirability. Because ants, in turn, deposit pheromone on the path they are following, this behavior results in a self-reinforcing process leading to the formation of paths marked by high pheromone concentration. By modeling and simulating ant foraging behavior, brood sorting, nest building, self-assembling, etc., algorithms can be developed to be used for complex, combinatorial optimization problems.

The first ant algorithm, named “Ant System” (AS), was developed in the nineties by **Dorigo et al.**(1996) and tested successfully on the well-known benchmark Travelling Salesman Problem. The ACO metaheuristic was developed to generalize the overall method of solving combinatorial problems by approximate solutions based on the generic behavior of natural ants. ACO is structured into three main functions as follows:

- 1) **Ant Solutions Construct:** This function performs the solution construction process where the artificial ants move through adjacent states of a problem according to a transition rule, iteratively building solutions.
- 2) **Pheromone Update:** performs pheromone trail updates. This may involve updating the pheromone trails once complete solutions have been built or updating after each iteration. In addition to pheromone trail reinforcement, ACO also includes pheromone trail evaporation. Evaporation of the pheromone trails helps ants to ‘forget’ bad solutions learned early in the algorithm run.
- 3) **Daemon Actions:** is an optional step in the algorithm which involves applying additional updates from a global perspective (for this, no natural counterpart exists). This may include applying additional pheromone reinforcement to the best solution generated (known as offline pheromone trail update).

An alternative approach, called the ant colony system (ACS) has been introduced by **Dorigo** and **Gambardella** in 1997 to improve the performance of ant system. It is based on four modifications of ant system: a different transition rule, a different pheromone trail update rule, the use of local updates of pheromone trail to favor exploration, and the use of candidate list to restrict the choice [34].

#### 2.4.4 Firefly Algorithm

The firefly algorithm proposed by Yang [41] can be considered an unconventional swarm-based heuristic algorithm for constrained optimization tasks inspired by the flashing behavior of fireflies. The algorithm constitutes a population-based iterative procedure with numerous agents (perceived as fireflies) concurrently solving a considered optimization problem. Agents communicate with each other via bioluminescent glowing, enabling them to explore cost function space more effectively than in standard distributed random search. The intelligence optimization technique is based on the assumption that the solution of an optimization problem can be perceived as an agent (firefly), which glows proportionally to its quality in a considered problem setting. Consequently, each brighter firefly attracts its partners (regardless of their sex), which makes the search space being explored more efficient.

The firefly algorithm has three particular idealized rules which are based on some of the basic flashing characteristics of real fireflies. They are the following:

- All fireflies are unisex, so one firefly will be attracted to other fireflies regardless of their sex.
- Attractiveness is proportional to a firefly’s brightness. Thus for any two flashing fireflies, the less brighter one will move toward the brighter one. The attractiveness is proportional

to the brightness, both of which decrease as their distance increases. If there is no brighter one than a particular firefly, it will move randomly.

- The brightness of a firefly is affected or determined by the landscape of the objective function.

The basic steps of the FA can be summarized as the pseudo code shown in algorithm 2.

```

1 Generate an initial population of n fireflies  $x_i(i = 1, 2, \dots, n)$ .
2 Light intensity  $I_i$  at  $x_i$  is determined by  $f(x_i)$ .
3 Define light absorption coefficient  $\gamma$ .
4 while  $t < MaxGeneration$  do
5     for  $i = 1 : n$  (all n fireflies) do
6         for  $j = 1 : n$  (all n fireflies) (inner loop) do
7             if  $I_i < I_j$  then
8                 | Move firefly i towards j.
9             end
10            Evaluate new solutions and update light intensity.
11            Vary attractiveness with distance r via  $exp[-\gamma r^2]$ .
12        end
13    end
14    Rank the fireflies and find the current global best  $g^*$ .
15 end
16 Postprocess results and visualization.

```

**Algorithm 2:** Pseudo code of the firefly algorithm (FA) [1]

Yang et al. [42] presented a study regarding the application and recent advancement in the firefly algorithm. First, they discussed the efficiency of the firefly algorithm. One of the reasons for its efficiency is the automatic subdivision in which the whole population is subdivided into various groups that swarm around to get the best solution. Also, fireflies can find all the optima simultaneously. Hence, his subdivision makes FFA fit for multimodal optimization problems. Second, they discussed that the firefly algorithm could be used to provide an equilibrium between the component's exploration and exploitation, which was necessary for the efficiency of the metaheuristic algorithm. Finally, they also discussed various firefly algorithm applications, including digital image compression, solving engineering design problems, NP-hard scheduling problems, performing classification and clustering, and optimization.

### 2.4.5 Artificial Bee Colony Algorithm (ABC)

Based on the behavior of the bees in nature, various swarm intelligence algorithms are available. These algorithms are classified into two: foraging behavior and mating behavior. Examples of algorithms simulating the foraging behavior of the bees include the Artificial Bee Colony (ABC), the Virtual Bee algorithm proposed by Yang, the Bee Colony Optimization algorithm proposed by **Teodorovic** and **Dell'Orco**, the BeeHive algorithm proposed by **Wedde et al.** The Bee Swarm Optimization algorithm proposed by Dries et al. and the Bees algorithm proposed by Pham et al. An individual entity (e.g., a bee in a bee colony) exhibit a simple set of behavior policies (e.g., migration, replication, death). However, a group of entities (e.g., a bee colony) shows complex emergent behavior with valuable properties such as scalability and adaptability [34].

Artificial Bee Colony is a predominant algorithm simulating the intelligent foraging behavior of a honeybee swarm, proposed by Karaboga and Basturk [43]. In the ABC algorithm, the colony of artificial bees contains three bees: employed bees, onlookers, and scouts.

A bee waiting in the dance area to choose a food source is called an onlooker, and one going to the food source visited by it before is named employed bee. The other kind of bee is the scout

bee that carries out a random search for discovering new sources. The position of a food source represents a possible solution to the optimization problem, and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. A swarm of virtual bees is generated and started to move randomly in a two-dimensional search space. Bees interact when they find some target nectar, and the solution of the problem is obtained from the intensity of these bee interactions.

A randomly distributed initial population solutions ( $x_i = 1, 2, D$ ) is being dispreaded over the  $D$  dimensional problem space. An employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) and tests the nectar amount (fitness value) of the new source (new solution). Provided that the nectar amount of the new one is higher than that of the previous one, the bee memorizes the new position and forgets the old one. After all employed bees complete the search process, they share the nectar information of the food sources and their position information with the onlooker bees in the dance area.

In the next phase, Reproduction, based on the probability value associated with the food source,  $P_i$ , the artificial onlooker bee, chooses a food source  $P_i = \frac{fit}{\sum_{n=1}^N fit_n}$  Where  $N$  is the number of food sources (that is the number of employed bees),  $fit_i$  is the fitness value of the solution  $i$ , proportional to the nectar amount of the food source in position  $i$ .

In the last phase, Replacement of bee and Selection, if a position can not be improved further through a predetermined number of cycles, then that food source is assumed to be abandoned. The value of a predetermined number of cycles is an important control parameter of the ABC algorithm called 'limit' for abandonment. After each candidate source position is produced and then evaluated by the artificial bee, its performance is compared with its old one. If the new food has an equal or better nectar than the old source, it replaces the old one in the memory. Otherwise, the old one is retained in the memory [34].

The local search performance of the ABC algorithm depends on neighborhood search and greedy selection mechanisms performed by employed and onlooker bees. The global search performance of the algorithm depends on random search process performed by scouts and neighbor solution production mechanism performed by employed and onlooker bees.

Ruby Dhurve, M. Seth [44] propose improving the classification methods and detecting the polarity of reviews using a machine learning approach. The objective is to select the best feature selection methods for sentiment analysis. Bag of a noun, bag of words, stop word removal, stemmer are used for feature selection. ABC algorithm is used to classify text into three classes: negative, positive, and neutral. The main aim of the thesis is to compute the result of the SVM and ABC classifier. As a result, nature inspired ABC classifier BON to give better results than the BOW, SVM with BON and BOW.

### 2.4.6 Other Algorithms

The literature is expanding rapidly, and the number of nature-inspired algorithms has increased dramatically. The article by Xin-She Yang. indicated that there are more than 100 nature-inspired algorithms.[45]

## 2.5 Conclusion

Discrete and combinatorial optimization problems in the real world are generally difficult and require intensive computational algorithms. Bio-inspired algorithms have been an interesting alternative to provide satisfactory solutions. In this chapter, we gave an overview of optimization problems and resolution methods. We presented metaheuristics and bio-inspired techniques in



general. In the next chapter, we will present an approach inspired by grey wolves that is the GWO algorithm.

## Chapter 3

# Proposed Approach

### 3.1 Grey Wolf Optimizer

#### 3.1.1 Introduction

The Grey Wolf Optimizer, which is developed by **Mirjalili et al** in 2014, the GWO algorithm is inspired by grey wolves in nature that search for the optimal way for hunting preys. Is one of the recent swarm intelligence algorithms that has attracted many researchers' attention in different optimisation areas. There are many features that distinguish GWO from other SI algorithms. It has few parameters to adjust, a good balance between exploration and exploitation can be achieved in a simple way, and a favourable convergence can be achieved. Also, GWO is simple, easy to use, flexible and scalable.

This section presents the GWO algorithm and describes its main components. It also includes discussions on the exploration, exploitation and convergence of this algorithm.

#### 3.1.2 Inspiration

GWO is a swarm intelligence technique. The inspiration of the GWO algorithm is the social intelligence of grey wolf packs in leadership and hunting. In each pack of grey wolves, there is a common social hierarchy that dictates power and domination (Figure 3.1) The most powerful wolf is alpha, which leads the entire pack in hunting, migration and feeding. When the alpha wolf is not in the pack, ill, or dead, the strongest wolf from the beta wolf takes the pack's lead. The power and domination of delta and omega are less than alpha and beta, as shown in Figure 3.1. This social intelligence is the main inspiration of the GWO algorithm. Another inspiration is the hunting approach of grey wolves. When hunting prey, grey wolves follow a set of efficient steps: chasing, encircling, harassing and attacking. That allows them to hunt big preys.

#### 3.1.3 Mathematical models of GWO

##### Encircling prey

As mentioned above, the first step of hunting is to chase and encircle. To mathematically model this, GWO considers two points in an n-dimensional space and updates the location of one of them based on that of another. The following equation has been proposed to simulate this:

$$X(t+1) = X(t) - A \cdot D \tag{3.1}$$

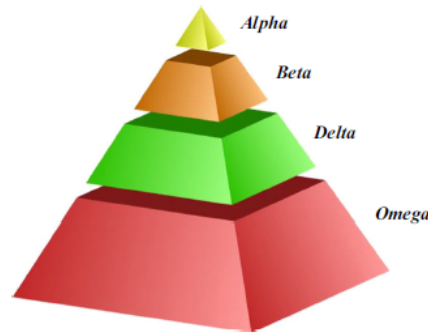


Figure 3.1: Social hierarchy of grey wolves [46]

where  $X(t+1)$  is the next location of the wolf,  $X(t)$  is current location,  $A$  is a coefficient matrix and  $D$  is a vector that depends on the location of the prey  $X_p$  and is calculated as follows:

$$D = |C \cdot X_p(t) - X(t)| \quad (3.2)$$

where  $C = 2 \cdot r_2$

$r_2$  is a randomly generated vector from the interval  $[0,1]$ . With these two equations, a solution can relocate around another solution. Note that the equations use vectors, so this is applied to any number of dimension. An example of a grey wolf's possible positions concerning a prey is shown in Figure 3.2.

The random components in equation simulate different step sizes and movement speeds of grey wolves. The equations to define their values are as follows:

$$A = 2a \cdot r_1 - a \quad (3.3)$$

where  $a$  is a vector where its values are linearly decreased from 2 to 0 during the course of run.  $r_1$  is a randomly generated vector from the interval  $[0,1]$ . The equation to update the parameter  $a$  is as follows:

$$a = 2 - t \left( \frac{2}{T} \right) \quad (3.4)$$

where  $t$  shows the current iteration and  $T$  is the maximum number of iterations.

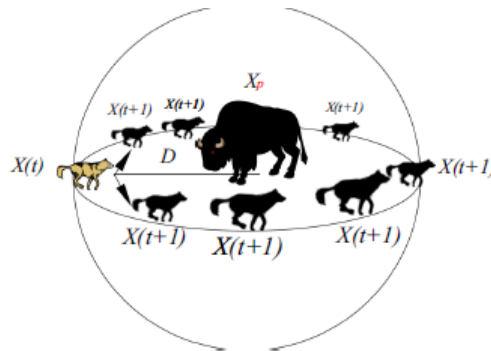


Figure 3.2: How the mathematical equations allow position updating around a pivot point. Equation 3.1 mathematically models the position updating of a grey wolf ( $X(t)$ ) around a prey ( $X_p$ ). Depending on the distance between the wolf and the prey ( $D$ ), a wolf can be relocated in a circle (in a 2D space), sphere (in a 3D space), or a hypersphere (in an N-D space) around the prey ( $X_p$ ) using Equation 3.1. [47]

### Hunt

With the equations presented above, a wolf can relocate to any points in a hypersphere around the prey. However, this is not enough to simulate the social intelligence of grey wolves. It was discussed above that social hierarchy plays a crucial role in the hunt and the survival of the packs. To simulate social hierarchy, the three best solutions are considered to be alpha, beta and delta. Although in nature, there might be more than one wolf in each category, it is considered that there is only one solution belong to each class in GWO for the sake of simplicity.

The concepts of alpha, beta, delta and omega are illustrated in Figure 3.3. Note that the objective is to find the minimum in this search landscape. It may be seen in this figure that alpha is the closest solution to the minimum, followed by beta and delta. The rest of the solutions are considered omega wolves. There is just one omega wolf in Figure 3.3, but there can be more.

In GWO, it is assumed that alpha, beta and delta are always the three best solutions obtained so far. The global optimum of optimization problems is unknown, so it has been assumed that alpha, beta and delta have a good idea of their location, which is reasonable because they are the best solutions for the entire population. Therefore, other wolves should be obliged to update their positions as follows:

$$X(t + 1) = \frac{X_1 + X_2 + X_3}{3} \tag{3.5}$$

where  $X_1$  and  $X_2$  and  $X_3$  are calculated with Equation 3.6.

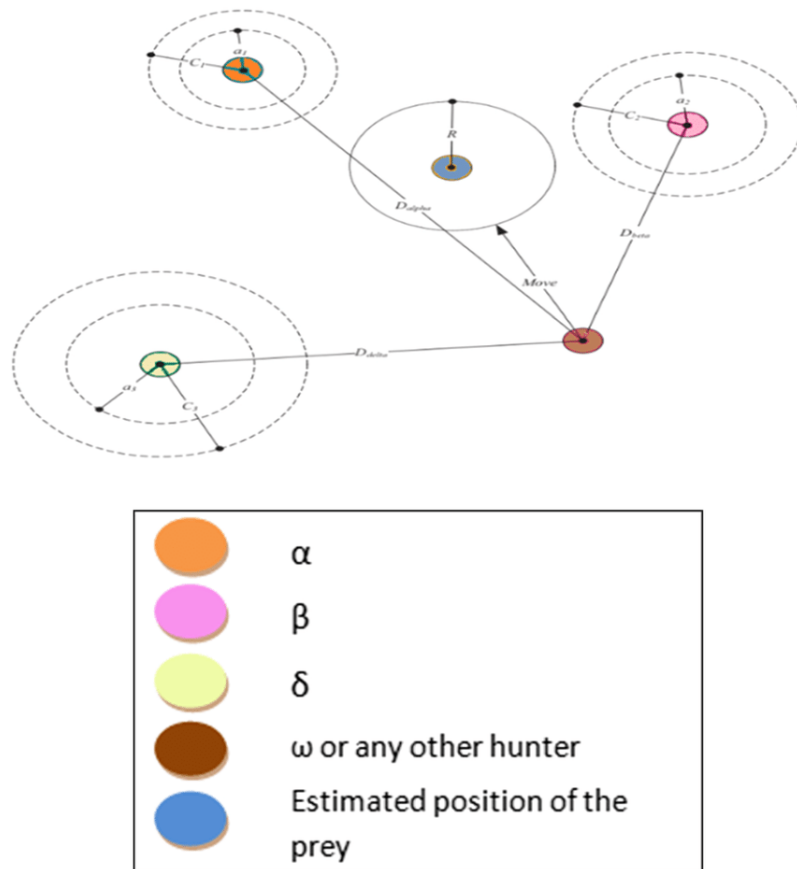


Figure 3.3: How alpha, beta, delta and omega are defined in GWO [48]

$$\begin{aligned}
 X_1 &= X_\alpha(t) - A_1 \cdot D_\alpha \\
 X_2 &= X_\beta(t) - A_2 \cdot D_\beta \\
 X_3 &= X_\delta(t) - A_3 \cdot D_\delta
 \end{aligned}
 \tag{3.6}$$

$D_\alpha$ ,  $D_\beta$  and  $D_\delta$  are calculated using Equation 3.7.

$$\begin{aligned}
 D_\alpha &= |C_1 \cdot X_\alpha - X| \\
 D_\beta &= |C_2 \cdot X_\beta - X| \\
 D_\delta &= |C_3 \cdot X_\delta - X|
 \end{aligned}
 \tag{3.7}$$

Exploration and exploitation are two conflicting processes [49] that an algorithm might show when optimizing a given problem. In the exploration process, the algorithm tries to discover new parts of the problem search space by applying sudden changes in the solutions since the main the objective is to discover the promising areas of the search landscape and prevent solutions from stagnating in a local optimum.

### 3.1.4 Exploration and exploitation in GWO

In exploitation, the main objective is to improve the estimated solutions achieved in the exploration process by discovering the neighbourhood of each solution. Therefore, gradual changes in the solutions should be made to converge towards the global optimum. The main challenge here is that exploration and exploitation are in conflict. Therefore, an algorithm should be able to address and balance these conflicting behaviour during optimization to find an accurate estimation of the global optimum for a given problem.

- The main controlling parameter of GWO to promote exploration is the variable  $C$ . This parameter always returns a random value in the interval of  $[0, 2]$ . It changes the contribution of the prey in defining the next position. This contribution is strong when  $C > 1$ , the solution gravitates more towards the prey. Since this parameters provides random values regardless of the iteration number, exploration is emphasized during optimization in case of any local optima stagnation.
- Another controlling parameter that causes exploration is  $A$ . The value of this parameter is defined based on  $a$ , which linearly decreases from 2 to 0. Due to the random components in this parameter, the range changes in the interval of  $[-2, 2]$  for the parameter  $A$ . Exploration is promoted when  $A > 1$  or  $A < -1$ , whereas there is emphasize on exploitation when  $-1 < A < 1$ .

As mentioned above, a good balance between exploration and exploitation is required to find an accurate approximation of the global optimum using stochastic algorithms. This balance is done in GWO with the decreasing behavior of the parameter  $a$  in the equation for the parameter  $A$ . This is illustrated in Figure 3.4 This figure shows five line charts calculated for  $A$ . It may be seen that although this parameter is stochastic, it results in exploration in the first half of the iterations and then exploitation in the second half. The GWO pseudocode is represented in the algorithm algorithm 3.

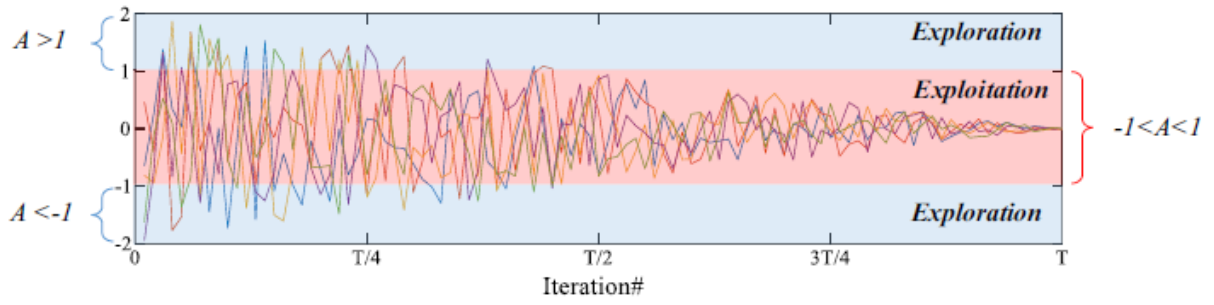


Figure 3.4: Impact of  $A$  on exploration and exploitation. Note that the value of  $A$  when running GWO five times is given in this figure. It is evident that the parameter  $A$  fluctuates adaptively from the first to the last iterations, while the range is always in the interval of  $[-2, 2]$ . Considering Equation 3.1, a wolf moves towards the prey when  $-1 < A < 1$ . [46]

```

1 Initialize the grey wolf population  $X_i(i = 1, 2, \dots, n)$ 
2 Initialize  $a$ ,  $A$ , and  $C$ 
3 Calculate the fitness of each search agent
4  $X_\alpha \leftarrow$  the best search agent
5  $X_\beta \leftarrow$  the second best search agent
6  $X_\delta \leftarrow$  the third best search agent
7 while  $t < \text{Max number of iterations}$  do
8   for each search agent do
9     | Update the position of the current search agent by Equation 3.5
10  end
11  Update  $a$ ,  $A$ , and  $C$ 
12  Calculate the fitness of all search agents
13  Update  $X_\alpha$ ,  $X_\beta$ , and  $X_\delta$ 
14   $t = t + 1$ 
15 end
16 return  $X_\alpha$ 

```

**Algorithm 3:** Pseudo code of the GWO algorithm

## 3.2 Support Vector Machine

Support vector machine (SVM) is an efficient tool for data mining and classification [50, 16]. Due to the vast volumes of data in business, primarily e-commerce, efficient use of data mining techniques becomes a necessity. SVM can also be considered as an optimization tool, as its objective is to maximize the separation margins between data sets. The proper combination of SVM with metaheuristics could be advantageous.

### 3.2.1 Support Vector Machine (SVM)

SVM is defined as a supervised learning algorithm which can solve linear and nonlinear two class binary classification problems for class 2. The aim of SVM is to separate class data by means of maximal margin hyper plane. Thus, the SVM guarantees to maximize the distance between data that are closest to the separation plane [51]. If the input data can be separated linearly, the separation hyper plane can be given in Equation 3.10:

$$f(x) = w^T x + b \quad \text{where} \quad w^T = \frac{w}{\|w\|} \quad (3.8)$$

Where  $w$  is the  $n$ -dimensional weight vector and  $b$  is the scalar multiplier or bias value. This equation finds a maximum margin to separate positive class from negative class.

When training the classifier, we want to ensure that the examples with positive labels are on the positive side of the hyperplane, i.e.,

$$w^T x_n + b \geq 1 - \xi_i \quad \text{when} \quad y_i = +1 \quad (3.9)$$

and the examples with negative labels are on the negative side, i.e.,

$$w^T x_n + b \geq -1 + \xi_i \quad \text{when} \quad y_i = -1 \quad (3.10)$$

These two conditions are often presented in a single equation presented in Equation 3.11.

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i \quad i = 1, \dots, n \quad (3.11)$$

The testing data is evaluated according to Equation 3.11. If one testing example satisfies this condition, it is assumed that this example is classified correctly. In Figure 3.5, squares and rings show two-class data, and stars show support vectors closest to the separation hyperplane. The margin at Figure 3.5(b) has a shorter distance than Figure 3.5(a). Figure 3.5(a), dashed separation line provides good classification performance while facing previously unseen data. Classifier with a smaller margin will have a higher expected risk. If the sample data is not separated linearly, some slack variables are introduced. Hence, to construct a maximal margin classifier (the distance between the two planes).

$$\max \rho = \frac{2}{\|w\|} \quad (3.12)$$

Such maximization of  $\rho$  is equivalent to the minimization of  $\|w\|$  or, more conveniently  $\|w\|^2$ . From the optimization point of view, the maximization of margins can be written as

$$\min \frac{1}{2} \|w\|^2 \quad (3.13)$$

This essentially becomes an **optimization problem**

$$\begin{aligned} \min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \\ \text{subject to} \quad y_i(w \cdot x_i + b) \geq 1 - \xi_i \\ \xi_i \geq 0, \quad (i = 1, \dots, n) \end{aligned} \quad (3.14)$$

Where  $C > 0$  is a parameter to be chosen appropriately. Here, the term  $\sum_{i=1}^n \xi_i$  is essentially a measure of the upper bound of the number of misclassifications on the training data.

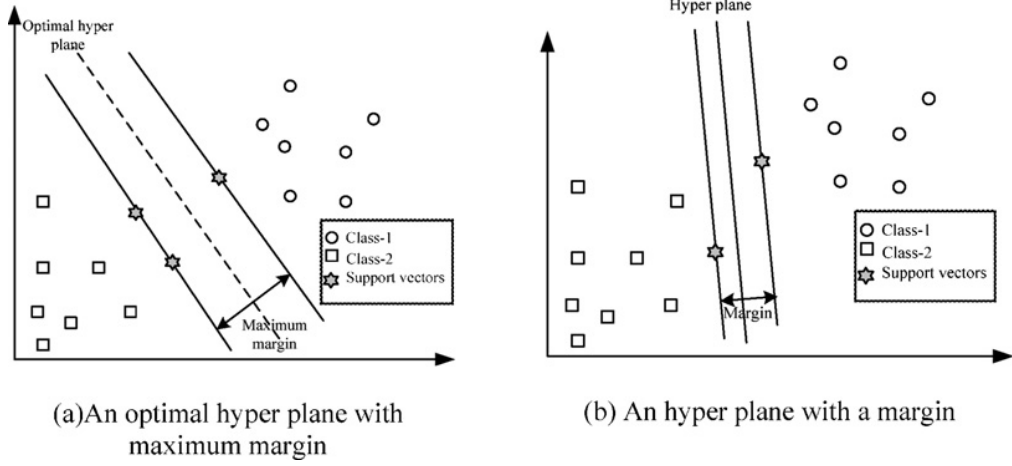


Figure 3.5: (a and b) Hyper plane and support vectors [52]

### 3.2.2 Convex Duality via Lagrange Multipliers

Recall the primal soft margin SVM Equation 3.14. We call the variables  $w$ ,  $b$ , and  $\xi$  corresponding to the primal SVM the primal variables. We use  $\alpha_i \geq 0$  as the Lagrange multiplier corresponding to the first constraint in Equation 3.14 that the examples are classified correctly and  $\beta_i > 0$  as the Lagrange multiplier corresponding to the non-negativity constraint (second constraint in Equation 3.14) of the slack variable. The Lagrangian is then given by

$$\mathcal{L}(w, b, \xi, \alpha, \beta) = \frac{1}{2} w^T w + C \sum_{i=1}^n \xi_i + \sum_{i=1}^n \alpha_i (1 - \xi_i - y_i (w^T x_i + b)) - \sum_{i=1}^n \beta_i \xi_i \quad (3.15)$$

By differentiating the Lagrangian Equation 3.15 with respect to the three primal variables  $w$ ,  $b$ , and  $\xi$  respectively, we obtain

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_{i=1}^n \alpha_i y_i x_i \quad (3.16)$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{i=1}^n \alpha_i y_i \quad (3.17)$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \alpha_i - \beta_i \quad (3.18)$$

The maximum of the Lagrangian by setting each of these partial derivatives to zero. By setting Equation 3.16, Equation 3.17, Equation 3.18 to zero, we find  $w = \sum_{i=1}^n \alpha_i y_i x_i$ ,  $\sum_{i=1}^n \alpha_i y_i = 0$ , and  $C = \alpha_i + \beta_i$  respectively.

$$\begin{aligned} \mathcal{L}(w, b, \xi, \alpha, \beta) &= \frac{1}{2} \sum_{i=1}^n \alpha_i y_i x_i^T \sum_{j=1}^n \alpha_j y_j x_j \\ &+ C \sum_{i=1}^n \xi_i + \sum_{i=0}^n \alpha_i \left( 1 - \xi_i - y_i \left[ \left( \sum_{j=1}^n \alpha_j y_j x_j^T \right) x_i + b \right] \right) - \sum_{i=1}^n \beta_i \xi_i \end{aligned} \quad (3.19)$$

$$\begin{aligned} \mathcal{L}(w, b, \xi, \alpha, \beta) &= - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (x_i x_j) \\ &+ C \sum_{i=1}^n \xi_i - \sum_{i=1}^n \xi_i (c - \alpha_i) + \sum_{i=1}^n \alpha_i - \sum_{i=1}^n \xi_i \alpha_i \end{aligned} \quad (3.20)$$



$$\mathcal{L}(w, b, \xi, \alpha, \beta) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (x_i x_j) \quad (3.21)$$

$$\begin{aligned} \max_{\alpha} & -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (x_i x_j) + \sum_{i=1}^n \alpha_i \\ & \text{subject to } \sum_{i=1}^n y_i \alpha_i = 0 \end{aligned} \quad (3.22)$$

$$0 \leq \alpha_i \leq C \quad \text{for all } i = 1, \dots, n$$

From the coefficients  $\alpha_i$ , we can write the final classification or decision function as.

$$f(x) = \text{sgn} \left[ \sum_{i=1}^n \alpha_i y_i (x \cdot x_i) + b \right] \quad (3.23)$$

where  $\text{sgn}$  is the classic sign function.

### 3.2.3 Kernels

As most problems are nonlinear in business applications, and the above linear SVM cannot be used. Ideally, we should find some nonlinear transformation  $\phi$  so that the data can be mapped onto a high-dimensional space where the classification becomes linear. The transformation should be chosen in a certain way so that their dot product leads to a kernel-style function  $K(x, x_i) = \phi(x) \cdot \phi(x_i)$ . In fact, we do not need to know such transformations, and we can directly use the kernel functions  $K(x, x_i)$  to complete this task. This is the so-called kernel function trick. Now the main task is to choose a suitable kernel function for a given, specific problem. If the problem cannot be solved in linear space, the original input space is mapped into a high-dimensional dot product space called feature space. This space has a complex dot product. The kernel function is used to find the optimal separation plane in the high-dimensional feature space. There are many kernel functions, and the most popular kernel functions are given in Table 3.1. In Table 3.1,  $d$ ,  $\sigma$ , and  $b$  are kernel parameters for each kernel function and these parameters affect the performance of SVM.

Type of classifier	Kernel function
Linear kernel	$K(x, x_i) = (x^T x_i)$
Polynomial kernel	$K(x, x_i) = [(x^T x_i) + 1]^d$
Radial based kernel	$K(x, x_i) = \exp\left(-\frac{\ x - x_i\ ^2}{2\sigma^2}\right)$
Sigmoid kernel	$K(x, x_i) = \tanh((x^T x_i) + b)$

Table 3.1: Popular kernel functions

## 3.3 Proposed Approach

As discussed before, kernel function  $C$ ,  $b$ , and control the SVM algorithm performance. For that reason, the GWO has been used to optimize these factors using Multi-start Strategy. The scheme of the proposed approach (GWO-SVM) algorithm is described in Figure 3.6. The GWO, alpha, beta, and delta are the top three solutions to examine the prey location. The GWO will stop when the final criterion is determined (the number of runs equals the maximum number of runs, and the iteration number equals the max number of iterations).

**Description of the proposed approach**

The proposed classification approach consists of six main phases.

**Preprocessing:** First, we preprocessed the data and split it into train and test data. We will discuss this in the next chapter.

**choosing kernel:** Secondly, we select kernel (Linear, Radial basis function, Polynomial)

**Initialization:** in this phase we set population size equals 50, maximum number of runs equals three, maximum number of iteration equals 100 hyperParameter C is  $10^{-3}$ ,  $10^{-2}$ ,  $10^{-1}$ , 1, 10,  $\gamma$  is  $10^{-4}$ ,  $10^{-3}$ ,  $10^{-2}$ ,  $10^{-1}$ , and degree is natural numbers in 3, 4.

**Evaluation:** Fourthly, if condition is satisfied, we evaluate the fitness function Equation 3.14. Next, we calculate  $X_\alpha$ ,  $X_\beta$ ,  $X_\delta$  and F1-score. Then update the position of each agent then update C, A, and  $\alpha$  using Equation 3.2, Equation 3.3, and Equation 3.4 respectively.

**Extraction optimized Parameter:** in the fifth phase, if conditions ( $iter > MaxIter$ ) are satisfied, extract the optimized parameters and predict the testing data using them, else we repeat the process until satisfied the condition.

**Output the results:** Finally, when the first criteria are determined, we get all results and choose the best kernel and parameters.

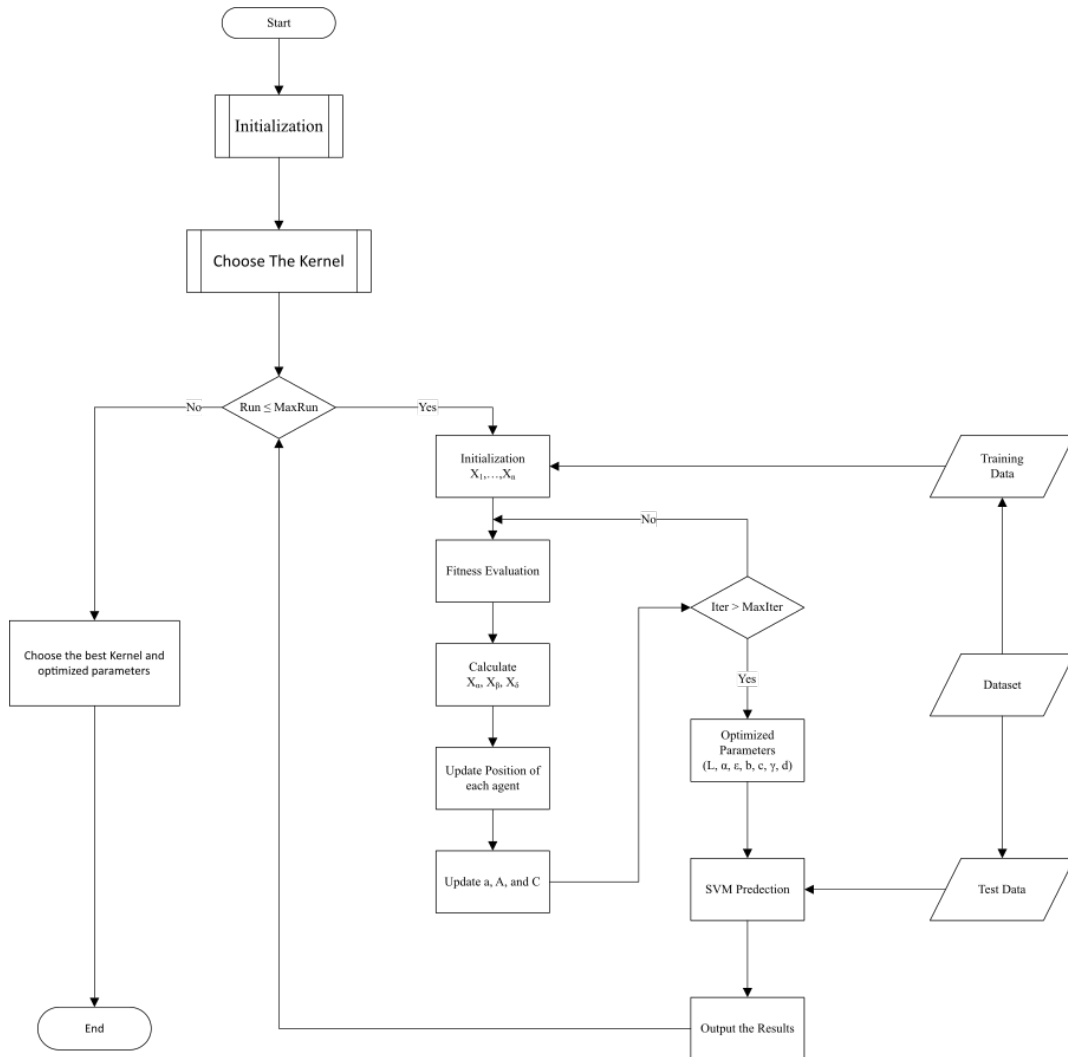


Figure 3.6: Flowchart of optimized SVM with GWO algorithm

### 3.4 Conclusion

In this chapter, we reviewed the proposed approach, starting with Grey Wolf Optimizer, by talking about inspiration, the mathematical models, then exploration and exploitation in GWO. Then we reviewed the Support vector machine through mathematical proof. Finally, we detailed the proposed approach by describing the steps of the proposed method and drawing the flowchart for an easier and deeper understanding.

## Chapter 4

# Experiments and results

### 4.1 Introduction

In this chapter, we present the results we obtained after having conducted several experiments using machine learning methods. We describe the corpus we used, the development environment and tools, and the different natural language processing (NLP) tasks performed during our experiments. We also present a hybrid method combining SVM & GWO for classification. To evaluate the quality of classification algorithms, five main measures are used, namely **matthew coefficient correlation**, accuracy, precision, recall, and F1-Score. In addition, during the learning and testing stages, the computation time was measured, which is also used in the performance analysis of the algorithms. Overall, the results are discussed, analyzed, and compared to other related work. Finally, based on the information obtained, the conclusion on the most efficient algorithm is given.

### 4.2 Data set

#### 4.2.1 Description of the Data Set

We collected data from several sources, first from sentiment analysis's most popular data set, the sentiment140 dataset. It contains 1,600,000 tweets extracted using the Twitter API. The tweets have been annotated (0 = negative, 4 = positive). Moreover, the suicide data set from this repository in GitHub <sup>1</sup>, which contains two columns [ids, text] and 52616 rows, has been labelled by -1. We reduced the dataset to 30,000 rows and divided them equally at the expense of the two categories (suicidal and non-suicidal). The data is marked with 4 as non-suicidal, and suicidal is marked with -1. The total data collection contains 30,000 tweets with 15,000 positive tweets and 15,000 negative tweets (The post mentions references to self-harm, suicidal ideation, apologies, negative feelings like worthlessness, self-hatred, guilt, etc.). It contains the following five fields:

- target: the polarity of the tweet (0 = negative, 1= positive)
- ids: The id of the tweet (1467838362)
- date: the date of the tweet (Mon Apr 06 22:26:58 PDT 2009)
- user: the user that tweeted (megan\_rice)
- text: the tweet's text (@Alliana07 it didn't make any sense to me, the suicide thing. I refuse to believe that that is actually what happened.)

---

<sup>1</sup><https://github.com/IE-NITK/TwitterSuicidalAnalysis>

Text	Label
all too familiar so why are we back here again tell me you know how it feels to be killing a friend screaming in silence theres no way of saving myself.	0
i am gonna say goodbye i can't take it anymore soo i am leaving.	0
@EmilyyBrowningg OMG! I can't wait to work with you.I gonna buy The Uninvited tonight with Vanessa I haven't seen it.	1
@DavidArchie Hey! Glad you arrived there safe. We'll miss you here at the Philippines! =^-^=	1

Table 4.1: Example of the Dataset

The following Table 4.1 shows an excerpt from the dataset, specifically the two fields ‘Text’ and the ‘Label (Sentiment)’.

### 4.3 Development tools and environment

#### 4.3.1 Programming language

To carry out our work, we used Python as a programming language. Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first published in 1991, the design philosophy of Python emphasizes the readability of the code, with Python’s design philosophy emphasizes code readability with its notable use of ample white space. Its language constructs and object-oriented approach aim to help programmers to write clear and logical code for both small and large scale projects [53].

Python supports several programming paradigms, including structured (in particular, procedural), object-oriented, and functional programming. Python is often described as a “battery included” language because of its extensive and rich standard library, which covers everything from asynchronous processing to zip files [54].

Python 3.0, released in 2008, was a significant overhaul of the language that is not fully backward compatible, and much of the Python 2 code (which was officially discontinued in 2020) does not work unmodified on Python 3 [55].

#### 4.3.2 why Python?

Python is preferred to other programming languages because it has a big community due to its characteristics and properties that have attracted many researchers and developers, especially in artificial intelligence and machine learning. Among these properties, we can mention :

- – Extensibility: the ability to extend legacy code by adding new features and gluing several components.
- – Python is commonly used in the context of data analysis and machine learning.
- – Elegant syntax (pseudo-code syntax).
- – Promotes the readability and maintainability of the code (intensive work with less code).

- – Availability of a large number of libraries.
- – Python is cross-platform (Platform independent).
- – Availability of data structures such as lists, tables, and dictionaries.
- – Dynamic typing and dynamic linking.

The decision to choose Python depends on our concerns. If we are interested in speed of development and lazy coding, then Python is a good choice. On the other hand, it is not recommended when the main concern is to reduce the cost of executing the code [55].

### 4.3.3 Tools and Libraries

**Scikit-learn:** Scikit-learn [56] (also known as sklearn) is a Python module integrating a wide range of supervised and unsupervised machine learning algorithms. This package focuses on making machine learning available to non-specialists using a high-level, general-purpose language. The focus is on ease of use, performance, documentation, and API consistency. It has minimal dependencies and is distributed under the simplified BSD license, encouraging its use in academic and commercial circles. Scikit-learn is open source. Source code, binaries, and documentation are available at: <https://scikit-learn.sourceforge.net>.

**NLTK:** NLTK<sup>2</sup> (Natural Language Toolkit) [57] is a complete Python library for natural language processing and text analysis. Originally designed for teaching, it was adopted in industry for research and development because of its usefulness and power. NLTK supports several features like classification, tokenization, stemming, morpho-syntactic labeling, semantic analysis and reasoning, etc.

**spaCy:** spaCy is a library for advanced Natural Language Processing<sup>3</sup> in Python and Cython. It's built on the very latest research, and was designed from day one to be used in real products. spaCy comes with pretrained pipelines and currently supports tokenization and training for 60+ languages. It features state-of-the-art speed and neural network models for tagging, parsing, named entity recognition, text classification and more, multi-task learning with pretrained transformers like BERT, as well as a production-ready training system and easy model packaging, deployment and workflow management. spaCy is commercial open-source software, released under the MIT license.

**WordNet:** WordNet [58] is a lexical database for English. It can be considered a sizeable electronic dictionary. It contains information on some 155,000 nouns, verbs, adjectives, and adverbs. WordNet resembles a thesaurus in that it groups words according to their meaning. However, there are some important distinctions. First, WordNet interconnects not only word shapes but specific meanings of words. As a result, words close to each other in the network are semantically ambiguous. Second, WordNet labels semantic relationships between words, while grouping words in a thesaurus follows no explicit pattern other than the similarity of meaning.

**Emojis for Python:** Emojis<sup>4</sup> is an open-source library for python that allows the manipulation of emoticons. It contains valuable features for the processing of emoticons in strings such as decoding (Decoding Unicode Emoji values in alias Emoji), encoding (Encoding emoticon aliases in Unicode Emoji values.), extracting emoticons from a string of characters Etc.

---

<sup>2</sup><https://www.nltk.org>

<sup>3</sup><https://spacy.io>

<sup>4</sup><https://github.com/alexandrevicenzi/emojis>

**pandas:** Pandas stands for “Python Data Analysis Library” is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. It is free software released under the three-clause BSD license. Source code, binaries, and documentation are available at: <https://pandas.pydata.org>.

**NumPy:** NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. It is free software released under the three-clause MIT license.

**SciPy:** SciPy is open-source software for mathematics, science, and engineering. SciPy library depends on NumPy, which provides convenient and fast N-dimensional array manipulation. The SciPy<sup>5</sup> library is built to work with NumPy arrays, and provides many user-friendly and efficient numerical routines such as routines for numerical integration and optimization. Together, they run on all popular operating systems, are quick to install, and are free of charge. NumPy and SciPy are easy to use, but powerful enough to be depended upon by some of the world’s leading scientists and engineers.

**Matplotlib:** Matplotlib is a cross-platform data visualization, and graphical plotting library for Python and its numerical extension NumPy and it is distributed under a BSD-style license.

**Jupyter:** Jupyter is an open-source, browser-based tool that integrates interpreted languages, libraries, and visualization tools [59]. A Jupyter notebook can run on-premises or in the cloud. Each document comprises multiple cells, where each cell contains a scripting language or markup code, and the output is embedded in the document. Typical outputs include text, tables, charts, and graphs. The use of this technology facilitates the sharing and reproduction of scientific work, as experiments and results are presented autonomously [60].

#### 4.3.4 Execution environment (Hardware)

**CPU:** Intel(R) Xeon(R) Platinum 8272CL CPU @2.60GHz 2 Cores 4 Logical processors

**RAM:** 16GB

**Storage:** 126 GB

**OS:** Windows 10 Pro Build 19042

## 4.4 Implementation

### 4.4.1 Data preprocessing

Preprocessing is one of the most common tasks in many machine learning applications and a fundamental step for all-natural language machine processing (NLP) tasks. The steps required for text-based preprocessing typically depend on the requirement or targeted application. In our case, we perform a preprocessing of tweets to analyse feelings and precisely for the automatic

---

<sup>5</sup><https://scipy.org>

detection of suicidal ideation. In this section, we describe how the tweets were preprocessed. Figure 4.1 shows the different steps performed during this phase.

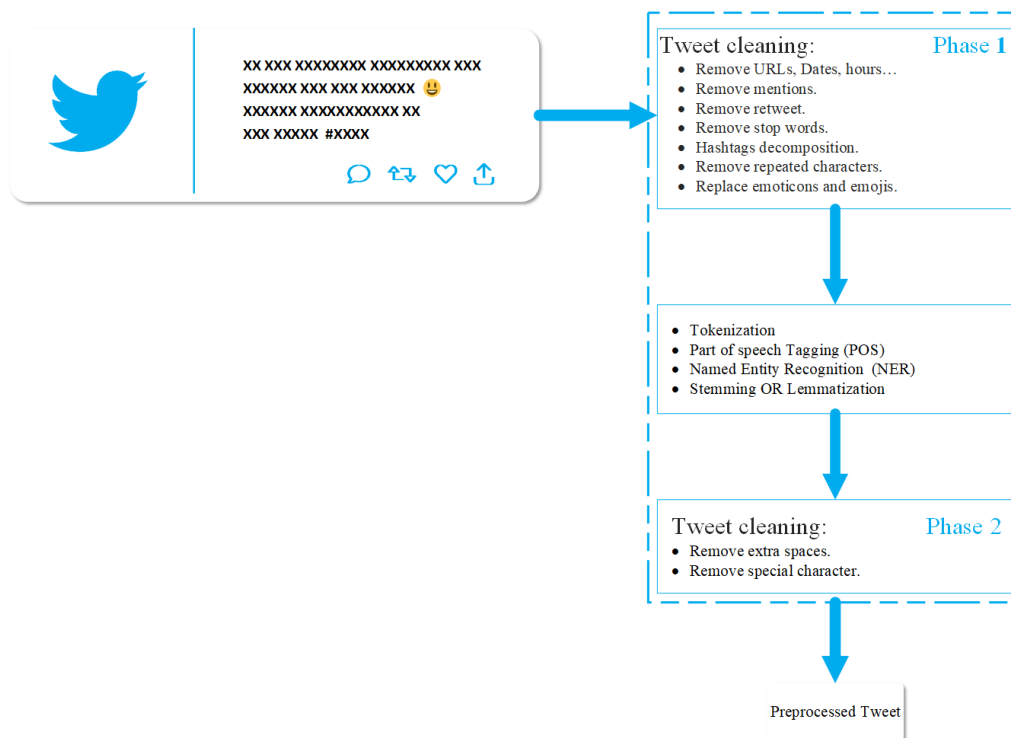


Figure 4.1: Tweets preprocessing phases

**Clean up tweets:** To clean up the tweets. This step includes removing URLs (which start with either “http://” or “https://”, user mentions (for example, ‘@user’), times, dates, numbers, extra spaces, etc. Indeed, they do not add any information indicating whether the tweet could express hatred or not. In particular, in the case of user mentions, if the relationship between the author of the tweet and the tagged person is known, this information can be valuable. However, since no background is provided regarding the author and tagged person, we consider that the use of tags is not useful for our preprocessing phase.

**Hashtag breakdown:** Hashtags are used to indicate the relevance of a tweet to a specific topic. Some of these hashtags are often small, meaningful phrases and their decomposition can be helpful for our study. So, and unlike Urls and user mentions, hashtags will not be removed during preprocessing. In a separate step and using the ‘Wordninja’ library, each hashtag is extracted and broken down into words that make it up. For example, the hashtag “#iwillkillmyself” will express “I will kill myself”. Furthermore, that way, we will lose less information during the data preprocessing phase.

**Replace emoticons and emojis:** Since Twitter users express their feelings with emoticons, emoticons play an important role in identifying the feeling of the tweet. The emoticons in a tweet are perceived and then decoded, that is to say, mapped with their aliases (Meanings) in natural language, and this using the library “Emot & Emojis for Python” described previously. The following table illustrates this mapping through some examples of emojis & emoticons.

**Capture elongated words:** Twitter users tend to be very informal in their language, and most lengthen the words. For example, the term ‘happyyyyyyyyy’ is an elongated form of ‘happy’, and this is to avoid having several repeated words (happyyyyy, happy, haaappyyyy, . . .) with the same meaning (happy).





Emojis & Emoticons	Alias
:)	happy_face_or_smiley
	persevere
>: (	frown_sad_andry_or_poute
	woozy_face

Table 4.2: Mapping of emoticons and emojis

**Remove punctuation and return the text to lowercase:** It is vital to have the whole word in a consistent case when classifying texts to ensure that all terms match the corresponding functionality independently, whether the letters of this term are in upper or lower case. This is extremely important for our study because it is prevalent to find irregular shapes (such as "I WiLL KIll mY SElf tONight") in microblogs. In addition to putting all the words of the text in lowercase, we remove punctuation and special characters in the text.

**Eliminate stopwords:** Stopwords contain little information and are less useful in the analysis of the tweet. They are usually articles, conjunctions or prepositions that do not help us find the context or the true meaning of a sentence. These are words that can be deleted without any negative consequences on our final model. Commonly used terms in English are 'is', 'and', 'are', etc. In our case, the Spacy library stopwords corpus was used to eliminate stopwords. The stopwords corpus includes 305 stopwords for the English language.

**Tokenization:** To process textual data, it is essential to find the boundaries of the words. Tokenization is the process of breaking down texts into words, phrases and symbols called 'Tokens'. The generated 'Tokens' are used to analyze and perform other exploration tasks. Tokenization is performed using the Spacy library.

**Part of Speech Tagging** This process consists of breaking down the sentence into words and assigning a label to each word, whether the word is a noun, a verb, an adjective etc. Morpho-syntactic labelling facilitates identifying the meaning behind the word and the relationship between them. Both NLTK and Spacy libraries are used for morpho-syntactic tagging of tweets.

**Named-entity recognition:** Another related sibling of POS tagging is NER tagging. A named entity is a word or a phrase that clearly identifies one item from a set of other items that have similar attributes [61]. Examples of named entities are organization, person, and location names in a public domain; gene, protein, drug and disease names in the biomedical domain. NER is the process of locating and classifying named entities in text into predefined entity categories. NER acts as an important pre-processing step for a variety of downstream applications such as information re-trieval, question answering, machine translation, etc. Here, we use semantic search as an example to illustrate the importance of NER in supporting various applications. Semantic search refers to a collection of techniques, which enable search engines to understand the concepts, meaning, and intent behind the queries from users [62].

**Stemming:** It is defined as the process which produces variants of a root/base word. In simple words, it reduces a base word to its stem word. We use stemming to shorten the look-up and normalize the sentences for a better understanding. In our work, Porter's algorithm [63] is used for the stemming stage.

**Lemmatization:** It is the process of assembling the inflected parts of a word such that they can be recognized as a single element, called the word’s lemma or it’s vocabulary form.[64] This process is the same as stemming but it adds meaning to particular words. In simple words, it connects text with the same meanings to a single word. It is defined as an algorithm technique of finding the lemma of a word which is a root word rather than a root stem.[65] It is based on the intended meaning the word is trying to convey.

At the end of the preprocessing, we perform a check and filtering to eliminate tokens composed of a single character or those composed of non-alphabetic characters.

#### 4.4.2 Features extraction

Feature Extraction is a general term for methods of deriving values (features) intended to be informative, from an initial set of measured data. The set of extracted features is called Feature Vector. Feature extraction is related to dimensionality reduction [66].

#### 4.4.3 Representation of the tweet:

We extract Unigram, Bigram, and Trigram characteristics from preprocessed tweets and weight them according to their TF-IDF and BOW values. The purpose of using TF-IDF and BOW is to reduce the effect of less informative terms that appear very frequently in the data corpus. This feature can be a standard feature.

**Bag-of-words** The simplest approach to convert text into structured features is using the bag of words approach. Bag of words simply breaks apart the words in the review text into individual word count statistics. Figure 4.2 presents how Turning raw text into a bag of words representation

**TF-IDF** The term frequency-inverse document frequency (also called TF-IDF), is a well-recognized method to evaluate the importance of a word in a document. Term Frequency of a particular term ( $t$ ) is calculated as number of times a term occurs in a document to the total number of words in the document. IDF (Inverse Document Frequency) is used to calculate the importance of a term. The mathematical representation of the weight of a term ‘ $t$ ’ in a tweet (document) ‘ $d$ ’ by TF-IDF is given by equations 4.1, 4.2, and 4.3.

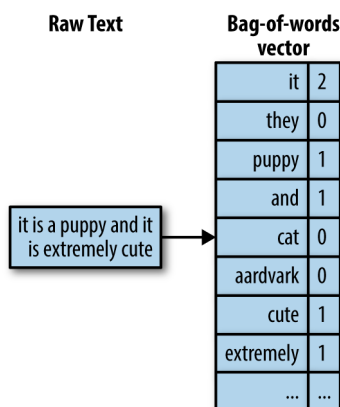


Figure 4.2: Turning raw text into a bag of words representatio [67]

$$TF - IDF(d, t) = tf(t) * idf(d, t) \quad (4.1)$$

Where

$$tf(t) = \frac{\text{Number of times 't' appears in 'd'}}{\text{Total number of terms in 'd'}} \quad (4.2)$$

$$idf(t) = \frac{\text{Total number of tweets}}{\text{Number of tweets containing the term 't'}} \quad (4.3)$$

#### 4.4.4 Classification

Supervised classification is the prediction of the value of a qualitative response variable. This response variable is commonly referred to as a category or class. Many classification algorithms (also called classifiers) and their performance depend on the problem and its nature. Since there is no classifier that is the best for all kinds of problems (no free lunch theorem [68]), for our work, a set of models was evaluated:

**Support Vector Machine** SVM are classifiers whose result is based on a decision boundary generated by support vectors, the points closest to the decision boundary. The shape of the boundary is determined by a function of the kernel. In this way, it is possible to solve problems that cannot be resolved by a linear boundary. Intuitively, a good separation is obtained by the hyperplane with the greatest distance to the support vectors. In general, the greater the margin, the lower the generalization error of the classifier [69].

**Logistic Regression:** The logistic regression classification algorithm uses a sigmoid function, expressed by Equation 4.4, as a prediction function that returns a probability value that can then be mapped to discrete classes. The results of the sigmoid function are probability estimates that range from 0 to 1. To predict the label of a data point, the class with the highest score or probability is chosen [70].

$$P(x) = \frac{1}{1 + e^{-x}} \quad (4.4)$$

**Decision Tree (DT):** The Decision tree is one of the classification techniques in which classification is done by the splitting criteria. The decision tree is a flow chart like a tree structure that classifies instances by sorting them based on the attribute (feature) values. Each and every node in a decision tree represents an attribute in an instance to be classified. All branches represent an outcome of the test, each leaf node holds the class label. The instance is classified based on their feature value. There are numerous methods for finding the feature that best divide the training data such as information gain, gain ratio, Gini index etc. The most common way to build decision trees by using top down greedy method partitioning, starting with the training set and recursively finding a split feature that maximizes some local criterion. Decision tree generates the rule for the classification of the [71].

**Naïve Bayes** NB is a simple probabilistic classifier based on the application of Baye's theorem with strong assumptions of independence. For text classification, this algorithm calculates the posterior probability of the document belongs to different classes, and it assigns the document to the class with the highest posterior probability. This probability model would be an independent entity model so that the presence of an entity does not affect other entities in classification tasks [72].

#### 4.4.5 Performance Parameters (Evaluation)

There are many valuation metrics used to evaluate the models. All the data have been randomly divided into ten disjoint subsets (folders) in this work, each containing approximately the same number of instances. In each experiment, nine folders have been used as training data, i.e., to set up the classifier. In contrast, the remaining folder was used as validation, i.e., to evaluate the classification results. This process was repeated ten times for each different choice of validation folder. The ten results were then averaged to produce a single estimation. The classification performance of each classifier is evaluated using the metrics described below [73].

**Confusion matrix:** A confusion matrix shows the combination of actual and predicted classes. Each row in the matrix represents a predicted class, while each column represents an actual class. This is a good measure of whether models can account for overlapping class properties and understand which classes are most easily confused. Figure 4.3 illustrates the confusion matrix for our classification problem.

		Actual Values			
		Suicidal (0)	No suicidal (1)		
Predicted Values	Suicidal (0)	<b>TN</b>	<b>FP</b>		
	No suicidal (1)	<b>FN</b>	<b>TP</b>		
		Suicidal (0)	No suicidal (1)		

Figure 4.3: Confusion matrix for our classification problem

Given a binary classifier and an instance, there are four possible outcomes. If the instance is positive (No-suicidal tweet) and is classified as positive, it is counted as a true positive (TP); if it is classified as negative (suicidal tweet), it is counted as a false negative (FN). If the instance is negative and is classified as negative, it is counted as a true negative (TN); if it is classified as positive, it is counted as a false positive (FP). Given a classifier and a set of instances (the test set), a two-by-two confusion matrix (also known as a contingency table) can be constructed representing the layouts of the instance set. This matrix forms the basis of many common metrics.

**Accuracy:** that is the portion of correctly classified instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.5)$$

**Sensitivity:** also called Recall or True Positive Rate - TPR, that measures the portion of actual positives that are correctly identified as such:

$$Sensitivity = \frac{TP}{TP + FN} \quad (4.6)$$

**Specificity:** also called True Negative Rate - TNR, that measures the portion of negatives that are correctly identified as such:

$$Specificity = \frac{TN}{TN + FP} \quad (4.7)$$

**Precision:** also called positive predictive value, that is a measure of actual positives with respect to all the instances classified as positive:

$$Precision = \frac{TP}{TP + FP} \quad (4.8)$$

**F-Measure (F1-score):** that is the harmonic mean of Precision and Sensitivity. It can be used as a single performance measure:

$$F1 - score = \frac{2 * Sensitivity * Precision}{Sensitivity + Precision} \quad (4.9)$$

**AUC (Area under ROC curve):** that is an estimation of the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one.

$$AUC = \frac{Sensitivity + Specificity}{2} \quad (4.10)$$

**MCC (Matthews Correlation Coefficient):** that correlates the observed and predicted binary classifications by simultaneously considering true and false positives and negatives. It can assume a value between -1 and +1, where +1 represents a perfect prediction, 0 no better than random prediction and -1 indicates total disagreement between prediction and observation:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}} \quad (4.11)$$

## 4.5 Experiments and results

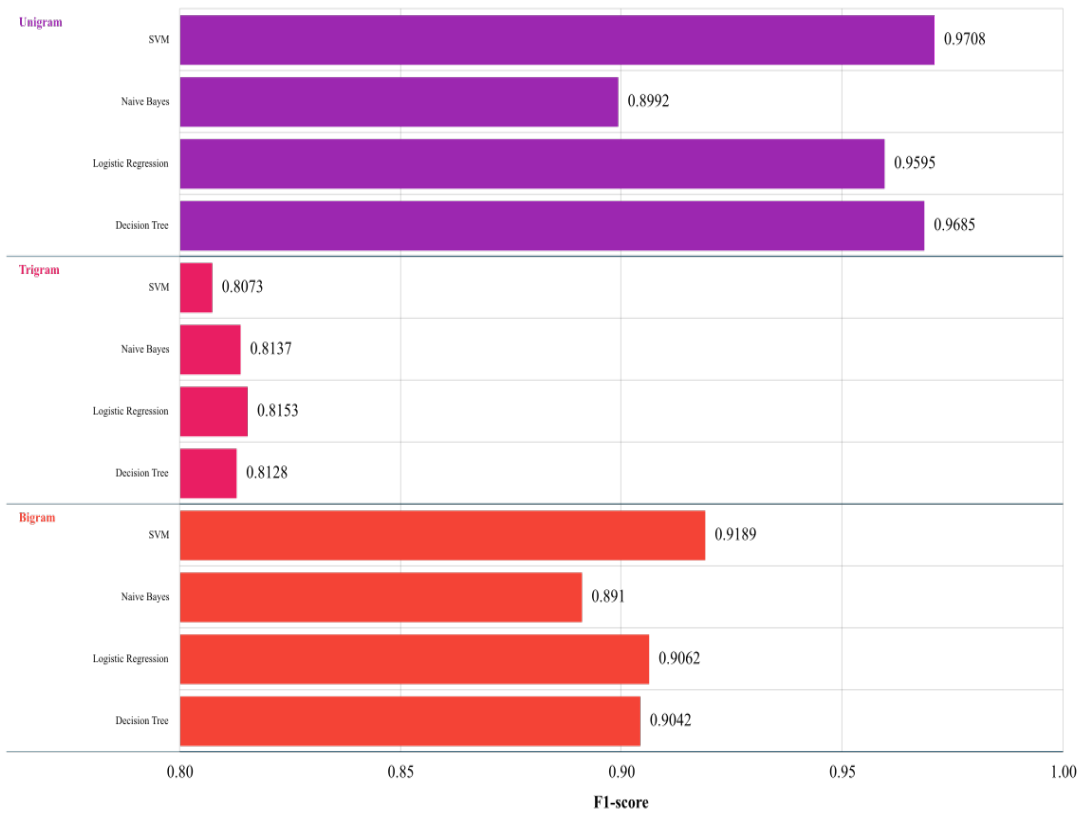
At the end of the preprocessing phase, our data set will be transformed into a matrix of **29,458** rows representing the samples (tweets) and **15,000** columns representing the extracted characteristics. The preprocessing and feature extraction phase using Lemmatization took **223, 19.3** secondes. Moreover, using Stemming took **34.8, 21.4** secondes respectively.

### 4.5.1 Classification

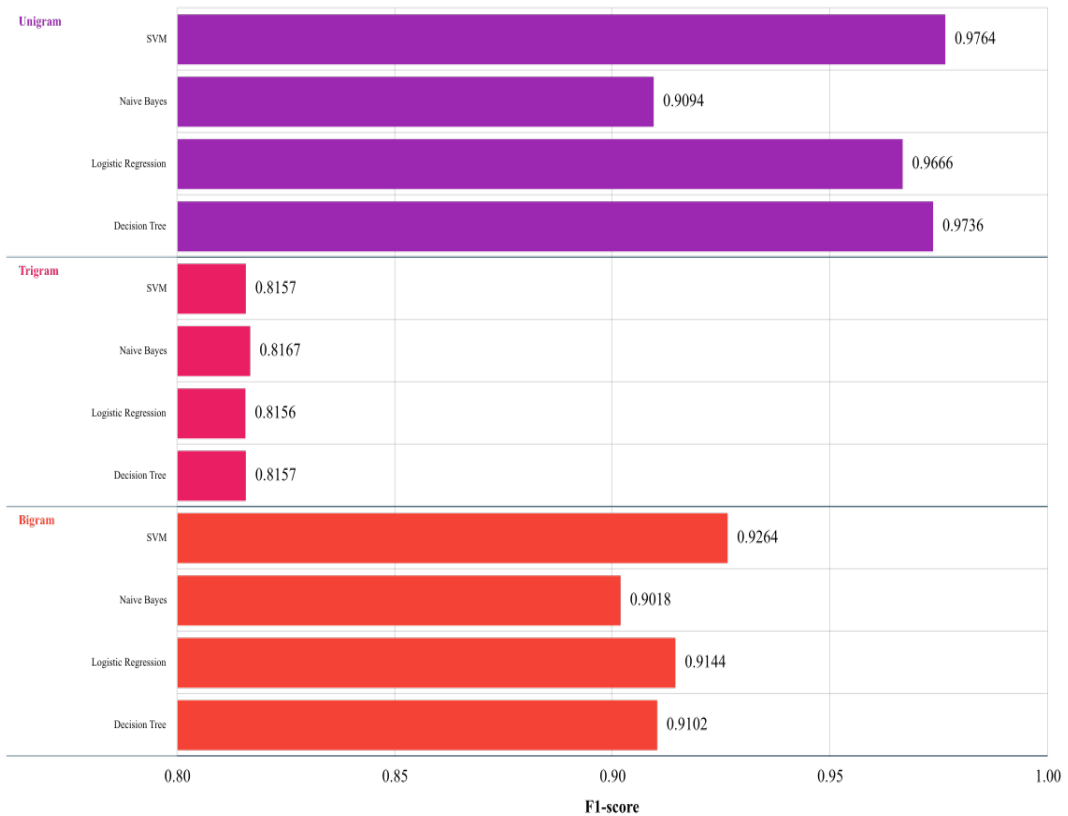
In this subsection, we will review the results of the comparative analysis of logistic regression (LR), Multinomial Naïve Bayes (MNB), decision tree (DT), and support vector machines (SVC) using Stemming and Lemmatization and TF-IDF, BOW representation.

#### TF-IDF

The results of the comparative analysis of the four algorithms using TF-IDF representation are detailed in Tables 4.3, 4.4.

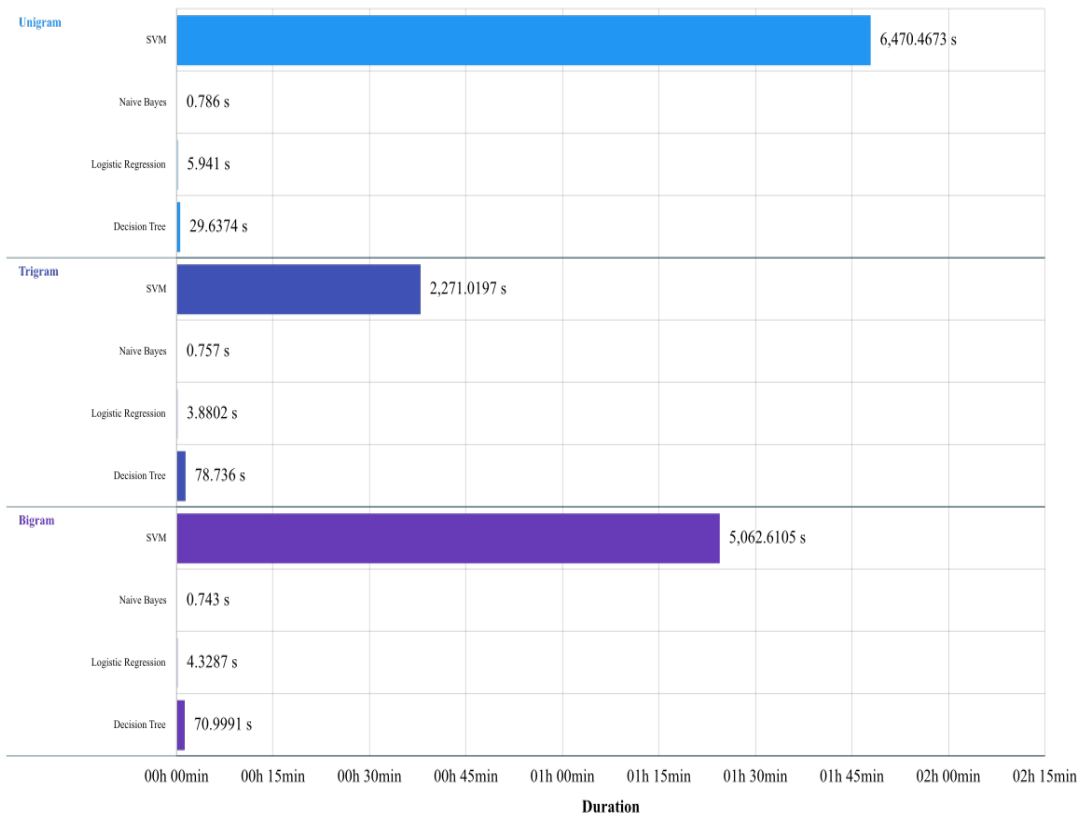


(a) results of the classification using lemmatization

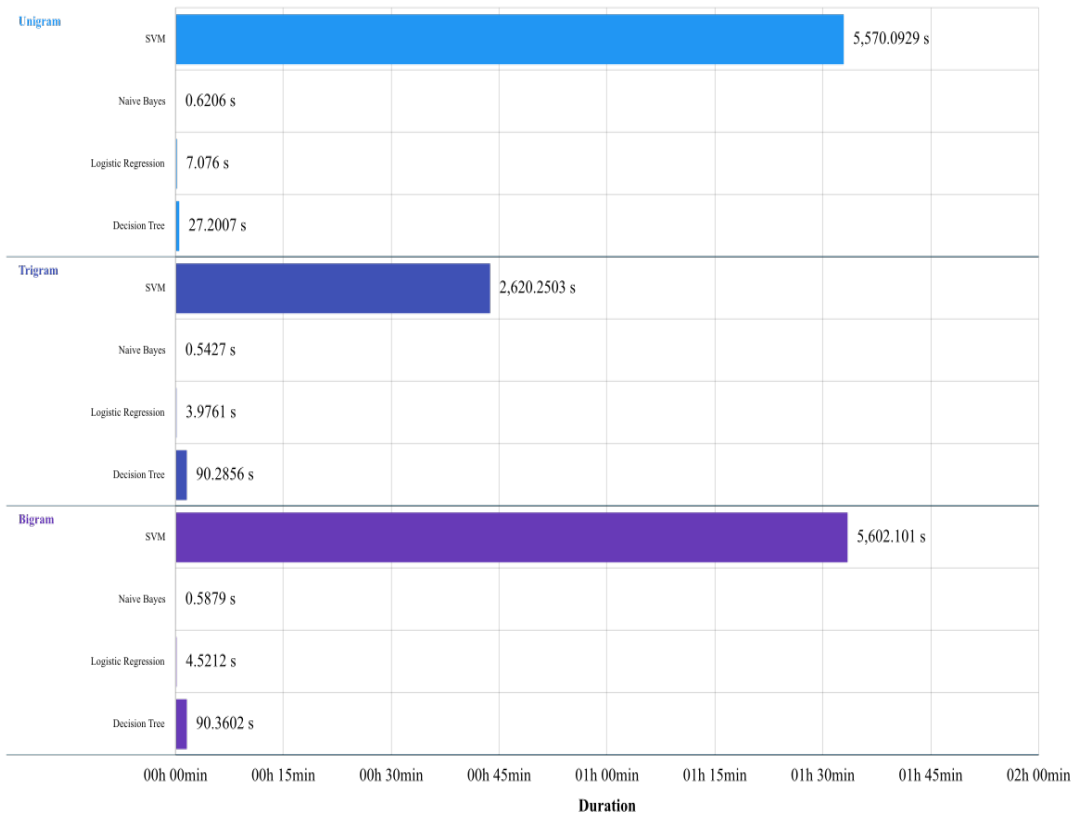


(b) results of the classification using stemming

Figure 4.4: Comparison of four classification algorithms using lemmatization and TF-IDF representation



(a) Duration of classification using lemmatization



(b) Duration of classification using stemming

Figure 4.5: Training and prediction time for the four classification algorithms using TF-IDF representation

Classifier		Ten-fold cross validation results (means)					
		F-Measure	Recall	Precision	Accuracy	AUC	MCC
Uni-Gram	DT	0.9685	0.9685	0.9684	0.9679	0.9693	0.9358
	LR	0.9594	0.964	0.9549	0.9585	0.9895	0.9171
	NB	0.8991	0.8484	0.9562	0.9031	0.9706	0.812
	SVM	0.9708	0.9671	0.9745	0.9704	0.9925	0.9408
Bi-Gram	DT	0.9042	0.9278	0.8817	0.8999	0.8929	0.8008
	LR	0.9062	0.9555	0.8617	0.8993	0.9628	0.8032
	NB	0.8909	0.8783	0.904	0.8906	0.9387	0.7816
	SVM	0.9189	0.9482	0.8913	0.9148	0.963	0.8312
Tri-Gram	DT	0.8128	0.9439	0.7137	0.7787	0.7883	0.5878
	LR	0.8153	0.9681	0.7042	0.7768	0.8649	0.5963
	NB	0.8137	0.9482	0.7126	0.779	0.753	0.5902
	SVM	0.8073	0.7963	0.8187	0.8066	0.8519	0.6136

Table 4.3: Comparison of four classification algorithms for the suicidal ideation detection task using lemmatization and TF-IDF representation

Classifier		Ten-fold cross validation results (means)					
		F-Measure	Recall	Precision	Accuracy	AUC	MCC
Uni-Gram	DT	0.9735	0.9725	0.9746	0.9731	0.9754	0.9462
	LR	0.9665	0.9689	0.9642	0.9659	0.9939	0.9318
	NB	0.9093	0.86	0.9647	0.9127	0.9783	0.8309
	SVM	0.9764	0.9732	0.9795	0.976	0.9959	0.9521
Bi-Gram	DT	0.9101	0.9314	0.8899	0.9064	0.905	0.8136
	LR	0.9143	0.9644	0.8692	0.908	0.9696	0.8209
	NB	0.9018	0.8889	0.9151	0.9015	0.9495	0.8035
	SVM	0.9263	0.9463	0.9072	0.9234	0.9693	0.8475
Tri-Gram	DT	0.8157	0.961	0.7086	0.779	0.7798	0.5963
	LR	0.8156	0.9777	0.6996	0.775	0.8676	0.5987
	NB	0.8167	0.9646	0.7081	0.7796	0.7467	0.5991
	SVM	0.8156	0.8096	0.8218	0.8138	0.859	0.6277

Table 4.4: Comparison of four classification algorithms for the suicidal ideation detection task using stemming and TF-IDF representation



**Bag of word**

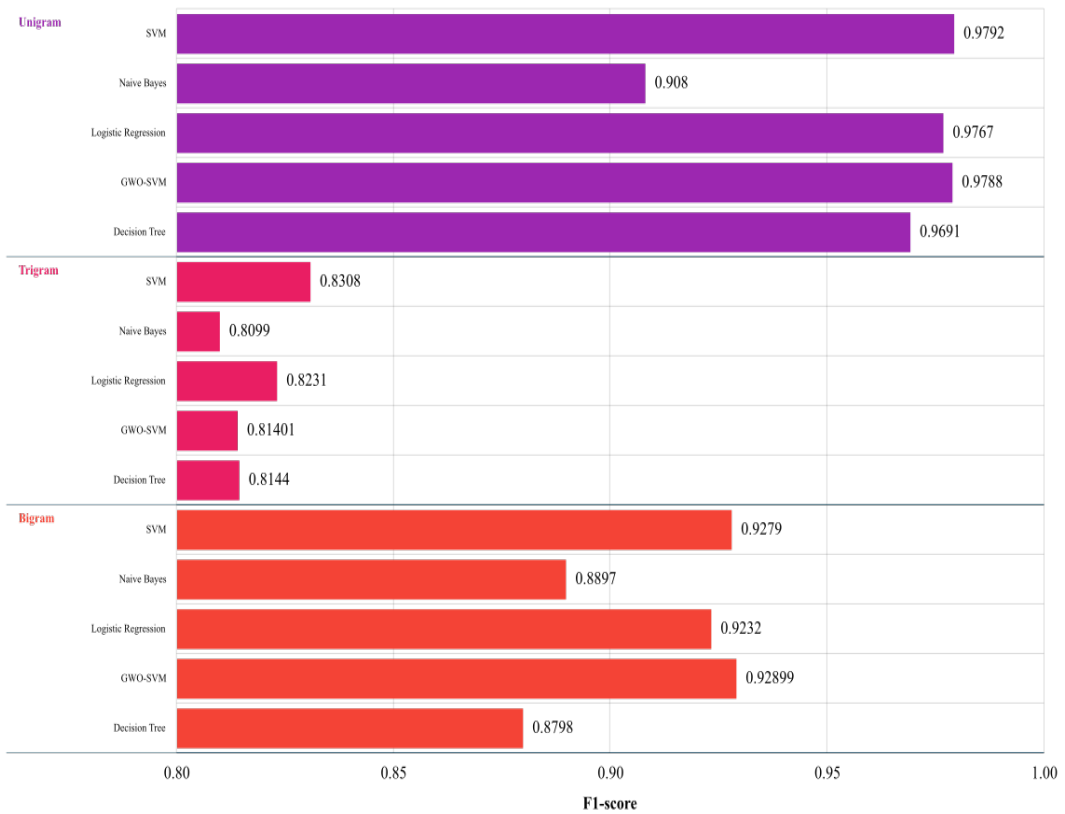
The results of the comparative analysis of the four algorithms using Bag of word representation are detailed in Tables 4.5, 4.6.

Classifier		Ten-fold cross validation results (means)					
		F-Measure	Recall	Precision	Accuracy	AUC	MCC
Uni-Gram	DT	0.969	0.9647	0.9734	0.9686	0.9714	0.9373
	LR	0.9766	0.9749	0.9784	0.9763	0.9953	0.9526
	NB	0.9079	0.8609	0.9604	0.9112	0.9728	0.8273
	SVM	0.9792	0.9706	0.9879	0.979	0.9949	0.9582
Bi-Gram	DT	0.8797	0.9235	0.8399	0.8715	0.8707	0.7464
	LR	0.9232	0.968	0.8824	0.918	0.9687	0.8399
	NB	0.8896	0.8723	0.9077	0.8899	0.935	0.7805
	SVM	0.9278	0.9461	0.9103	0.9251	0.9688	0.8508
Tri-Gram	DT	0.8143	0.9397	0.7185	0.7819	0.7909	0.5916
	LR	0.823	0.9706	0.7144	0.7876	0.8666	0.6155
	NB	0.8098	0.956	0.7024	0.7715	0.7319	0.5813
	SVM	0.8307	0.8939	0.776	0.8146	0.8655	0.636

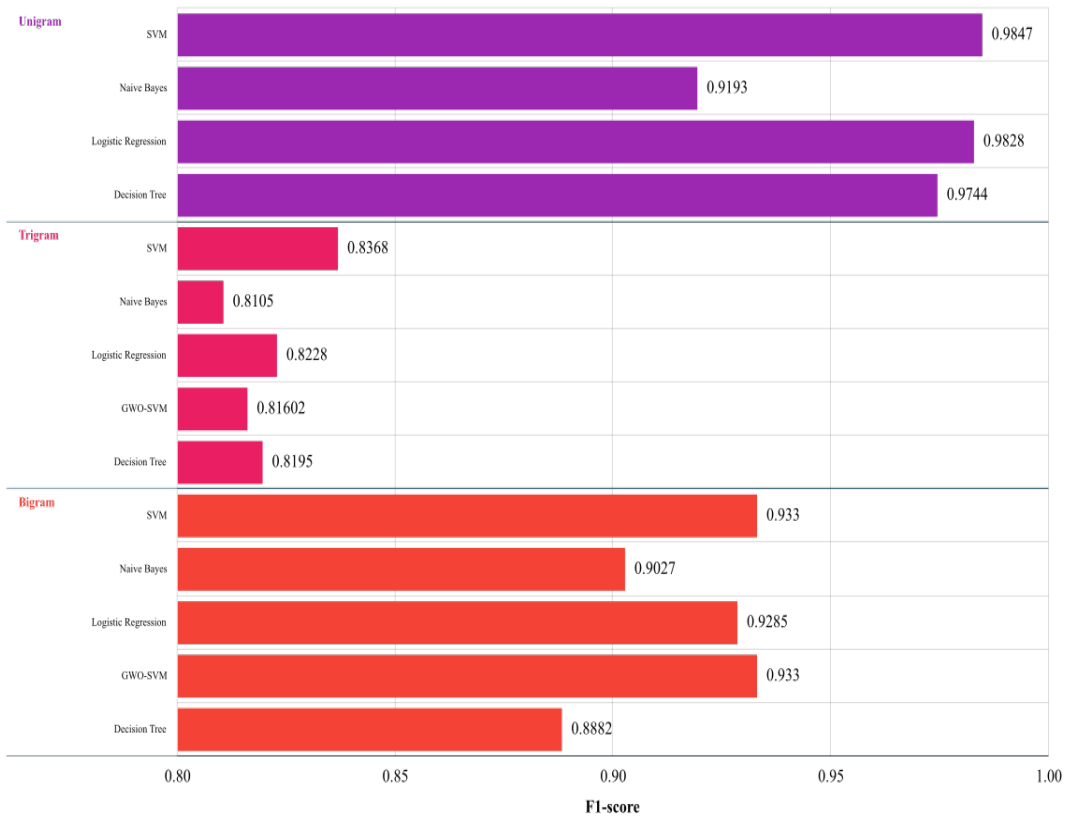
Table 4.5: Comparison of four classification algorithms for the suicidal ideation detection task using lemmatization and BOW representation

Classifier		Ten-fold cross validation results (means)					
		F-Measure	Recall	Precision	Accuracy	AUC	MCC
Uni-Gram	DT	0.9743	0.9729	0.9757	0.9739	0.9746	0.9479
	LR	0.9828	0.9816	0.9839	0.9825	0.9974	0.965
	NB	0.9192	0.8737	0.9697	0.9219	0.9794	0.8483
	SVM	0.9847	0.9786	0.9908	0.9845	0.997	0.9691
Bi-Gram	DT	0.8881	0.9284	0.8512	0.881	0.873	0.7648
	LR	0.9284	0.9747	0.8864	0.9235	0.9747	0.8513
	NB	0.9026	0.8865	0.9193	0.9027	0.9497	0.806
	SVM	0.933	0.9484	0.9181	0.9307	0.9737	0.8618
Tri-Gram	DT	0.8195	0.9592	0.71536	0.7850	0.784	0.6053
	LR	0.8227	0.9791	0.7094	0.7853	0.8659	0.6163
	NB	0.8105	0.9704	0.6958	0.7691	0.7191	0.5848
	SVM	0.8367	0.9119	0.773	0.8189	0.867	0.6476

Table 4.6: Comparison of four classification algorithms for the suicidal ideation detection task using stemming and BOW representation

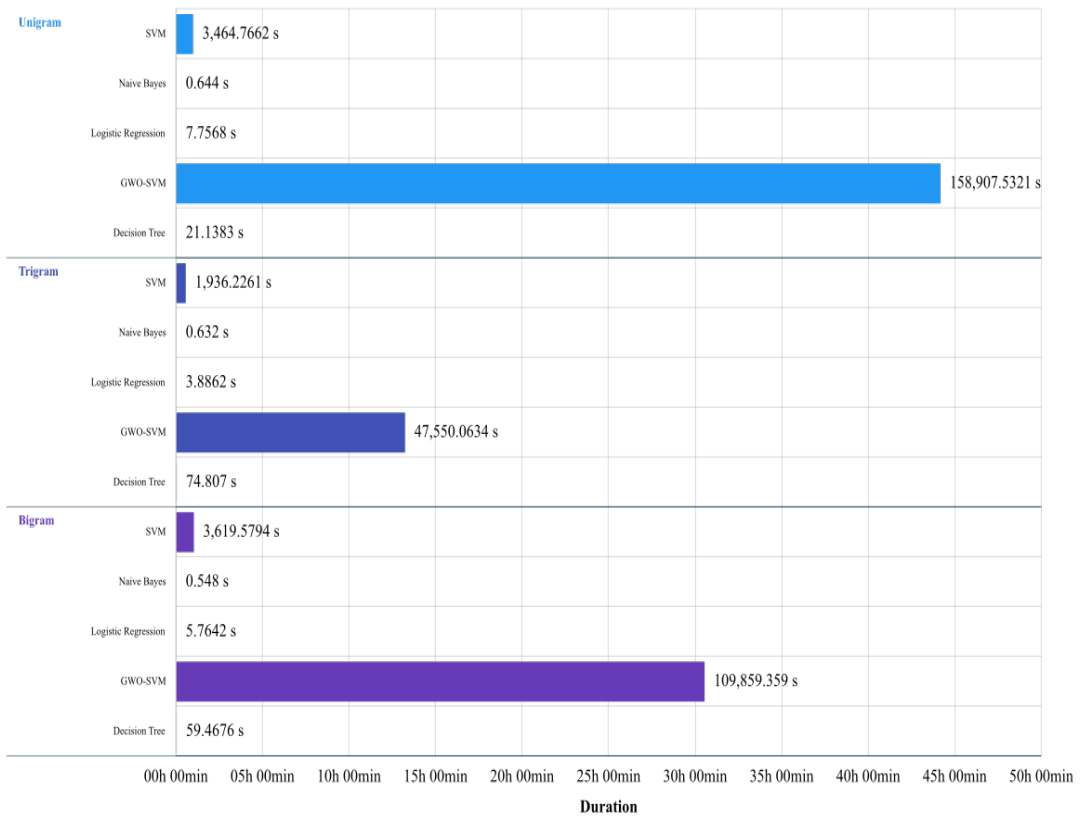


(a) results of the classification using lemmatization

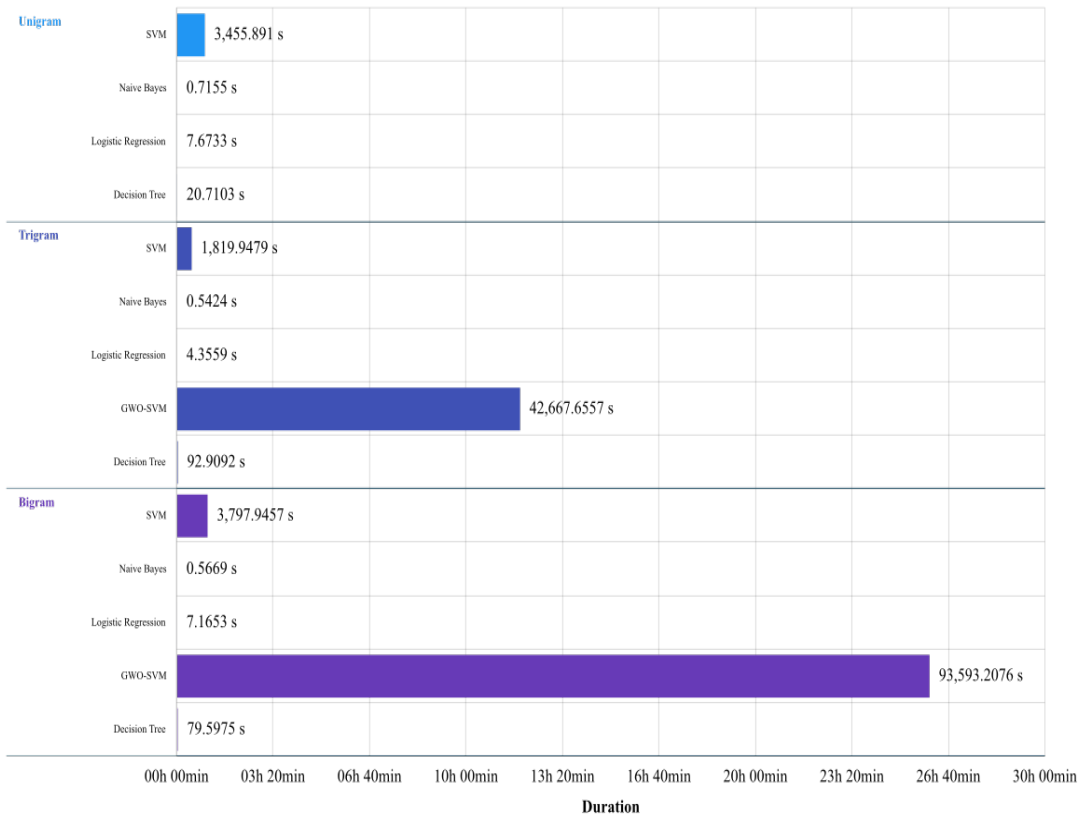


(b) results of the classification using stemming

Figure 4.6: Comparison of four classification algorithms using lemmatization and BOW representation



(a) Duration of classification using lemmatization



(b) Duration of classification using stemming

Figure 4.7: Training and prediction time for the four classification algorithms using BOW representation

### 4.5.2 Optimization (Bio-inspired model)

In this subsection, we preview the results of the proposed method using the bag of word ponderation. We illustrated the fixed parameter of the GWO-SVM model in Table 4.7, and then the execution time ( training + prediction +optimization) and f1-measure (max (f1-score macro) ) are shown in Table 4.8. After that, we illustrated the convergence curve of the model in figures 4.8, 4.9.

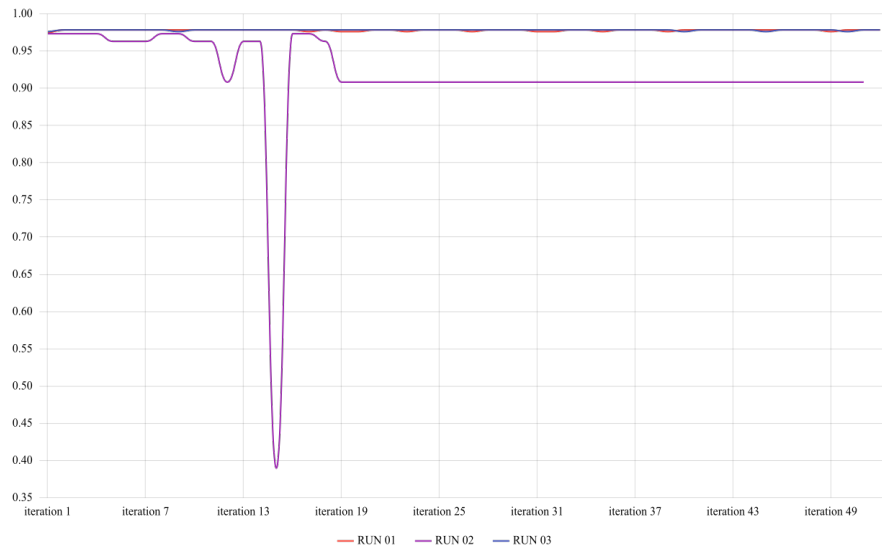
Parameter	Description	Value
Population	Number of gray wolves (Initial population), number of search agents to find the optimum	30
Max_itterations	The maximum number of iterations which can be optimized	100
Max_runs	The number of independent optimization runs to be done	3
Max_stagnating_pop	If the score remains unchanged for the number of generations defined by this parameter, the optimization will be stopped for that particular run	50
C	Hyperparameter how much to avoid misclassifying each training example	$\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$
gamma	The gamma parameter is the inverse of the standard deviation of the RBF kernel	$\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$
degree	It is the degree of the polynomial kernel function	$\{3, 4\}$

Table 4.7: Comparison of four classification algorithms for the suicidal ideation detection task using stemming and BOW representation

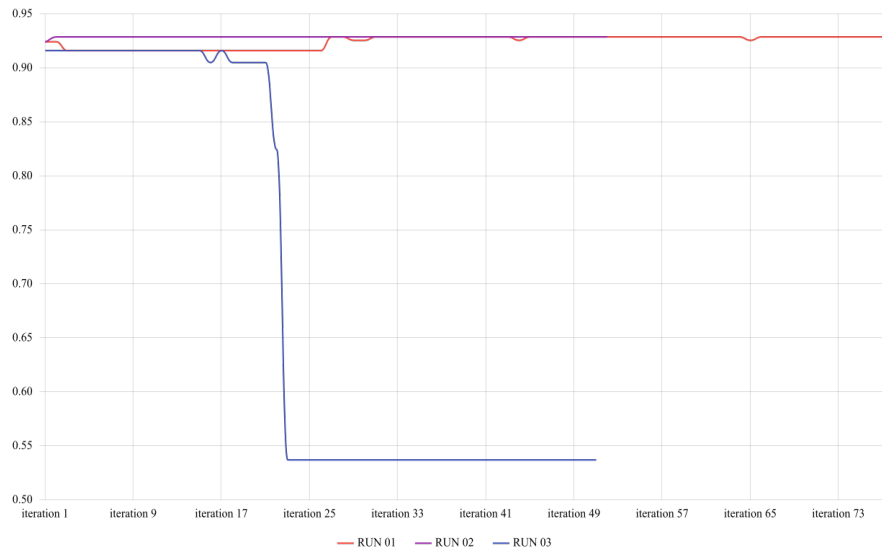
		Uni-Gram	Bi-Gram	Tri-Gram
Lemmatization	F1-mesure	0.9788	0.9289	0.814
	Durration	158,907.53	109,589.35	47,550.06
Stemming	F1-mesure	–	0.933	0.816
	Durration	–	93,593.2	42,667.65

Table 4.8: Comparison of classification results(execution time and f1-score 'macro') for the suicidal ideation detection task using stemming and lemmatization by proposed model GWO-SVM

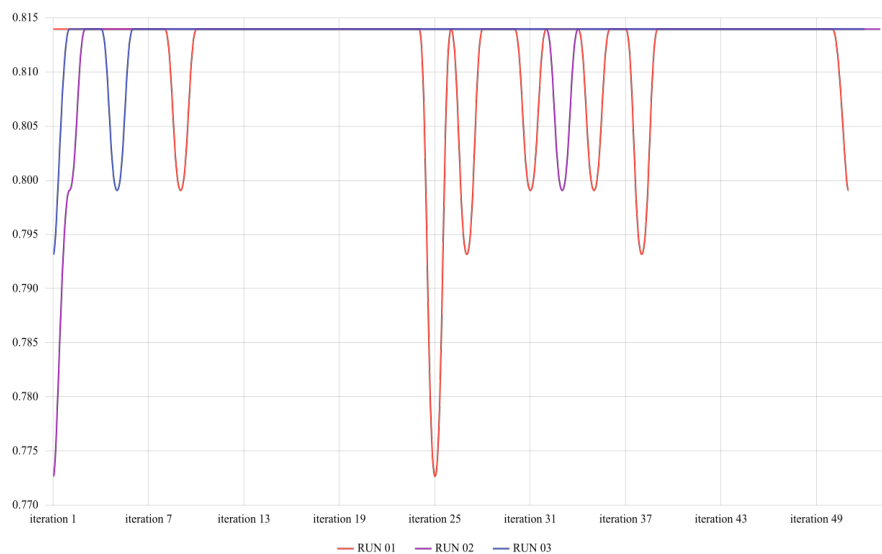
The convergence curve of the bio-inspired model using lemmatization and bag of word ponderation is illustrated in Figure 4.8. Subfigures (a), (b), and (c) show the convergence curves of the model using Uni-Gram, Bi-Gram, and Tri-Gram, respectively, with applying a multi-start strategy, in which execute the suggested method three times (3 runs).



(a) Convergence curve of the GWO-SVM model using Uni-Gram

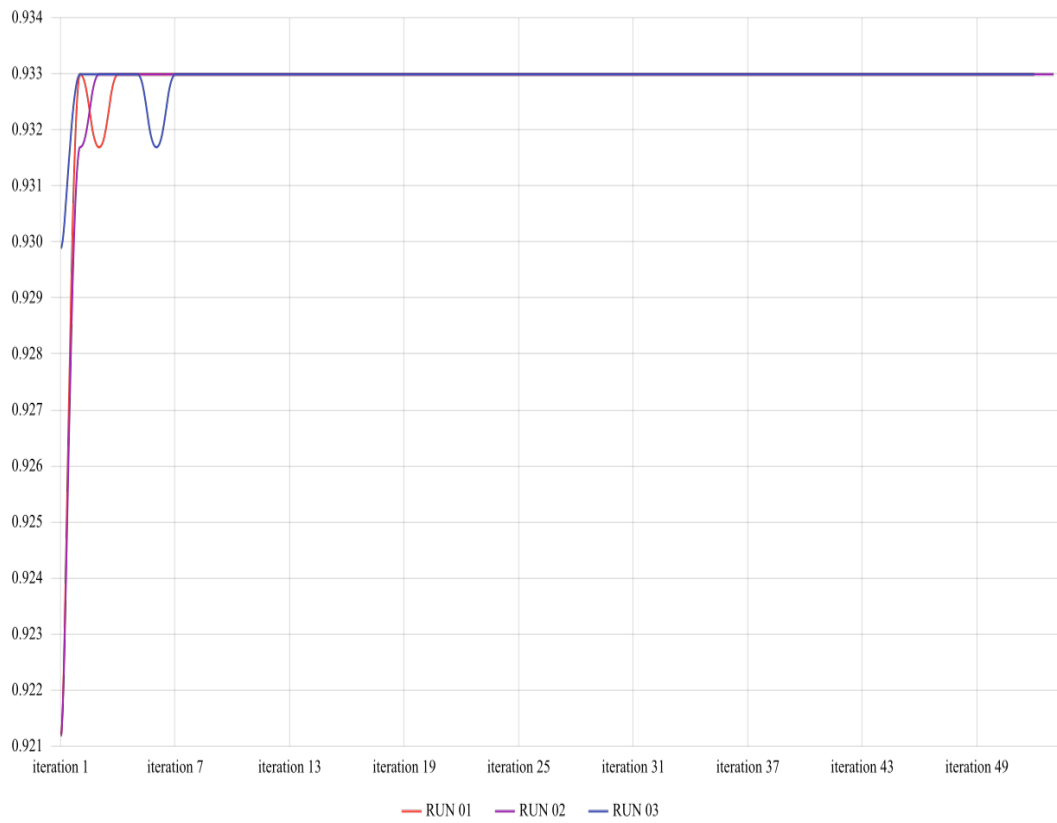


(b) Convergence curve of the GWO-SVM model using Bi-Gram

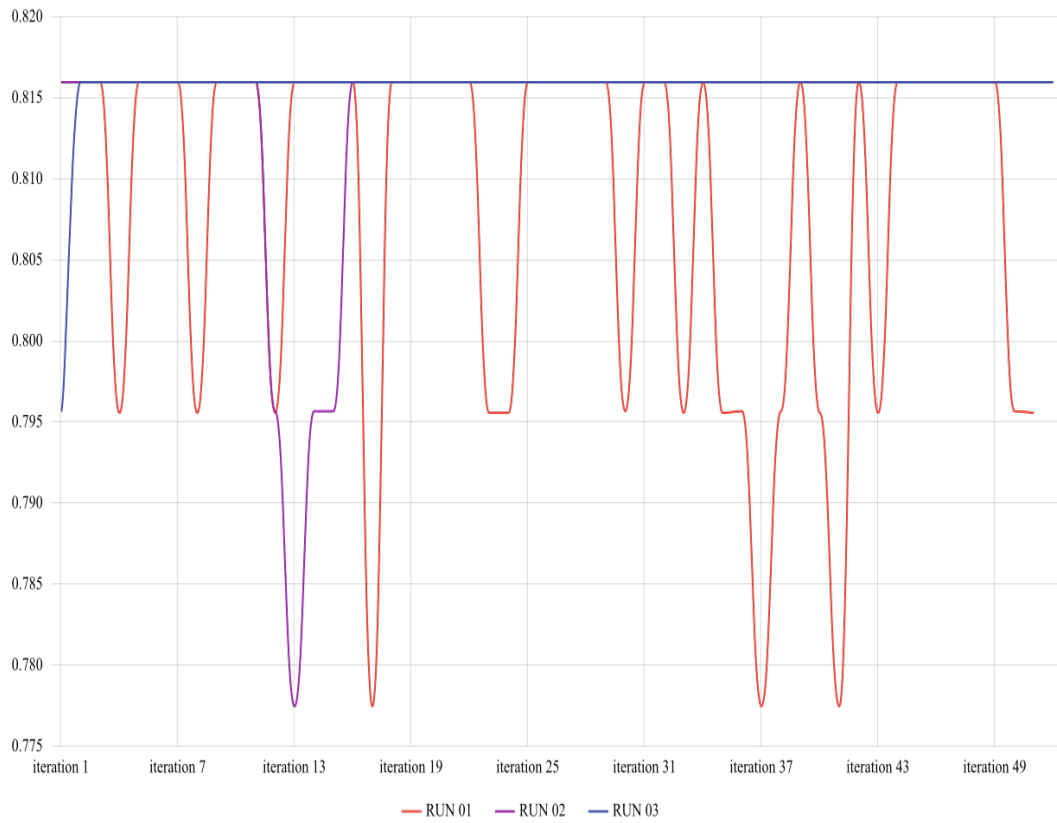


(c) Convergence curve of the GWO-SVM model using Tri-Gram

Figure 4.8: Convergence curve of the GWO-SVM model using Lemmatization



(a) Convergence curve of the GWO-SVM model using Bi-Gram



(b) Convergence curve of the GWO-SVM model using Tri-Gram

Figure 4.9: Convergence curve of the GWO-SVM model using Stemming

## 4.6 Interpretation of results and comparative study

After reviewing the classification results using four supervised learning classifiers and a bio-inspired model and measuring with f1-score, we took, for instance, classification results with SVM and GWO-SVM using lemmatization and bow weighing, bio-inspired model improved the classification results (0.001 %). As for the rest of the experiences, we note that the suggested method Didn't get better results than SVM.

As for the execution time, we noticed a huge difference (approximately 58 times slower than SVM), which means this approach is very expensive. Especially when compared with logistic regression.

## 4.7 Application (Web version)

The application of this work (web version) is implemented on Flask framework, HTML, CSS, Javascript.

The Figure 4.10 illustrates the word cloud of the most important word (token) after preprocessing text data based on the label of tweets (positive or negative). Subfigure 01 represents the mean of ten-fold cross-validation of the classification results using the four classifier plus bio-inspired model. Moreover showing the execution time of each classifier.

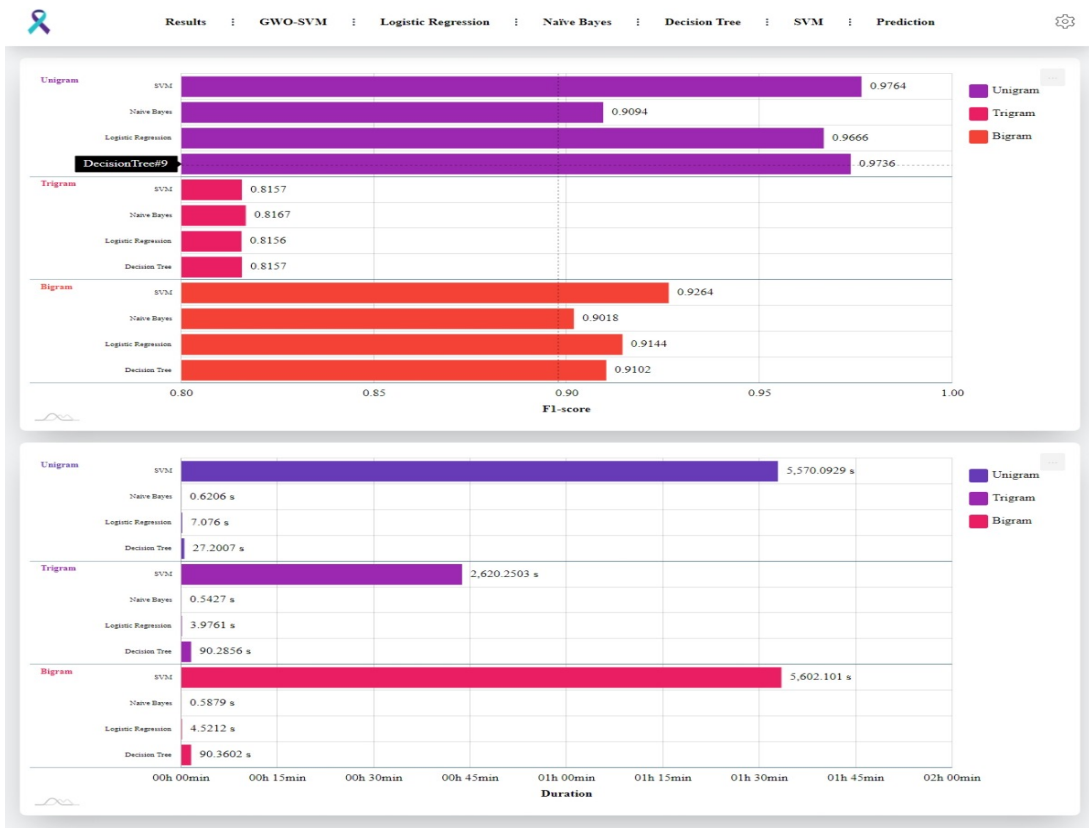
The 4.11a is shown the execution of the proposed method, and 4.11b, 4.11c show the results obtained from the classifier(Confusion Matrix, ROC, mean of [MCC, f1-score, Accuracy, Recall, Precision])

The result of the classification of ML classifiers and the GWO-SVM model, For instance, the death note shows the **real** suicide letter, and Figure 4.12 represents the prediction results.

suicide letter: "Alcohol and other drugs provided a way to ignore the realities of my situation. It was easy to spend the night drinking and forget that I had no future to look forward to. I never liked what alcohol did to me, but it was better than facing my existence honestly. I haven't touched alcohol or any other drug in over seven months (and no drugs or alcohol will be involved when I do this) and this has forced me to evaluate my life in an honest and clear way. There's no future here. The darkness will always be with me."



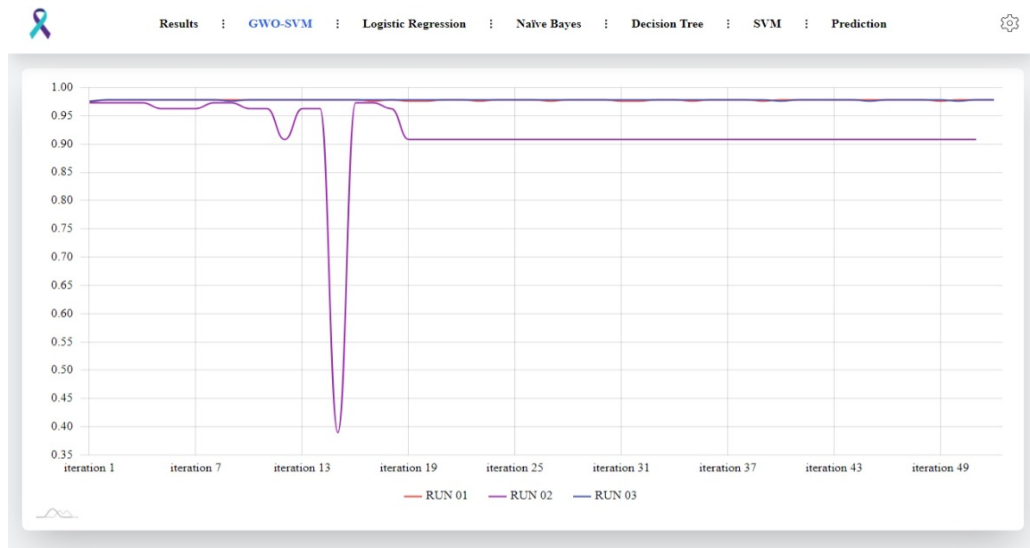
(a)



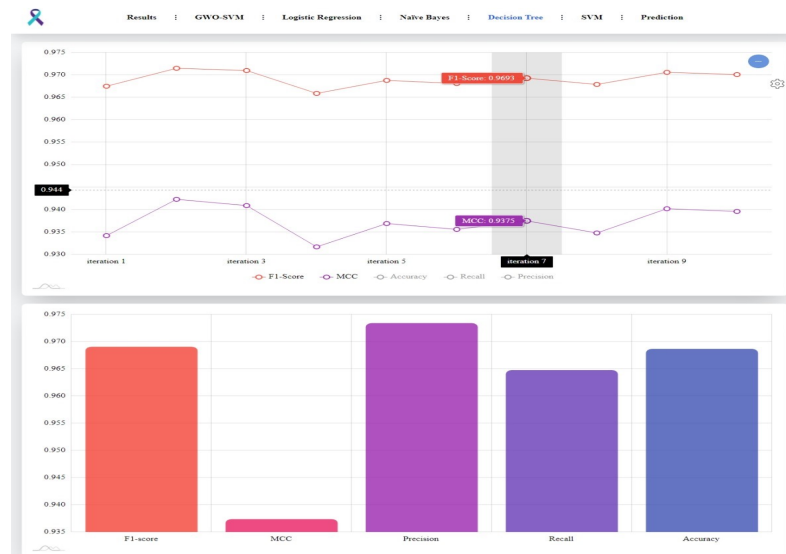
(b)

Figure 4.10: Word Cloud & mean of f1-score and execution time of classifiers

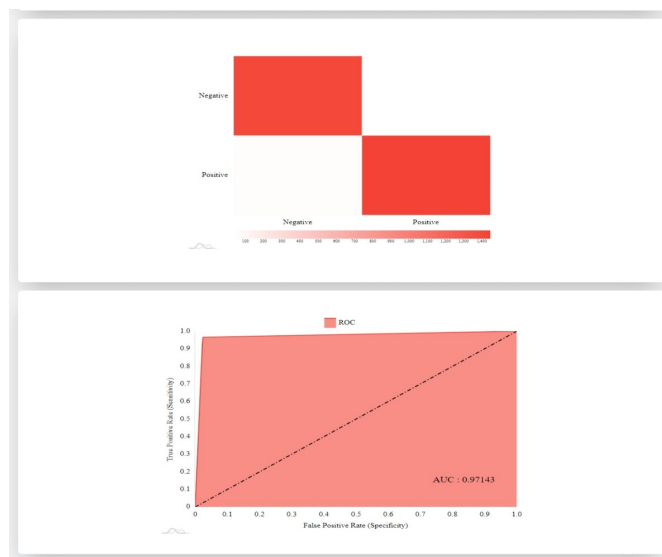




(a)



(b)



(c)

Figure 4.11: Classification results & GWO-SVM convergences



Figure 4.12: The prediction section

## 4.8 Conclusion

As the detection of suicidal ideation in social media continues to be a topical issue, the need for automatic suicidal thoughts detection systems becomes more evident. Throughout this chapter, we have presented our different experiments carried out in the context of sentiment analysis for suicidal thought detection using supervised machine learning algorithms and the proposed bio-inspired model (GWO-SVM).

# General conclusion

Integrating Machine Learning methods into suicide care offers new directions for the improvement of detection of suicide ideation and the possibility for early suicide prevention. Our work takes part in this journey towards the technological improvement in convolutional linguistics to be shared within the research community and successfully implemented in mental health care.

Our study presented an approach to recognize suicide ideation signs in Twitter social media and focused on detecting the most effective performance improvement solutions. We built our system on a Twitter data corpus created by suicide-indicative and non-suicidal posts for such purposes. We used different data representation techniques to reformulate the text of the posts into the presentation that our system can recognize. In particular, we characterized a closer connection between suicidal thoughts and language usage by applying various NLP and text classification techniques such as stemming, lemmatization TF-IDF, BOW with Uni-Gram, Bi-Gram, and Tri-Gram representation.

Based on our experiment, the proposed GWO-SVM hybrid model considerably does not improve text classification accuracy. The main reason why the model is not superior to other machine learning classifiers is that it takes much time to train, predict and optimize.

Using this approach, we can ensure that the hybrid model cannot effectively improve the prediction results we try to prove in our experiment.

According to our comparative evaluation, we specifically demonstrated the weakness of GWO-SVM. It resulted in the lowest performance among other classification approaches chosen for our experiment. We achieved a no-improvement through the hyper-parameter optimization based on the adaptive hyper-parameter tuning for the support vector machine.

Although our research findings show that the performances of applied classification approaches are reasonably good, the absolute value of the metrics indicates that this is a challenging task worthy of further study. We might try to access a larger dataset with suicide ideation content and a new dataset with related topics in our future work. We might examine the correlation between suicidal ideation and family environment, weather, etc. Both datasets will be collected from different social media sources to demonstrate further and compare our proposed hybrid model. In addition, the performance of the datasets will be applied for further investigation with other machine learning, deep learning classifiers, such as XGBoost, Random Forest, C-LSTM, RNN and their combined models, accompanied by various parameter optimization evaluations.

The limitation of our experiment can be found in its data deficiency and annotation bias. Data deficiency is one of the most critical issues of current research, where mainly supervised learning techniques are applied. They usually require a manual annotation. However, there are not enough annotated data to support further research. Another issue is the annotation bias caused by manual labelling with some predefined annotation rules. In some cases, the annotation

may lead to bias of labels resulting in misleading evidence to confirm the suicide action of the authors.

We believe that our study can contribute to future machine learning research for building an easily accessible and highly effective suicide detection and reporting system implemented in social media networks as an efficient intervention point between at-risk individuals and mental health services.

# Bibliography

- [1] Xin-She Yang. *Nature-Inspired Optimization Algorithms*. 1st. NLD: Elsevier Science Publishers B. V., 2014. ISBN: 0124167438.
- [2] Anthony Liew. “Understanding Data, Information, Knowledge And Their Inter-Relationships”. In: *Journal of Knowledge Management Practice* Vol. 7 (June 2007).
- [3] Sandro Saitta. *Data, Information, Knowledge and Wisdom*. 2007. URL: <http://www.dataminingblog.com/data-information-knowledge-and-wisdom/>.
- [4] Arun K Pujari. *Data mining techniques*. Universities press, 2001, pp. 45–47.
- [5] H. Chen et al. *Medical Informatics: Knowledge Management and Data Mining in Biomedicine*. Integrated Series in Information Systems. Springer US, 2005. ISBN: 9780387243818. URL: <https://books.google.dz/books?id=jMpSav7-WFOC>.
- [6] R. Lefébure and G. Venturi. *Gestion de la relation client*. Solutions d’entreprise. Eyrolles, 2004. ISBN: 9782212113310. URL: <https://books.google.dz/books?id=Z8-FhuB9k0YC>.
- [7] Margaret H. Dunham. *Data Mining: Introductory and Advanced Topics*. USA: Prentice Hall PTR, 2002. ISBN: 0130888923.
- [8] Santosh M Tondare Shital Gheware Anand Shivdas Kejkar. *Data Mining :Task, Tools, Techniques and Applications*. [https://www.researchgate.net/publication/275638874\\_Data\\_Mining\\_Task\\_Tools\\_Techniques\\_and\\_Applications](https://www.researchgate.net/publication/275638874_Data_Mining_Task_Tools_Techniques_and_Applications). 2014.
- [9] *Data Mining – Techniques and Applications*. <https://ifourconsultancy.wordpress.com/tag/content-management-systems/>.
- [10] ©. Tan and Steinbach. “Data Mining Classification : Basic Concepts , Decision Trees , and Model Evaluation”. In: 2004.
- [11] Sai Deepesh. 2020. URL: <https://www.datamining365.com/2020/02/data-generalization-summarization.html>.
- [12] S. Sumathi and S.N. Sivanandam. *Introduction to Data Mining and its Applications*. Studies in Computational Intelligence. Springer Berlin Heidelberg, 2006. ISBN: 9783540343516. URL: <https://books.google.dz/books?id=06L1BwAAQBAJ>.
- [13] Gardarin Georges. *Bases de données. Objet et relationnel*. 1999, p. 258.
- [14] Usama M Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, et al. “Knowledge Discovery and Data Mining: Towards a Unifying Framework.” In: *KDD*. Vol. 96. 1996, pp. 82–88.
- [15] Yongjian Fu. “Data mining”. In: *Potentials, IEEE* 16 (Nov. 1997), pp. 18 –20. DOI: [10.1109/45.624335](https://doi.org/10.1109/45.624335).
- [16] Christopher J.C. Burges. “A Tutorial on Support Vector Machines for Pattern Recognition”. In: *Data Mining and Knowledge Discovery* 2.2 (1998), pp. 121–167. ISSN: 1573-756X. DOI: [10.1023/A:1009715923555](https://doi.org/10.1023/A:1009715923555). URL: <https://doi.org/10.1023/A:1009715923555>.

- [17] Shili Peng et al. “Improved support vector machine algorithm for heterogeneous data”. In: *Pattern Recognition* 48.6 (2015), pp. 2072–2083. ISSN: 0031-3203. DOI: <https://doi.org/10.1016/j.patcog.2014.12.015>. URL: <https://www.sciencedirect.com/science/article/pii/S0031320314005263>.
- [18] Mirza Cilimkovic. “Neural Networks and Back Propagation Algorithm”. In: 2010.
- [19] Sung Eun Kim and Il Seo. “Artificial Neural Network ensemble modeling with conjunctive data clustering for water quality prediction in rivers”. In: *Journal of Hydro-environment Research* 9 (Apr. 2015). DOI: [10.1016/j.jher.2014.09.006](https://doi.org/10.1016/j.jher.2014.09.006).
- [20] Xizhao Wang and Weipeng Cao. “Non-iterative approaches in training feed-forward neural networks and their applications”. In: *Soft Computing* 22.11 (2018), pp. 3473–3476. ISSN: 1433-7479. DOI: [10.1007/s00500-018-3203-0](https://doi.org/10.1007/s00500-018-3203-0). URL: <https://doi.org/10.1007/s00500-018-3203-0>.
- [21] Renguang Zuo. “Machine Learning of Mineralization-Related Geochemical Anomalies: A Review of Potential Methods”. In: *Natural Resources Research* 26.4 (2017), pp. 457–464. ISSN: 1573-8981. DOI: [10.1007/s11053-017-9345-4](https://doi.org/10.1007/s11053-017-9345-4). URL: <https://doi.org/10.1007/s11053-017-9345-4>.
- [22] Chetna Kaushal and Deepika Koundal. “Recent trends in big data using hadoop”. In: *International Journal of Informatics and Communication Technology (IJ-ICT)* 8 (Apr. 2019), p. 39. DOI: [10.11591/ijict.v8i1.pp39-49](https://doi.org/10.11591/ijict.v8i1.pp39-49).
- [23] Wei Chen et al. “Visual analysis of user-driven association rule mining”. In: *Journal of Visual Languages & Computing* 42 (2017), pp. 76–85. ISSN: 1045-926X. DOI: <https://doi.org/10.1016/j.jvlc.2017.08.007>. URL: <https://www.sciencedirect.com/science/article/pii/S1045926X17300071>.
- [24] “Introduction to Optimization”. In: *Practical Genetic Algorithms*. John Wiley & Sons, Ltd, 2003. Chap. 1, pp. 1–25. ISBN: 9780471671749. DOI: <https://doi.org/10.1002/0471671746.ch1>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/0471671746.ch1>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/0471671746.ch1>.
- [25] Christian Blum and Andrea Roli. “Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison”. In: *ACM Comput. Surv.* 35.3 (2003), pp. 268–308. ISSN: 0360-0300. DOI: [10.1145/937503.937505](https://doi.org/10.1145/937503.937505). URL: <https://doi.org/10.1145/937503.937505>.
- [26] Jeffrey Tuhtan. “Cost Optimization of Small Hydropower”. PhD thesis. Oct. 2007.
- [27] Ke-Lin Du and M. N. S. Swamy. *Search and Optimization by Metaheuristics: Techniques and Algorithms Inspired by Nature*. 1st. Birkhäuser Basel, 2016. ISBN: 3319411918.
- [28] Jucemar Monteiro. “Algorithms to improve area density utilization, routability and timing during detailed placement and legalization of VLSI circuits”. PhD thesis. May 2019.
- [29] Roya Amiri, Javad Majrouhi Sardroud, and Borja García de Soto. “BIM-based Applications of Metaheuristic Algorithms to Support the Decision-making Process: Uses in the Planning of Construction Site Layout”. In: vol. 196. June 2017. DOI: [10.1016/j.proeng.2017.08.030](https://doi.org/10.1016/j.proeng.2017.08.030).
- [30] Fred Glover. “Future Paths for Integer Programming and Links to Artificial Intelligence”. In: *Comput. Oper. Res.* 13.5 (1986), pp. 533–549. ISSN: 0305-0548. DOI: [10.1016/0305-0548\(86\)90048-1](https://doi.org/10.1016/0305-0548(86)90048-1). URL: [https://doi.org/10.1016/0305-0548\(86\)90048-1](https://doi.org/10.1016/0305-0548(86)90048-1).
- [31] Christos Voudouris and Edward P. K. Tsang. “Guided Local Search”. In: *Handbook of Metaheuristics*. Ed. by Fred Glover and Gary A. Kochenberger. Boston, MA: Springer US, 2003, pp. 185–218. ISBN: 978-0-306-48056-0. DOI: [10.1007/0-306-48056-5\\_7](https://doi.org/10.1007/0-306-48056-5_7). URL: [https://doi.org/10.1007/0-306-48056-5\\_7](https://doi.org/10.1007/0-306-48056-5_7).

- [32] Helena Ramalhinho Lourenço, Olivier C. Martin, and Thomas Stützle. “Iterated Local Search: Framework and Applications”. In: *Handbook of Metaheuristics*. Ed. by Michel Gendreau and Jean-Yves Potvin. International Series in Operations Research & Management Science. Springer, 2019. Chap. 0, pp. 129–168. DOI: [10.1007/978-3-319-91086-4](https://doi.org/10.1007/978-3-319-91086-4).
- [33] Hari Dubey, Bijaya Panigrahi, and Manjaree Pandit. “Bio-inspired optimisation for economic load dispatch: A review”. In: *International Journal of Bio-Inspired Computation* 6 (Mar. 2014), pp. 7–21. DOI: [10.1504/IJBIC.2014.059967](https://doi.org/10.1504/IJBIC.2014.059967).
- [34] S Binitha, S Siva Sathya, et al. “A survey of bio inspired optimization algorithms”. In: *International journal of soft computing and engineering* 2.2 (2012), pp. 137–151.
- [35] Farkhund Iqbal et al. “A Hybrid Framework for Sentiment Analysis Using Genetic Algorithm Based Feature Reduction”. In: *IEEE Access* 7 (2019), pp. 14637–14652. DOI: [10.1109/ACCESS.2019.2892852](https://doi.org/10.1109/ACCESS.2019.2892852).
- [36] Riccardo Poli, James Kennedy, and Tim Blackwell. “Particle swarm optimization”. In: *Swarm Intelligence* 1.1 (2007), pp. 33–57. ISSN: 1935-3820. DOI: [10.1007/s11721-007-0002-0](https://doi.org/10.1007/s11721-007-0002-0). URL: <https://doi.org/10.1007/s11721-007-0002-0>.
- [37] M. Clerc and J. Kennedy. “The particle swarm - explosion, stability, and convergence in a multidimensional complex space”. In: *IEEE Transactions on Evolutionary Computation* 6.1 (2002), pp. 58–73. DOI: [10.1109/4235.985692](https://doi.org/10.1109/4235.985692).
- [38] Xinmiao Li, Jing Li, and Yukeng Wu. “A Global Optimization Approach to Multi-Polarity Sentiment Analysis”. In: *PLOS ONE* 10 (Apr. 2015), e0124672. DOI: [10.1371/journal.pone.0124672](https://doi.org/10.1371/journal.pone.0124672).
- [39] Abd. Samad Hasan Basari et al. “Opinion Mining of Movie Review using Hybrid Method of Support Vector Machine and Particle Swarm Optimization”. In: *Procedia Engineering* 53 (2013). Malaysian Technical Universities Conference on Engineering Technology 2012, MUCET 2012, pp. 453–462. ISSN: 1877-7058. DOI: <https://doi.org/10.1016/j.proeng.2013.02.059>. URL: <https://www.sciencedirect.com/science/article/pii/S1877705813001781>.
- [40] Marco Dorigo and Gianni Di Caro. “The Ant Colony Optimization Meta-Heuristic”. In: *New Ideas in Optimization*. GBR: McGraw-Hill Ltd., UK, 1999, pp. 11–32. ISBN: 0077095065.
- [41] Xin-She Yang. *Firefly Algorithms for Multimodal Optimization*. 2010. arXiv: [1003.1466](https://arxiv.org/abs/1003.1466) [math.OA].
- [42] Xin-She Yang and He Xingshi. “Firefly Algorithm: Recent Advances and Applications”. In: *International Journal of Swarm Intelligence* 1 (Aug. 2013). DOI: [10.1504/IJSI.2013.055801](https://doi.org/10.1504/IJSI.2013.055801).
- [43] Dervis Karaboga and Bahriye Basturk. “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm”. In: *Journal of global optimization* 39.3 (2007), pp. 459–471.
- [44] Ruby Dhurve and M. Seth. “Weighted Sentiment Analysis Using Artificial Bee Colony Algorithm”. In: 2015.
- [45] Xin-She Yang. “Nature-inspired optimization algorithms: Challenges and open problems”. In: *Journal of Computational Science* 46 (2020). 20 years of computational science, p. 101104. ISSN: 1877-7503. DOI: <https://doi.org/10.1016/j.jocs.2020.101104>. URL: <https://www.sciencedirect.com/science/article/pii/S1877750320300144>.
- [46] Hossam Faris et al. “Grey wolf optimizer: a review of recent variants and applications”. In: *Neural Computing and Applications* 30.2 (2018), pp. 413–435. ISSN: 1433-3058. DOI: [10.1007/s00521-017-3272-5](https://doi.org/10.1007/s00521-017-3272-5). URL: <https://doi.org/10.1007/s00521-017-3272-5>.



- [47] Yazid Tikhamarine, Doudja Souag-Gamane, and Ozgur Kisi. “A new intelligent method for monthly streamflow prediction: hybrid wavelet support vector regression based on grey wolf optimizer (WSVR–GWO)”. In: *Arabian Journal of Geosciences* 12.17 (2019), p. 540. ISSN: 1866-7538. DOI: [10.1007/s12517-019-4697-1](https://doi.org/10.1007/s12517-019-4697-1). URL: <https://doi.org/10.1007/s12517-019-4697-1>.
- [48] Prerna Sharma et al. “The health of things for classification of protein structure using improved grey wolf optimization”. In: *The Journal of Supercomputing* 76 (Feb. 2020). DOI: [10.1007/s11227-018-2639-4](https://doi.org/10.1007/s11227-018-2639-4).
- [49] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. “Exploration and Exploitation in Evolutionary Algorithms: A Survey”. In: *ACM Comput. Surv.* 45.3 (July 2013). ISSN: 0360-0300. DOI: [10.1145/2480741.2480752](https://doi.org/10.1145/2480741.2480752). URL: <https://doi.org/10.1145/2480741.2480752>.
- [50] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.
- [51] Lipo Wang. *Support vector machines: theory and applications*. Vol. 177. Springer Science & Business Media, 2005.
- [52] Xin-She Yang, Suash Deb, and Simon Fong. “Accelerated Particle Swarm Optimization and Support Vector Machine for Business Optimization and Applications”. In: vol. 136. Mar. 2012. ISBN: 978-3-642-22184-2. DOI: [10.1007/978-3-642-22185-9\\_6](https://doi.org/10.1007/978-3-642-22185-9_6).
- [53] D. Kuhlman. *A Python Book: Beginning Python, Advanced Python, and Python Exercises*. Platypus Global Media, 2011. ISBN: 9780984221233. URL: <https://books.google.dz/books?id=1FL-ygAACAAJ>.
- [54] G. Lindstrom. “Programming with Python”. In: *IT Professional* 7 (2005), pp. 10–16.
- [55] Guido van Rossum. “Python Programming Language”. In: *Proceedings of the 2007 USENIX Annual Technical Conference, Santa Clara, CA, USA, June 17-22, 2007*. Ed. by Jeff Chase and Srinivasan Seshan. USENIX, 2007, p. 36.
- [56] Fabian Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830. URL: <http://jmlr.org/papers/v12/pedregosa11a.html>.
- [57] Edward Loper and Steven Bird. “NLTK: the Natural Language Toolkit”. In: *CoRR* cs.CL/0205028 (July 2002). DOI: [10.3115/1118108.1118117](https://doi.org/10.3115/1118108.1118117).
- [58] CHRISTIANE FELLBAUM. “The Encyclopedia of Applied Linguistics”. In: Nov. 2012. ISBN: 9781405194730. DOI: [10.1002/9781405198431.wbeal1285](https://doi.org/10.1002/9781405198431.wbeal1285).
- [59] Fernando Perez and Brian E. Granger. “IPython: A System for Interactive Scientific Computing”. In: *Computing in Science Engineering* 9.3 (2007), pp. 21–29. DOI: [10.1109/MCSE.2007.53](https://doi.org/10.1109/MCSE.2007.53).
- [60] Bernadette M. Randles et al. “Using the Jupyter Notebook as a Tool for Open Science: An Empirical Study”. In: *Proceedings of the 17th ACM/IEEE Joint Conference on Digital Libraries*. JCDL '17. Toronto, Ontario, Canada: IEEE Press, 2017, pp. 338–339. ISBN: 9781538638613.
- [61] R. Sharnagat. “Named entity recognition: A literature survey,” in: *Center For Indian Language Technology* (2014).
- [62] Krisztian Balog. *Entity-oriented search*. Springer Nature, 2018.
- [63] Martin F Porter. “An algorithm for suffix stripping”. In: *Program* (1980).
- [64] Vimala Balakrishnan and Lloyd-Yemoh Ethel. “Stemming and Lemmatization: A Comparison of Retrieval Performances”. In: *Lecture Notes on Software Engineering* 2 (Jan. 2014), pp. 262–267. DOI: [10.7763/LNSE.2014.V2.134](https://doi.org/10.7763/LNSE.2014.V2.134).

- 
- [65] Péter Halácsy. “Benefits of deep NLP-based Lemmatization for Information Retrieval”. In: *CLEF*. 2006.
- [66] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.
- [67] Alice Zheng and Amanda Casari. *Feature engineering for machine learning: principles and techniques for data scientists*. ” O’Reilly Media, Inc.”, 2018.
- [68] D. H. Wolpert and W. G. Macready. “No Free Lunch Theorems for Optimization”. In: *Trans. Evol. Comp* 1.1 (Apr. 1997), pp. 67–82. ISSN: 1089-778X. DOI: [10.1109/4235.585893](https://doi.org/10.1109/4235.585893). URL: <https://doi.org/10.1109/4235.585893>.
- [69] Juan Carlos Pereira-Kohatsu et al. “Detecting and Monitoring Hate Speech in Twitter”. In: *Sensors* 19.21 (2019). ISSN: 1424-8220. DOI: [10.3390/s19214654](https://doi.org/10.3390/s19214654). URL: <https://www.mdpi.com/1424-8220/19/21/4654>.
- [70] Nika Mostahinic. “Respiratory Rate Estimation Using WiFi Channel State Information - A Machine Learning Approach”. PhD thesis. June 2020. DOI: [10.13140/RG.2.2.30773.29924](https://doi.org/10.13140/RG.2.2.30773.29924).
- [71] Seema Sharma et al. “Machine learning techniques for data mining: A survey”. In: *2013 IEEE International Conference on Computational Intelligence and Computing Research*. 2013, pp. 1–6. DOI: [10.1109/ICCIC.2013.6724149](https://doi.org/10.1109/ICCIC.2013.6724149).
- [72] Irina Rish et al. “An empirical study of the naive Bayes classifier”. In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. Vol. 3. 22. 2001, pp. 41–46.
- [73] Gianni D’Angelo and Salvatore Rampone. “Feature extraction and soft computing methods for aerospace structure defect classification”. In: *Measurement* 85 (2016), pp. 192–209. ISSN: 0263-2241. DOI: <https://doi.org/10.1016/j.measurement.2016.02.027>. URL: <https://www.sciencedirect.com/science/article/pii/S0263224116001044>.

