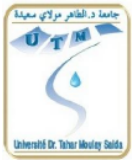
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE



UNIVERSITÉ DR TAHAR MOULAY -SAIDA-

FACULTÉ DE TECHNOLOGIE

MEMOIRE DE MASTER

OPTION : MICR

THEME

La détection automatique du diabète basée sur le modèle essaim de poisson.

Encadreur :
HAMOU REDA MOHAMED
Co-Encadreur :
LOKBANI AHMED CHAOUKI

Présenté par :
HABBAZ ABDELKRIM

Promotion : JUIN 2016

Remerciements

En premier lieu, je remercie DIEU de m'avoir aidé et donner la force et la volonté pour achever ce modeste travail. Par la suite ce travail a été réalisé sous la direction du monsieur **Ahmed Chaouki LOKBANI** qu'il trouve ici ma profonde reconnaissance et mes sincères remerciements, pour ses encouragements, son aide ses conseils précieux et aussi ses idées pour la réalisation de ce mémoire. J'adresse mes vifs remerciements à Monsieur **HAMOU Reda**. Je tiens aussi à remercier, les membres du jury qui ont accepté de juger ce travail. Je remercie aussi l'ensemble des enseignants ayant intervenu aux cours de notre première année de post-graduation, trouvent ici l'expression de ma gratitude. Également, je dois à mes camarades et à l'ensemble de mes collègues . Et enfin, je tien à remercier mes chères parents, mes frères et mes oncle pour leurs encouragements, leurs aides et leurs grande patience avec moi.

Dédicace

Je dédie ce travail à celle qui m'a donné la vie, le symbole de tendresse qui s'est sacrifiée pour mon bonheur et ma réussite, à ma mère, à mon père ,à mon frère et Soeurs, à La Famille HABBAZ et KADOUS , à mes amis, à tous mes camarades de promotion MICR2 ,à mes 2 encadreurs Mr LOKBANI Ahmed Chaouki et MR HAMMOU Reda MED, à tous l'ensemble des étudiants du département d'informatique et à tous ceux qui m'aiment.

HABBAZ Abdelkrim

Résumé

Comme on le sait bien le diabète est un problème de santé majeur et un maladie qui peut être très dangeureuse dans les pays qui sont en cours de développement et les pays développés et son incidence augmente de façon spectaculaire.dans cette memoire on va proposer un systeme de detection automatique de diabete basée sur un modele bioinspiré nommé essaim de poisson (en englais :FISH SWARM ou AFSA).AFSA (artificial fish-swarm algorithm) est l'une des meilleures méthodes d'optimisation parmi les algorithmes intelligence en essaim. Cet algorithme est inspiré par le collectif, le mouvement des poissons et de leurs différents comportements sociaux.dans le but d'atteindre des resultats acceptables il ya plusieurs parametres a ajuster dans le modele AFSA Parmi ces paramètres, le seuil visuel globale et le seuil visuel locale sont très significatifs, compte tenu du fait que le poisson artificiel essentiellement se déplacer en fonction de ces paramètres. Dans la norme AFSA, ces deux paramètres restent constants jusqu'à la fin de l'algorithme. Les grandes valeurs de ces paramètres augmentent la capacité de l'algorithme de recherche globale, tandis que les petites valeurs d'améliorer la capacité de recherche locale de l'algorithme.Cet algorithme a de nombreux avantages, y compris une grande vitesse de convergence, la flexibilité et une grande précision. dans cette memoire on va evaluer notre modele de AFSA dans le but de detection automatique de diabete.

mots clés : diabète , detection automatique, modele bioinspiré, essaim de poisson,FISH SWARM ,AFSA ,méthodes d'optimisation ,précision.

Abstract

As is well known Diabetes is a major health problem and a disease that can be very dangerous in countries that are being developed and developed countries, and its incidence increases so spectaculaire.in this work we will propose a Automatic system of the detection of the diabètes based on a model named fish swarm .AFSA (artificial fish-swarm algorithm) is one of the best methods of optimization algorithms among the swarm intelligence. This algorithm is inspired by the collective, the movement of fish and their different social behaviors.in order to achieve acceptable results there are several parameters to adjust in the AFSA model among These parameters we will use only step and visual witch are signifiant, considering the fact that the artificial fish essentially move in function of these parameters.these two parameters are kept constant until the end of the algorithm. in this work we will evaluate our AFSA model for the purpose of automatic detection of diabetes.

Textbf keywords : diabetes, automatic detection, bioinspired model FISH SWARM, AFSA, optimization methods, precision .

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Introduction generale

Introduction

Le diagnostic de la maladie de diabète via une interprétation abstraite des données est un problème de classification importante. Le diabète survient lorsque le corps est incapable de produire ou de répondre correctement à l'insuline qui est nécessaire pour réguler le glucose. Le diabète est non seulement un facteur contribuant à la maladie cardiaque, mais augmente également les risques de développer une maladie du rein, la cécité, lésions nerveuses, et les dommages des vaisseaux sanguins. Les statistiques montrent que plus de 80 pour cent des personnes atteintes de diabète meurent d'une certaine forme de maladies cardiaques ou des vaisseaux sanguins. Actuellement, il n'y a pas de remède pour le diabète ; cependant, il peut être contrôlé par l'injection d'insuline, en changeant les habitudes alimentaires, et faire des exercices physiques.

Les méta-heuristiques et les méthodes biomimetiqes constituent une classe de méthodes qui fournissent des solutions de bonnes qualités en un temps raisonnable à des problèmes difficiles pour lesquels il n'existe pas des méthodes classiques plus efficaces. Elles sont généralement des méthodes stochastiques itératives qui progressent vers un optimum global en évaluant une fonction « objectif ». L'Optimisation Par Essaim de Poisson est une branche de l'intelligence en essaim (une méta-heuristique basée sur la population), inspirée par le comportement des Poisson réelles.
L'algorithme de AFSA est bien adapté pour des problèmes d'optimisation discrets tels que les problèmes d'affectation quadratiques, la bioinformatique et la fouille de données.

Problématique

Les données du dataset sont des données abstraite pour la machine . Que nous apprennent ces données et quel est le degré de fiabilité des informations qui en découlent ? est la problématique de ce mémoire de master est aussi de voir quel est l'apport des Métaheuristiques dans l'amélioration des tâches de la fouille de données (Data Mining). Plus précisément on résout un problème d'optimisation NP-difficile qui est la detection de la maladie du diabéte et améliorer la classification supervisée (un taux de précision accéptable (élevé)) de la fouille de données, en utilisant les méthodes Bio-inspirée et les métaheuristiques .
Une solution prometteuse est d'utiliser une méthode Bio inspiré (AFSA) qui affecte une catégorie à un ensemble de données, classer ces données en deux classes différentes, la première classe contient des données des personnes malade et la seconde contient des données des personnes normales.

Objectif du travail

L'objectif de ce travail est proposer une techniques Bio inspiré pour la resolution d'une tache de Data Mining capable a donnée de bonnes résultats par rapport a d'autre obtenu par d'autres méthode de classification classique.
Il ya d'autres travailes qui utilisent cette algorithmes mais aucun de ces travail n'a l'utiliser pour la classification c'st l'avantage de notre travail .
Dans ce travail, nous avons utilisé une méthode Bio inspiré son principe consiste à apprendre à ranger des éléments dans des catégories prédéfinies en fonction de leur caractéristiques.
Savoir attribuer automatiquement une catégorie à un élément peut être très utile et en calculer des mesure de validation en essayant d'améliorer les mesures de validation obtenues par les algorithmes de classification classique.

Organisation du mémoire

Afin de répondre à cet objectif et mieux comprendre ce qui est ecrit, ce mémoire est structuré de la façon suivante :
Le deuxième chapitre est une présentation de la maladie du diabète avec des définitions et des explications sur les différentes types du diabète et leur affections sur le monde entier.

Le troisième chapitre est une présentation générale on ce qui concerne le domaine de DATA MINING historique et définition, les techniques utilisé .Nous parlerons aussi des outils a utilisé dans ce domaine et les but a atteindre.

Dans Le quatrième chapitre on parle des méthodes metaheuritique qui ont un but de résoudre des problème np-difficiles dans un temps reél ar l'utilisation des heuristique ,on va voir les définition est les meilleurs algorithmes utilisé dans les metaheuristique avec des exemples et des solutions.

Le cinquième chapitre concerne l'implémentation de notre algorithme proposé avec les résultats obtenus de l'expérimentation.
Et enfin on a une petite conclusion générale qui mets en œuvre un résumé générale de tous ce qu'on a vu .

CHAPITRE I :

PRÉSENTATION DE DIABÉTE

Chapitre 1

PRÉSENTATION DE DIABÉTE

1.1 Introduction

Le diabète est une maladie lourde de conséquences par ses complications. C'est pourquoi il constitue un problème de santé publique au niveau national et international dont le poids humain et économique augmente graduellement. En effet, ses complications en font une maladie dont la morbidité et la mortalité sont fortement accrues par rapport a la population générale. Actuellement, il y a 200 millions de diabétiques dans le monde. Plus de 90 % des diabétiques ont le diabète de type II , et seulement 10 % présentent le diabète de type I. En 2030, il y aura 330 millions de diabétiques dans le monde.
Il est la quatrième cause de décès. L'augmentation du nombre de diabétique est tellement rapide que l'organisation mondiale de la santé (OMS) l'a identifié comme étant une épidémie.
Cette épidémie du diabète est surtout due aux modifications du mode de vie, l'occidentalisation, l'alimentation très calorique, la sédentarité, le dépistage plus actif du diabète, le vieillissement de la population.D'après l'OMS, le nombre d'adulte diabétiques atteindra les 300 millions de malades en 2025 .

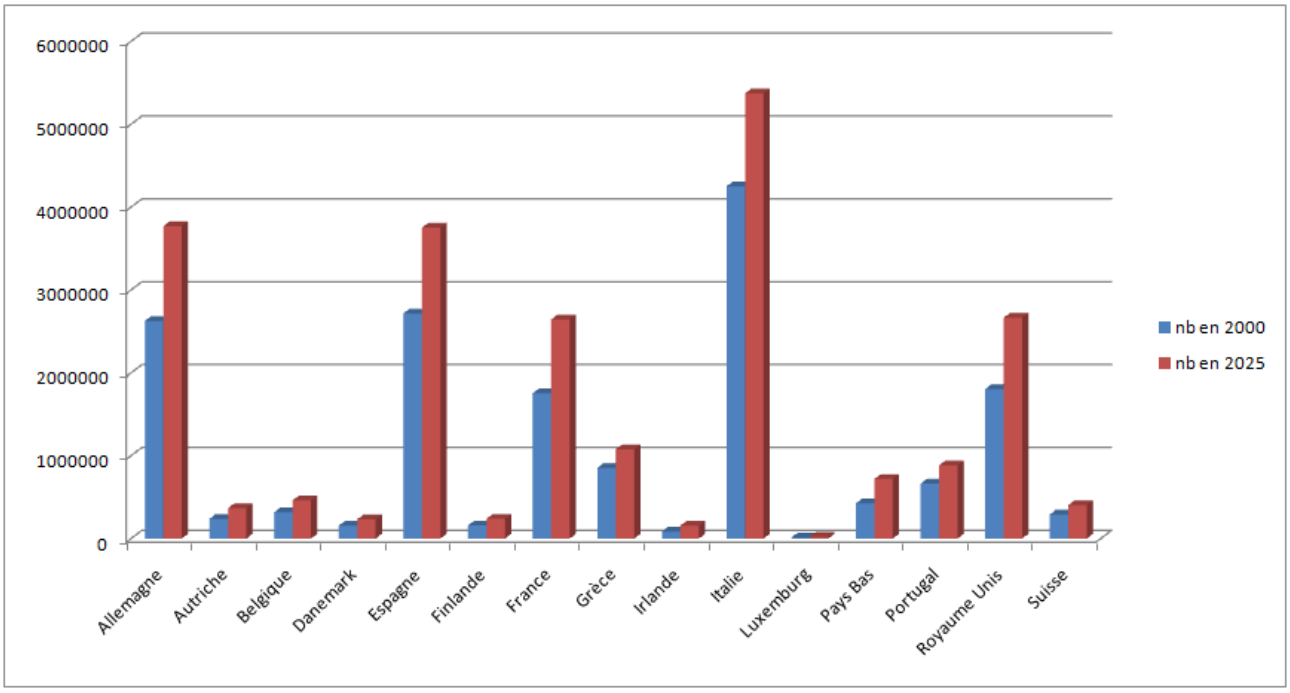


FIGURE 1.1 – Taux de diabétiques en 2000 et les prévisions pour le 2025

1.2 Définition

L'Organisation mondiale de la Santé définit le diabète comme un trouble du métabolisme d'étiologies multiples, caractérisé par une hyperglycémie chronique avec des troubles du métabolisme des glucides, lipides et de protéines résultant de défauts de sécrétion d'insuline, d'action de l'insuline, ou les deux.
Le diabète est une maladie chronique incurable causée par une carence ou un défaut d'utilisation de l'insuline

entrainant un excès de sucre dans le sang. Produite par le pancréas, l'insuline est une hormone qui permet au glucose (sucre) contenu dans les aliments d'être utilises par les cellules du corps humain. Les cellules disposent de toute cette énergie dont elles ont besoin pour fonctionner.
Si l'insuline est insuffisante ou si elle ne remplit pas son rôle adéquatement, comme c'est le cas dans le diabète, le glucose (sucre) ne peut pas servir de carburant aux cellules il s'accumulent dans le sang et ensuite déversé dans l'urine.A la longue ,l'hyperglycémie provoqué par la présence excessive de glucose dans le sang entraines certaines complications , notamment au niveau des yeux ,des reins,des nerfs,du coeur et des vaisseaux sanguins**[HAMIDI Yacine Nacer Eddine,2012]**.

1.3 Cause du diabète

La prévalence de cette maladie a été multipliée par cinq en moins de cinquante ans. Cette augmentation progressive est due à divers facteurs :

- le vieillissement global de la population ;
- l'augmentation de l'espérance de vie du diabétique ;
- l'augmentation de la fécondité des femmes diabétique ;
- l'augmentation de l'obésité ;
- l'incrémentation de la consommation des sucres raffinés.

Ainsi que d'autre facteurs qui peuvent servir comme déclencheur tels que :

- le sédentarisme ;
- les régimes riches en graisse et protéine ;
- les régimes riches en graisse et protéine ;
- la consommation réduite de fibre ;
- une alimentation déficiente en hydrate de carbone complexe et vitamine E ;
- une alimentation déficiente en hydrate de carbone complexe et vitamine E ;
- le stress chronique ;
- le tabagisme qui peut causer l'apparition de l'insulinoresistance.

1.4 Classification du diabète

Les critères pour le diagnostic et la classification du Diabète sucré (Diabetes Mellitus) ont été développé par un comité d'expert de l'Association Américaine de Diabète (ADA) et par un comité de l'OMS.
La classification du diabète se base principalement sur son étiologie et caractéristique physiopathologie. Le diabète est classé en quatre types :

- Diabète type 1 (DM1) ;
- Diabète type 2 (DM2) ;
- Autre types spécifique de diabète ;
- Diabète gestationnel (DMG).

Fréquemment les personnes souffrant de DM2 finissent par nécessité de l'insuline à une étape de leur vie, d'un autre coté, certains malades de DM1 peuvent progresser lentement ou avoir de longues périodes de rémission sans avoir besoin d'insuline. C'est à cause de ces cas que les termes insulinodépendant et non insulinodépendant ont été éliminés.

1.4.1 Diabète de type 1

Quand la maladie est diagnostiquée, la sécrétion d'insuline est déficiente mais pas inexistante. La sécrétion d'insuline est insuffisante aussi bien à jeun qu'en réponse au différents stimulus, ceci est une conséquence de l'autodestruction progressive et sélective des cellules Béta des îlots de Langerhans, ce qui affecte l'utilisation des hydrates de carbone, protéine et graisse.
Puisque un pancréas sain sécrète une quantité d'insuline beaucoup plus élevé de ce dont le corps à besoin, des mois ou des années peuvent passer avant que la maladie soit diagnostiqué. La vitesse à laquelle les cellules se détruisent dépendra de l'âge du malade, étant plus rapide pour les bébés et les enfants et plus lente pour les adultes. Une fois le traitement avec insuline établie, l'organisme passera par une période, allant jusqu'à un an, durant laquelle les sécrétions d'insuline se réinitialisent et les besoins exogènes diminuent, sans pour autant abandonner le traitement. Après une dizaine d'années, les cellules Béta seront entièrement détruites, donc toute l'insuline nécessaire devra être administrée par injection.

1.4.2 Diabète de type 2

Dans ce type de diabète l'altération métabolique n'est pas aussi intense que pour DM1 et l'évolution de la maladie est progressive. Ce diabète est caractérisé par la résistance ou la faible sensibilité du corps à l'insuline, c'est-à-dire que le taux d'insuline endogène peut se trouver dans les paramètres normaux, mais les tissus sont incapables de l'assimiler et par conséquent le taux de glucose dans le sang augmente.
L'insuline agit à niveau cellulaire à travers de certains récepteurs de membrane . La liaison insuline-récepteur active un deuxième messager qui induit la synthèse des protéines et l'activation et inhibition des enzymes intra-cellulaire. Les malades souffrant de diabète de type 2 ont des altérations dans les mécanismes post-récepteurs, ce qui oblige l'organisme à augmenter la sécrétion d'insuline pour compenser et ceci peut conduire à l'épuisement des cellules Béta. Pour des personnes avec une certaines prédisposition les cellules ne seront pas capable de maintenir un taux de glucose normale, ce qui conduit à l'apparition du diabète. Même si les diabétiques de type 2 ne nécessitent pas les injections d'insuline pour survivre, prés du 40 % des malades finissent par en avoir besoin pour contrôler la glycémie. L'hyperinsulinisme de plus de 80 % de diabétique de type 2 est la conséquence de leur obésité (la graisse abdominal est la plus dangereuse).

Le tableau 2 présente un résumé sur les deux types de diabète DM1 et DM2 :

Caractéristiques	Diabète type1	Diabète type 2
Age d'apparition	Avant 30 ans	Après 30 ans
Sexe	Prédominante sur les mâles	Prédominance sur les femmes
Forme d'apparition	brusque	Lente, progressive et insidieuse
Indice de masse corporel	Normal	Augmenté, souvent avec obésité
Réserve pancréatique	Très peu ou nulle	Normal ou augmenté
Dépendance de l'insuline	Oui	Non, au moins pendant les premières années
Facteur immunologique	Présent	Absent
Hérédité	Dans quelque cas	Presque toujours
Concordance entre jumeaux	Presque 50 % des cas	Plus de 95 % des cas
Association avec d'autres maladies	Rarement	Souvent
Principal cause de décès	Insuffisance rénale	Infarctus du myocarde

TABLE 1.1 – Comparaison entre DM1 et DM2

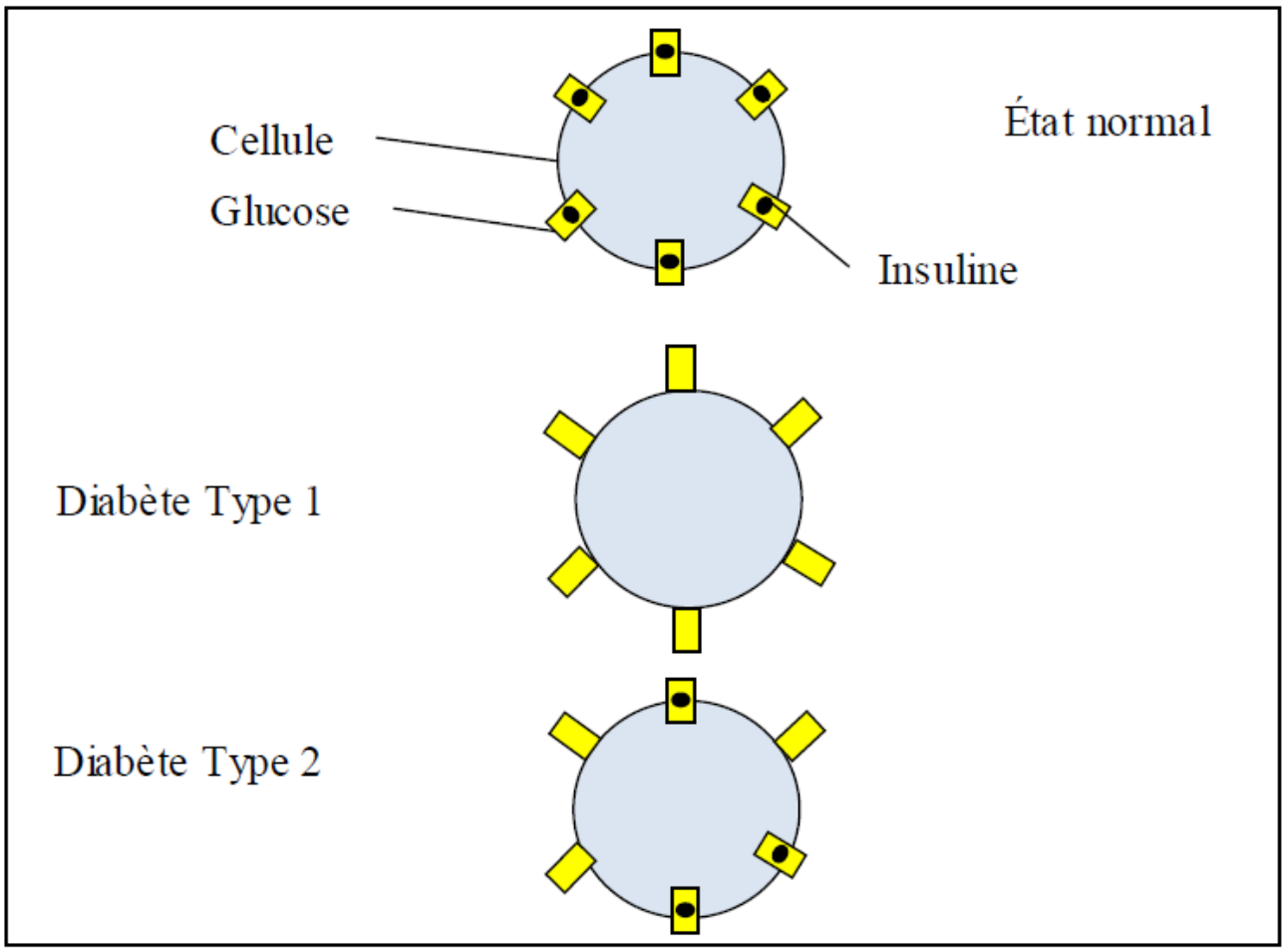


FIGURE 1.2 – Fonctionnement de l'insuline

1.5 Diagnostic

Durant plusieurs années, le taux de glucose dans le sang pour diagnostiquer un diabète été assez élevé mais en 1997 le niveau standard pour un taux de glucose normal fut réduit car beaucoup de personnes souffraient des complications dues au diabète même s'ils n'étaient pas diabétique selon les standards**[HAMIDI Yacine Nacer Eddine,2012]**. En 2003, le standard fut à nouveau modifié. Après beaucoup de débats, l'ADA publia le nouveau standard pour le diagnostic, en se basant sur les critères suivants :

- Glycémie plasmatique occasionnelle (GPO) : c'est le niveau de glucose quand le patient a mangé normalement avant les analyses. Elle doit être $\geq$ 200 mg/dl et accompagner de symptômes classiques du diabète.
- Glycémie plasmatique à jeun (GPJ) : c'est le niveau de glucose quand le patient n'a rien consommé durant les huit heures avant les analyses. Elle doit être $\geqslant$ 126 mg/dl.
- Hyperglycémie provoquée par voie orale (HGPO) : l'analyse est réalisé deux heures après avoir ingérer 75g de glucose par voie oral. Elle doit être $\geqslant$ 200 mg/dl.

Il est préconisé d'obtenir deux mesures consécutives du même test permettant de confirmer le diagnostic de diabète.

	Normal	Prédiabète	Diabète
Glycémie plasmatique occasionnelle			$\geqslant$ 200 mg/dl + symptômes
Glycémie plasmatique à jeun	$<$ 100 mg/dl	$\geqslant$ 100 mg/dl et $<$ 126 mg/dl	$\geqslant$ 126 mg/dl
HGPO	$<$ 140 mg/dl	$\geqslant$140 mg/dl et $<$ 200 mg/dl	$\geqslant$ 200 mg/dl

TABLE 1.2 – Testes de glycémie.

1.6 Conclusion

Nous avons vu dans ce chapitre les types de diabète, les différents traitements et testes ainsi que les complications dû à cette maladie. Même s'il existe des méthodes de prévention qui permettent de réduire le risque d'avoir le diabète, parfois il impossible de l'éviter comme pour le diabète de type 1. Dans ces cas là, la seule solution est de pouvoir le diagnostiquer très tôt et faire tout son possible pour combattre les complications.

Dans ce travail nous présentons un classifieur basé sur les essaims de poissons artificiel pour la reconnaissance du diabète.

CHAPITRE II :

DATA MINING

Chapitre 2

DATA MINING (fouille de données)

2.1 Introduction

Les techniques de data mining sont utilisé de façon augmentaté dans le domaine économique. Tels que la prédiction de certains indicateurs économiques, la découverte des informations cachées, des problèmes ou de trouver des problèmes dans le secteur industriel, ainsi que dans les relations avec les clients à travers l'étude de leurs données et leurs comportements afin d'améliorer le rapport coût-efficacité de la relation avec les clients ou d'attirer de nouveaux clients. Dans ce chapitre nous voulons reconnaitre les différentes techniques de data mining afin d'avoir un aperçu complet sur eux.

Ce chapitre a pour objectif de présenter le processus de l'extraction de connaissances à partir des données, la fouille de données, ses tâches et plus précisément la classification supervisée. La fouille de données désigne en réalité un ensemble de traitements différents menant à la découverte de connaissances variées. Ces traitements sont la classification, la segmentation et les règles d'association.
La classification est une tâche centrale de l'étape de fouille de données dans le processus d'extraction de connaissances à partir de données. Elle consiste à construire un classifieur à partir d'un ensemble d'exemples étiquetés par leur classe (phase d'apprentissage) et ensuite à prédire la classe de nouveaux exemples avec le classifieur (phase de classement).

2.2 Définitions du data mining

2.2.1 Definition 1

Data mining signifie l'extraction de connaissance a travers l'analyse d'une grande quantité de données pour utiliser ces connaissances dans le processus de décision. On peut trouver les données stocker et organisé dans les datamarts et les entrepôts de données, ou dans autre sources non structurées**[RIGHI, EL-AMIN,2015]**.

2.2.2 Definition 2

Le datamining est un processus qui applique les techniques de l'intelligence artificielle dans le but de découvrir des modèles au sein des données, et il est reconnu pour être particulièrement puissant pour identifier les clients qui partagent les mêmes caractéristiques .

2.2.3 Definnition 3

Data Mining (fouille de données) est un processus de découvertede règle, relations, corrélations et/ou dépendances à travers une grande quantité de données, grâce à des méthodes statistiques, mathématiques, de reconnaissances de formes, ...etc

il nécessite de très puissants systèmes informatiques, généralement multiprocesseurs, de manière à autoriser tous les calculs, filtres, synthèses et interprétations possibles. « La DM est souvent critiquée par les chercheurs académiques parce que les praticiens ont tendance à utiliser ces techniques comme des "boîtes noires". Les outils d'extraction de données génèrent des résultats tout en donnant peu d'informations sur la façon dont ceux-ci sont obtenus et sur leur robustesse » **[Alaoui,2015]**.

2.3 Historique

Le “data mining” que l'on peut traduire par “fouille de données” apparaît au milieu des années 1990 aux États-Unis comme une nouvelle discipline à l'interface de la statistique et des technologies de l'information : bases de données, intelligence artificielle, apprentissage automatique (« machine learning »).

David Hand (1998) en donne la définition suivante : « Data Mining consists in the discovery of interesting, unexpected, or valuable structures in large data sets».

La métaphore qui consiste à considérer les grandes bases de données comme des gisements d'où l'on peut extraire des pépites à l'aide d'outils spécifiques n'est certes pas nouvelle. Dès les années 1970 Jean-Paul Benzécri n'assignait-il pas le même objectif à l'analyse des données ? : « L'analyse des données est un outil pour dégager de la gangue des données le pur diamant de la véridique nature ».

On a pu donc considérer que bien des praticiens faisaient du data mining sans le savoir.

On confondra ici le « data mining », au sens étroit qui désigne la phase d'extraction des connaissances, avec la découverte de connaissances dans les bases de données . Comme l'écrivent Hébrail et Lechevallier :

- l'accroissement exponentiel dans les entreprises, de données liés à leur activité (données sur la clientèle , les stocks, la fabrication, la comptabilité ...) qu'il serait dommage de jeter car elles contiennent des informations-clé sur leur fonctionnement (...) stratégiques pour la prise de décision. ;

- Les progrès très rapides des matériels et des logiciels (...) L'objectif poursuivi par le data mining est donc celui de la valorisation des données contenues dans les systèmes d'information des entreprises. » ;

Les premières applications se sont faites dans le domaine de la gestion de la relation client qui consiste à analyser le comportement de la clientèle pour mieux la fidéliser et lui proposer des produits adaptés. Ce qui caractérise la fouille de données (et choque souvent certains statisticiens) est qu'il s'agit d'une analyse dite secondaire de données recueillies à d'autres fins (souvent de gestion) sans qu'un protocole expérimental ou une méthode de sondage ait été mis en oeuvre.

Quand elle est bien menée, la fouille de données a apporté des succès certains, à tel point que l'engouement qu'elle suscite a pu entraîner la transformation (au moins nominale) de services statistiques de grandes entreprises en services de data mining.

La recherche d'information dans les grandes bases de données médicales ou de santé (enquêtes, données hospitalières etc.) par des techniques de data mining est encore relativement peu développée, mais devrait se développer très vite à partir du moment où les outils existent. Quels sont les outils du data mining et que peut-on trouver et prouver ? .c'est ce qu'ont vas le voir dans les sections suivants.

Le Data Mining (fouille de données) renvoie à l'ensemble des méthodes et algorithmes pour l'exploration et l'analyse de gros volumes de données (bases de données informatiques) dans la perspective d'une aide à la prise de décision.

Le Data Mining (fouille de données) repose sur la mise en évidence de règles, de tendances invisibles pour un analyste humain.

La convergence de Data Mining a plusieurs disciplines comme l'indique le scema si dessous :

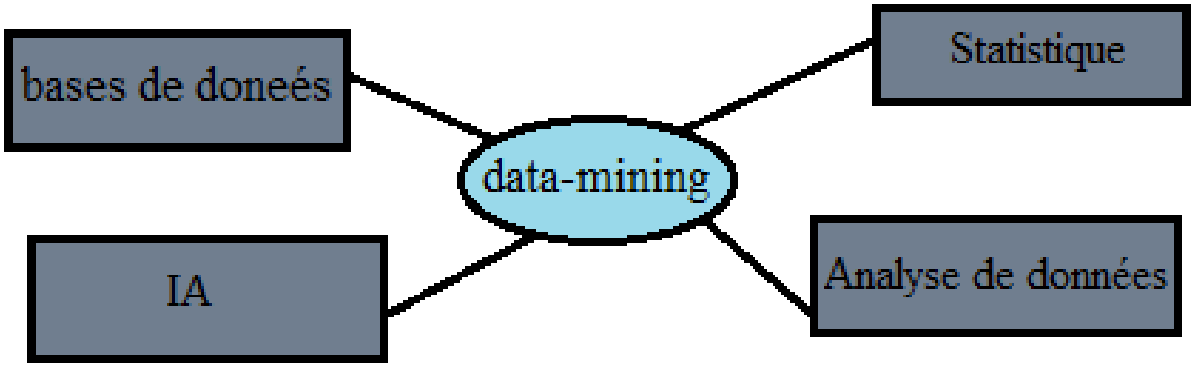


FIGURE 2.1 – Les différente relations du DATA MINING

2.4 Processus du Data Mining

Il est très important de comprendre que le data mining n'est pas seulement le problème de découverte de modèles dans un ensemble de donnée.
Ce n'est qu'une seule étape dans tout un processus suivi par les scientifiques, les ingénieurs ou toute autre personne qui cherche à extraire les connaissances à partir des données. En 1996 un groupe d'analystes définit le data mining comme étant un processus composé de cinq étapes sous le standard CRISP-DM (Cross-Industry Standard Process for Data Mining) comme schématisé ci-dessous :

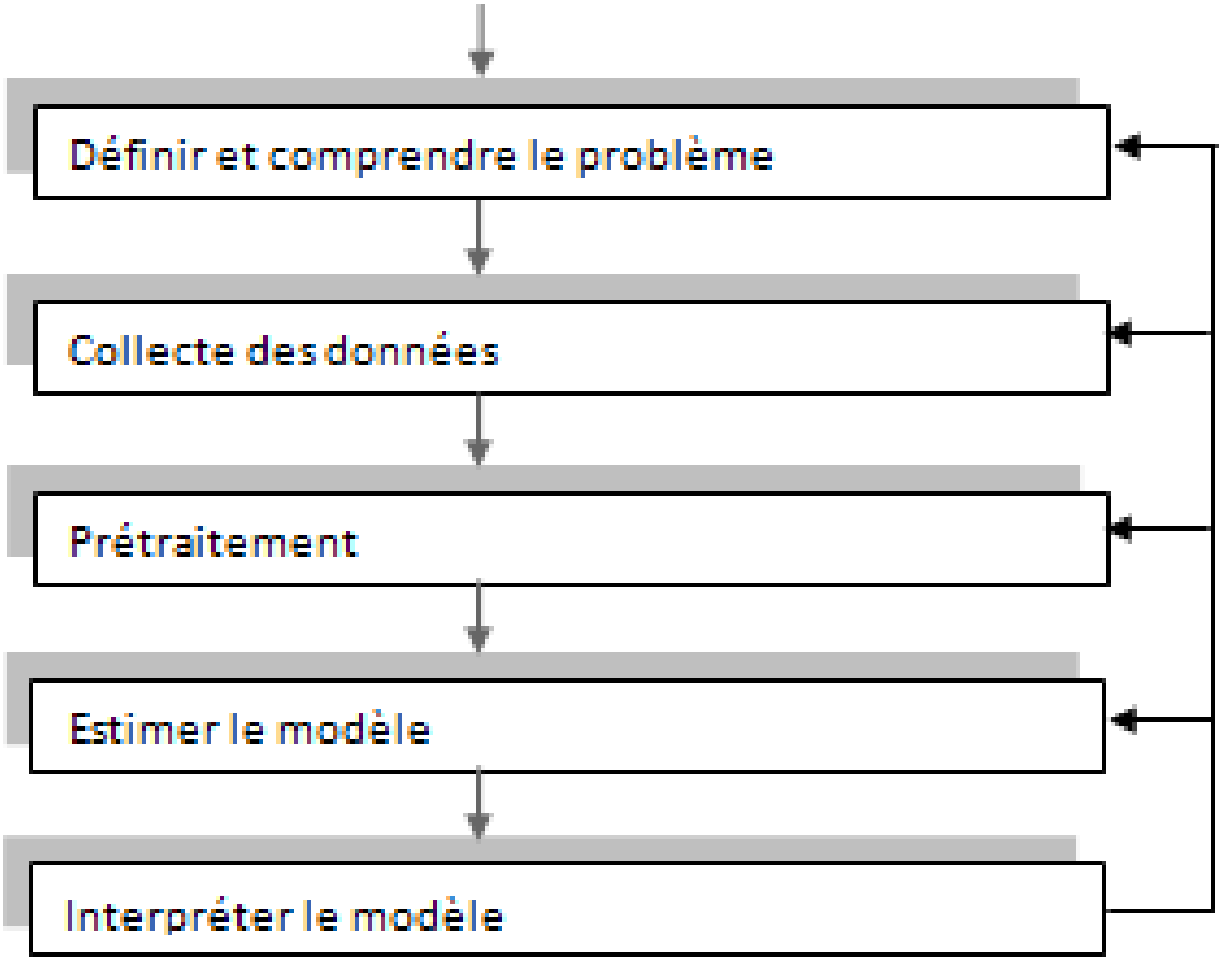


FIGURE 2.2 – Le Processus du Data Mining 01

2.4.1 intégration et collecte de données

Cette étape consiste à regrouper un ensemble de données relatives au système cible de l'étude, en unique source : l'entrepôt de données pour faciliter les différents traitements. Ces données sont le plus souvent hétérogènes. Cette première phase se concentre sur la collecte des données nécessaires pour mener à bien un processus ECD.

2.4.2 sélection, nettoyage et transformation

Cette étape consiste à sélectionner, dans l'entrepôt, les données qui seront retenues pour construire le modèle.
Le nettoyage de données consiste à gérer la qualité des données et plus particulièrement les imprécisions et les incertitudes qu'elles peuvent contenir. Le développeur pourra choisir de corriger ou d'ignorer les données manquantes et erronées. Quelques unes des tâches de cette phase sont :

2.4.2.1 la détection des valeurs erronées

Les valeurs incohérentes parce qu'il y a des erreurs (délibérées ou non), des omissions, des duplications qui ont été introduites, ... (erreur de mesure par exemple). Elles peuvent être dues à une erreur dans les données, comme des cas particuliers. Selon les tâches qu'on va effectuer après la détection de ces anomalies, soit à prendre en compte ou à éliminer.

2.4.2.2 le traitement des valeurs manquantes

Les valeurs manquantes sont inconnues, non nécessaires, non enregistrées, ... Les valeurs manquantes doivent être traitées spécialement, parfois une valeur manquante doit être détectée et ajoutée. Il est intéressant de les conserver car elles signifient qu'il y a un problème dans les données (examen médical par exemple) et dans certains cas, on peut ignorer les données avec les valeurs manquantes.

2.4.2.3 l'échantillonnage

Dans le cas où la quantité des données est importante, on peut travailler avec un échantillon des données. Cela permet d'augmenter l'efficacité du processus de la fouille de données

2.4.2.4 la sélection d'attributs

La sélection d'attributs est une étape de prétraitement qui permet d'améliorer l'efficacité du processus et de donner une plus grande lisibilité du problème à étudier.

La transformation consiste à préparer les données brutes et à les convertir en données appropriées. Elle structure les données dans un formalisme attendu par les algorithmes qui seront appliqués par la suite. Les opérations les plus connues sont la discrétisation des variables continues, la binarisation des variables nominales, l'agrégation de données. Cette étape facilite les tâches de la fouille de données. Plusieurs algorithmes de fouille de données sont contraignants sur la forme des données qu'ils acceptent La discrétisation de variables continues est un exemple de transformation d'attributs. Il s'agit de transformer un attribut continu en divisant son domaine en intervalles finis. Ainsi, le domaine de l'attribut transformé devient un ensemble discret.

L'agrégation de données est un autre type de transformation. L'agrégat d'un attribut est la transformation de ce dernier par une règle ou une équation. Par exemple, on veut analyser les salaires annuels des employés, et que l'on dispose seulement des salaires mensuels, un nouvel attribut agrégat serait le salaire multiplié par douze.

2.4.3 la fouille de données)

La fouille de données est le coeur du processus car elle permet d'extraire l'information (utile et inconnue) de gros volumes de données stockées dans des bases ou des entrepôts de données. La section 3 est consacrée à la fouille de données et ces tâches (la classification, ...).

2.4.4 évaluation et interprétation

Les résultats obtenus par la phase précédente doivent être évalués pour mesurer la qualité des modèles construits. Il n'y a pas de critère unique d'évaluation, mais il en existe plusieurs. Le choix dépend du contexte dans lequel nous sommes : par exemple dans la classification, on utilise souvent la précision et la régression prédictive, le critère utilisé est l'erreur quadratique moyenne de la valeur prédite à partir de la valeur réelle.
L'interprétation concerne l'analyse des résultats de l'étape de la fouille de données par un expert du domaine assisté par le développeur. Celui-ci pourra décider de modifier les réglages effectués lors des étapes précédentes (données sélectionnées, seuils de discrétisation, ...) et de relancer le processus.

2.4.5 diffusion

Une fois le modèle est construit, il y a plusieurs tâches à effectuer : Soit de prendre des décisions futures, soit d'appliquer sur un ensemble de nouvelles données.
Dans le cours du temps, on doit mesurer l'évolution du modèle. Il peut émerger de nouveaux cas qui provoquent

une dégradation progressive du modèle. Par exemple dans les applications liées au commerce, où les goûts des consommateurs sont influencés par des facteurs externes non prévisibles.

2.5 Les outils

On y retrouve des méthodes statistiques bien établies, mais aussi des développements récents issus directement de l'informatique. Sans prétendre à l'exhaustivité, on distinguera les méthodes exploratoires où il s'agit de découvrir des structures ou des comportement inattendus, de la recherche de modèles prédictifs où une « réponse » est à prédire, mais on verra plus loin que l'acception du terme « modèle » diffère fondamentalement de son sens habituel.

2.6 Classification des méthodes de fouille de données

Les méthodes de la fouille de données peuvent être classées selon :

2.6.1 le type d'apprentissage utilisé dans les méthodes de la fouille

2.6.1.1 Fouille supervisée :

comprenant à la fois des données d'entrée et de sortie dont le but est de classer correctement un nouvel exemple.

2.6.1.2 Fouille non supervisée

c'est un processus dans lequel l'apprenant reçoit des exemples d'apprentissage (pas notion de classe) ne comprenant que des données d'entrées dont le but est de regrouper les exemples en segment (cluster) d'exemples similaires.

2.6.2 les méthodes de fouille de données selon les objectifs

2.6.2.1 Classification

elle permet de prédire si une instance de donnée est membre d'un groupe ou d'une classe prédéfinie.

2.6.2.2 Segmentation

elle consiste à partitionner logiquement la base de données en clusters (Former des groupes homogènes à l'intérieur d'une population).

2.6.2.3 Règles d'associations

elle consiste à déterminer les corrélations (ou relations) entre les attributs. Des exemples des méthodes de fouille de données classés selon cette organisation sont présentés dans le tableau si dessous :

	Supervisées	Non supervisées
Classification	▪ Arbres de décision ▪ Réseaux de Neurones avec perceptron. ▪ Modèles/ Réseaux Bayésiens. ▪ Machines à vecteur supports (SVM).	▪ K plus proches voisins (ppv-raisonnement à partir de cas). ▪ Règles temporelles ▪ Reconnaissance des formes.
Segmentation		▪ K-means. ▪ K plus proches voisins (ppv-raisonnement à partir de cas). ▪ Réseaux de Neurones avec cartes de kohonen.
Association		▪ Règles d'Association

FIGURE 2.3 – Quelques méthodes de fouille de données

On peut classer les méthodes de fouille de données selon USAMA Alfayyad en deux grandes familles MÉTHODES PRÉDICTIVES ET MÉTHODES DESCRIPTIVES.

- Les méthodes descriptives : elles permettent de décrire la situation actuelle, elles caractérisent les propriétés générales des données dans la base de données et mettent l'accent sur la compréhension et l'interprétation de ces dernières. ;
- Les méthodes prédictives : qui, en apprenant sur le passé, simulent le futur. Elles utilisent les données avec des résultats connus pour développer des modèles permettant de prédire les valeurs des autres données. ;

2.7 APPRENTISSAGE

Puisque le sujet central traité dans ce mémoire s'inscrit dans le domaine de l'intelligence collective nous allons plutôt focaliser sur les méthodes d'apprentissage les plus importantes, adoptées dans le domaine de l'intelligence artificiel.
Depuis les débuts de cette discipline. l'approche traditionnelle de programmation des robots la plus utilisée est basée sur l'analyse et la géométrie. Cette approche a permis de résoudre un grand nombre de problèmes. et d'obtenir des preuves de convergence et de stabilité.

Malgré ceci de nombreux problèmes persistent. Le principal étant celui d'envisager l'ensemble des cas possibles pendant la programmation. En effet. la réaction d'un système pré-programné face à une situation inconnue est incertaine. Il a donc fallu trouver des solutions alternatives à l'approche mathématique (qui est de nature déterministe) pour permettre aux machines d'être plus autonomes. Ces solutions sont inspirées du vivant. Les techniques qui ont le plus de succès dans ce sens.

2.7.1 Définition 01

l'apprentissage est l'acquisition de savoirs ou de savoir-faire ou de savoir-être (attitudes)

2.8 Qu'est ce qu'une donnée

Une donnée est un enregistrement au sens des bases de données, que l'on nomme aussi « individu » (au sens des statistiques) ou « instance » (terminologie orientée objet au sens informatique). Une donnée est caractérisée

par un ensemble d'attributs.

2.8.1 Les différentes natures d'attributs

Un attribut peut être de nature qualitative ou quantative en fonction de l'ensemble des valeurs qu'il peut prendre. Un attribut est qualitatif si on ne peut pas en faire une moyenne, sa valeur est d'un type défini en extension (une couleur, une marque de voiture, ...) sinon, l'attribut est de nature quantative : un entier, un réel, ... ; il peut représenter un salaire, une surface, un nombre d'habitants, ... ; on peut donc appliquer les opérateurs arithmétiques habituels sur les attributs quantatifs, ce qui n'est pas le cas des attributs qualitatifs.
Un attribut peut également être un enregistrement (une date par exemple), donc composé lui-même de sous attributs (jour, mois et année dans le cas d'une date), sur lequel on peut définir les opérateurs arithmétiques habituels :
Donc quantatif ne signifie pas forcément « numérique » et, réciproquement, numérique ne signifie pas forcement quantatif : un code postal est numérique, mais pas quantatif.

2.8.2 Les différentes natures de valeurs d'attributs

2.8.2.1 attributs à valeurs nominales

les valeurs sont des symboles (des noms) dont aucune relation (ordre ou distance) existe entre les nominaux.
Exemple01 : Les valeurs de temps : ensoleillé, neigeux, pluvieux, gris. Une opération (règle) qu'on peut exécuter : Si temps = neigeux alors voyage = non.

2.8.2.2 attributs à valeurs ordinales

une notion d'ordre s'impose sur les ordinaux mais il n'est pas possible de calculer directement des distances entre les valeurs ordinales, les opérations d'addition et de soustraction ne sont pas possibles.
Exemple02 :
Les températures sont décrites par les adjectifs (chaud, froid, moyen)*et* chaud > moyen > froid, Une règle qu'on peut exécuter : si température > froid alors voyage = oui

2.8.2.3 attributs de type intervalles

les intervalles impliquent une notion d'ordre, et les valeurs sont mesurées dans des unités spécifiques et fixées. La somme, la différence et le produit de deux intervalles ne sont pas possibles (car le point zéro n'existe pas).
Exemple03 : La température est exprimée en degrés Celsius ou Fahrenheit.

2.8.2.4 attributs de type rapport (ratio)

toutes les opérations mathématiques sont autorisées sur les attributs de ce type.
Exemple04 : L'attribut distance, par exemple on peut comparer deux distances

2.9 Les tâches de fouille de données

2.9.1 Classification supervisée

Elle permet de prédire si un élément est un membre d'un groupe ou d'une catégorie donnée.

2.9.2 Segmentation (clustering)

Elle consiste à former des groupes (clusters) homogènes à l'intérieur d'une population. Pour cette tâche, il n'y a pas de classe à expliquer ou de valeur à prédire définie à priori, il s'agit de créer des groupes homogènes dans la population (l'ensemble des enregistrements). Après l'application de l'algorithme et lorsque les groupes ont été construits, d'autres techniques ou une expertise doivent dégager leur signification et leur éventuel intérêt. Parmi les algorithmes de clustering, il y a K-means.
La segmentation est une tâche d'apprentissage non supervisée car on ne dispose d'aucune autre information préalable que la description des exemples. Elle joue un rôle exceptionnel dans les applications de fouilles de données telles que l'exploration des données scientifiques, la fouille de texte, les applications de bases de données spatiales, l'analyse du web, le marketing, la biologie et d'autres.

2.9.3 Règles d'Association

La découverte des règles d'association (la recherche des corrélations fréquentes entre un ensemble d'objets) à partir d'une base de données transactionnelle est proposée par les auteurs **[agrawal et al, 1993]**. Cette approche est considérée aujourd'hui comme faisant parti des approches d'apprentissage symbolique non supervisé utilisé dans le domaine de fouille de données et d'extraction de connaissances.
Elle peut être appliquée à tout secteur d'activité pour lequel il est intéressant de rechercher des groupements potentiels des produits ou de services : services bancaires, services de télécommunication, l'organisation de l'accès aux sites internet. Par exemple, elle peut être également utilisée dans le secteur médical pour la recherche de complications dues à des associations de médicaments ou à la recherche de fraudes en recherchant des associations inhabituelles. Une caractéristique principale de la méthode est la clarté des résultats produits.
L'extraction de règles d'association est un processus itératif et interactif constitué de plusieurs phases allant de la sélection et la préparation de données jusqu'à l'interprétation des résultats.

2.10 Classification supervisée

Le fonctionnement de la classification supervisée se décompose en deux points :
Le premier est la phase d'apprentissage, tout ce qui est appris par l'algorithme est représenté sous la forme des règles de classification que l'on appelle le modèle d'apprentissage.
Le second point est la phase de la classification proprement dite, dans laquelle les données tests vont être utilisées pour estimer la précision des règles de classification générées pendant la première phase. Si la précision du modèle est considérée comme acceptable, la règle pourra être appliquée à des nouvelles données.

2.10.1 Déefinition 01

on dispose d'un ensemble x de N exemples, i.e. des couples (donnée, étiquette). Chaque donnée x_i $\in$ D est caractérisée par p attributs et par sa classe y_i $\in$ Y. Dans un problème de classification supervisée, la classe prend sa valeur parmi un ensemble y fini. Le problème consiste alors, en s'appuyant sur l'ensemble d'exemples X=(x_i,y_i)_i $\in$ 1,. ..., N, à prédire la classe de toute nouvelle donnée x $\in$ D
On parle d'une classification binaire quand le nombre de classes |y| est 2 et il peut être quelconque. Dans tous les cas, il s'agit d'un attribut qualitatif pouvant prendre un nombre fini de valeurs.
Si une donnée appartient à plusieurs classes, c'est le cas du problème multi classe.
L'apprentissage supervisé est l'utilisation d'un ensemble d'exemples pour prédire la classe de nouvelles données.
-hypothèse de représentativité des exemples. Les exemples dont on dispose sont représentatifs de l'ensemble de données. En général, il est difficile de déterminer si cette hypothèse est vérifiée pour un jeu d'exemples donné.
Typiquement, dans un problème de classification, le nombre d'exemples (les données pour lesquelles on dispose de leur classe) est petit : l'étiquetage des données est en général effectué à la main, ce qui entraine un cout élevé de cet étiquetage, donc l'obtention d'un nombre important d'exemples. C'est le problème de statistique où l'échantillon est petit.

2.10.2 Définition 02

un classifieur est un algorithme qui, à partir d'un ensemble d'exemples, produit une prédiction de la classe de toute donnée.
D'une manière générale, un classifieur procède par « induction » : à partir d'exemples (cas particuliers), on construit une connaissance plus générale. La notion d'induction de connaissance implique la notion de « généralisation » de la connaissance : à partir de connaissances éparses, les exemples, on induit une connaissance plus générale. Naturellement, même si l'on suppose que la classe des étiquettes n'est pas erronée, il y a un risque d'erreur lors de généralisation, ce risque est quantifié par la notion de « taux d'échec », ou d' «erreur de généralisation ». « Généraliser » : il faut déterminer, au moins implicitement, la pertinence d'attributs pour prédire la classe d'une donnée quelconque. Il implique de construire un modèle de données, la taille de ce modèle est un paramètre important. - sur apprentissage : la taille du modèle est plus gros que l'ensemble des exemples, on a rien appris rien généralisé et on est incapable d'effectuer une prédiction fiable pour une donnée qui ne se trouve pas dans l'ensemble des exemples.
- apprentissage difficile (on n'est plus capable d'effectuer une prédiction fiable pour une donnée particulière), on peut n'avoir appris que les propositions des différentes classes dans l'espace de données : par exemple, 1/3 des données sont bleu et les autres sont rouge, cela sans lien avec la description des données, prédire la classe revient alors à tirer la classe au hasard avec ces proportions un tiers/ deux tiers.
-entre les deux extrêmes, il y a un milieu où le modèle a pris du recul par rapport aux exemples, a su extraire les informations pertinentes du jeu d'exemples pour déterminer la classe de n'importe quelle donnée avec une

probabilité élevée de succès.
Le modèle est alors de taille modérée et la probabilité d'erreur de ce modèle est la plus faible que l'on puisse obtenir : on a un modèle optimisant par rapport qualité/prix, i.e. la probabilité d'effectuer une prédiction correcte/cout du modèle. La recherche d'un modèle optimisant ce rapport est l'objectif de l'apprentissage automatique lequel est l'un des outils indispensables pour la réaliser de la fouille de données.

2.11 Domaines d'application

Il existe plusieurs domaines d'application de ce problème,on peut citer :
L'attribution de crédit bancaire, la reconnaissance de gènes, la prédiction des sites archéologiques, le diagnostic médical, détection de fraudes fiscales,detection d'intrusion et comme notre modele detection de diabéte etc.

2.12 Validation croisée

Pour éviter l'apprentissage par coeur en classification supervisée et pour avoir une estimation non optimiste de l'erreur de classification, il faut utiliser en test des échantillons qui ont servi à l'apprentissage (utiliser une base d'apprentissage et une base de test). Les performances des méthodes de la classification dépendent généralement du nombre d'échantillons d'apprentissage : plus ce nombre est élevé, plus fiable seront les règles de classification. En même temps, il est nécessaire de conserver un nombre significatif d'échantillons de test pour que l'évaluation de ces performances soit significative. La technique de la validation croisée est fréquemment utilisée pour répondre à ces deux besoins : elle consiste à diviser l'ensemble de départ en un certain nombre de sous ensembles de taille égale, chaque sous ensemble étant alors utilisé comme base de test, alors que l'union de tous les autres sous ensembles est utilisée comme base d'apprentissage comme indique la Figure si dessous . Soit r ce nombre de sous ensembles, on parle alors de la validation croisée d'ordre r, naturellement r est supérieur ou égal à 2.
Soit D un ensemble de données de n échantillons (la classe est connue), chacun comportant d attributs. L'ensemble D est alors découpé en r sous ensembles disjoints de taille identique [E(n/r) $\pm$1], la représentation statistique de chaque classe étant préservée par rapport à celle de l'ensemble D. À chaque itération, le nombre d'erreurs de classification réalisées est comptabilisé et le taux d'erreur de la classification totale est obtenu en divisant ce nombre d'erreurs par le nombre d'échantillons n.
Etant donné que la classe de chaque échantillon est connue, nous pouvons également produire la matrice de confusion qui recense pour chaque classe réelle, le nombre d'échantillons que le classifieur a classé dans chaque classe ce qui permet de détecter les éventuelles confusions que pourrait faire le système. Si c est le nombre de classe de D, la matrice de confusion est une matrice de taille cxc où la cellule à l'intersection de ligne i et de la colonne j est le nombre d'éléments de classe réelle i classés par le système dans la classe j.
On appelle « Leave One Out » le cas particulier pour lequel le nombre de sous ensemble de validation croisée est égale au nombre de mesures de l'ensemble (r=n). L'apprentissage se fait alors sur la base complète sauf 1 échantillon et le test est effectué sur l'élément restant. Cela permet, dans le cas d'un faible ensemble de données, de maximiser l'ensemble d'apprentissage mais présente l'inconvénient de se rapprocher de l'apprentissage « par coeur ». La difficulté de cette méthode est de trouver la bonne valeur de d en fonction de la taille de l'ensemble de départ et de la complexité du problème : plus le problème est complexe, plus le système a besoin d'exemples pour apprendre.
-si r est petit, l'ensemble d'apprentissage sera trop faible et le système de classification ne généralisera pas assez, entrainant une augmentation du taux d'erreur. Par contre, il y a moins de phase d'apprentissage à réaliser et l'obtention des résultats est plus rapide. Cette solution est intéressante quand on dispose de beaucoup de mesures.
-si r est grand, l'ensemble de test sera plus faible par rapport à l'ensemble d'apprentissage et on risque le phénomène de sur-apprentissage (cas extrême : « Leave One Out »).

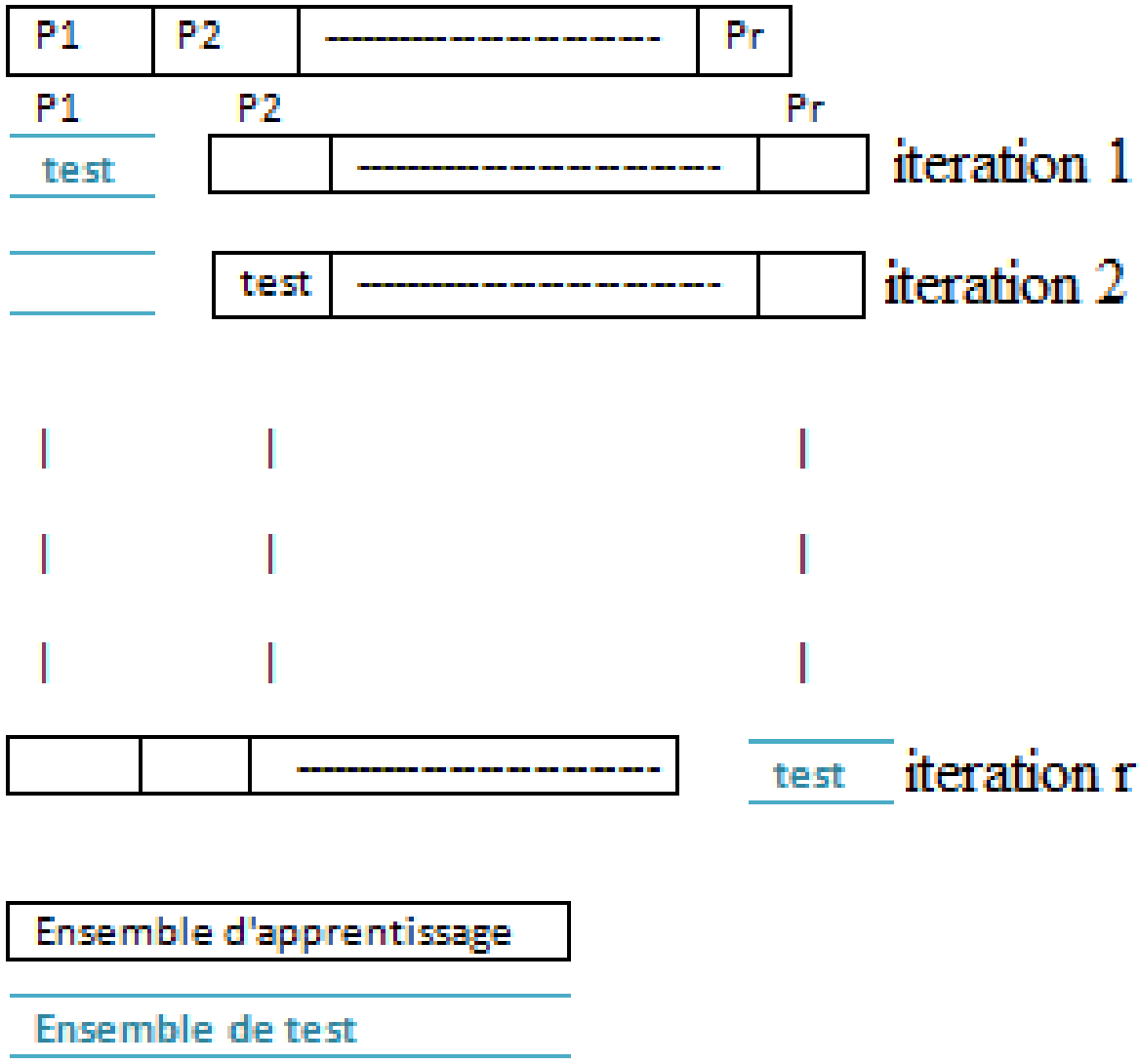


FIGURE 2.4 – Schéma de validation croisée ordre c

2.13 Les méthodes de la classification supervisée

2.13.1 Classifieur bayésien naïf

Le classifieur bayésien naïf fournit une approche simple, avec une sémantique claire de représenter, utiliser, et d'apprendre des connaissances probabilistes. La méthode est conçue pour l'utiliser dans des tâches d'apprentissage supervisé dont le but est de prédire la précision de la classe pour les données de test et dans lequel les données d'apprentissage incluent l'information de classes. C'est une forme de réseaux bayésien, le terme naïf est utilisé car le classifieur repose sur des hypothèses d'indépendance des données.
Il suppose que les attributs sont indépendants pour une classe donnée et que aucun des attributs cachés ou latents a une influence sur le processus de prédiction **[John et Langely, 1995]**, Malgré ses hypothèses, le classifieur performe généralement bien avec peu de données d'apprentissage. Puisque les données sont considérées indépendantes, seule la variance de chaque caractéristique doit être calculée (non pas la matrice de covariance en entier). Il trouve son fondement théorique dans le théorème de bayes **[bayes, 1763]**.
Etant donné les hypothèses d'indépendance des variables, le classifieur est défini comme suit :

$$classe(f_1, ..., f_n) = plusgrand(p(C = c)\Pi_{i=1}^{n}(F_i = f_i|C = c))$$

La fonction plusgrand ne fait que sélectionner la plus grande valeur parmi les probabilités que l'objet appartienne à chacune des classes. On obtient cette probabilité par la multiplication de la probabilité d'obtenir

cette classe (parmi les échantillons d'apprentissage) avec la probabilité d'obtenir chaque caractéristique pour cette classe, selon la valeur moyenne et la variance préalablement calculée. La classe qui obtient la plus grande probabilité en fonction des caractéristiques de l'objet présenté est choisie en tant que classe prédite pour l'objet.

2.13.2 Réseaux de Neurones

Les réseaux de neurones sont une très grande famille de techniques d'intelligence artificielle dont une branche très importante permet de faire de la classification. Les premières notions de réseaux de neurones sont apparues dans les années 1950 lorsque les chercheurs ont voulu simuler le fonctionnement du cerveau. Un réseau de neurones est constitué d'un graphe pondéré orienté dont les noeuds symbolisent les neurones. Ces neurones possèdent une fonction d'activation (fonction signe, fonction sigmoïde, ...) qui permet d'influencer les autres neurones du réseau. Les liens synaptiques sont les connexions entre les neurones, ces liens propagent l'activité de neurones avec une pondération caractéristique de la connexion (poids synaptique). C'est une technique très employée en classification, notamment dans le Domaine médical **[ye et al, 2002]**.
Il existe plusieurs types de réseaux de neurones, la plus courante est le perceptron multicouche. L'interconnexion entre les neurones permet un calcul global complexe qui en classification se traduit par des frontières de décision aux formes complexes. La phase d'entrainement de réseau consiste à régler les poids synaptiques grâce à l'ensemble d'apprentissage.

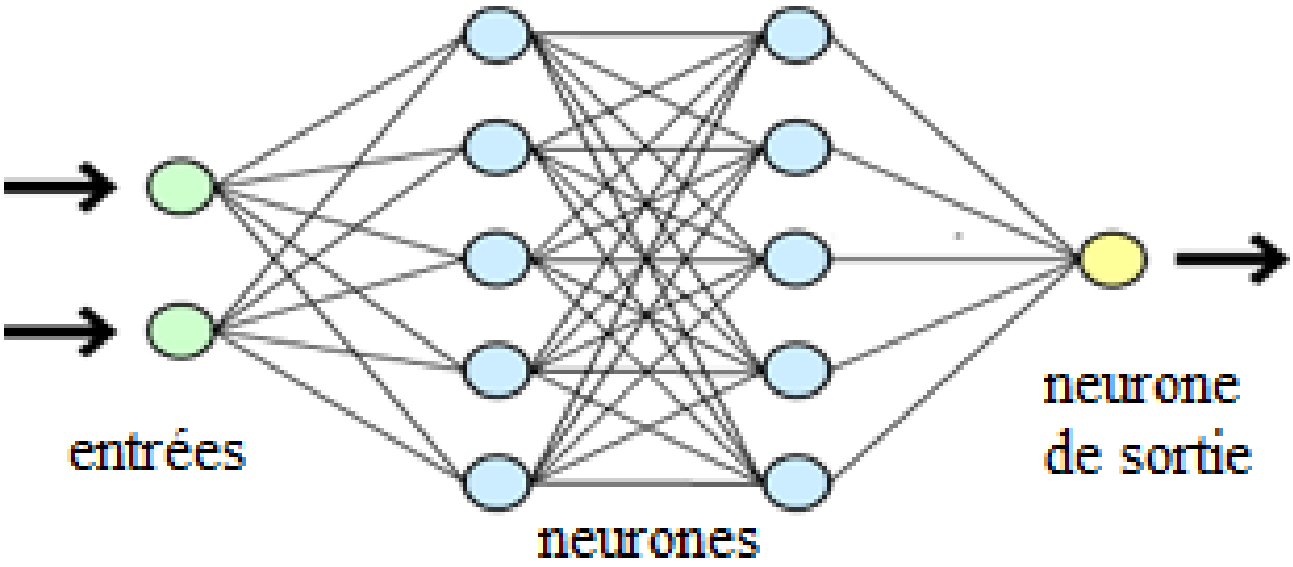


FIGURE 2.5 – Vue simplifiée d'un réseau artificiel de neurones

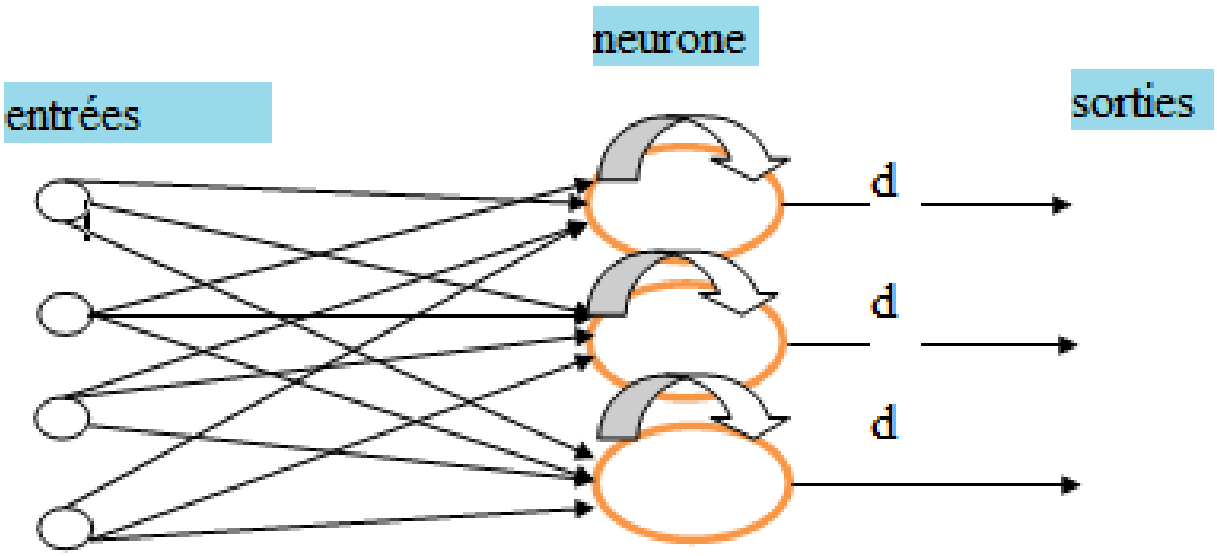


FIGURE 2.6 – Réseau de neurones avec rétroaction

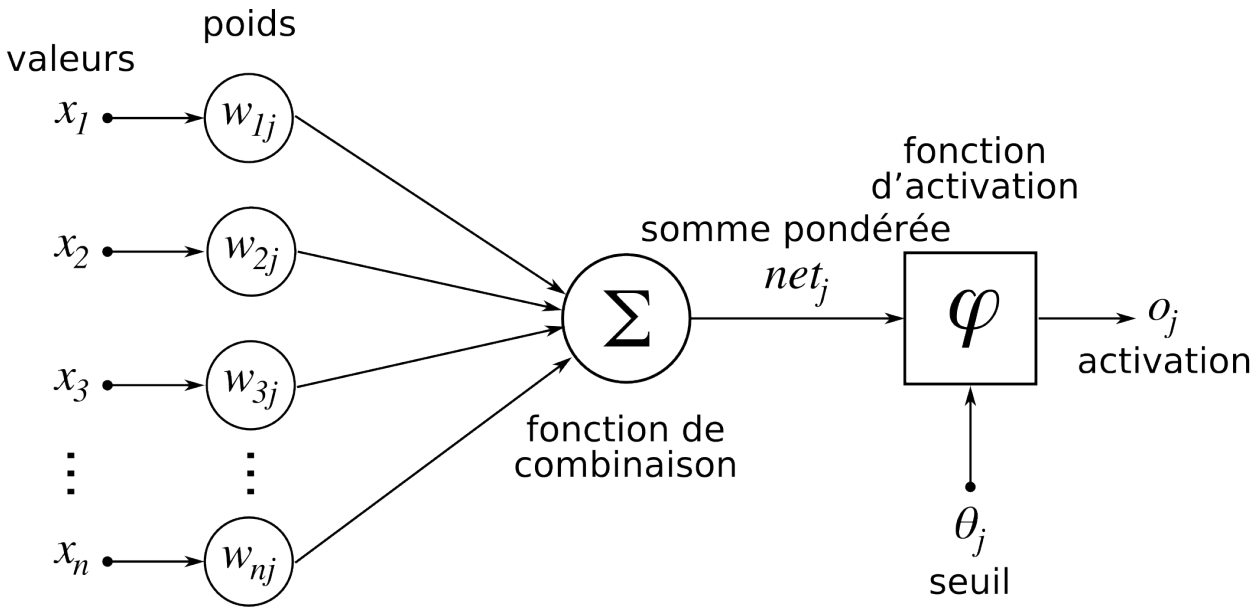


FIGURE 2.7 – Structure d'un neurone artificiel

Le neurone calcule la somme de ses entrées puis cette valeur passe à travers la fonction d'activation pour produire sa sortie.

2.13.3 Séparateurs à Vaste Marge (SVM)

Les SVM sont considérés comme l'une des méthodes robustes et précises **[vapnik, 1995]**, ils disposent d'un fondement théorique simple, les SVM ont été largement utilisés dans la reconnaissance des formes, la classification, la régression et l'estimation de densité.
Le principe de base est de trouver l'hyperplan optimal qui sépare deux classes dans l'espace de description. Cet espace est celui qui maximise la distance entre les deux classes (la marge), le SVM utilise des fonctions de noyau qui, dans un espace augmenté, permettent une séparation optimale des points en différentes catégories. Les données d'apprentissage sont utilisées pour découvrir l'hyperplan qui séparera au mieux les points. L'idée du SVM est d'utiliser sa fonction de noyau pour reconsidérer le même problème dans un espace de dimension plus élevée. Cet espace est caractérisé par la possibilité d'y trouver un séparateur linéaire qui permet de classer les points dans les deux groupes appropriés. Le séparateur linéaire peut ensuite être projeté dans l'espace d'origine

où il devient habituellement non linéaire. Le critère d'optimisation est la largeur de marge entre les classes (l'espace vide de chaque coté des frontières de décision). La largeur de marge est caractérisée par la distance jusqu'aux échantillons d'entrainement le plus près. Ces échantillons s'appellent vecteurs de supports, ils définissent la fonction discriminante qui permet la classification. Le nombre de vecteurs de support est minimisé en maximisant la marge.

2.13.4 Plus Proche Voisin (PPV)

Le plus proche voisin est l'un des algorithmes les plus populaires utilisés pour le classement dans les différents domaines de la reconnaissance des formes et de la fouille de données, c'est une méthode basée sur la notion de proximité (voisinage) entre exemples et sur le raisonnement à partir de cas similaire pour prendre une décision. Un objet est classifié selon un vote majoritaire par ses voisins, l'objet obtient la classe qui est la plus commune chez ses K plus proche voisins dans l'espace des caractéristiques. k doit donc un entier positif, généralement petit. On choisit souvent un k impair pour éviter l'égalité dans le vote. La distance utilisée pour le calcul de la proximité des voisins est le plus souvent la distance euclidienne. Les exemples d'apprentissage sont des vecteurs dans un espace multidimensionnel. La phase d'apprentissage consiste simplement à conserver ces données dans un format qui permettra le calcul efficace des distances et la recherche des voisins. Plusieurs algorithmes de KNN et ses variantes ont été présentés dans la littérature afin de diminuer la charge de calcul dédié à la recherche des voisins les plus proches d'un nouvel échantillon **[shakhnarovich et al, 2006]**.

2.13.5 Bagging

La méthode du Bagging a été introduite par **[Breiman, 1996]**. Le mot Bagging est la contraction des mots Bootstrap et Aggregating. Le principe du bagging est d'entraîner un algorithme d'apprentissage sur plusieurs bases d'apprentissage obtenues par tirage avec remise (ou bootstrap) de m exemples d'apprentissage dans l'échantillon d'apprentissage S. Pour chaque tirage b, une hypothèse h_b est obtenue. L'hypothèse finale est basée sur les moyennes des hypothèses obtenues. Son avantage est qu'on améliore la performance des classifieurs instables en calculant la moyenne de leurs réponses.
Ainsi, si les hypothèses h_b calculées pour chaque tirage b ont une variance importante, alors l'hypothèse finale aura une variance réduite. Un classifieur est dit instable si un petit changement dans les données d'apprentissage provoque un gros changement dans le comportement du classifieur. Le but du bagging est d'atténuer l'instabilité inhérente à certains classifieurs.

2.13.6 Boosting

Le boosting est une méthode d'agrégation de classifieurs faibles pour obtenir un classifieur performant **[Schapire, 1990]**. Son principe consiste à assigner un poids égal à chaque exemple d'apprentissage (1/n) (appelé pondération) où n est le nombre d'échantillons. Ce poids indique la difficulté de prédire la classe de cet exemple. L'algorithme s'exécute T fois construisant T classifieurs sur les exemples d'apprentissage pondérés.
Chaque fois qu'on produit un classifieur on change les poids des nouveaux exemples utilisés pour le classifieur suivant, on augmente le poids de ceux dont la classe est mal prédite et on diminue le poids des autres. Ensuite, on calcule l'erreur (e) du modèle sur l'ensemble pondéré. Si e est égale à zéro ou e est supérieure à 0.5 on termine la construction du classifieur. L'idée est de forcer le nouveau classifieur à diminuer l'erreur attendue. Le classifieur final se forme en utilisant un schéma de vote.

2.13.7 Arbres de Décisions

Il est essentiel de produire des procédures de classification compréhensibles par l'utilisateur. Dans l'exemple du diagnostic médical, le médecin doit pouvoir comprendre et interpréter les raisons du diagnostic. Les arbres de décisions répondent à cette contrainte car ils représentent graphiquement un ensemble de règles facilement interprétables. Bien sûr, souvent la taille de l'arbre (plusieurs centaines de noeuds) empêche de saisir la procédure de classification dans son ensemble mais en partant de la description d'un élément, sa classification est toujours compréhensible. L'induction avec des arbres de décision est l'une des formes d'algorithme d'apprentissage les plus simples et pourtant les plus efficaces **[russell et norvig, 2003]**. Les arbres de décisions sont une des techniques les plus populaires de l'apprentissage automatique et de la fouille de données.

2.13.7.1 Apprentissage des arbres de décisions

Le schéma général des algorithmes est le suivant :

Entrées : Language de description ; échantillon S
Debut
Initialiser l'aarbre a vide ;// la racine est le noed courant
Répeter
Decider si le noed courant est terminal ;
SI le noed est terminal
ALORS affecter une classe ;
SINON sélectionner un test et créer le sous-arbre ;
passer au noed suivant non exploré s'il existe ;
Jusqu'a obtenir un arbre de decision :
FIN

TABLE 2.1 – Algorithme d'apprentissage generique.

Le principe consiste à diviser récursivement et le plus efficacement possible les exemples de l'ensemble d'apprentissage par des tests définis à l'aide des attributs jusqu'à ce que l'on obtienne des sous-ensembles d'exemples ne contenant (presque) que des exemples appartenant tous à une même classe.
Dans toutes les méthodes, on trouve les trois opérateurs suivants :

1. Décider si un noeud est terminal, c'est-à-dire décider si un noeud doit être étiqueté comme une feuille. Par exemple : tous les exemples sont dans la même classe, il y a moins d'un certain nombre d'erreurs, ... ;
2. Sélectionner un test à associer à un noeud. Par exemple : aléatoirement, utiliser des critères statistiques, ... ;
3. Affecter une classe à une feuille. On attribue la classe majoritaire sauf dans le cas où l'on utilise des fonctions coût ou risque.

Les méthodes vont différer par les choix effectués pour ces différents opérateurs, c'est-à-dire sur le choix d'un test (par exemple, utilisation du gain et de la fonction entropie) et le critère d'arrêt (quand arrêter la croissance de l'arbre, soit quand décider si un noeud est terminal).
Avec un tel algorithme, on peut calculer un arbre de décision dont l'erreur apparente est faible, voire nulle. Un arbre de décision parfait est un arbre de décision tel que tous les exemples de l'ensemble d'apprentissage soient correctement classifiés. Un tel arbre n'existe pas toujours (s'il existe deux exemples tels que à deux descriptions identiques correspondent deux classes différentes). L'objectif est de construire un arbre d'erreur de classification la plus petite possible. Mais malheureusement :

- l'erreur apparente est une vision très optimiste de l'erreur réelle,
- trouver un arbre de décision d'erreur apparente minimale est, en général, un problème NP-complet.

2.13.7.2 Exemple de construction d'un arbre de décision

Le jeu de données utilisé : un ensemble de jours (un jour = un exemple), chacun caractérisé par un numéro et ses conditions météorologiques (température, humidité de l'air, force du vent, ciel), l'attribut cible étant « jouer au tennis ? », dont les valeurs possibles sont oui et non. Une fois l'arbre de décision construit, on pourra classer une nouvelle donnée pour savoir si on joue ou non ce jour-là.
La figure si dessous présente un exemple d'arbre de décision.

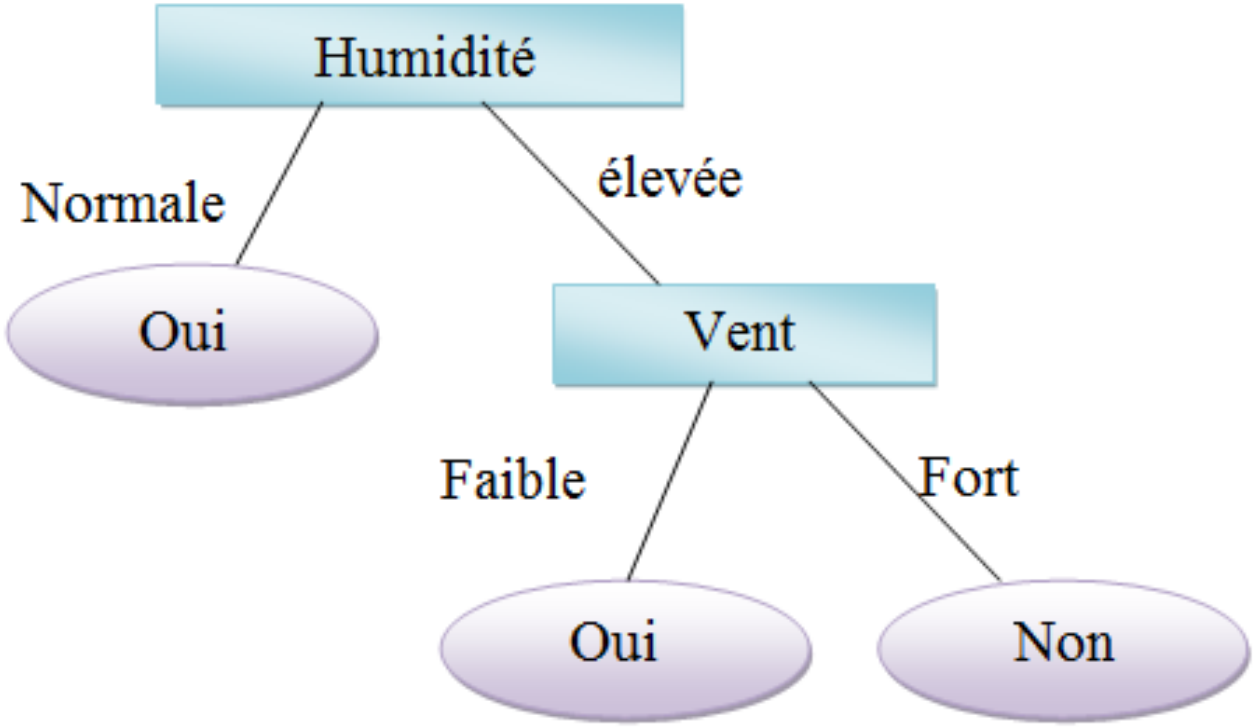


FIGURE 2.8 – Un exemple d'arbre de décision (jouer au tennis)

La donnée 3 du Tableau si dessous est prédite comme « oui » car son attribut « humidité » vaut « Elevée », donc on suit la branche droite partant de la racine et on atteint le noeud « Vent », et l'attribut « Vent » vaut « Faible », ce qui nous fait suivre la branche gauche de ce noeud et atteindre une feuille étiquetée « oui » (celle du milieu sur la figure). De même, la donnée 2 sera prédite comme de classe « non » et la donnée 5 sera prédite de classe « oui » en s'aiguillant directement depuis la racine vers la feuille « oui » à la gauche de la figure.

Jour	Ciel	Température	Humidité	Vent	Jouer au tennis ?
1	Ensoleillé	Chaude	Elevée	Faible	Non
2	Ensoleillé	Chaude	Elevée	Fort	Non
3	Couvert	Chaude	Elevée	Faible	Oui
4	Pluie	Tiède	Elevée	Faible	Oui
5	Pluie	Fraîche	Normale	Faible	Oui
6	Pluie	Fraîche	Normale	Fort	Non
7	Couvert	Fraîche	Normale	Fort	Oui
8	Ensoleillé	Tiède	Elevée	Faible	Non
9	Ensoleillé	Fraîche	Normale	Faible	Oui
10	Pluie	Tiède	Normale	Faible	Oui
11	Ensoleillé	Tiède	Normale	Fort	Oui
12	Couvert	Tiède	Elevée	Fort	Oui
13	Couvert	Chaude	Normale	Faible	Oui
14	Pluie	Tiède	Elevée	Fort	Non

TABLE 2.2 – Jeu de données (jouer au tennis)

Les principaux algorithmes de construction d'un arbre de décision sont : ID3 **[Quinlan, 1986]**, C4.5 **[Quinlan, 1993]** et CART **[Breimann et al, 1984]**.

2.13.8 Elagage

Un des problèmes connus lors de la phase de construction et de classement est que la taille de l'arbre grandit de manière linéaire avec la taille de la base d'apprentissage. De plus, les arbres de décisions complexes peuvent avoir des taux d'erreur très élevés à cause de sur-apprentissage qui peut se produire lorsque l'ensemble d'apprentissage contient des données bruitées ou qu'il ne contient pas certains exemples importants, ou encore lorsque les exemples sont trop spécifiques. L'élagage est l'une des solutions pour réduire ces taux d'erreurs en simplifiant l'arbre par suppression de quelques branches.
Il existe plusieurs techniques d'élagage **[Mingers, 1989]** pour éviter le sur-apprentissage.

2.14 Conclusion

Nous avons présenté une introduction à un domaine de recherche en plein essor « l'extraction de connaissances à partir de données ». En effet, les quantités de données collectées dans divers domaines deviennent de plus en plus importantes et leur analyse de plus en plus demandée. Nous avons cité le processus d'extraction de connaissances et nous avons insisté sur une étape importante qui est la fouille de données. Ensuite, nous sommes focalisés sur une tâche de cette étape de fouille de données, qui est la tâche de la classification supervisée.

CHAPITRE III :

Les Métaheuristiques

Chapitre 3

Les Métaheuristiques

3.1 Introduction

L'un des principes les plus fondamentaux de notre monde, qui occupe aussi une place importante dans le monde informatique, industriel, etc., est la recherche d'un état optimal. En effet, de nombreux problèmes scientifiques, sociaux, économiques et techniques ont des paramètres qui peuvent être ajustés pour produire un résultat plus souhaitable. Ceux-ci peuvent être considérés comme des problèmes d'optimisation et leur résolution est un sujet central en recherche opérationnelle. Des techniques ont été conçues pour résoudre ces problèmes, notamment les problèmes « difficiles » (par exemple ceux qui présentent de nombreux extremes locaux pauvres), en déterminant des solutions qui ne sont pas rigoureusement optimales, mais qui s'en approchent. Ces méthodes, appelées heuristiques et méta heuristiques, se basent généralement sur des phénomènes physiques, biologiques, sociopsychologiques, et peuvent faire appel au hasard.

L'heuristique est un terme de didactique qui signifie « l'art d'inventer, de faire des découvertes ».
Il existe un grand nombre de métaheuristiques d'optimisation. Elles se distinguent classiquement en deux groupes : les méthodes de recherche locale et les méthodes de recherche globale. Dans ce chapitre, nous allons définir les problèmes d'optimisation , l'heuristique et la métaheuristique suivis par une classification de cette dernière et on va voir que les métaheuristiques se prêtent à toutes sortes d'extensions. On va présenter un état de l'art sur les principales métaheuristiques connues dans la littérature en se basant sur la métaheuristique ACO (Ant Colony Optimization). Les algorithmes de colonie de Fourmis ont connu un essor important dans la résolution des problèmes d'optimisation difficile.

3.1.1 Métaheuristiques pour l'optimisation

3.1.1.1 Problème d'optimisation

Un problème d'optimisation au sens général est défini par un ensemble de variables, une fonction objectif f et un ensemble de contraintes d'égalité (ou d'inégalité) que les variables doivent satisfaire. L'ensemble des solutions possibles du problème forme l'espace de recherche E, où chaque dimension correspond à une variable. L'espace de recherche E est fini puisque le décideur précise exactement le domaine de définition de chaque variable entre autres pour des raisons de temps de calcul. Suivant le problème posé, nous cherchons à minimiser ou maximiser la fonction objective f . Un problème d'optimisation peut être statique ou dynamique (i.e. la fonction objective change avec le temps), monoobjectif ou multi-objectif (i.e. plusieurs fonctions objectives doivent être optimisées) et avec ou sans contraintes. Ainsi, un problème d'optimisation peut être, par exemple, à la fois continu et dynamique. Il existe de nombreuses méthodes déterministes (ou exactes) permettant de résoudre certains types de problèmes d'optimisation et d'obtenir la solution optimale du problème, en un temps raisonnable. Ces méthodes nécessitent que la fonction objective présente un certain nombre de caractéristiques telles que la convexité, la continuité ou la dérivabilité. Nous pouvons citer, parmi les méthodes les plus connues, les méthodes de programmation linéaire, quadratique et/ou dynamique, la méthode de Newton , la méthode du simplex ou encore la méthode du gradient .

3.1.1.2 Optimisation difficile

Les méthodes de résolution exacte ne sont pas adaptées à toutes les problématiques, et donc certains problèmes sont trop complexes à résoudre par ces mé-thodes. Parmi ces problématiques, nous pouvons citer l'existence de discontinuités, l'absence de convexité stricte, la non-dérivabilité, la présence de bruit ou encore la

fonction objectif peut ne pas être définie précisément (e.g. quand c'est un cout). En outre, les méthodes de résolution exacte peuvent avoir un temps de résolution trop long. Dans ce cas, le problème d'optimisation est dit difficile, car aucune méthode exacte n'est capable de le résoudre en un temps raisonnable. Il est alors nécessaire d'avoir recours à des heuristiques de résolution dites méthodes approchées, qui fournissent un résultat sans garantie de l'optimalité.
Plusieurs problèmes d'optimisation dépendent du choix d'une meilleure configuration de l'ensemble de variables pour atteindre ses objectifs, ils peuvent se découper en deux catégories : les problèmes où les solutions sont codées avec des valeurs réelles de variables et les problèmes où les solutions sont codées avec des variables discrètes, parmi les derniers, on trouve une classe des problèmes appelée les problèmes d'optimisation combinatoire (CO).
Nous sommes à la recherche d'un objet à partir d'un ensemble dénombrable fini ou infini, cet objet est généralement un nombre entier, un sous ensemble, une permutation ou une structure de graphe [Papadimitriou et Steiglitz, 1982].

Définition 01

un problème d'optimisation combinatoire p(S, f) peut être défini par :

- Un ensemble de variables X=x_1,...,n_n ;
- Les domaines variables D_1,..., D_n ;
- Les contraintes entre les variables ;
- Une fonction « objectif » à minimiser (ou à maximiser) telle que ;
- L'ensemble de toutes les affectations possibles à réaliser
  S={s= {(x_1,v_1),...,(x_n,v_n)}/v_i ∈ D_i, satisfait toutes les contraintes}.

S est généralement appelé un espace de recherche (solutions), chaque élément de l'ensemble peut être vu comme une solution candidate, pour résoudre un problème d'optimisation combinatoire on doit trouver une solution $s* \in S$ avec une fonction objectif de valeur minimale qui est $f(s*) < f(s)$.
$\forall s* \in S$, s* est nommé une solution optimale globale de (S, f) et $s \subseteq S$ est nommé l'ensemble de solutions optimales globales.
Parmi les ensembles des problèmes combinatoires, on cite :

- Le Problème du Voyageur de Commerce (PVC) qui consiste à trouver le plus court chemin et qui permet de passer par un certain nombre de villes et revenir à la ville de départ ;
- Le Problème d'Affectation Quadratique (PAQ) : ce problème possède de nombreuses applications telles que la répartition de bâtiments ou de services, l'affectation des portes d'aéroport, la synthèse d'images ;
- Les problèmes d'ordonnancement.

De nombreux algorithmes ont été développés pour résoudre les problèmes d'optimisation combinatoire en raison de leur importance pratique. Ces algorithmes peuvent être classés en algorithmes exactes et algorithmes approchés.

Les méthodes exactes permettent de résoudre certains problèmes en un temps fini **[Papadimitriou et Steiglitz, 1982]** et **[Nemhauser et Wolsey, 1988]**. Ces méthodes nécessitent généralement un certain nombre de caractéristiques de la fonction objectif, comme la stricte convexité, la continuité ou encore la dérivabilité. On peut citer par exemple : la méthode de la programmation linéaire, quadratique ou dynamique, la méthode du gradient, la méthode de newton, ... **[Dréo, 2006]**. Pour les problèmes d'optimisation combinatoire qui sont NP-difficiles (optimisation difficile) **[Garey et Johnson, 1979]**, il n'existe pas un algorithme exact polynomial, supposant que $P \neq NP$.
Quand le nombre de combinaisons possibles devient exponentiel par rapport à la taille du problème, le temps de

calcul devient rapidement critique (un temps de calcul trop élevé). L'utilisation des méthodes approchées pour résoudre les problèmes d'optimisation combinatoire a reçu de plus en plus d'attention au cours des dernières années.

Si les méthodes exactes permettent de trouver une ou plusieurs solutions dont l'optimalité est garantie, dans certaines situations, on peut obtenir de solutions de bonnes qualités sans garantie d'optimalité mais avec un temps de calcul réduit (les méthodes approchées). Parmi les méthodes approchées, on distingue les méthodes constructives et les méthodes de recherches locales.

Les méthodes constructives génèrent les solutions en ajoutant les composants jusqu'à ce que une solution soit complète, une solution partielle initialement vide, elles sont généralement plus rapide mais avec des solutions de qualité inférieure par rapport aux algorithmes de recherche locale.

Les algorithmes de recherche locale commencent par une solution initiale et essayent de remplacer la solution actuelle, d'une manière itérative, par une meilleure solution dans le voisinage qui est défini d'une manière formelle comme suit :

Définition

Une structure de voisinage est une fonction $N : S \to 2^s$ qui associe à chaque $s \in S$ un ensemble de voisinage $N(s) \subseteq S$. N(s) est appelé voisinage de S.

L'introduction d'une structure de voisinage nous permet de définir la notion du minimum local.

Définition

Une solution $s \in S$ est un minimum local relativement à la structure de voisinage N si $f(s) \leq f(s')\forall s' \in N(S)$.

Une solution $s \in S$ est un minimum global si $f(s) \leq f(s')\forall s' \in S$, voir scéma si dessous :

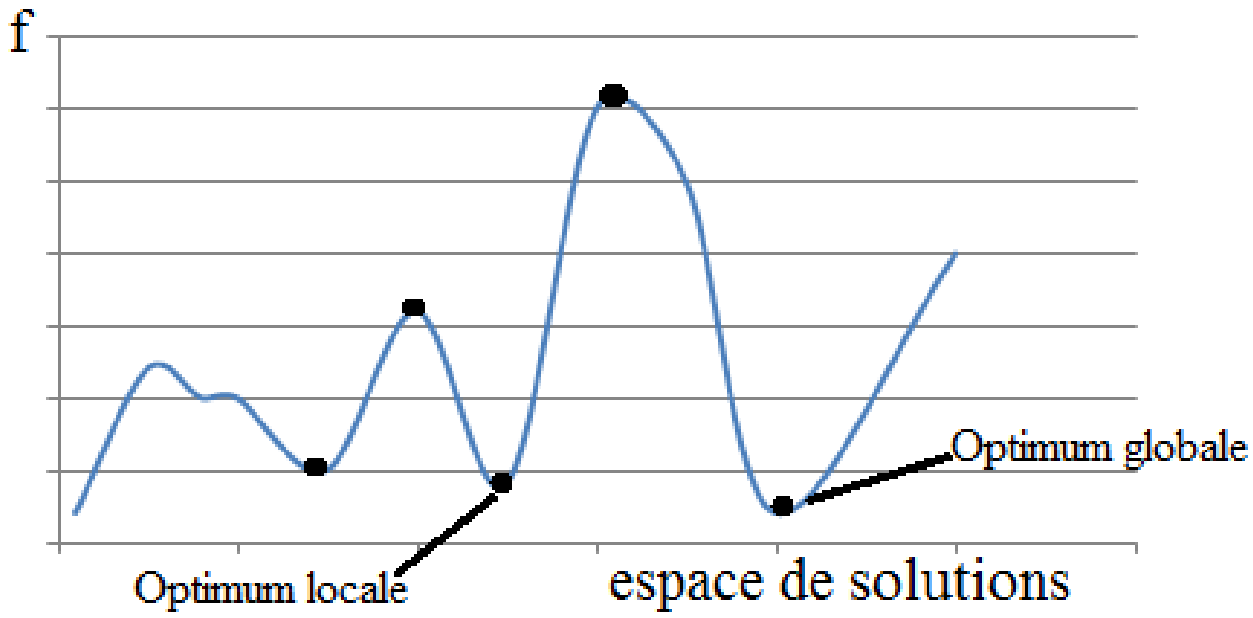


FIGURE 3.1 – Une solution locale minimale et une solution globale minimale

3.2 Algorithmes d'optimisation approchés

3.2.1 Heuristiques

L'heuristique est une méthode, conçue pour un problème d'optimisation donné, qui produit une solution non nécessairement optimale lorsqu'on lui fournit une instance de ce problème.

3.2.2 Métaheuristiques

Plusieurs définitions ont été proposées pour expliquer clairement la notion de métaheuristique, nous citons parmi elles :
« Un processus itératif qui subordonne et qui guide une heuristique, en combinant intelligemment plusieurs concepts pour explorer et exploiter tout l'espace de recherche. Des stratégies d'apprentissage sont utilisées pour structurer l'information afin de trouver efficacement des solutions optimales, ou presque optimales » **[Osman et Laporte, 1996]**.
« Les métaheuristiques sont généralement des algorithmes stochastique itératifs, qui progressent vers un optimum global, c'est-à-dire l'extremum global d'une fonction objectif » **[Verel, 2008]**.
Les métaheuristiques constituent une classe de méthodes qui fournissent des solutions de bonne qualité en un temps raisonnable à des problèmes combinatoires réputés difficiles pour lesquels on ne connait pas de méthode classique plus efficace. Elles sontgénéralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est à dire l'extremum global d'une fonction en évaluant une certaine fonction objectif. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème à fin d'en trouver une approximation de la meilleure solution d'une manière proche des algorithmes d'approximations. L'intérêt croissant apporté aux métaheuristiques est tout à fait justifié par le développement des machines avec des capacités calculatoires énormes, ce qui a permis de concevoir des métaheuristiques de plus en plus qui ont fait preuve d'une certaine efficacité lors de la résolution de plusieurs problèmes à caractère NP-difficile.

Les métaheuristiques sont souvent inspirées de processus naturels qui relèvent de la physique (l'algorithme du recuit simulé), de la biologie de l'évolution (les algorithmes génétiques) ou encore de l'éthologie (les algorithmes de colonies de fourmis ou l'optimisation par essaim particulaire). Les métaheuristiques se caractérisant par leur capacité à résoudre des problèmes très divers, elles se prêtent naturellement à des extensions. Pour illustrer celles-ci, nous pouvons citer :

1. **Les métaheuristiques pour l'optimisation multi objectif** : où il faut optimiser plusieurs objectifs contradictoires. Le but ne consiste pas ici à trouver un optimum global, mais à trouver un ensemble l'optimal, qui forment une surface de compromis pour les différents objectifs du problème ;
2. **Les métaheuristiques pour l'optimisation multimodale** : où l'on ne chercheplus l'optimum global, mais l'ensemble des meilleurs optima globaux et/ou locaux ;
3. **Les métaheuristiques pour l'optimisation de problèmes bruités** : où il existe une incertitude sur le calcul de la fonction objectif, dont il faut tenir compte dans la recherche de l'optimum ;
4. **Les métaheuristiques pour l'optimisation dynamique** : où la fonction objectif varie dans le temps, ce qui nécessite d'approcher l'optimum à chaque pas de temps ;
5. **Les métaheuristiques hybrides** : qui consistent à combiner différentes métaheuristiques, afin de tirer profit des avantages respectifs ;
6. **Les métaheuristiques parallèles** : où l'on cherche à accélérer le calcul, en répartissant la charge de calcul sur des unités fonctionnant de concert. Le problème revient alors à adapter les métaheuristiques pour qu'elles soient distribuées.

3.3 Classification des métaheuristiques

Il existe plusieurs façons de classer et de décrire les algorithmes de métaheuristiques. Selon les caractéristiques choisies, plusieurs classifications sont possibles, on peut distinguer **[blum et roli, 2003]** :

3.3.1 Inspirées de la nature vs non inspirées de la nature

Une manière, plus intuitive, de classer les métaheuristiques consiste à séparer celles qui sont inspirées d'un phénomène naturel, de celles qui ne le sont pas.
Les algorithmes génétiques et les algorithmes de colonies de fourmis sont inspirés de la nature tandis que l'algorithme de Recherche Tabou et l'algorithme de Recherche Locale Itérative sont non inspirés de la nature, d'après les auteurs [blum et roli, 2003], cette classification n'est pas significative :

- de nombreux algorithmes hybrides récents ne correspondent pas à ces deux classes (les deux classes en même temps) ;
- la difficulté de classer une méthode dans certains cas, par exemple, l'utilisation de la mémoire dans la recherche tabou n'est pas inspirée de la nature.

3.3.2 Basées sur la population des solutions vs une solution unique

Une autre manière de la classification est de partager les métaheuristiques en deux grandes classes : les métaheuristiques à solution unique (Recherche Locale, Recherche Tabou, Recuit Simulé,. . ..) et celle à une population de solutions (Algorithmes Evolutionnaires, Recherche par Dispersion, Optimisation par Essaim Particulaires, . . .). Les méthodes d'optimisation à population de solutions améliorent, au fur et à mesure des itérations, une population des solutions. L'intérêt de ces méthodes est d'utiliser la population comme un facteur de diversité.
Les méthodes d'optimisation à solution unique sont appelées les méthodes trajectoires (décrire une trajectoire en l'espace de recherche au cours du processus de recherche).

3.3.3 Dynamique vs une fonction objectif statique

Selon l'utilisation de la fonction 'objectif', on peut classer les métaheuristiques. Etant donné un problème d'optimisation consistant à minimiser une fonction f sur un espace s de solutions, les méthodes qui travaillent directement sur f sont statiques, les méthodes sont dynamiques s'ils utilisent une solution g obtenue à partir de f en ajoutant quelques composantes qui permettent de modifier la topologie de l'espace de solutions, ces composantes additionnelles pouvant varier durant le processus de recherche.

3.3.4 Une structure de voisinage vs des structures diverses de voisinages

Etant donné qu'un minimum local relativement à un type de voisinage ne l'est pas forcement pour un autre type de voisinage, il est peut être utile d'utiliser des métaheuristiques basées sur plusieurs types de voisinage c'est le cas de la Recherche à Voisinage Variable où la diversification est faite.

3.3.5 Utilisation de mémoire à long terme vs mémoire à court terme

Une caractéristique importante est l'utilisation de l'historique pour la classification, les algorithmes sans mémoire sont des processus markoviens puisque l'action à réaliser est totalement déterminée par la situation courante.
Une mémoire à long terme conserve une trace de mouvements récemment effectués, des solutions visitées ou en générale des décisions prises. Une mémoire à court terme est une accumulation de paramètres de synthèse sur la recherche.

3.4 Présentation des principales Métaheuristiques

Il existe un grand nombre de métaheuristiques d'optimisations. Nous allons nous appuyer sur la classification qui distingue les méthodes de trajectoires (qui exploitent séquentiellement un seul voisinage) des méthodes basées sur des populations de solutions (qui exploitent plusieurs à la fois).

3.4.1 Méthodes de trajectoires

3.4.1.1 Méthode de Descente (Hill Climbing)

Les Méthodes de Descente sont assez anciennes. Leur succès revient à leur simplicité et à leur rapidité **[Papadimitriou, 1976]**. La principale caractéristique de la méthode de descente est sa grande simplicité de mise en oeuvre. La plupart du temps, elle ne fait que calculer f(s+i)-f(s), où, i correspond à un déplacement élémentaire, et si cette expression peut se simplifier algébriquement, alors on pourra évaluer très rapidement cette différence. Le principe consiste à choisir une solution s' dans le voisinage d'une solution s en améliorant la recherche tel que f(s')< f(s).
On peut décider soit d'examiner toutes les solutions de voisinage et prendre la meilleure de toutes (ou prendre la meilleur trouvée), soit d'examiner un sous ensemble de voisinage.
La méthode de Descente peut être décrite comme suit :

Méthode de Descente
Procédure Descente Simple (solution initiale s)
Répéter :
Choisir s' dans N(s)
Si f(s')<f(s) alors $s \rightarrow s'$
Jusqu'à ce que $f(s') < f(s), \forall s' \in S$

L'ensemble S définit l'ensemble des points pouvant être visités durant la recherche. La structure de voisinage N donne les règles de déplacement dans l'espace de recherche.
La fonction f induit une topologie sur l'espace de recherche.
Une variante consiste à parcourir N(s) et à choisir la première solution s' rencontrée telle que f(s') <f(s) (pour autant qu'une telle solution existe).

3.4.1.2 Recuit Simulé (Simulated Annealing)

La méthode du Recuit Simulé est issue d'une analogie entre le phénomène physique de refroidissement lent d'un corps en fusion, qui le conduit à un état solide, de basse énergie. Il faut abaisser lentement la température, en marquant des paliers suffisamment longs pour que le corps atteigne l'équilibre thermodynamique à chaque palier de température pour les matériaux, cette énergie se manifeste par l'obtention d'une structure régulière, comme dans les cristaux et l'acier **[Syari et al, 1983]**.
Par analogie avec le processus physique, la fonction à minimiser est l'énergie du système E, on utilise aussi un paramètre « la température du système T ».
A partir de l'espace de solution, s_0 est choisi aléatoirement, on associe avec cette solution une énergie initiale $E = E\_0$ et une température initiale T=T_0 élevée.
La solution est modifiée à chaque itération de l'algorithme, ce qui entraine une variation ΔE de l'énergie du système. Si cette variation diminue l'énergie du système (ΔE est négative) on l'accepte, sinon on l'accepte avec une probabilité $e^{frac-\Delta ET}$, La température est restée constante.
Cet algorithme commence par s_0, et continue jusqu'à k_max (maximum d'étapes) ou jusqu'à un état ayant une énergie e_max, l'appel voisin (s) engendre un état aléatoire voisin d'un état s, l'appel aléatoire renvoie une valeur aléatoire dans l'intervalle [0,1]. L'appel temp(r) renvoie la température à utiliser selon la fraction r du temps total déjà dépensé.

Recuit Simulé

$s \longleftarrow s\_0$;

$e \longleftarrow E(s)$;

$k \longleftarrow 0$;

Tantque $k < k\_max$ **et** $e > e\_max$ $s \longleftarrow voisin(s)$;

$e\_n \longleftarrow E(s\_n)$;

si $e\_n < eoualatoire() < p(e\_n - e, temp(k/k\_max))$

alors

$s \longleftarrow s\_n$;

$e \longleftarrow e\_n$;

$k \longleftarrow k + 1$;

Retourner s

3.4.1.3 Recherche Tabou (Tabu Search)

La Recherche Tabou a été introduite par l'auteur Glover **[Gloer, 1986]**. Ainsi l'idée de base se trouve dans **[Hansen, 2000]**, elle n'a aucun caractère stochastique et utilise la notion de mémoire pour éviter de tomber dans un optimum local.
Dans la recherche Tabou et à partir d'une solution donnée, on examine toutes ses voisines, on retient la meilleure solution trouvée même s'elle reste non améliorante.
On ne s'arrête pas à un optimum trouvé pour la première fois. Cette stratégie peut entrainer des cycles, par exemple un cycle de longueur 3 pour éviter ce type de cycle, on utilise la notion de mémoire (liste Tabou), on sauvegarde les k dernières configurations visitées dans une mémoire à court terme et on interdit tout mouvement qui conduit à une de ces configurations. Cette mémoire nous permet d'éviter tous les cycles de longueur inférieure ou égale à k (une valeur qui dépend du problème à résoudre). La mémorisation de configurations serait trop couteuse en termes du temps de calcul et de place mémoire, ce qui diminue l'efficacité de la méthode. La liste Tabou mémorise des caractéristiques de configurations complètes. C'est-à-dire lorsqu'un mouvement vient d'être effectué, c'est généralement la caractéristique perdue par la configuration courante qui devient Tabou.
Lorsque les listes Tabou font intervenir des caractéristiques de modifications, les interdictions qu'elles engendrent peuvent s'avérer trop fortes et restreindre l'ensemble des solutions admises à chaque itération d'une manière jugée trop brutale. Un mécanisme particulier, l'aspiration, est utilisé pour résoudre ce problème, il permet de lever le statut Tabou d'une configuration, sans introduire un risque de cycles dans le processus de recherche. La fonction d'aspiration est définie de différentes manières. La plus simple consiste à révoquer le statut Tabou d'un mouvement si ce dernier permet d'atteindre une meilleure solution que celle trouvée.
Etant donnée s une solution courante, T : liste tabou $N^T(s)$ est un ensemble de solutions de N(s) vers laquelle la Recherche Tabou accepte de se diriger. Cet ensemble contient toutes les solutions qui ne sont pas taboues ainsi que celles qui le sont mais dont le statut tabou est levé en raison de critère d'aspiration.
Le critère d'arrêt, par exemple, est de fixer un nombre maximum d'itérations sans améliorer s^* ou de fixer un temps limite après lequel la recherche doit s'arrêter.

Recherche Tabou

$s \longleftarrow s\_0$;

$T \longleftarrow null$;

$s^* \longleftarrow s$;

Tantque(arrêt n'est pas satisfait) ;

Déterminer une solution s' qui minimise f(s') dans $N^T(s)$

si$f(s') < f(s^*)$ **alors**

$s* \longleftarrow s'$;

$s \longleftarrow s'$;

mise-à-jour T ;

Fin tq

3.4.1.4 Méthode GRASP (Greedy Randomized search Procedure)

La méthode GRASP est une méthode itérative introduite par les auteurs Feo et Resende **[FEO, 1989]** et **[FEO, 1994]**. GRASP est une hybridation puisqu'elle combine les avantages de la Méthode de Descente, la recherche aléatoire et la méthode de Voisinage.
C'est une méthode itérative qui se déroule en deux phases :
Une phase de construction d'une solution et une phase d'amélioration d'une solution.
Durant la phase de construction et d'une manière itérative une solution est construite : chaque itération ajoute un élément, l'élément rajouté est choisi à partir d'une liste des candidats, on utilise une liste des meilleurs candidats retenus avec un critère donné (fonction gloutonne ou une heuristique permettant de mesurer le bénéfice qu'on peut espérer).
Cette liste est appelée RCL (Restricted Candidate List), RCL est dynamiquement mise à jour.
- La phase de construction continue jusqu'à ce qu'une solution complète soit obtenue.
- La phase d'amélioration : à partir de la solution trouvée par la phase précédente, une Méthode de Descente, Recuit Simulé ou Recherche Tabou est appliquée pour améliorer la solution.

Méthode GRASP

$f^* \longleftarrow \infty$(une valeur de meilleur solution rencontrée)

Tant que (critère d'arrêt n'est pas vérifié)

$S \longleftarrow ensemblevide$;

Tant que(arrêt n'est pas satisfait) ;

- évaluer l'ensemble des éléments pouvant etre rajoutés a s

- déterminer la liste RCL

- choisir un meilleur élément dans RCL et l'ajouter a S

Fin tq

- appliquer une méthode de Descente,Recuit Simulé ou de Recherche Tabou a $s : s^{'} \longleftarrow s$

Si $f(s^{'}) < f(s^*)$

Alors

$s^{'} \longleftarrow s^*$

$s^*) \longleftarrow f(s^{'}$

FIN Tq

Une procédure de GRASP répète les deux phases et retourne à la fin une meilleure solution trouvée. GRASP est une méthode simple qui permet d'améliorer les solutions par l'intervention d'une heuristique spécialisée. Mais elle est moins performante par rapport à d'autres Métaheuristiques (elle peut tomber sur le minimum local et

ceci est du à l'absence de mémoire).

3.4.1.5 Recherche à Voisinages Variables (VNS)

L'idée de base de la méthode de VNS est le changement systématique de la structure de voisinage avec une méthode heuristique de recherche locale au lieu d'utiliser une structure de voisinage unique, elle a été proposée par **[Mladenovic et Hansen, 1997]**.
Etant donné un ensemble de structures de voisinages (qui sont souvent imbriquées), une solution est générée aléatoirement dans le voisinage de la solution courante dont laquelle une Méthode de Descente locale est effectuée. Si l'optimum local obtenu n'est pas le meilleur que la solution courante, on passe vers le voisin suivant (répétition de procédure de recherche). On recommence la recherche à partir du premier voisinage où il y a d'autres solutions meilleures qui n'ont pas été explorées ou on explore chaque structure de voisinage . Tel que $N^{(t)}(S)$ est l'ensemble des solutions dans le t^{me} voisinage de s.

Recherche a voisinages Variables

choisir une solution $s \in S$;

$t \longleftarrow 1$;

Tant que(arrêt n'est pas vérifié) ;

- choisir aléatoirement une solution $s^{'} dans N^{t}(S)$

- Appliquer une procédure de recherche locale a $s^{'}$

$s^{"} \longleftarrow s^{'}$;

Si $f(s^{"}) < f(s^{*})$

Alors

$s \longleftarrow s^{"}$; $t \longleftarrow 0$;

Si $t < T$**Alors**$t \longleftarrow t + 1$;

SINON

$t \longleftarrow 1$

FIN Tq

3.4.2 Méthodes basées sur les populations

3.4.2.1 Algorithmes de Colonies de Fourmis (Ant Colony Optimization)

Une colonie de fourmis communiquent indirectement via des modifications dynamiques des pistes de phéromones et construisent une solution à un problème donné. La description de l'algorithme de colonie de Fourmis sera présentée dans une prochaine section.

3.4.2.2 Algorithmes Génétiques (Genetic Algorithm)

Les Algorithmes Génétiques sont des algorithmes d'optimisation qui s'appuient sur des techniques dérivées de la génétique et de l'évolution naturelle : sélection, croisement, mutation. Ils ont été inventés dans les années de 60 **[Holland, 1962]**. L'auteur Goldberg a étudié les Algorithmes Génétiques et il a enrichit sa théorie par **[Goldberg, 1989]** :

- Un individu est lié à un environnement par son code d'ADN ;
- Une solution est liée à un problème par son indice de qualité ;
- Une bonne solution à un problème donné peut être vue comme un individu susceptible de survivre dans un environnement donné.

Les éléments suivants sont nécessaires pour utiliser un Algorithme Génétique :

1. Le principe du codage d'un élément d'une population : généralement après une phase de modélisation du problème à étudier, on associe à chaque point de l'espace d'états une structure de données. Le succès de l'Algorithme Génétique est lié à la qualité du codage. Il y'a le codage réel (utilisé pour l'optimisation du problème à variables réelles) et le codage binaire qui est le plus utilisé ;
2. La génération d'une population initiale : le choix d'une population initiale est nécessaire pour les générations futures et principalement pour converger rapidement vers un optimum global.
3. Les opérateurs permettent la diversification de la population au cours des générations et l'exploration d'espace d'états. L'opérateur de croisement recompose les gènes d'individus existant dans la population. L'opérateur de mutation a pour but de garantir l'exploration de l'espace d'états.
4. Les paramètres de dimensionnement : la taille de la population, le critère d'arrêt, les probabilités d'application des opérateurs de croisement et de mutation.

Un Algorithme Génétique recherche le ou les extrema d'une fonction définie sur un espace de données.

Algorithme Génétique
Générer aléatoirement une population d'individus.
Pour passer d'une génération k a une génération$k+1$, on applique les trois opérations pour tous les éléments
de la population k :
1-sélectionner les couples de parents$p_1 et p_2$ en fonction de leur adaptation ;
2-appliquer l'opérateur de croisement avec une probabilité p_c et générer les couples d'enfants $c_1 et c_2$, certains
éléments sont choisis en fonction de leur adaptation , on 3-applique l'opérateur de mutation avec une probabilité $p_m$et
évaluer$c_1 et c_2$, pour l'insérer dans la nouvelle population.
le critère d'arrêt : Soit le nombre de génération est fixé a l'avance ou un temps limité pour trouver une solution ou
il n'y a pas d'évolution

Voici le corps principal d'une itération d'un algorithme génétique :

- Évaluer la qualité (fitness) des individus et leurs chances de survie
- Sélectionner les individus pour la reproduction
- Effectuer la reproduction
- Remplacer l'ancienne population par la nouvelle

qualité (fitness) d'un individu permet d'illustrer avec une valeur numérique (et donc plus facile à manipuler) la qualité des gènes qui forment l'individu. Plus la qualité d'un individu est grande, plus ses chances d'être sélectionné pour la reproduction sont grandes.
En général, on calcule les chances de reproduction d'un individu en regard de sa qualité en relation avec la qualité totale des individus de la population. La reproduction se fait en croisant deux individus. On applique sur les deux individus choisis des opérateurs génériques, habituellement l'enjambement (cross-over) et la mutation. La reproduction donne deux enfants (offspring), qui sont placés dans la nouvelle population. On répète la reproduction jusqu'à ce qu'on ait rempli la nouvelle population (la taille de la population devrait rester constante). On remplace alors l'ancienne population par la nouvelle, et on recommence le processus le nombre de générations voulu.

EXPLICATIONS DES ÉLÉMENTS D'UN ALGORITHME GÉNÉTIQUE

Individu : Les individus correspondent aux « solutions » du problème à optimiser. Ces solutions doivent être « codées » pour que le traitement puisse être effectué par l'algorithme génétique. Cette représentation codée d'une solution est appelée chromosome, et est composée de gènes. Chaque gène peut représenter une variable, un

élément de la solution, ou encore une partie plus abstraite. La manière la plus utilisée de codage par algorithme génétique est le codage en vecteurs.

Chaque solution est représentée par un vecteur. Ce vecteur peut être binaire ou encore de n'importe quel type discret dénombrable (entier, caractères, etc...).

On pourrait également utiliser un type continu (ex : nombres réels), mais dans ce cas, il faut également revoir les opérations qui modifient le contenu des chromosomes (la fonction qui crée aléatoirement les chromosomes et les opérateurs génétiques).

La simplicité veut que les chromosomes soient uniformes, c'est-à-dire que tous les gènes sont du même type. Cependant, si on tient compte encore une fois des opérations qui modifient le contenu des chromosomes, on peut assez aisément construire des vecteurs d'éléments de type différents. On demande habituellement que les chromosomes soient tous de même longueur, basés sur la même architecture, les gènes homologues étant au même endroit sur leur chromosome respectif. De fait, le codage par vecteur est si utilisé (très grande simplicité comparée aux autres méthodes de codage de chromosome) que les algorithmes sont souvent identifiés comme étant des méthodes de traitement vectoriel. Cette affirmation n'est pas tout a fait vraie car d'autres types de codage existent, bien que n'étant pas très fréquents. Par exemple, l'utilisation des algorithmes génétiques pour faire de la programmation génétique utilise un codage en arbre (ce qui permet entre autres d'avoir des chromosomes de longueurs différentes).

Population :
C'est l'ensemble des individus, ou encore l'ensemble des chromosomes d'une même génération. Habituellement, la taille de la population reste constante tout au long de l'algorithme génétique.

Générer :
Habituellement, au départ d'un algorithme génétique, il faut créer une population d'individus. Ces individus sont générés par une fonction simple. Cette fonction affecte à chaque individu qu'elle crée une valeur aléatoire pour chacun de ses gènes. L'algorithme génétique peut également utiliser comme population de départ une population déjà créée a priori.

Qualité, fitness, d'un individu : Le calcul de la qualité d'un individu est essentiel aux algorithmes génétiques. Cette fonction donne, en valeur numérique (habituellement réelle), la qualité d'un individu. C'est selon cette valeur numérique qu'est calculée les chances de sélection de cet individu. La fonction de fitness doit avoir 0 comme plancher, pour ne pas fausser le calcul des pourcentages. Les algorithmes génétiques étant une technique d'optimisation, ils cherchent la qualité maximale, donc l'optimisation de la fonction de qualité. Si on cherche plutôt à minimiser une fonction, il faudra la modifier de sorte que la fonction de qualité se maximise. Il serait bien entendu possible de conserver un fonction de qualité qui fonctionne à l'envers et de modifie à la place le calcul des probabilités, mais ceci rendrait l'algorithme beaucoup plus difficile à décoder pour les utilisateurs externes.

Sélection :
Selon la qualité des individus, chacun se voit attribuer un pourcentage de chances d'être choisi pour la reproduction, qui correspond à l'importance relative de la qualité de l'individu par rapport à la qualité totale de la population.

Reproduction :
La reproduction s'effectue généralement en croisant deux individus, ce qui produit deux nouveaux individus à placer dans la nouvelle population. De la manière classique, la reproduction consiste à appliquer les opérateurs génétiques sur les deux chromosomes sélectionnés et mettre les deux chromosomes résultant dans la nouvelle population. Les deux opérateurs génétiques courants sont l'enjambement (cross-over) et la mutation. Le premier est la véritable reproduction. Chaque chromosome enfant reçoit environ la moitié des gènes de chacun de ses parents. La probabilité d'enjambement est presque toujours de 50 De cette façon, après l'enjambement, on obtient deux enfants qui sont complémentaires par rapport à leurs parents (si un enfant a un gène d'un tel parent, l'autre enfant tiendra ce même gène de l'autre parent). L'opérateur de mutation, bien qu'ayant une probabilité bien moindre (habituellement entre 0,5En effet, les gènes des enfants sont limités par les gènes des parents, et si un gène n'est pas présent dans la population initiale(ou s'il disparaît à cause des reproductions), il ne pourra jamais se développer chez les descendants. L'opérateur de mutation est là pour contourner ce problème.
Chaque gène possède une faible probabilité de muter, c'est-à-dire d'être aléatoirement remplacé par une autre incarnation de ce gène. Cette précaution permet de conserver ce qu'on appelle la diversité génétique.
Habituellement, la mutation crée des individus faibles, peu aptes à survivre. Cependant, un certaine mutation

pourrait se révéler « géniale » et permet d'augmenter grandement l'évolution de la population.
Ces opérateurs génétiques peuvent habituellement s'accommoder de la plupart des situations sans trop de modifications. Toutefois, afin de conserver la cohérence des individus, on doit parfois modifier grandement, voire changer totalement les opérateurs génétiques. Il n'y a pas de limite quant au nombre d'opérateurs génétiques. D'une manière générale, on considère comme opérateur génétique, n'importe quelle fonction qui modifie le contenu génétique d'un ou de plusieurs chromosomes. Il parle aussi quelques fois de l'opérateur génétique de clonage. Celui-ci existe (un individu est passé directement, sans modifications, à la génération suivante) et peut être incorporé dans un algorithme génétique. Cependant, le potentiel évolutif de cet opérateur est secondaire. On assiste également à un clonage lorsque les autres opérateurs génétiques n'ont pas d'effet (cas très rare).

Validité et cohérence :
Selon la méthode de codage et sa signification, on doit être toujours certain que les individus de notre population soient valides. En effet, que ferait-on d'une solution donnant une bonne note de qualité, mais n'ayant aucun sens pratique une fois interprété. Il faut donc s'assurer que la fonction de création des individus crée toujours des individus valides, et que les opérateurs génétiques conservent la validité des individus traités. Ceci vise à conserver la cohérence générale de l'algorithme.

Critère de terminaison :
Généralement, un algorithme génétique se termine après un certain nombre de générations, mais on peut également terminer l'exécution de l'algorithme lorsqu'une certaine condition soit atteinte, par exemple lorsque la qualité d'un individu dépasse un certain seuil.

EXEMPLE SIMPLE D'ALGORITHME GÉNÉTIQUE

Problématique : On veut trouver l'entier (de 0 à 16) qui maximise la fonction .

$F(x) = 1/4|15x^2 - x^3| + 4$

Population : 6 chromosomes de 4 bits (codage), représentant sous forme binaire les entiers (individus) qu'ils représentent.
Fonction de qualité :

$F(x) = 1/4|15x^2 - x^3| + 4$
Opérateurs génétiques :
enjambement : Il y a 50de chance qu'un bit soit échangé avec son homologue sur l'autre chromosome.
Mutation : Chaque bit a 3de chance de muter (passer de 1 à 0 ou de 0 à 1).

étiquette du chromosome	chaine du chromosome	entier décodé	qualité (fitness)	chances de reproduction
$X1$	0100	4	48	13.58%
$X2$	0000	0	4	1.13%
$X3$	1110	14	53	14.99%
$X4$	0001	1	7.5	2.13%
$X5$	1000	8	116	31.81%
$X6$	1011	11	125	35.36%

Moyennes de la qualité (average fitness) : 58,917.
Sélection et opérateurs génétiques : Chromosome enjambement –> mutation –> chromosomes finaux

Couple 1 :
X5 1000 1000 1010 1010
X3 1110 1110 1100 1100
Couple 2 :
X5 1000 1000 1011 1011
X6 1011 1011 1000 1000
Couple 3 :
X6 1011 1011 1000 1001
X1 0100 0100 0111 0111

Étiquette du Chromosome	Chaîne du chromosome	Entier décodé	Qualité (fitness)	Chances de reproduction
X1'	1010	10	129	18,18 %
X2'	1100	12	112	15,79 %
X3'	1011	11	125	17,62 %
X4'	1000	8	116	16,35 %
X5'	1001	9	125,5	17,69 %
X6'	0111	7	102	14,38 %

Moyennes de la qualité (average fitness) : 118,25.

La population de la deuxième génération peut maintenant servir pour engendrer une troisième génération. Bien qu'ici, on ait trouvé la solution optimale (10) en une seule génération, on remarque quand même que les la moyenne des chances de survie a augmenté considérablement. De cette façon, on peut conclure que l'algorithme génétique encourage le « perfectionnement de la race ».

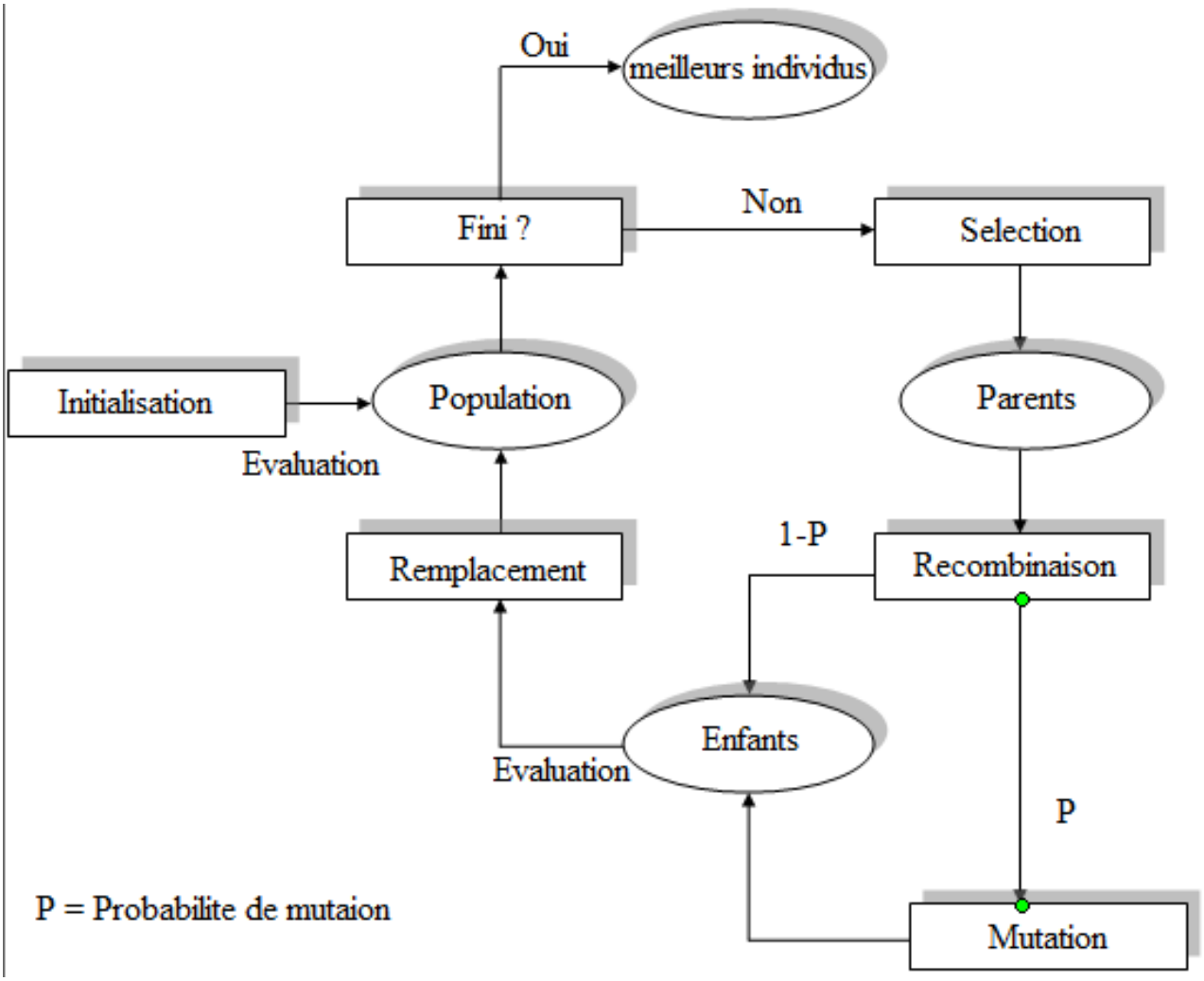


FIGURE 3.2 – Schéma global d'un algorithme génétique.

3.4.2.3 Algorithme d'Estimation de Distribution (EDA)

Cet algorithme a été introduit par l'auteur Baluja en 1994 **[Baluja, 1994]** dans ses travaux sur PBI (Apprentissage Incrémental Basé sur Population) et puis par les auteurs muhlenbein et Paab en 1996 **[Mühlenbein et paass, 1996]** Cet algorithme utilise les informations collectées tout au long du processus d'optimisation pour construire des modèles probabilistes des bonnes régions de l'espace de recherche et pour engendrer des nouvelles solutions [Larranaga et Lozano, 2001]. Elle n'utilise pas les opérateurs de croisement et de mutation utilisés

par les algorithmes évolutionnaires. On construit une nouvelle solution à partir des solutions des générations précédentes.

Algorithme d'estimation de distribution

Initialisation aléatoirement une population X^0

Calculer le modéle $p^0 : X^t \longleftarrow X^0$;

Tant que (le critère d'arrêt n'est pas vérifié)

- Sélection de la sous famille $X^t_{parent} de X^t$

- Calcul de la sous famille X^t_{parent}

- Echantillonnage de $p^{t+1} pour creer X^t_{offspring}$

- Remplacement de certains éléments de X^t par les éléments de X^t par les éléments de $X^t_{offspring}$ pour créer X^{t+1}

$t \longleftarrow t+1$;

Fin tq

- On génère une famille de solutions X^0.
- On construit un modèle probabiliste p^0 .
- La boucle tant que regroupe quatre étapes :

- choix d'une sous famille de meilleures solutions.
- construction du modèle probabiliste p^1 à partir de cette sous famille.
- génération d'une nouvelle famille de solution $X^1\_offspring$ par l'échantillonnage du modèle p^1.
- remplacement de certains éléments de X^0 par les éléments de $X^1\_offspring$ pour créer X^1.

- on itère le processus jusqu'à ce que le critère d'arrêt soit satisfait.

La distribution probabiliste est calculée par différentes manières suivant le modèle considéré (gaussienne, ...) **[Larranaga et Lozano, 2001]**.

3.4.2.4 Optimisation par Essaims particulaires (Particle Swarm Optimization)

L'optimisation par Essaims particulaires a été introduite par les auteurs Russel et James en 1995 **[Kennedy et Eberhart, 1995]**. Cet algorithme s'inspire à l'origine du monde du vivant. Il s'appuie notamment sur un modèle développé par le biologiste Craig Reynolds à la fin des années 1980 pour la simulation du déplacement d'un groupe d'oiseaux.

L'algorithme PSO déplace un essaim de particules dans l'espace de recherche. Le déplacement de chaque particule est influencé par sa vitesse, la meilleure position qui a été retenue et la meilleure position connue par toutes les particules de l'essaim.

Soit x_i un vecteur de position de la i^{me} particule de l'essaim v_i un vecteur de vitesse de cette particule. D la dimension de ce problème. x_i et v_i sont des vecteurs à D éléments dont la j^{me} est notée respectivement x_{ij} et v_{ij} Soit p_i un vecteur de dimension D qui correspond à la meilleure position atteinte par la particule i et p_{ij} sa coordonnée sur la dimension j. on note g le vecteur de dimension D qui correspond à la meilleure position connue de l'essaim.

Optimisation par Essaim Particules
Initialiser aléatoirement un essaim
Évaluer la fonction (objectif) pour chaque essaim
$x_i \longleftarrow p_i; i = 1, ..., N(tailledel'essaim)$
Calcul g
Tant que (le critère d'arrêt n'est pas vérifié)
Mis-a-jour $x_i etv_i$ selon les equations (01) et (02)
Evaluer la fonction objectif
mis-a-jour p_i
mis-a-jour g
Fin tq

A l'itération t+1 le déplacement des particules est calculé à l'aide des équations :

$$v_{ij}(t+1) = w.v_{ii}(t) + c_1.r_1.(p_{ij} - x_{ij}(t)) + c_2.r_2.(g_j - x_{ij}(t)) \quad (01)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (02)$$

w : coefficient de l'inertie
c_1, c_2 : coefficients d'accélération.
$r_1, r_2 \in [0, 1]$ (générés aléatoirement).
La méthode d'optimisation par essaim particulaires partage beaucoup de similarité avec l'Algorithme Génétique dans le sens où les propriétés d'un individu sont influencées par les caractéristiques des autres.

3.4.2.5 Recherche par Dispersion (Scatter Search)

C'est une stratégie évolutionnaire, elle a été proposée par l'auteur Glover **[Glover, 1977]**. Elle est appliquée à un grand nombre de problèmes d'optimisation. Cet algorithme démarre avec une population de solutions dont lequel un sous ensemble de solutions est sélectionné pour un ensemble de références ensRef qui évolue par les mécanismes d'intensification et de diversification. Les solutions de cet ensemble sont combinées pour générer des nouvelles solutions mettant à jour l'ensemble de références (la combinaison est guidée, elle n'est pas aléatoire contrairement à l'Algorithme Génétique). L'ensemble de références dans l'algorithme de Recherche par Dispersion est de taille petite que la taille des populations dans les algorithmes évolutionnaires.

Recherche par dispersion
Procédure
Début
- Générer une population (InitPOP)
- Générer un ensemble de référence (RefSet)
Répéter
Répéter
Sélection d'un sous ensemble (subset) ;
Amélioration (s,s+)
Jusqu'a (le critére 01 est satisfait)
mise ajour ensemble de références (refSet)
Jusqu'a (le critére 02 est satisfait)
Fin

3.5 Algorithmes de Colonies de Fourmis (ACO)

3.5.1 Optimisation naturelle par les fourmis

Les fourmis, les abeilles et les termites sont des insectes sociaux qui présentent un comportement de groupe intelligent et complexe. Le comportement collectif des insectes apparait à partir d'une interaction simple et indirecte des membres d'un groupe appelé les colonies. Les fourmis étant semi ou complètement aveugles, elles communiquent entre elles utilisant un produit chimique volatil appelé phéromone (produite par les glandes situées dans les abdomens des fourmis). Elles sont attirées par ces substances, qu'elles perçoivent grâce à des récepteurs situés dans leurs antennes. La phéromone est utilisée pour des alertes de sécurité et pour guider les autres fourmis vers la source de nourriture. Le comportement coopératif de recherche de nourriture était un intérêt particulier pour les chercheurs qui ont constaté que la colonie est en mesure de trouver le plus court chemin entre le nid et une source de nourriture même si les fourmis individuelles n'ont pas une vision globale du chemin.
La Figure si dessous illustre un exemple d'une colonie de fourmis qui est connectée à une source de nourriture par deux branches différentes. Les fourmis effectuent une recherche autour du nid. Elles empruntent les différents chemins dont initialement il n'y a pas de traces de phéromones .

FIGURE 3.3 – Etape expérimentale pour observer le comportement des fourmis

FIGURE 3.4 – Situation de début de recherche-comportement de fourragement

Quand une fourmi atteint la source de nourriture, elle retrace son chemin de retour vers le nid par une quantité de phéromone pour avertir les autres fourmis qu'il y a de la nourriture à la fin du chemin. Les fourmis les plus rapidement arrivées au nid après avoir visité la source de nourriture sont celles qui empruntent le chemin le plus court, aussi la quantité de phéromone présente sur le plus court chemin est plus importante que celle présente sur le chemin le plus long. Une piste qui présente une plus grande concentration en phéromone est plus attirante par les fourmis, elle a une probabilité plus grande d'être empruntée. La piste courte va alors être plus renforcée que la plus longue, et elle sera choisie par la quasi-totalité des fourmis .

FIGURE 3.5 – Comportement du fourragement après un certain temps

Cette approche est initialement adaptée pour l'optimisation par les auteurs Colorni, dorigo et Maniezzo **[colorni et al., 1992]**.

3.6 Exemple du problème du voyageur de commerce (PVC)

Le problème du PVC consiste à trouver le plus court chemin permettant de relier un ensemble de villes en ne passant qu'une et une seule fois par une ville. Le problème revient à trouver le cycle hamiltonien minimal dans un graphe complet pondéré où les sommets sont les villes et les arcs sont les chemins entre les villes.
La résolution par Ant System (AS) [colorni et al, 1992] :
Le TSP a fait l'objet de la première implémentation d'un algorithme de colonie de fourmis, par suite une description de l'algorithme :
Soit G = (V,E) un graphe complet.
A chaque itération t chaque fourmi k=1,. . ., m parcourt le graphe et construit un trajet complet de n=|V| étapes (|V| étant le cardinal de l'ensemble de sommets V). Pour chaque fourmi, le trajet entre la ville i et la ville j dépend de :

- La liste des villes que la fourmi k peut visiter quand elle est sur la ville i (c'est-à-dire les villes qui sont reliées à i par une arrête et que la fourmi k n'a pas encore visité) : J_i^k ;
- L'inverse de la distance entre les villes : $\eta_{ij} = 1/d_{ij}$ appelée la visibilité ou la valeur heuristique ;
- La quantité de phéromones déposée sur l'arrête reliant les deux villes : τ_{ij} .

On utilise des lois de transition aléatoires où $p_{ij}^k(t)$ est la probabilité que la fourmi k, située sur la ville i à l'itération t se déplace vers la ville j

$$P_{ij}^k(t) = \{\frac{(\tau_{ij}(t))^\alpha.(\eta_{ij(t))^\beta}}{\Sigma le J_t^k((\tau_{il}(t))^\alpha.\eta_{il}(t))^\beta)}\} sij \in J_i^k$$
$$0 \text{ Sinon}$$

Où $\alpha et \beta$ sont deux paramètres à ajuster contrôlant l'importance relative de l'intensité de $\tau_{ij}(t)$ et de la visibilité η_{ij}
$\alpha = 0$, seule la visibilité de la ville qui a l'influence.
B=0 , seule les pistes de phéromones influent.
Pour éviter une sélection trop rapide d'un trajet, on doit procéder au réglage de ces paramètres en jouant sur les comportements d'intensification et de diversification.
Après un tour complet, chaque fourmi laisse une quantité de phéromone $\delta\tau_{ij}^k(t)$ sur l'ensemble de son parcours. Cette quantité dépend de la qualité de la solution trouvée.

$$\Delta_{ij}^k(t) = \{\frac{Q}{L^k(t)} si(i,j) \in T^k(t)\}$$
$$0 \text{ Sinon}$$

Où $T^k(t)$ est le trajet effectué par la fourmi k à l'itération t
$L^k(t)$ est la longueur du tour et Q un paramètre fixé.
Enfin, le processus d'évaporation entre en jeu pour éviter de piéger dans une solution sous optimale en faisant

oublier au système les mauvaises solutions. A chaque itération la règle de mise à jour est comme suit :

$$\tau_{ij}(t+1) = (1-p).\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$$

Où p : est le taux d'évaporation et m : nombre de fourmis.
La quantité de phéromones sur chaque arrête est une distribution uniforme d'une petite quantité.
Le pseudo code de Ant system est :

Algorithme de Colonies de Fourmis de base : le Ant system

Pour $t = 1, ..., t_{max}$

Pour chaque fourmi $k = 1, ..., m$

Choisir une ville au hasard

Pour chaque ville non visités i

Choisir une ville j , dans la liste J_i^k de villes restants , selon la formule (03)

Fin Pour

Déposer une piste$\Delta\tau_{ij}^{k}$sur le trajet$T^k(t)$

Fin Pour

Evaporer les pistes selon la formule (05)

Jusqu'a (le critére 02 est satisfait)

Fin Pour

3.6.1 Variantes

Ant system et élitisme : Cet algorithme a été introduit par les auteurs **[Dorigo et al, 1996]**. Il utilise des fourmis élitistes. La fourmi qui effectue un chemin plus court dépose une quantité plus grande de phéromone pour l'exploration de la meilleure solution par les autres fourmis.

Ant-Q : cette approche utilise une règle de mise à jour locale inspirée de Qlearning **[Gambardella et Dorigo, 1995]**, c'est une préversion de « Ant Colony System »

Ant Colony System : Cet algorithme a été introduit par les auteurs **[Dorigo et Gambardella, 1997]**. C'est une variante la plus connue de Ant System pour améliorer les performances de cet algorithme sur des problèmes de grandes tailles où une nouvelle règle de déplacement (règle pseudo-aléatoire proportionnelle) s'ajoute un processus de mise à jour locale des éléments des pistes de phéromones, l'objectif de ce mécanisme étant d'augmenter la diversification de la recherche.

Max Min Ant System : c'est une variante plus efficace de « Ant System », où seules les meilleures fourmis tracent des pistes et le dépôt de phéromone est limité par une borne supérieure (empêchant une piste d'être trop renforcée) et une borne inférieure (laissant la possibilité d'être explorée à n'importe quelle solution).
Cet algorithme atteint de meilleurs résultats que l'original, et évite notamment une convergence prématurée **[Stutzle and Hoos, 1997].**

3.7 Formalisation d'un algorithme de Colonie de Fourmis

3.7.1 Définition formelle d'un problème d'optimisation

Etant donné un problème d'optimisation (s,Ω, f) :
- S : est un espace de recherche défini sur un ensemble fini de variables de décision discrètes
- Ω : est un ensemble de contraintes sur les variables.
- $f : s \longrightarrow R^+$ une fonction objectif à minimiser.

L'ensemble S_Ω des solutions réalisables est l'ensemble des éléments de S qui vérifient toutes les contraintes de Ω. Un élément $s^* \in S\Omega$ est un optimum global ssi :
$s \in S_\Omega, f(s^*) \leqslant f(s)$.

L'espace de recherche S est défini sur un ensemble de variables discrètes $Xi(i = 1, \ldots, n)$ ayant chacune $v_i^j \in v_i^1, \ldots, v_i^{|Di|}$. L'ensemble$S_\Omega^*$.

Optimisation par colonie de fourmi

Résoudre un problème d'optimisation combinatoire par colonie de fourmis revient à parcourir un graphe de construction complet. On note c_{ij} chaque variable instanciée $x_i = v_i^j$ et C l'ensemble des c_{ij} .
On obtient le graphe de construction $G_c(V, E)$ en associant C à l'ensemble des sommets V ou l'ensemble des arrêtes E. une trace de phéromone τ_{ij} est associée à chaque c_{ij} .
Les fourmis se déplacent de sommet en sommet le long des arrêtes du graphe de construction en exploitant les informations que fournissent les traces de phéromones, et construisent ainsi peu à peu à une solution. De plus, chaque fourmi dépose une certaine quantité de phéromones pour chaque c_{ij} rencontré lors de son parcours. Cette quantité $\Delta\tau$ peut dépendre de la qualité de la solution trouvée. Les fourmis suivantes utilisent l'information fournie par les traces de phéromones pour se guider et atteindre les régions les plus prometteuses de l'espace de recherche.

Optimisation par Colonies de Fourmis

Définir les paramètres , initialiser les traces de phéromones

Tant que (critére d'arret n'est pas vérifié)

- Construction des solutions par les fourmis

- Actions spécifiques (facultatif)

- Mise a jour de phéromones

Fin tq

L'algorithme de colonies de fourmis se décompose en trois étapes qui se répètent à chaque itération :

- Construction des solutions par les fourmis
- Actions Spécifiques
- Mise à jour de phéromones

Généralement, le critère d'arrêt est défini par un nombre maximal d'itérations ou un temps CPU maximal.
Construction des solutions par les fourmis : Dans cette étape, m fourmis artificielles construisent des solutions à partir de l'ensemble des composantes de solution possibles
$C = c_{ij}(i = 1, \ldots, n; j = 1, \ldots, |Di|$, n étant soit le nombre de sommets, soit le nombre d'arêtes, selon que l'on a associé C à V ou à E).
La construction d'une solution commence par une solution partielle vide $s^p = \varnothing$. Puis, à chaque étape de la construction, cette solution partielle s^pest étendue en y ajoutant une composante de solution parmi l'ensemble des voisins réalisables $N(s^p) \subseteq C$. Le choix d'une composante dans $N(s^p)$ se fait de manière probabiliste à chaque étape de la construction. Chaque composante $c_{ij} \in N(s^p)$ a une probabilité $p(c_{ij}/s^p)$ d'être choisie. Les lois de probabilités dépendent de la variante utilisée.

Actions Spécifiques : Une fois l'étape de construction effectuée par les fourmis, et avant l'étape de mise à jour des traces de phéromones, des actions spécifiques au problème ou des actions centralisées (qui ne peuvent pas être effectuées séparément par chaque fourmi) peuvent être effectuées. Le plus souvent, ces actions consistent en une recherche locale parmi les solutions construites, et où seules les solutions localement optimisées sont utilisées dans la mise à jour des traces de phéromones.

Mise à jour de phéromones : Le but de la mise à jour des traces de phéromones, dernière étape d'une itération, est d'augmenter les valeurs de phéromones associées à de bonnes solutions, tout en réduisant celles associées à des mauvaises.
En général, la mise à jour se fait :

- en réduisant toutes les valeurs de phéromones par un procédé appelé évaporation ;

- en augmentant les valeurs de phéromones associées à un ensemble de bonnes solutions choisies S_{upd} .

A la fin d'une itération t, on effectue donc la mise à jour suivante :

$$\forall (i,j) \in E, \tau_{ij}(t+1) = \underbrace{(1-p)\tau_{ij}(t)}_{} + \rho \sum_{\delta \in S_{upd} | c_{ij} \in s} F(s)$$

Où $\rho \in [0,1]$ est le taux d'**évaporation** et $F : S \longrightarrow R^+$ est une fonction, appelée la fonction d'évaluation ou fonction fitness, telle que :

$$\forall s \neq s' \in S, f(s) < f(s') \Longrightarrow F(s) \geqslant F(s')$$

Stigmergie (une forme de communication indirecte)
Le choix de la méthode d'implémentation des pistes de phéromone est important pour obtenir les meilleurs résultats. Ce choix est lié principalement aux possibilités de représentation de l'espace de recherche, chaque représentation pouvant apporter une façon différente d'implémenter les pistes. Les pistes de phéromone décrivent à chaque pas l'état de la recherche de la solution par le système. Les agents modifient la façon dont le problème va être représenté et perçu par les autres agents. Cette information est partagée par le biais des modifications de l'environnement des fourmis, grâce à la stigmergie.

Intensification/Diversification
L'intensification est l'exploitation de l'information rassemblée par le système à un moment donnée (dans le cas des Algorithmes de Colonies de Fourmis, cela revient à favoriser les pistes de phéromones).
La diversification est au contraire l'exploration de régions de l'espace de recherche imparfaitement prises en compte. Le piège d'un mauvais réglage entre intensification et diversification serait de tomber rapidement dans une solution sous-optimale (intensification trop forte), ou de ne jamais trouver une solution (diversification trop forte). Dans le cas des Algorithmes à Colonies de Fourmis, plus la valeur de α sera élevée, plus l'intensification sera forte (les fourmis favoriseront les arcs avec beaucoup de phéromones). Inversement, plus α sera faible, plus la diversification sera forte.
Le paramètre β influe de la même manière sur l'intensification/diversification.

Parallélisme
Du fait que les solutions émergent des interactions indirectes ayant cours dans le système et que chaque fourmi ne prend en compte que des informations locales de son environnement (les pistes de phéromones), la parallélisation des algorithmes de fourmis est facile. On peut imaginer une solution multi-agent dans laquelle les fourmis seraient des agents purement réactifs.

3.8 Coclusion

Dans ce chapitre, nous avons décrit un état de l'art des méthodes d'optimisation basées sur les métaheuristiques les plus connues. Dans un premier temps nous avons défini les problèmes d'optimisation difficile, l'heuristique et la métaheuristique. Un intérêt particulier a été porté à la méthode d'optimisation par colonie de fourmi. Cet algorithme forme une classe de métaheuristique proposée pour les problèmes d'optimisation difficile. L'algorithme de Colonie de Fourmis s'inspire des comportements collectifs de dépôt et de suivi de pistes observées. Dans le chapitre suivant, nous allons voir l'implementation de notre algorithme proposé avec les resultats obtenue apres l'experimentation.

CHAPITRE IV :

Implémentation et résultats

Chapitre 4

Implémentation et résultats

4.1 Introduction

AFSA est un algorithme d'optimisation qui simule le comportement des essaims poissons , tels que la recherche de nourriture et de mouvement. Par exemple, la position de la plupart des poissons dans un bassin est typiquement la position à laquelle la plupart des denrées alimentaires peut être obtenue. L'AFSA comprend trois étapes principales, qui sont suivre, deplacer en essaim, et nouriture. Dans le AFSA, ces trois étapes sont répétées afin de déterminer la solution optimale. Semblable à d'autres algorithmes bioinspirés, l'AFSA est utilisé pour déterminer la solution optimale ou la plus satisfaisante dans un temps limité en recherchant constamment des solutions possibles en utilisant un métaheuristique. Dans le AFSA, la position de chaque poisson est considéré comme une solution, et chaque solution a une valeur de remise en forme qui est évaluée en utilisant la fonction de remise en forme. La fonction de remise en forme change lorsque des objectifs différents sont établis.

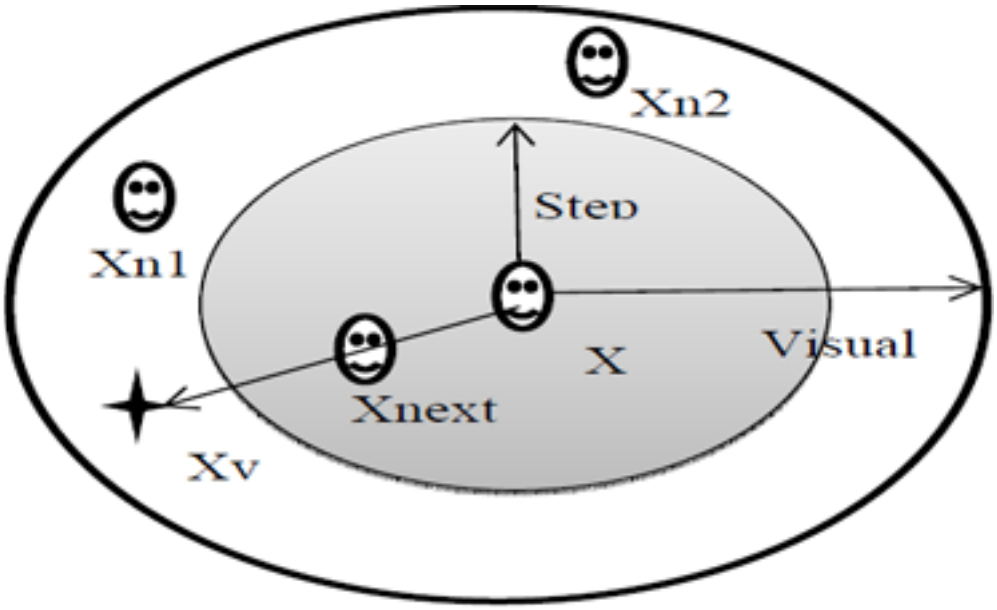


FIGURE 4.1 – concept Vision du poissons artificielle

4.2 Introduction a AFSA

Supposons que le vecteur d'état d'essaim de poissons artificiel est $X = (x_1, x_2..., x_n)$où $(x_1, x_2..., x_n)$est l'état du poisson. Visual est la distance visuelle, le poisson artificiel ne se produit pas les différents actes que dans le rayon intérieure du cercle à la longueur du champ de vision . La concentration alimentaire dans la position du poisson est exprimé sous la forme $y = f(x)$ où y est la valeur de la fonction objective.
la distance entre le poisson artificiel est $d_{i,j} =\mid X_i - X_j \mid$, i et j est un poisson aléatoire. Step est l'etape maximum des poissons artificiels.δ est le degré de facteur de congestion.

supposons que X_v est la position visuelle a un moment.X_{next} est la nouvelle position.alors que la procedure de movement est definit comme cela :

$$X_v = X_i + Visual * rand(), i \in 0,] \ (1)$$

$$X_{next} = X + \frac{X_v - X}{\|X_v - X\|} * step * rand() \ (2)$$

Ou *rand()* produit des nombres aléatoire entre 0 et 1.les comportements de base du posson artificielle sont :

4.2.1 le comportement de nourriture (Prey behavior)

Ceci est un comportement biologique de base qui tend à la nourriture. Censé l'état de artificielle poisson est X_i, sélectionnez un étatX_j dans sa plage de détection au hasard. SiX_j est supérieure à X_j, alors déplacer à X_j ; au contraire, choisi de façon aléatoire l'étatX_j et déterminer si elle doit répondre à la condition de terminer les processus, répété plusieurs fois, si toujours les conditions des déplacer vers l'avant ne sont pas réalise , déménagement une étape au hasard.

$$X_j = x_i + Visual * rand()$$

si $Y_i < Y_j$, alors deplacer une etape vers cette direction.

$$X_i^{t+1} = X_i^t + \frac{X_j - X_i^t}{\|X_j - X_i^t\|} * step * rand()$$

4.2.2 le comportement Essaimer (Swarm Behavior)

Supposons que l'etat actuel du poisson artificielle est $X_i(d_{i,j} < Visual), n_f$ est le nombre des poisson artificielle,Si $n_f < \delta$ indique que les partenaires ont plus de nourriture et moins de monde, if Y_c est mieux queY_i, alors aller a l'avant vers le centre de la direction de la société, sinon retourner vers le comportement de nourriture .

$$X_i^{t+1} = X_i^t + \frac{X_c - X_i^t}{\|X_c - X_i^t\|} * step * rand()$$

4.2.3 le comportement de Suivre (Follow Behavior)

Censé l'état du poisson artificiel est X_i , explorer son état optimale X_{max} des voisins visuels, le nombre de partenaires de X_{max} est n_f , Si $n_f < \delta$ indique que près de la distance ont plus de nourriture et pas trop de monde, aller plus loin à l'avant de la position X_{max} ; autrement efectuer un processus de recherche de nouriture .

$$X_i^{t+1} = X_i^t + \frac{X_j - X_i^t}{\|X_j - X_i^t\|} * step * rand()$$

4.3 Processsus de AFSA proposé dans ce travail

le f_i represent le poisson$_i, etC_i$ represent le centre du $poisson_i$.le processus est le suivant :

(1) **Initialisation**. Encoder le problème d'optimisation à intégrer avec AFSA, créer la fonction fitness et les poissons initial au hasard, et comprennent la position et les paramètres.

Initialiser lessaim de poisson
tq (i=0 ; i < NUMdepoisson ; i++)
calculer FITNESS des poissons
DO étape suivre
SI (suivre échec) **ALORS**
DO étape essaimer
SI (essaimer échec) **ALORS**
DO étape nourriture
SI (suivre échec) **ALORS**
FIN
FIN
FINPOUR
FINTq
la sortie solution optimale

TABLE 4.1 – Pseudocode de AFSA.

(2) Évaluer la Fitness. Utilisez la fonction de remise en forme pour évaluer la condition physique de chaque poisson.

(3) Mouvement de poisson . Traiter les mouvements Suivre, essaimer et nouriture de tous les poissons et déterminer la solution optimale.

Suivre. À cette étape, les f_i sont comparé avec les poissons voisins sur la base de la valeur de fonction fitness ; si la forme optimale de son voisin est supérieure et le degré de monde de ce poisson ne dépasse pas le degré bondé maximal, alors f_i déplace vers la position du poisson voisin, ce qui indique que la fonction sous-ensemble de f_i est remplacé par celui du poisson voisin. Cela indique également que l'étape de suivi est terminé. Si l'étape de suivi échoue, alors mettre en œuvre essaimer ou Suivre le poisson suivant.

Essaimer. À cette étape, les f_i sont comparé par rapport a la valeur de remise en forme de leur propreC_i ; si la valeur de remise en forme des C_i est supérieure et le degré de monde de f_in'est pas plus grand que le degré maximal de monde, alors lef_i déplace vers le C_i ; cela indique que la fonctionnalité du sous-ensemble de f_i est remplacée par leC_i et ici l'étape essaimer est terminée. Si l'étape de essaimer échoue, mettre en œuvre l'étape nourriture pour le prochain poisson.

nourriture. À cette étape, le poisson na rien a predire que la nouriture se situe ent milieu de l'essaim alors qu'il va deplacer vers le milieu ce qui veut dire prendre les caracteristique du poisson du milieu de l'essaim,don la nouriture est terminée. Si l'étape nouriture échoue, l'algorithme répète cette étape jusqu'à ce que le nombre répété atteint le nombre maximal d'essai.

4.4 Les Parameters de AFSA.

4.4.1 Distance

La distance entref_i, f_j est obtenue par la formule (1). Ces deux poissons ont le même nombre de fonctionnalités k, et si la première caractéristique def_i est 0 et la première caractéristique de f_j0, alors la distance entre f_i, f_j restera la même. Mais si la première caractéristique def_i est différente de la première caractéristique de$f_j$la distance entre sera plus un. La distance entre deux poissons est la somme des différences de chaque fonction :

$$Distance(F_i, F_j) = \sum_{k=1}^{k} \|F_i(k) - F_j(k)\| \quad (1)$$

La visibilité d'un poisson et aussi la distance maximale que ce poisson peut se déplacer. En d'autres termes, il est le nombre maximum de caractéristiques que l'on peut choisir le poisson.

4.4.2 Le voisin

Le voisin de f_i est toutes poissons qui sont dans la vision de f_i ; si la distance entre $f_k$et f_i est supérieur à 0 et inférieur ou égal à la vision,f_k est voisin def_i. Elle est obtenue par la formule (2) :

$$Neighbor(F_i) = \{F_k | 0 < Distance(F_i, F_k) \leqslant vision\} \quad (2)$$

Le centre def_i est le centre du voisin de f_i. Il peut être considéré comme un poisson ; la fonction de centre est obtenu par la formule (3) ; si la fonction plus de la moitié des voisins def_i sont 0, alors le centre de f_i sera 0, et vice versa :

$$F_{center}(i) = \begin{cases} 0, & \sum_{k=1}^{k} F_k(i) < \frac{k}{2} \\ 1, & \sum_{k=1}^{k} F_k(i) \geq \frac{k}{2} \end{cases} \quad (3)$$

4.4.3 Le degré d'encombrement

Le degré d'encombrement des f_i est de representer la densite de position def_i.Elle est obtenue par la formule (4) :

$$CrowdedDegree(F_i) = \frac{NeighboursofF_i}{TotalnumberofFishes} \quad (4)$$

l'étape d'initiation de AFSA

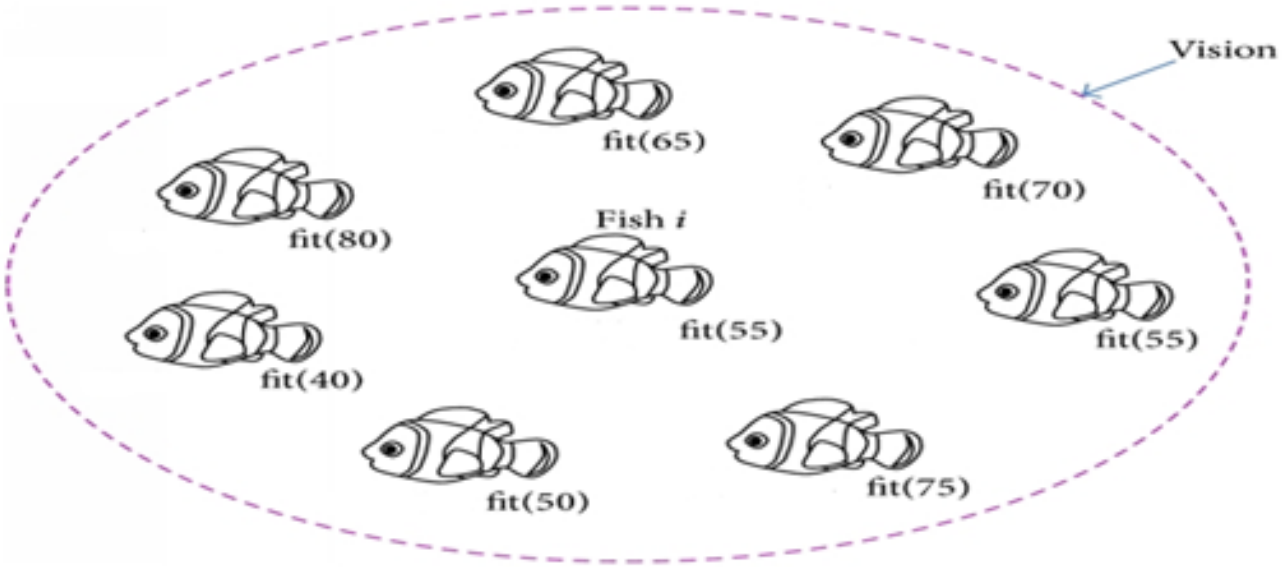


FIGURE 4.2 – l'étape d'initiation de AFSA.

l'étape de suivre de AFSA

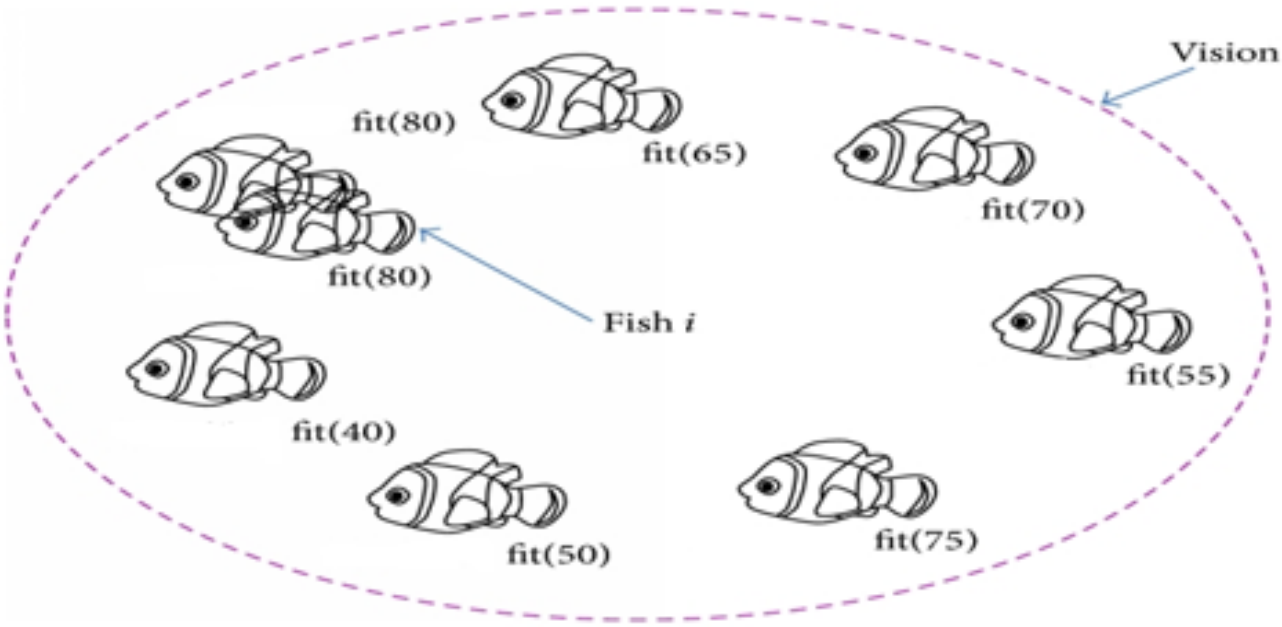


FIGURE 4.3 – l'étape de suivre de AFSA.

4.4.4 Organigramme de la méthode proposée

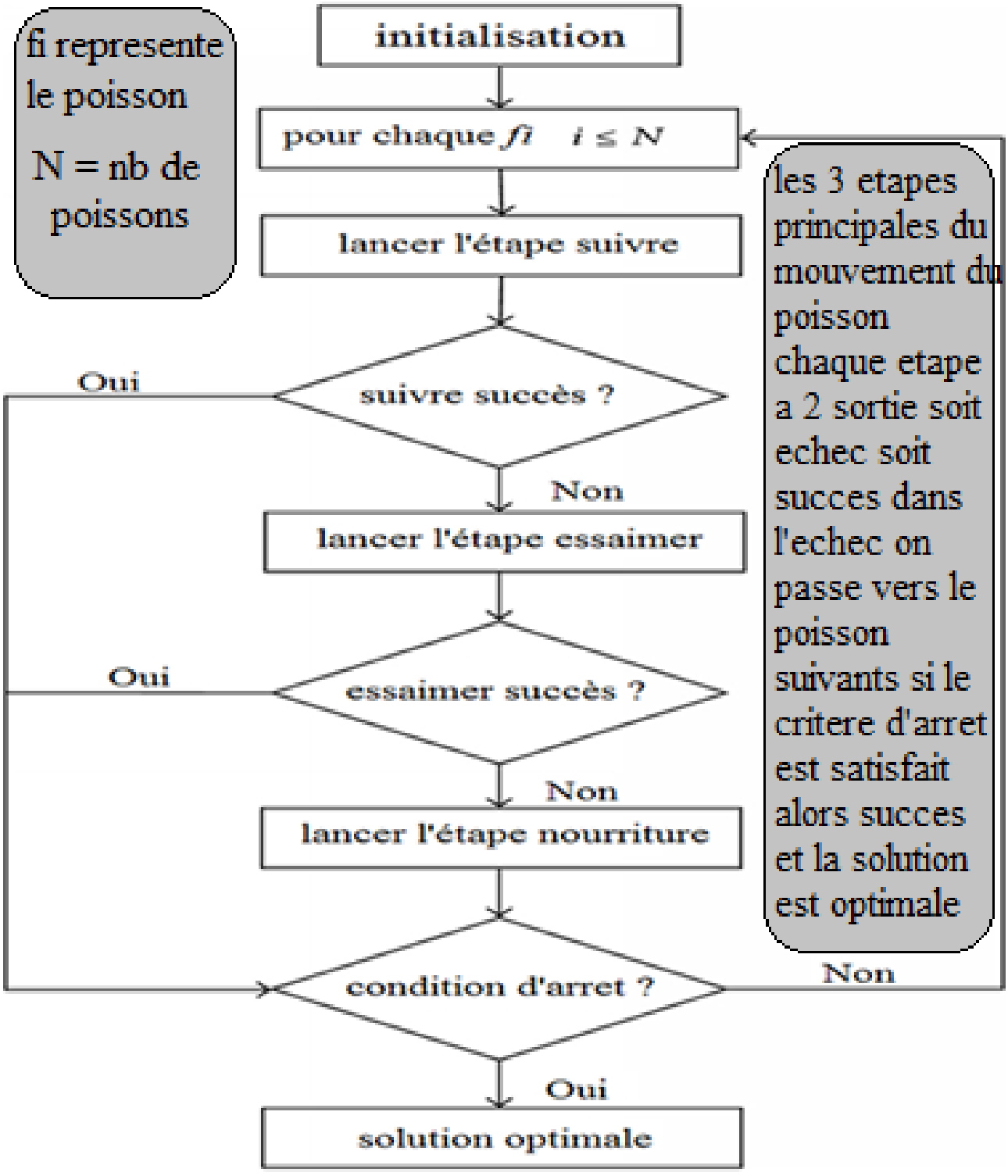


FIGURE 4.4 – Organigramme de la méthode proposée.

4.5 Informations sur le Dataset du Diabete utilise

Dans cette étude, nous utilisons le dataset UCI indian piima dataset.un ensemble de données introduites par Black CL **[Blake C. L., 1998]**. Cette base de données contient 768 échantillons, chaque échantillon a 8 caractéristiques qui sont huit résultats cliniques :

1. Nombre de fois enceinte .

2. la concentration plasmatique de glucose a 2 heures dans un test de tolérance au glucose oral .
3. Pression sanguine diastolique (mm Hg) .
4. épaisseur du pli de la peau du triceps (mm) .
5. l'insuline sérique à 2 heures (mu U/ml) .
6. Indice de masse corporelle BMI (kg/m^2) .
7. fonction de pedigree du diabète .
8. Age .

Ces caractéristiques sont détaillées dans le tableau suivant. Tous les patients de ce data set sont femmes indiennes Pima âgé au moins 21 ans et vivant près de Phœnix, Arizona, États-Unis. La variable cible binaire prend les valeurs '0' ou '1' . Alors que «1» signifie un test positif pour le diabète, '0' est un test négatif. Il y a 268 cas dans la classe «1» et 500 cas dans la classe «0» .

les caracteristiques	Moyenne	La deviation standard	Min/max
1	3.8	3.4	0/17
2	120.9	32.0	0/199
3	69.1	19.4	0/122
4	20.5	16.0	0/99
5	79.8	115.2	0/846
6	32.0	7.9	0/67.1
7	0.5	0.3	0.0 78/2.42
8	33.2	11.8	21/81

TABLE 4.2 – Brève analyse statistique du dataset des Pima Indienne de Diabetes

4.6 Le modèle biologique Vs Le modèle artificiel

Le modèle biologique	Le modèle artificiel
Ensemble de poissons	Ensemble des données
Choix d'un poisson au hasard	Choix d'un malade
seuil visuel globale	Les malades qui sont incluent dans le svg
Recherche de cible	Le choix d'une place ou bien d'une classe
Seuil visuel locale	Les malades qui sont incluent dans le svl
Nouvelle position: Etape essaimé Etape déplacé Etape nourriture	Générer une solution Etape essaimé Etape déplacé Etape nourriture
S'il Ya plusieurs poisson alors rester sinon déplacé vers un autre essaim	Solution

FIGURE 4.5 – Le modèle biologique Vs Le modèle artificiel

4.7 Le langage de programmation : Java

Le langage Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, avec le soutien de Bill Joy ,employés de Sun Microsystems, présenté officiellement le 23 mai 1995 au SunWorld.
Beaucoup d'applications et de sites Web ne fonctionnent pas si Java n'est pas installé et leur nombre ne cesse de croître chaque jour. Java est rapide, sécurisé et fiable. Des ordinateurs portables aux centres de données, des consoles de jeux aux superordinateurs scientifiques, des téléphones portables à Internet, la technologie Java est présente sur tous les fronts.

Java a donné naissance à un système d'exploitation (JavaOS), à des environnements de développement (eclipse/JDK), des machines virtuelles (MSJVM (en), JRE) applicatives multiplate-forme (JVM), une déclinaison pour les périphériques mobiles/embarqués (J2ME), une bibliothèque de conception d'interface graphique (AWT/Swing), des applications lourdes (Jude, Oracle SQL Worksheet, etc.), des technologies web (servlets, applets) et une déclinaison pour l'entreprise (J2EE). La portabilité du bytecode Java est assurée par la machine virtuelle Java, et éventuellement par des bibliothèques standard incluses dans un JRE. Cette machine virtuelle peut interpréter le bytecode ou le compiler à la volée en langage machine. La portabilité est dépendante de la qualité de portage des JVM sur chaque OS.

telechargement
vous pouvez télécharger Java gratuitement avec la dernière version sur le site officielle java

`https://www.java.com/fr/download/`
Si vous construisez un dispositif imbriqué ou grand public et que vous souhaitez y intégrer Java, contactez Oracle pour plus d'informations sur la façon d'inclure Java dans votre dispositif.
La dernière version de Java comprend d'importantes améliorations en matière de performances, de stabilité et de sécurité pour les applications Java exécutées sur votre ordinateur. L'installation de cette mise à jour gratuite garantit que les applications Java sont toujours exécutées de manière sécurisée et efficace.

FIGURE 4.6 – image du JAVA

4.7.1 L'editeur Netbeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL (Common Development and Distribution License) et GPLv2. En plus de Java, NetBeans permet également de supporter différents autres langages, comme C, C++, JavaScript, XML, Groovy, PHP et HTML de façon native ainsi que bien d'autres (comme Python ou Ruby) par l'ajout de greffons. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

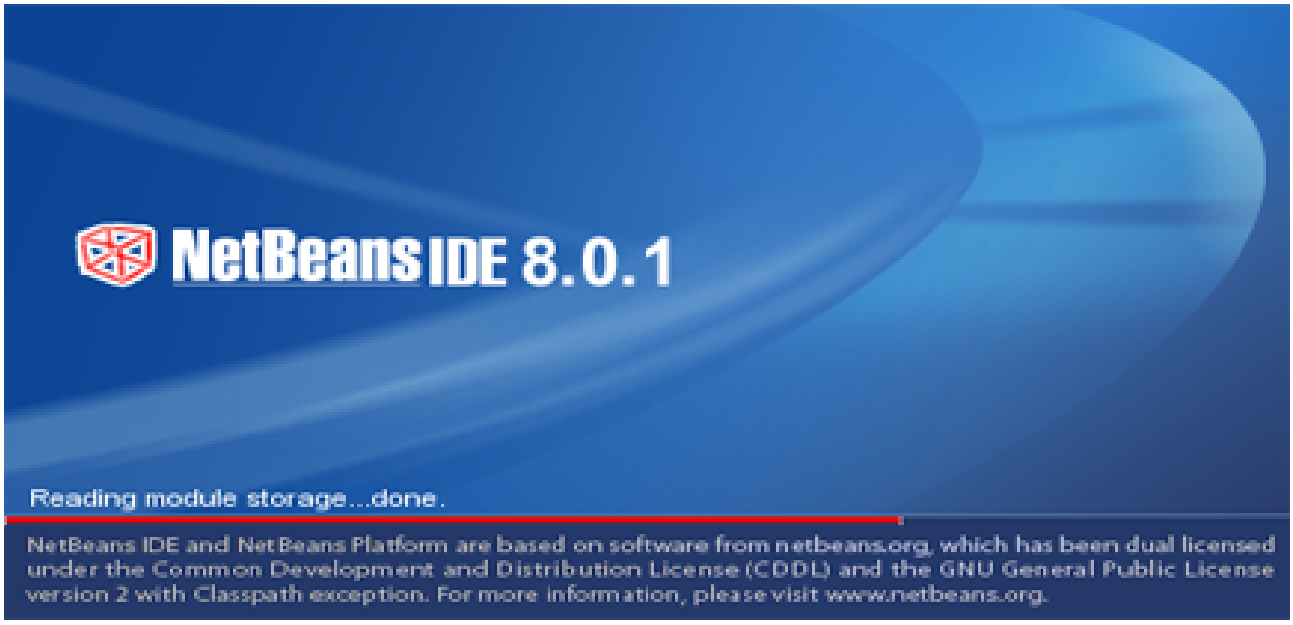


FIGURE 4.7 – image du netbeans IDE

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). Un environnement Java Development Kit JDK est requis pour les développements en Java.

NetBeans constitue par ailleurs une plate forme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plate forme.

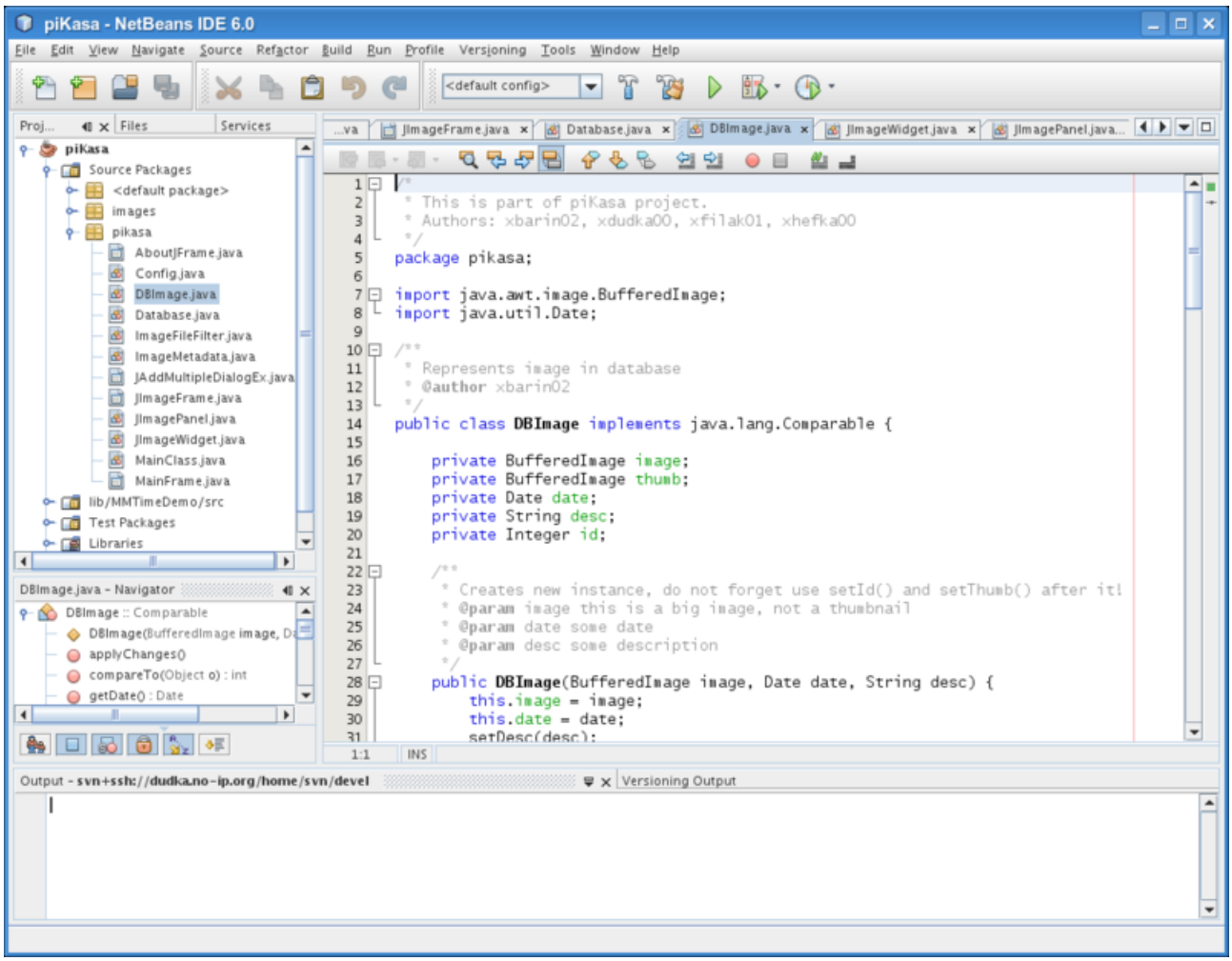


FIGURE 4.8 – image du netbeans IDE

4.8 Distance Utilisé Dans le Modèle

-Dans notre modèle on a utilisé une seule distance pour le calcule ,cette distance est la distance euclidienne :

4.8.1 Distance euclidienne

Dans un hyperespace, espace à n dimensions, la distance euclidienne entre les points $X(x_1, x_2..., x_n) et Y(y_1; y_2...; y_n)$ est :

$$\sqrt{\sum_i^n =_1, (x_i - y_i)^2}$$

4.9 les mesures de performance de classifieurs

Nous considérons ici un problème simple de classification pour lequel nous nous intéressons à une classe unique C et nous voulons évaluer un système qui nous indique si une instance peut être associé ou non à cette classe C.
- Ce problème est un problème de classification à deux classes (C et non C noté ¬ C). Si on peut maitriser ce problème simple(bi-classe) , on peut aussi maitriser les mesures de validation de plusieurs classe ou bien multi-classe.

4.9.1 Matrice de contingence(matrice de confusion)

Pour évaluer un système de classification , nous utilisons un corpus étiqueté (corpus d'apprentissage et même pour le test) pour lequel on connaît la vraie catégorie de chaque connexion ou instance, et le résultat obtenu par

le classifieur.Pour ce corpus, nous pouvons construire la matrice de contingence pour chaque classe, qui fournit 4 informations essentielles :

- **Vrai Positif (VP) :**est une personne qui est malade et qui présente un test positif.
- **Faux Positif (FP) :**est une personne qui n'est pas malade et qui présente un test positif.
- **Faux Négatif (FN) :**est une personne qui est malade et qui présente un test négatif.
- **Vrai Négatif (VN) :**est une personne qui n'est pas malade et qui présente un test négatif.

Catégorie C_i		Jugement d'expert	
		Normal	Anomaly
Jugement du classifieur	Normal	VP_i	FP_i
	Anomaly	FN_i	VN_i

FIGURE 4.9 – Matrice de contingence.

4.9.2 Précision et Rappel

- Certains principes d'évaluation sont utilisés de manière courante dans les différents domaine . Les performances en termes de classification sont généralement mesurées à partir de deux indicateurs traditionnellement utilisés c'est les mesures de rappel et précision. Initialement elles ont été conçues pour les systèmes de recherche d'information, mais par la suite la communauté de classification de textes les a adoptées.
- Formellement, pour chaque classeC_i, on calcule deux probabilités qui peuvent être estimées à partir de la matrice de contingence correspondante, ainsi ces deux mesures peuvent être définies de la manière suivante :
- **Le rappel** étant la proportion de documents correctement classés dans par le système par rapport à tous les documents de la classeC_i.

$$Rappel(C_i) = \frac{Nombrededocumentsbienclasssdans C_i}{Nombrededocumentsdelaclasse C_i}$$

$$R_i = \frac{VP_i}{VP_i + FN_i}$$

- Le rappel mesure la capacité d'un système de classification à détecter les documents correctement classés. Cependant, un système de classification qui considérerait tous les documents comme pertinents obtiendrait un rappel de 100%. Un rappel fort ou faible n'est pas suffisant pour évaluer les performances d'un système.Pour cela, on définit la **précision**.

- **La précision** est la proportion de documents correctement classés parmi ceux classés par le système dans C_i .

$$Precision(C_i) = \frac{Nombrededocumentsbienclasssdans C_i}{Nombrededocumentsclassdans C_i}$$

$$P_i = \frac{VP_i}{VP_i + FP_i}$$

- La précision mesure la capacité d'un système de classification à ne pas classer un document dans une classe, un document qui ne l'est pas. Comme elle peut aussi être interprétée par la probabilité conditionnelle qu'un document choisi aléatoirement dans la classe soit bien classé par le classifieur. Ces deux indicateurs pris l'un indépendamment de l'autre ne permettent d'évaluer qu'une facette du système de classification : la qualité ou la quantité.

4.9.3 Taux de succès et taux d'erreur

Le taux de succès ou l'exactitude Acc (Accuracy rate) est une mesure souvent utilisée par la communauté de l'apprentissage automatique. Le taux de succès désigne le pourcentage d'exemples bien classés par le classifieur, tandis que le taux d'erreur désigne le pourcentage d'exemples mal classés. Les deux taux sont estimés comme suit :

$$Acc = \frac{VP+VN}{VP+VN+FP+FN}$$
$$Err = \frac{VP+FN}{VP+VN+FP+FN}$$

4.9.4 Bruit et silence

Le Bruit (B) et le Silence (S) sont respectivement les notions complémentaires de la précision et du rappel.
- On utilise aussi la notion de bruit qui présente Le bruit est le pourcentage de textes incorrectement associés à une classe par le système , Le silence est le pourcentage de connexion à associer à une classe incorrectement non classés par le système .

$$Bruit = 1 - Prcision(P) = \frac{fP}{VP+FP}$$
$$Silence = 1 - Rappel(R)\frac{FN}{VP+FN}$$

4.9.5 Taux de chute et la spécificité

$$\text{Taux de chute}= \frac{fP}{FP+VN}$$
$$\text{spécificité}= \frac{VN}{FP+VN}$$

4.9.6 L'overlap et la généralité

$$overlap = \frac{VP_i}{VP_i+FP_i+FN_i}$$
$$\text{généralité}= \frac{VP}{VP+VN+FP+FN}$$

4.9.7 TP rate et FP rate

$$TPrate = \frac{VP_i}{VP_i+FP_i+FN_i}$$
$$FPrate = \frac{FP_i}{VP_i+FP_i}$$

4.9.8 F-mesure et entropie

- Observés conjointement, les indicateurs les plus célèbres à savoir le rappel et la précision, sont une estimation courante de la performance d'un système de classification.
- Cependant plusieurs mesures ont été développées afin de synthétiser cette double information. La F-mesure est la mesure de synthèse communément adoptée depuis les années 80 pour évaluer les algorithmes de classification de données textuelles à partir de la précision et du rappel.
- Elle est employée indifféremment pour la classification (Non supervisé) ou la catégorisation (Supervisé), pour la problématique de recherche d'information ou de classification. Elle permet donc, de combiner, selon un paramètre!, rappel et précision.la mesure est ainsi notée :

$$P_i = \frac{2*prcision*rappel}{prcision+rappel}$$

Entropie : c'est la perte d'information , elle se calcule par la formule :

$$E = -log(prcision)$$

4.10 Interface d'application

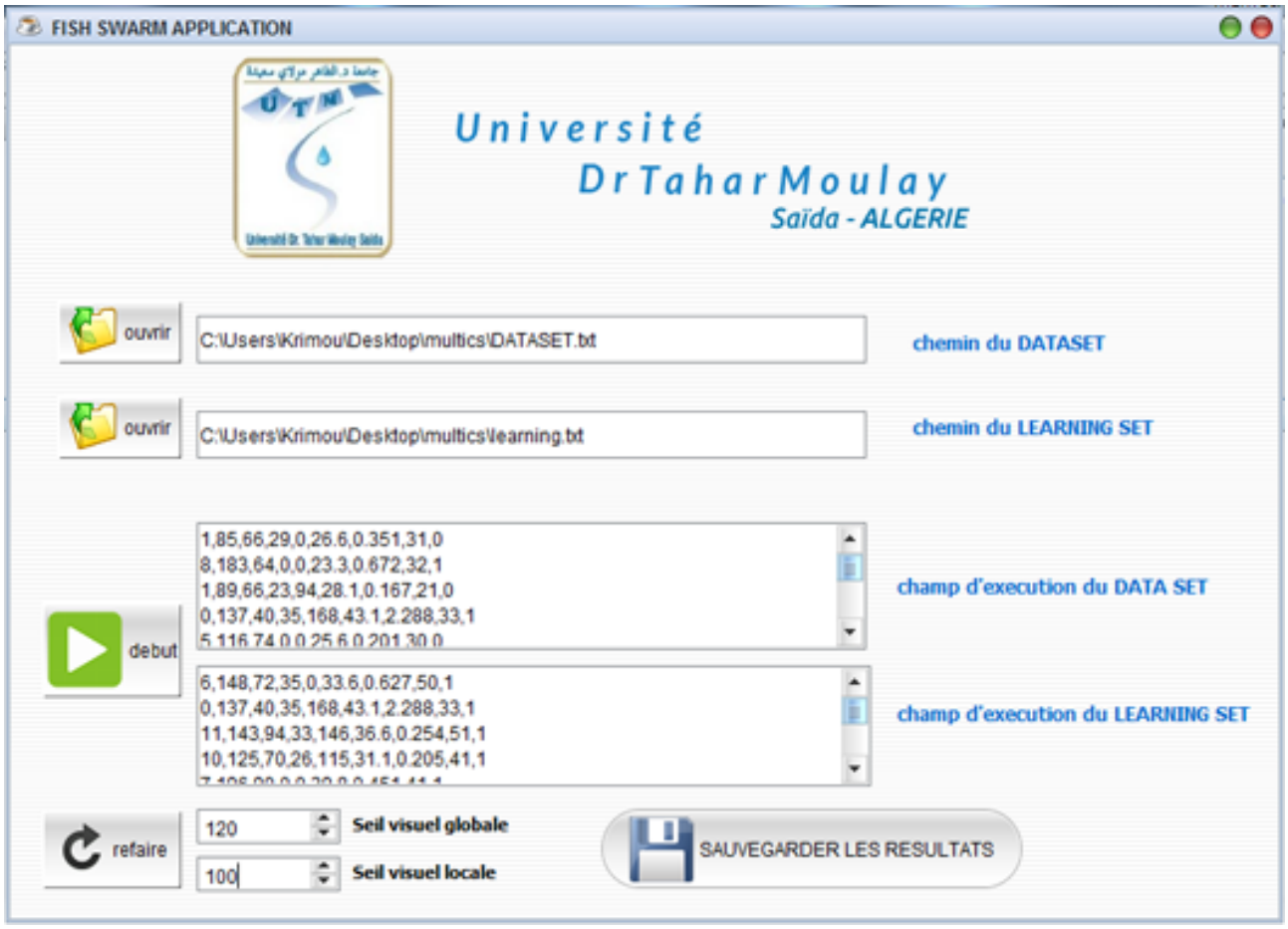


FIGURE 4.10 – Interface d'application.

4.10.1 Contenu de l'interface d'application

- 2 buttons de type Filechooser dans le haut une pour faire entrer le data set a classifier et lautre pour faire entrer le training set .
- 2 TextBox pour afficher les données du data set et training set .
- une button de Debut pour lancer le travail et une autre Refaire pour refaire le travail .
- une dernière button pour sauvegarder les résultats sous forme d'un fichier txt format textuel.

Partie Résultat de l'interface

FIGURE 4.11 – Interface d'application.

4.11 Les résultats

on va essayer plusieurs expérimentation pour obtenir des résultats satisfaisants . c'est ce qu'on va voir dans la section suivante .

4.11.1 Mesures de validation

1- Les résultats obtenue de l'incrémentation parallèle du seuil visuel locale et globale .

10-11 , 20-21 , 30-31 , 40-41 , 50-51 , 60-61 ,70-71 , 80-81 , 90-91 .

SVI +SVG / Mesures	5-6		10-11		21-20		31-30		41-40		51-50		61-60		71-70		81-80		91-90	
Rappel	0,247		0.388		0,611		0.712		0,739		0.755		0.758		0,759		0.76		0.762	
Précision	0,155		0.291		0,595		0.747		0,786		0.808		0,811		0.813		0.813		0.813	
F-mesure	0,19		0.332		0,603		0.728		0,762		0.781		0.784		0.785		0.786		0.787	
Entropie	1,86		0.234		0,518		0.295		0,239		0.212		0.209		0.206		0.206		0.206	
Bruit	0,844		0.708		0.404		0,255		0.213		0,191		0.188		0.186		0.186		0.186	
Silence	0,752		0.611		0.388		0,287		0.26		0.244		0.241		0,24		0.239		0.237	
Accuracy	0,143		0.241		0,491		0.638		0,68		0.705		0.709		0.71		0.712		0.713	
Overlap	0,105		0.199		0,432		0.578		0,615		0.64		0.644		0.646		0.647		0.648	
Temps d'exécution	3sec		1sec		1sec		1sec		1sec		1sec		1sec		1sec		1sec		1sec	
Matrice de contingence	70	380	131	319	268	181	335	115	354	96	364	86	365	85	365	85	366	84	366	84
	213	29	206	36	170	72	135	107	125	117	118	124	116	126	116	126	115	127	114	128

FIGURE 4.12 – les résultats de la 1 iere exprimentation.

Illustration graphique des résultats

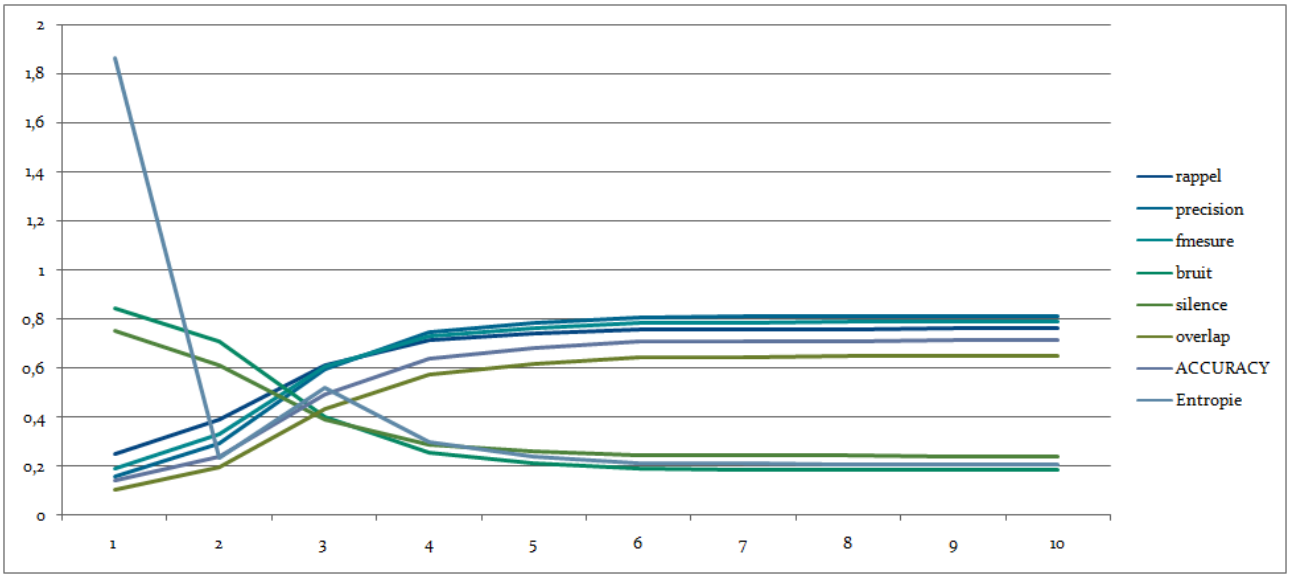


FIGURE 4.13 – Illustration graphique des résultats.

Graphes des Valeurs d'évaluation

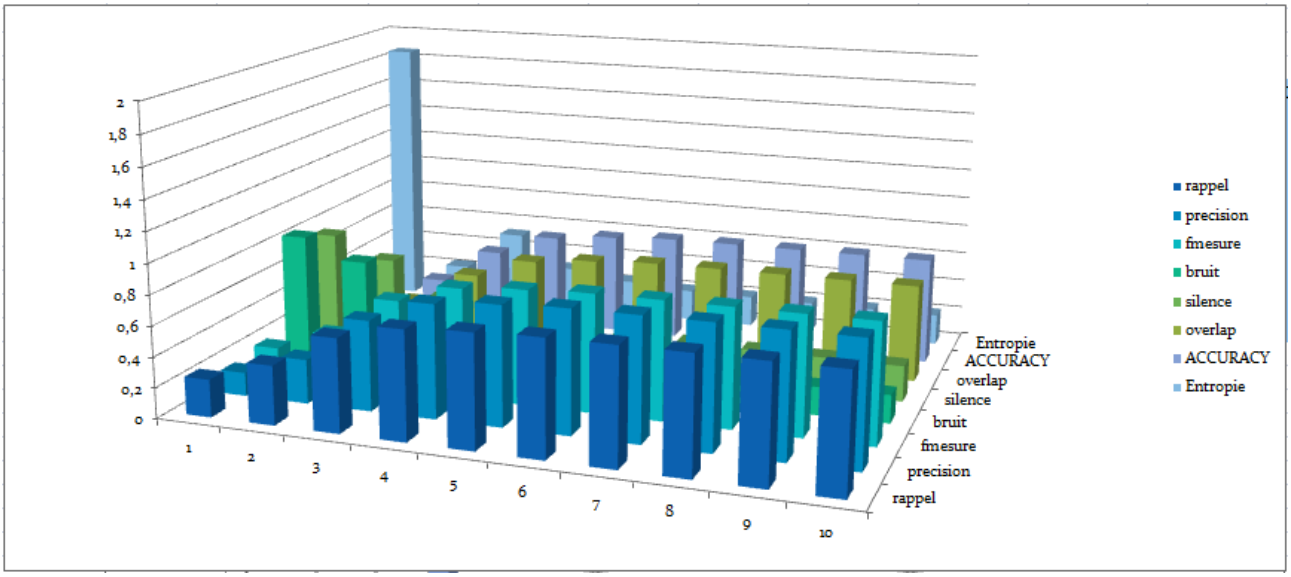


FIGURE 4.14 – Graphes des Valeurs d'évaluation.

Comme on a vu dans les trois figures précédentes , avec l'incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie que la precision s'augmente progressivement de (6-5) à (71-70) , et à partir de (71-70) elle commence a se stabiliser avec un pourcentage de 81.3% un peut prêt .

- Pour le rappel , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'il augmente progressivement de (6-5) à (71-70) , et à partir de (71-70) il commence a se stabiliser avec un pourcentage de 76.2% un peut prêt .

- Pour la f-mesure , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'elle augmente progressivement de (6-5) à (71-70) , et à partir de (71-70) elle commence a se stabiliser avec un pourcentage de 78.7% un peut prêt .

- Pour le taux de succès , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'il augmente progressivement de (6-5) à (71-70) , et à partir de (71-70) il commence a se stabiliser avec un pourcentage de 71.3% un peut prêt .

- Pour l'entropie , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'elle diminue progressivement de (6-5) à (21-20) , et à partir de (21-20) elle commence a s'augmente après dans (31-30) elle commence a se diminue a nouveau jus-qu'elle stabilise à partir de (61-60) avec un pourcentage de 20.6% un peut prêt donc la perte d'informations se diminue progressivement .

- Pour le bruit et le silence , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'elle se diminue progressivement de (6-5) à (71-70) , et à partir de (71-70) elle commence a se stabiliser avec un pourcentage de 18.6% et 23.7% un peut prêt .

NB : jusqu'à ici on n'a pas pu connaitre les meilleurs paramètres pour de bonne résultats , et qu'elle paramètre et le plus important , pour cela on va essayer d'autres expérimentations.

2- Les résultats obtenue de l'incrémentation du seuil visuel globale et avec un seuil visuel locale stable de 80 .

l'incémentation du seuil lobale sera comme suit : 81 , 91 , 101 , 111 , 121 .

SVl +SVG / Mesures	80-81		80-91		80-101		80-111		80-121		80-131	
Rappel	0.76		0.76		0.76		0.76		0.76		0.76	
Précision	0,813		0,813		0,813		0,813		0,813		0,813	
F-mesure	0,786		0,786		0,786		0,786		0,786		0,786	
Entropie	0,206		0,206		0,206		0,206		0,206		0,206	
Bruit	0,186		0,186		0,186		0,186		0,186		0,186	
Silence	0,239		0,239		0,239		0,239		0,239		0,239	
Accuracy	0,712		0,712		0,712		0,712		0,712		0,712	
Overlap	0,647		0,647		0,647		0,647		0,647		0,647	
Temps d'exécution	3sec		1sec		1sec		1sec		1sec		1sec	
Matrice de contingence	366	84	366	84	366	84	366	84	366	84	366	84
	114	128	114	128	114	128	114	128	114	128	114	128

FIGURE 4.15 – les résultats de la 2 ème expérimentation.

Illustration graphique des résultats

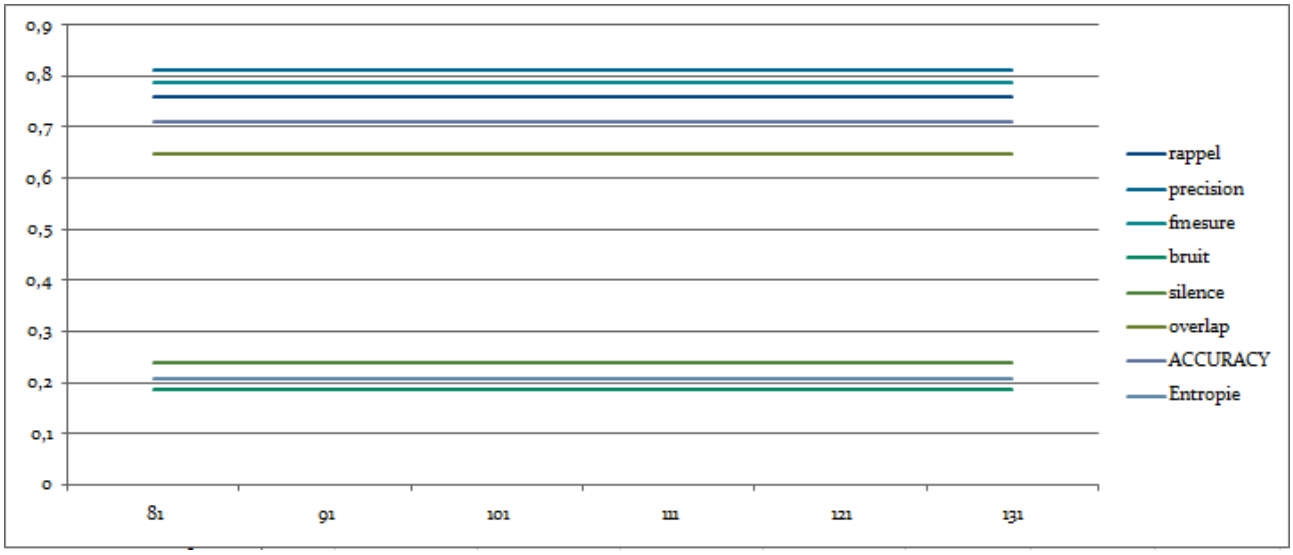


FIGURE 4.16 – Illustration graphique des résultats.

Graphes des Valeurs d'évaluation

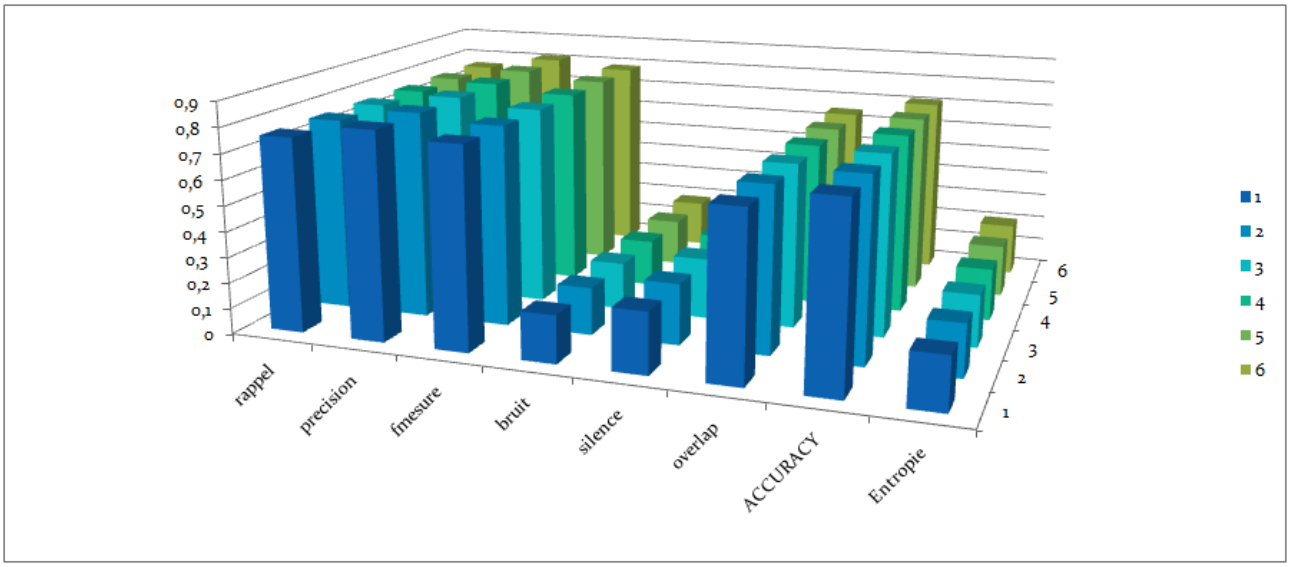


FIGURE 4.17 – Graphes des Valeurs d'évaluation.

Comme on a vu dans les trois figures précédentes , avec l'incrémentation a chaque fois le seuil visuel globale et un seuil visuel locale stable on voie que tous les mesures de validité reste stable avec le même pourcentage dans tous les essaye .

NB : cette étape nous a permet de connaitre que la l'icrementation seule du seuil globale ne donne pas des résultats et aussi qu'un seuil globale seul n'est important .

3- Les résultats obtenue de l'incrémentation du seuil visuel locale et un seuil globale stable .

seuil visuel globale de 120 .
et valeur de seuil globale : 5 , 10 , 20 , 30 , 40 , 50 ,60 , 65 , 70 , 75 , 80, 90 , 100 .

SVL / Mesures	5	10	20	30	40	50	60	65	70	75	80	100
Rappel	0.23	0.366	0.6	0.71	0.739	0.754	0.758	0.759	0.759	0.76	0.76	0.762
Precision	0.142	0.268	0.584	0.74	0.876	0.811	0.811	0.813	0.813	0.813	0.813	0.813
F-mesure	0.175	0.31	0.592	0.724	0.762	0.781	0.784	0.785	0.785	0.786	0.786	0.787
Entropie	1.75	1.31	0.537	0.301	0.239	0.209	0.209	0.206	0.206	0.206	0.206	0.206
Bruit	0.857	0.73	0.415	0.26	0.213	0.188	0.188	0.186	0.186	0.18	0.18	0.18
Silence	0.769	0.63	0.399	0.289	0.26	0.245	0.24	0.24	0.24	0.23	0.23	0.23
Accuracy	0.13	0.233	0.476	0.634	0.68	0.705	0.71	0.71	0.71	0.71	0.712	0.713
Overlap	0.09	0.18	0.42	0.568	0.615	0.541	0.644	0.646	0.646	0.647	0.647	0.65
Temps d'execution	3 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC	1 SEC
Matrice de contingence	64 386	121 329	263 187	333 117	354 96	365 85	366 84	366 84	366 84	366 84	366 84	366 84
	214 28	208 34	175 67	136 106	125 117	119 123	116 126	116 126	116 126	115 127	115 127	114 128

FIGURE 4.18 – les résultats de la 3 iéme exprimentation.

Illustration graphique des résultats

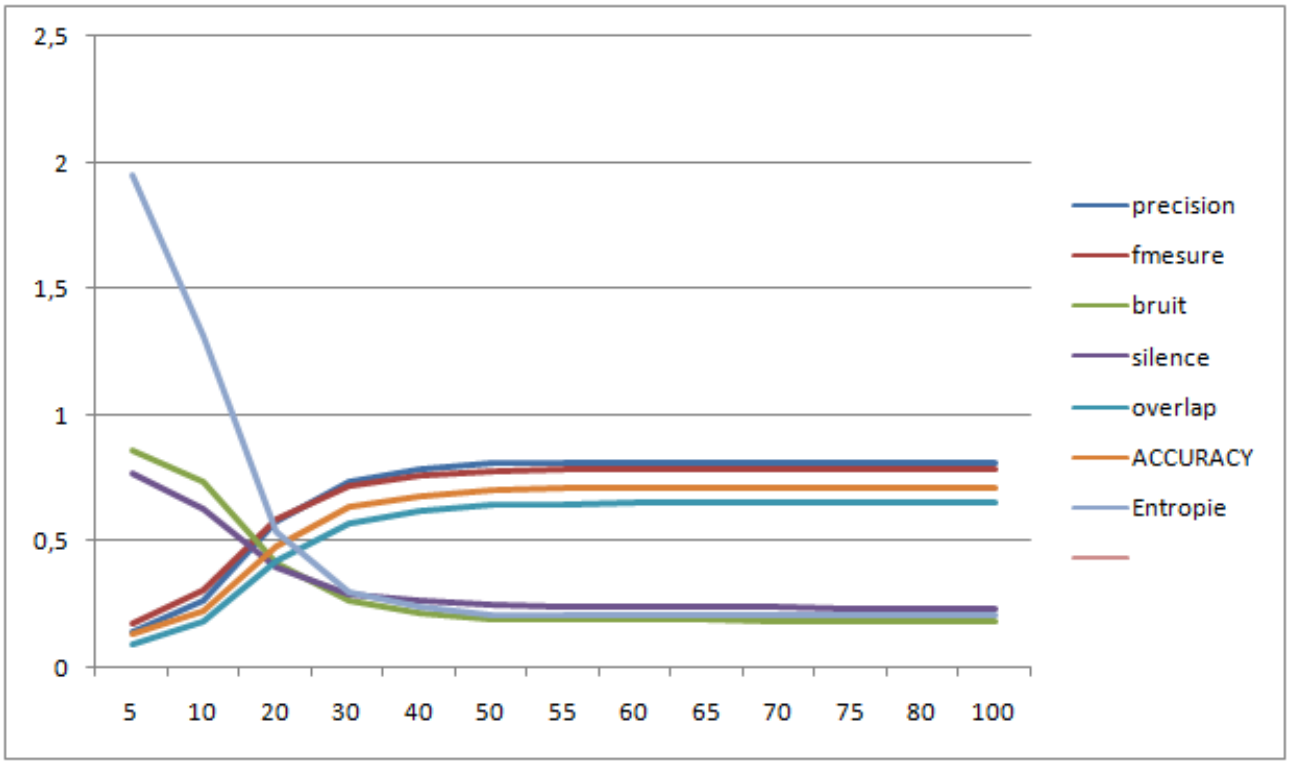


FIGURE 4.19 – Illustration graphique des résultats.

Graphes des Valeurs d'évaluation

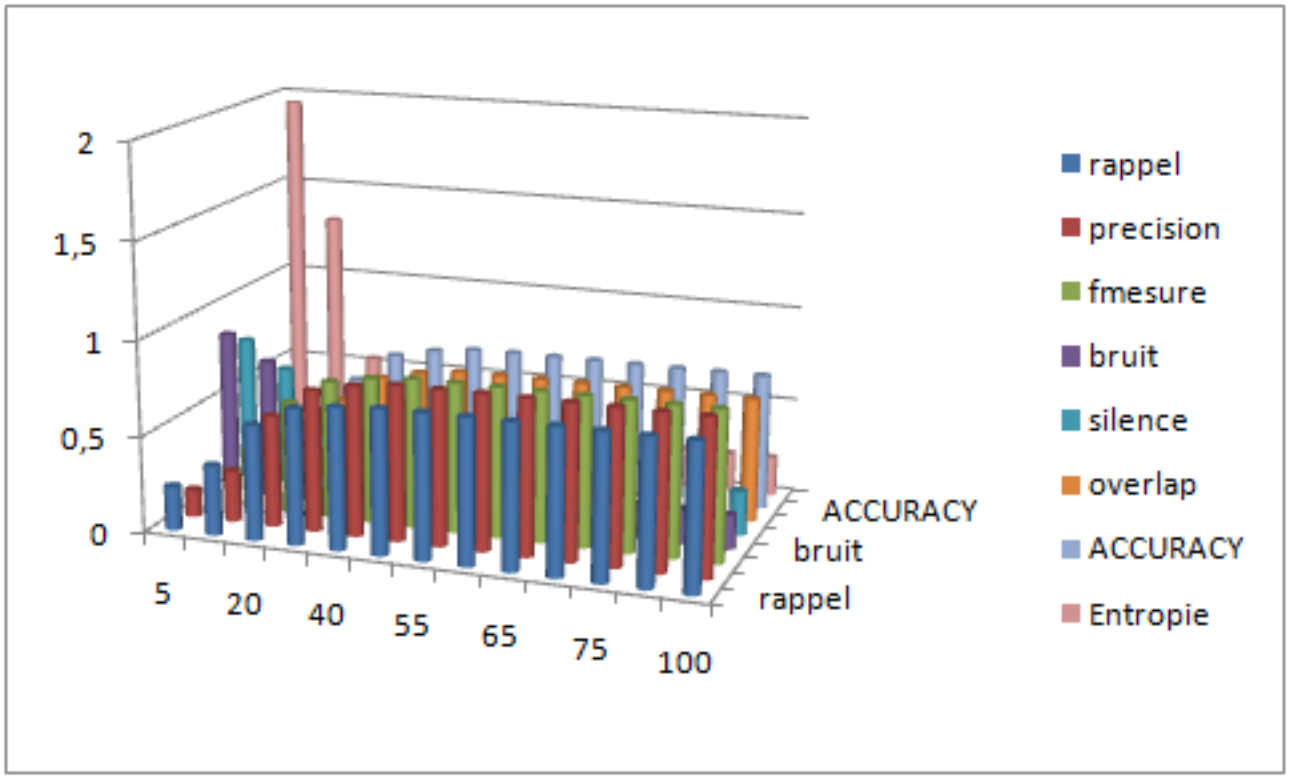


FIGURE 4.20 – Graphes des Valeurs d'évaluation.

Comme on a vu dans les trois figures précédentes , avec l'incrémentation a chaque fois le seuil visuel locale et un seuil visuel globale stable on voie que la precision s'augmente progressivement de (120-5) à (120-60) , et à partir de (120-70) elle commence a se stabiliser avec un pourcentage de 81.3% un peut prêt .

- Pour le rappel , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'il augmente progressivement de (120-5) à (120-60) , et à partir de (120-70) il commence a se stabiliser avec un pourcentage de 76.2% un peut prêt .

- Pour la f-mesure , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'elle augmente progressivement de (120-5) à (120-70) , et à partir de (120-70) elle commence a se stabiliser avec un pourcentage de 78.7% un peut prêt .

- Pour le taux de succès , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'il augmente progressivement de (120-5) à (120-50) , et à partir de (71-60) il commence a se stabiliser avec un pourcentage de 71.3% un peut prêt .

- Pour l'entropie , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'elle augmente progressivement de (120-5) à (120-60) , et à partir de (120-70) elle commence a se stabiliser avec un pourcentage de 20.6% un peut prêt donc la perte d'informations se diminue progressivement .

- Pour le bruit et le silence , avec incrémentation a chaque fois le seuil visuel locale et le seuil visuel globale on voie qu'elle se diminue progressivement de (120-5) à (120-70) , et à partir de (120-70) elle commence a se stabiliser avec un pourcentage de 18.6% et 23.7% un peut prêt .

NB : dans cette étape on a bien connait que l'utilisation des 2 paramètres seuil visuel globale et seuil visuel locale et obligatoire mais on aussi vue que la première étape est presque identique que la troisième ca veut dire que le seuil visuel locale est plus important que le seuil visuel globale .

pour validé nos résultats on va faire un dernier test d'incrementation parallèle avec des nombre un peut plus grand .

l'essaye avec les couples : 80-80 , 120-80 , 120-20 , 394-394 , 597-597 , 999-999 .

SVI +SVG / Mesures	80-80		120-80		120-120		394-394		597-597		999-999	
Rappel	0.76		0.76		0.762		0.762		0.762		0.762	
Précision	0,813		0,8133		0,8133		0,8133		0,8133		0,8133	
F-mesure	0,786		0,786		0,787		0,787		0,787		0,787	
Entropie	0,206		0,2066		0,2066		0,2066		0,2066		0,2066	
Bruit	0,186		0,186		0,1866		0,1866		0,1866		0,1866	
Silence	0,239		0,237		0,237		0,237		0,237		0,237	
Accuracy	0,712		0,712		0,713		0,713		0,713		0,713	
Overlap	0,647		0,647		0,648		0,648		0,648		0,648	
Temps d'exécution	3sec		1sec		1sec		1sec		1sec		1sec	
Matrice de contingence	366	84	366	84	366	84	366	84	366	84	366	84
	116	126	115	127	114	128	114	128	114	128	114	128

FIGURE 4.21 – les résultats de la dernière expérimentation.

Illustration graphique des résultats

FIGURE 4.22 – Illustration graphique des résultats.

Graphes des Valeurs d'évaluation

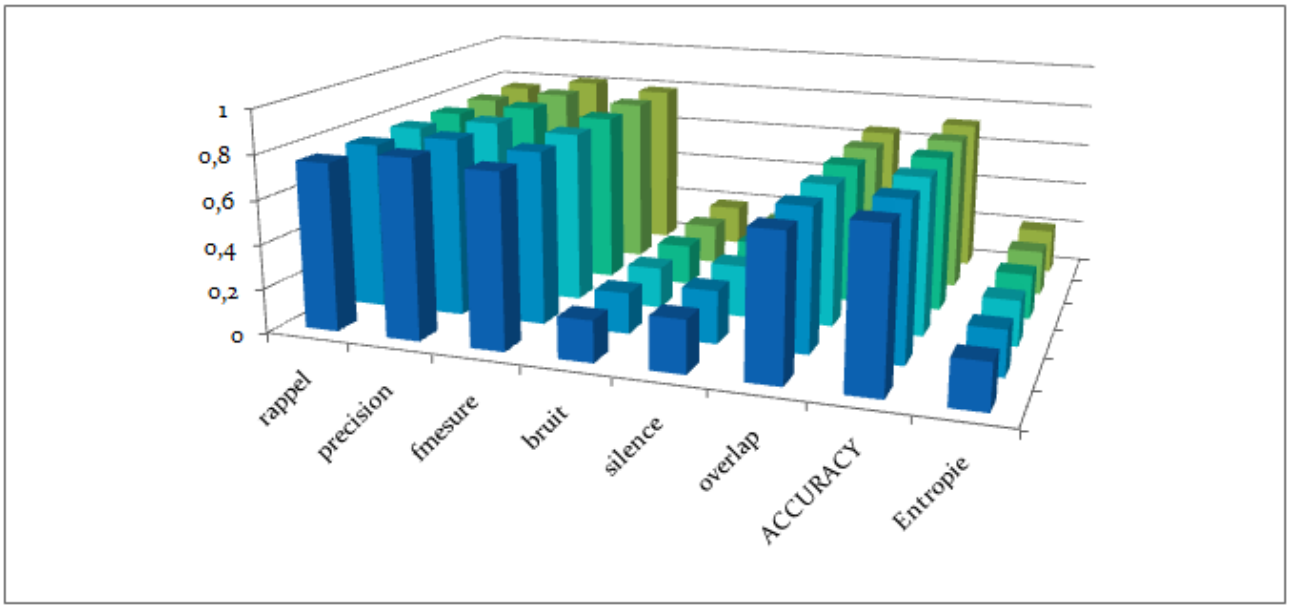


FIGURE 4.23 – Graphes des Valeurs d'évaluation.

comme l'indique les 3 figures ci dessous exactement les premier tableau par l'incérmentation des valeurs de seuil globale et le seuil locale les mesures de validation augmente peut a peut de (80-80) à (120-80) , et à partir de (120-120) elle commence a se stabiliser avec le même pourcentage dans tous les autres cas.

cela nous confirme la stabilité des mesures et aussi confirme nos resultats obtenue dans les 3 premier essaye .

4.11.2 Matrice de contingence

voici la matrice de contingences la plus optimale selon nos essayes .

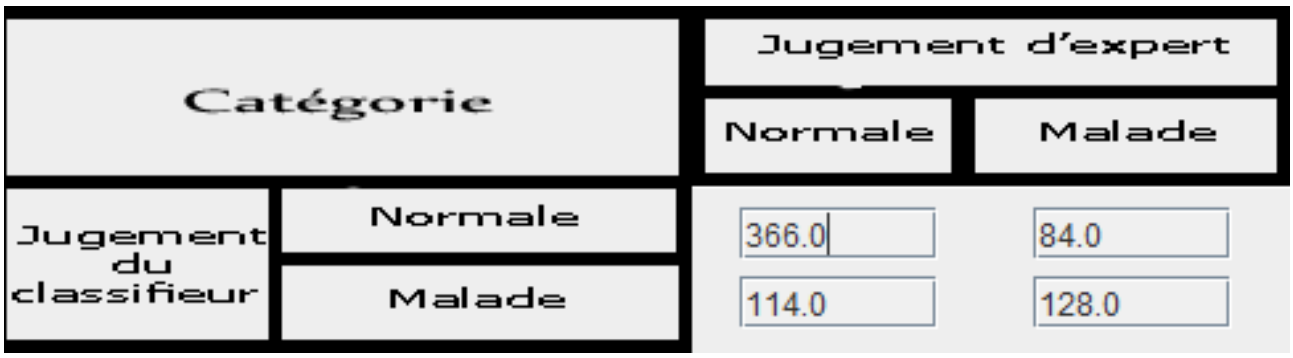


FIGURE 4.24 – Matrice de contingence.

4.12 Comparaison des résultats par rapport a autres algorithme

Résultats obtenus pour un échantillon de 50 non malade et 28 malade .

notre algorithme est presque bon par rapport a d'autres algorithme connue comme l'indique les 2 tableaux ci dessous .

Sortie désiré	Non-diabétiques	Diabétiques	Méthodes
Non-diabétiques	**42**	**8**	**AFSA (notre méthode)**
Diabétiques	**14**	**14**	
Non-diabétiques	44	6	LS-SVM (Polat et al., 2008)
Diabétiques	11	17	
Non-diabétiques	45	5	PCA-LS-SVM
Diabétiques	9	19	
Non-diabétiques	44	6	PCA-MI-LS-SVM
Diabétiques	4	24	
Non-diabétiques	45	5	PCA-PSO-LS-SVM
Diabétiques	4	24	
Non-diabétiques	48	2	MI-MCS-SVM
Diabétiques	3	25	

TABLE 4.3 – Comparaison des résultats par rapport a autres algorithme.

Autre comparaison selon le taux de succès (Accuracy) :

Auteurs	Méthodes	accuracy
Ster et al.	QDA	59.5
Zarndt	C4.5 rules	67
Yildirim et al.	RBF	68.23
HABBAZ.A (notre methode)	**AFSA**	**71**
Bennet et al.	C4.5 (5xCV)	72
Zarndt	Bayes	72.2
Statlog	Kohonen	72.8
Ster et al.	ASR	74.3
Shang et al.	DB-CART	74.4
Friedman	Naïve Bayes	74.5
Zarndt	CART DT	74.7
Statlog	BP	75.2
Ster et al.	SNB	75.4
Ster et al.	NB	75.5
Grudzinski	KNN	75.5
Zarndt	MML	75.5
Statlog	RBF	75.7
Ster et al.	LVQ	75.8
Friedman	Semi-Naïve Bayes (5xCV)	76
Ster et al.	MLP + BP	76.4
Ster et al.	FDA	76.5
Ster et al.	ASI	76.6
Statlog	SMART	76.8
Bennet et al.	GTO DT (5xCV)	76.8
Yildirim et al.	BFGS quasi Newton	77.08
Yildirim et al.	LM	77.08
Statlog et al.	LDA	77.5
Yildirim et al.	GD	77.6
Bennet et al.	SVM (5xCV)	77.6
Polat et al.	GDA-LS-SVM	79.16
Yildirim et al.	GRNN	80.21
Calisir et al.	LDA-MWSVM	89.74

TABLE 4.4 – une autre comparaison selon Accuracy.

4.13 problème majeur et limites

le problème majeur de cette algorithme est le seuil visuel globale parce qu'il est le seuil maximum a atteindre par les poissons .

4.14 Conclusion et perspectives

Nous avons présenté et détaillé dans ce chapitre l'approche AFSA .Cette approche vise à sélectionner le meilleur solutions à partir d'un ensemble de données volumineux qui contient des données redondantes, bruitées ou non pertinentes. Le but est d'améliorer une des tâches de la fouille de données qui est la classification supervisée. Avec les résultats expérimentaux, nous avons pu remarquer que l'approche proposée est compétitive comparant à d'autres métaheuristiques.

Ce travail présente un modèle automatique pour diagnostiquer la maladie du diabète basé sur ESSaiM DE POISON. Les résultats montrent que le modèle est satisfaisant . Les résultats démontrent que le modèle proposé est plus rapide et beaucoup plus fiables.on peut merger pluieurs methodec avec celle si pour atteindre a des resultats tres performants . Cette méthode peut être couplé avec des logiciels médicaux pour aider les médecins à prendre des décisions plus précises sur la maladie du diabète.

Conclusion Générale

Conclusion générale

Au cours de ce travail de master, nous avons vu un des apports le plus important des Métaheuristiques dans l'amélioration des tâches de la fouille de données (DataMining). Nous avons proposé une approche pour améliorer la tâche de la classification supervisée de la fouille de données. Ces dernières années le volume de données de toutes sortes croît de plus en plus. Avec tant de données disponibles, il est nécessaire de développer des algorithmes qui peuvent extraire des informations significatives à partir de ce vaste volume. La recherche des pépites utiles d'information parmi des quantités énormes de données est devenue connue comme le champ de la fouille de données. Ce champ interdisciplinaire tire ces techniques et tâches de plusieurs autres domaines. Une tâche en particulier est la classification supervisée.

L'étude de ce problème se justifie par le fait qu'une recherche exacte a un cout exponentiel en temps de calcul et en espace mémoire en effet cela fait partie des problèmes NP-difficiles et qui peuvent être résolus par les méta heuristiques.

Nous avons proposé dans ce mémoire de master la résolution d'un problème d'optimisation NP-difficile la detection du diabete par la metaheuristique de AFSA essaim de poissons pour l'amélioration de la tâche de la classification supervisée de la fouille de données en améliorant la précision et la compréhension du classifieur. Les expérimentations sont réalisées sur des bases de données de l'UCI (University of California, Irvine). Les résultats expérimentaux montrent que cette approche est compétitive.

bibliographie

RÉFÉRENCES

[1] AwadEl-bayoumy,Rashad Elsoud3,El-dosuky «Fast Artificial Fish Swarm Algorithm», 2013.

[2] Zhehuang Huang et Yidong Chen « An Improved Artificial Fish Swarm Algorithm based on Hybrid Behavior Selection », 2013.

[3]X. -x. Wei, H. -w. Zeng and Y. -q. Zhou, "Hybrid Artificial Fish School Algorithm for Solving Ill-conditioned Linear Systems of Equations", IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS), (2010).

[4] K. Zhu, M. Jiang et Y. Cheng, "Niche Artificial Fish Swarm Algorithm Based on Quantum Theory", IEEE 10th International Conference on Signal Processing (ICSP), (2010).

[5] K. Zhu and M. Jiang, "Quantum Artificial Fish Swarm Algorithm", IEEE 8th World Congress on Intelligent Control and Automation, Jinan, China, (2010) July 6-9.

[6]M. Jiang and Y. Cheng, "Simulated Annealing Artificial Fish Swarm Algorithm", IEEE 8th World Congress on Intelligent Control and Automation, Jinan, China, (2010) July 6-9.

[7] Davar Giveki, Hamid Salimi, GholamReza Bahmanyar, Younes Khademian, "Automatic Detection of Diabetes Diagnosis using Feature Weighted Support Vector Machines based on Mutual Information and Modified Cuckoo Search", China,2012.

[8] Reza Azizi, "Empirical Study of Artificial Fish Swarm Algorithm", iran,mars 2014.

[9] Mehdi Neshat , Ghodrat Sepidnam , Mehdi Sargolzaei , Adel Najaran Toosi, "Artificial fish swarm algorithm : a survey of the stateof-the-art, hybridization, combinatorial and indicative applications", iran,mars 2012.

[10] HAMIDI Yacine Nacer Eddine, "L'utilisation de l'approche boosting pour le diagnostic du diabéte.",1 juillet 2012.

[11] RIGHI, EL-AMIN, "Techniques de data mining pour la gestion de la relation client dans les banques",04 juin 2014 .

[12] R. Agrawal, T. Imielinski, et A. Swami, "Mining association rules between sets of items in large databases",In Proceedings of the ACM SIGMOD International Conference on Management of Data, 207-216, 1993 .

[13] T. Bayes, "An Essay towards solving a Problem in the Doctrine of Chances. Philosophical Transactions of the Royal Society of London",1763.

[14] G.H. John et P. Langley, "Estimating Continuous Distributions in Bayesian Classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence",San Mateo,1995.

[15] Z. Ye, J. Yang, D.Y. Geng, "Fuzzy rules to predict degree of malignancy in brain glioma. Med Biol Eng Comput",p. 145-52, 2002.

[16] C. Cortes et V. Vapnik, "vector network. Learning machine", 1995.

[17] Shakhnarovich, Gregory, T. Darrell et P. Indyk, "Nearest-Neighbor Methods in Learning and Vision",. The MIT Press, 262 p, 2006.

[18] L. Breiman, "Bagging predictors",Machine Learning, 24 :123–140, 1996.

[19] R.E. Schapire, "The strength of weak learning",Machine Learning,1990.

[20] S.J. Russell et P. Norvig, "Artificial Intelligence : A Modern Approach",2003.

[21] J. R. Quinlan, "Induction of decision trees. Machine Learning",1986.

[22] J. R. Quinlan, "C4.5 : Programs for Machine Learning",Morgan Kaufmann,San Diego, 1993.

[23] Mingers, J, "An empirical comparison of pruning methods for decision tree induction",1989.

[24] C. H. Papadimitriou et K. Steiglitz, "Combinatorial Optimization - Algorithms and Complexity",New York, 1982.

[25] C. H. Papadimitriou et K. Steiglitz, "The complexity of combinatorial optimization problems. PhD",New York, 1976.

[26] G. L. Nemhauser et A. L. Wolsey, "Integer and Combinatorial Optimization",New York, 1988.

[27] J. Dréo, "Adaptation de la méthode des colonies de fourmis pour l'optimisation en variables continues",2006.

[28] M. R. Garey et D. S. Johnson, "Computers and intractability ; a guide to the theory of

NPcompleteness",1979.

[29] I. H. Osman et G. Laporte, "Metaheuristics : A bibliography. Annals of Operations Research",1996.

[30] S. Verel, "Métaheuristiques pour l'Optimisation Difficile",ScoBi, Université de Nice Antipolis 2008, cours.

[31] C. Blum et A. Rol, "Metaheuristics in Combinatorial Optimization : Overview and Conceptual Comparison",2003.

[32] K.P. Syari et C.D Gelatt, "Optimization by Simulated Annealing",1983.

[33] T.A. FEO, "A probabilistic heuristic for a computationally difficult set covering problem",1989.

[34] T.A. FEO, "Greedy randomized adaptive search procedures". Journal of Global Optimization,1994.

[35] N. Mladenovic, et P. Hansen, "Variable Neighborhood Search",1997.

[36] J. Holland, et P. Hansen, "Outline for a logical theory of adaptive systems",1962.

[37] D. E. Goldberg, "Genetic Algorithms - in Search",1989.

[38] S. Baluja, "Population-based incremental learning : A method for integrating genetic search based function optimization and competitive learning",1994.

[39] H. Mühlenbein et G. Paass, "From recombination of genes to the estimation of distributions i",1996.

[40] P. Larrañaga et J.A. Lozano, "Estimation of Distribution Algorithms, a new tool for evolutionary computation",2001.

[41] J. Kennedy et R.C. Eberhart, "Particle Swarm Optimisation",1995.

[42] F. Glover. Eberhart, "Heuristics for integer programming using surrogate constraints",1977.

[43] F. Glover. Eberhart, "Heuristics for integer programming using surrogate constraints",1977.

[44] Alaoui Abdiya, "Application des techniques des métaheuristiques pour l'optimisation de la tâche de la classification de la fouille de données", 2012.

[45] ABDESSEMED MOHAMED RIDA, "proposistion f'une methode de classification dans un environement de robotique collective", iran,mars 07 juin 2006.

[46] SAIDI Meryem, "Traitement de données médicales par un Système Immunitaire Artificiel, Reconnaissance Automatique du Diabète", telemcen,28 juin 2011.

[47] Zeyneb KAMECHE, "Sélection des Web Services à Base Des Essaimes Particulaires", telemcen,27 septembre 2011.

[48] Karabatak, M., Ince, M.C., 2009. An expert system for detection of breast cancer based on association rules and neural network.

[49] MrSaeedFarzi, "Efficient Job Scheduling in Grid Computing with Modified Artificial Fish Swarm Algorithm", 2009.

[50] Yazdani, D., Toosi, A.N., Meybodi, "Fuzzy Adaptive Artificial Fish Swarm Algorithm, Adelaide (2010)

[51] Hu, J., Zeng. X., Xiao, J., "Artificial Fish SwarmAlgorithm for Function Optimization", (2010)