UNIVERSITE KASDI MERBAH OUARGLA

Faculté des Nouvelles Technologies de l'Information et de la communication

Département D'informatique et de technologie de l'information

Mémoire

MASTER ACADEMIQUE

Domaine :Mathématique et Informatique

Filière : Informatique

Spécialité : Informatique Industriel/Fondamentale

Présenté par :

GHANEM Kaoutar

MIADI May Sihem

Thème

Résolution des Problèmes de Classification Par Les Machines à Vecteurs Supports

Encadreur :Mme.MARREF NADIA

Année universitaire 2019/2020

Dédicaces Kaoutar

Je dédie ce modeste travail :

A mes très chers parents pour leur soutient et encouragement durant toutes mes années d'études et sans lesquels je n'aurais jamais réussi et à ma famille.

A Ma grand-mère et Mon grand-père mon modèle dans ma vie que Dieu ait pitié de lui.

À mes sœurs Meriem, Amina,Amani,Ouarda,Mariya et Manar

À mes amis proches :Maroua et Safia ,Asma

A tous ***mes amis****(es).*

À tous les professeurs et enseignants que j'ai eu durant tout mon cursus scolaire et qui m'ont permis de réussir Dans mes études.

À toute personne ayant contribué à ce travail de près ou de loin.

Dédicaces May sihem

Je dédie ce mémoire :

A mes très ***chers parents*** *pour leur soutient et encouragement durant toutes mes années d'études et sans lesquels je n'aurais jamais réussi et à* ***ma famille.***

A tous ***mes professeurs*** *et* ***enseignants*** *que j'ai eu durant tout mon cursus scolaire et qui m'ont permis de réussir dans mes études.*

A tous ***mes amis****(es).*

A toute personne ayant contribué à ce travail de près ou de loin.

Remerciements

Nos remerciements et nos profondes gratitudes vont à notre promotrice Mme MARREF ***Nadia*** *pour son encadrement, son suivi et ses conseils tout au long de cette période.*

Nos remerciements et notre gratitude vont aux ***Professeurs*** *et* ***enseignants*** *du département* ***d'informatique*** *ainsi que* ***ses étudiants*** *et son personnel côtoyés tout au long de notre cursus universitaire.*

Nous tenons aussi à remercier mesdames et messieurs les membres du ***jury*** *pour leur précieux temps accordé à l'étude de notre mémoire.*

Que toute personne ayant œuvré de près ou de loin à la réalisation de ce projet par une quelconque forme de contribution, trouve ici le témoignage de notre plus profonde reconnaissance.

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations :

AA:Apprentissage automatique
ACP, Analyse des composants principaux
AD:Arbres de décision
IA : Intelligence Artificielle
KNN:K-Nearest Neighbors
QPProbleme quadratique,
RBF:Radial Basis functions
SI:les systèmes informatiques
SMO:Algorithme d'optimisation minimale séquentielle
sv : vecteur support
SVM : Machines à Vecteurs de Supports

INTRODUCTION GENERALE

« Il n'y a pas d'intelligence sans apprentissage », affirme **Yann Le Cun**, patron de **l'Intelligence Artificielle (IA)** chez Facebook. C'est dans cette logique que s'inscrit **l'apprentissage automatique**, une technique d'Intelligence Artificielle inventée en 1959 par **Arthur Samuel** et qui est aujourd'hui en plein essor avec l'avènement du Big Data (Hadoop, MapReduce, etc.)**[1].**

La faculté d'apprendre est essentielle à l'être humain pour reconnaître une voix, une personne, un objet... On distingue en général deux types d'apprentissage : l'apprentissage «**par cœur**» qui consiste à mémoriser telles quelles des informations, et l'apprentissage **par généralisation** où l'on apprend à partir d'exemples un modèle qui nous permettra de reconnaître de nouveaux exemples. Pour **les systèmes informatiques(SI)**, il est facile de mémoriser un grand nombre de données (textes, images, vidéos...), mais difficile de généraliser.

La construction de machines capables d'apprendre automatiquement à partir des expériences accumulées, constituait depuis bien longtemps une préoccupation des chercheurs en intelligence artificielle et les techniques développées, dans ce sens, ne cessent de se poursuivre.

Le besoin à de telles techniques se justifie par le fait que plusieurs tâches et problèmes ne peuvent être résolus par les techniques classiques de programmation à cause de la difficulté. Les techniques **d'apprentissage automatique(AA)** sont utilisées dans de très nombreux domaines.

Dans notre travail, nous nous intéressons en particulier à une **méthode de classification** supervisée nommée les **Machines à Vecteurs Supports (SVM)** .Le succès de cette méthode est justifié par les solides bases théoriques qui la soutiennent. Il existe en effet un lien direct entre la théorie de l'apprentissage statistique et l'algorithme d'apprentissage du **SVM**.

C'est dans ce contexte que s'est développée, dans les deux dernières décennies, la méthode des machines à vecteurs supports appelée aussi machines à vaste marge (**SVM**). Cette méthode dérive de **l'apprentissage statistique**, un fondement théorique solide contrairement à son ancêtre les réseaux de neurones. Durant les années 90, la méthode **SVM** a connu un

progrès très intéressant dans son principe, son implémentation et son extension aux problèmes **multi classes**. Durant la dernière décennie, les recherches ont été concentrées sur l'adaptation de la méthode à des problèmes particuliers tel que la détection des nouveautés et le clustering ainsi que sur l'amélioration de ses performances (optimisation et parallélisassions) et son application dans différents domaines tels que l'imagerie, le son, les banques, la biologie, ... etc.

De ce fait, plusieurs applications ont connu un réel succès avec l'utilisation des **SVMs.** On peut citer la reconnaissance de l'écriture manuscrite, la reconnaissance des visages, le diagnostic des maladies, la planification financière, ...etc.

L'entrainement d'un **SVM** consiste à résoudre le problème d'optimisation quadratique convexe. Des techniques standards de programmation quadratiques telle que la méthode du gradient conjugué ou la méthode des points intérieurs, … peuvent résoudre le problème du **SVM** mais pour les problèmes de grandes taille, ces méthodes deviennent inenvisageables. Il ya cependant d'autres techniques dédiées aux **SVM** qui servent à décomposer le problème d'optimisation en dessous problèmes de petites tailles (Algorithme de Joachims et Algorithme de John Platt(**SMO**)) ,et nous avons choisi d'implémenter l'algorithme **SMO**.

Notre étude est articulée autour de trois chapitres principaux.

Le **premier chapitre** est consacré à la présentation générale de L'apprentissage automatique et **les méthodes de classification**. En premier, nous étudierons L'apprentissage automatique d'une façon générale : les nombreux domaines que sont utilisés Les **techniques d'apprentissage** automatique comme (secteur de la santé, Services financiers, Jeux : appliqué au jeu d'échec…). on parle sur Les différents **types d'apprentissage** automatique (Apprentissage supervisé, non supervisé, semi-supervisé et l'apprentissage par renforcement), Nous décrivons aussi Les **problèmes auxquels s'applique l'apprentissage automatique** (La classification, La régression , Le regroupement)et la différence entre eux , ainsi que les approches algorithmiques courantes en apprentissage automatique, notamment l'algorithme k-voisin le plus proche, l'apprentissage par arbre de décision K-Nearest Neighbors (**KNN**) et Machines à vecteurs supports (**SVM**) .

Le **deuxième chapitre** est composé de deux volets. Le premier concerne **SVM à deux classes** il est composé en deux parties : Dans la première partie nous présentons Le but de l'apprentissage statistique qui est représenté en apprendrons une fonction qui correspond aux exemples vus et qui prédit les sorties pour les entrées qui n'ont pas encore été vues. Dans le

cas des machines à vecteurs supports, la fonction recherchée est de forme linéaire. Nous avons abordé les **SVM** à marge dure et **SVM** à marge souple ou dans ce dernier les données sont affectées par un bruit(C). Nous discutons dans les deux cas le meilleur hyperplan, la Maximisation de la distance entre l'hyperplan et la donnée la plus proche (la Marge) et la formulation du problème dual et primal. La deuxième partie concerne le problème de l'absence de séparateur linéaire (**SVM non-linéaires**), en utilisant l'idée de l'astuce du noyau (Noyau linéaire, Noyau polynomial, Noyau RBF), et le deuxième volet concerne les **SVM** à **plusieurs classes**, On trouve dans la littérature plusieurs méthodes de décomposition : Méthode Une-contre-reste (1vsR), Méthode Une contre Une (1vs1)...

Le troisième chapitre décrit en détail l'algorithme d'optimisation basé sur les **SVM** qui est l'algorithme **SMO** que nous avons choisi dans notre Application, et en utilisant une librairie pour Python spécialisée dans l'apprentissage automatique (module **SVC**) dans le but de faire une comparaison avec notre implémentation de l'algorithme le **SMO** sur deux types de données (artificielles et réelles).

Enfin, une conclusion générale et des perspectives de travail viennent clôturer ce mémoire.

I. CHAPITRE I : L'apprentissage automatique et les méthodes de Classification

I.1. Introduction

Au cours des deux dernières décennies, l'apprentissage automatique est devenu un élément majeur des technologies de l'information et de l'analyse des données. Avec cette croissance, il est devenu une partie cachée régulière de nos vies. Avec l'augmentation des quantités de données disponibles, il y a de bonnes raisons de croire que l'analyse intelligente des données deviendra de plus en plus courante en tant que composante essentielle du progrès technologique.

La construction de machines capables d'apprendre automatiquement à partir des expériences accumulées, constituait depuis bien longtemps une préoccupation des chercheurs en intelligence artificielle et les techniques développées, dans ce sens, ne cessent de se poursuivre. Le besoin à de telles techniques se justifie par le fait que plusieurs tâches et problèmes ne peuvent être résolus par les techniques classiques de programmation à cause de la difficulté, voire l'impossibilité, d'établir des modèles mathématiques qui peuvent les décrire. Les exemples sont flagrants : les informaticiens ne peuvent jusqu'à maintenant écrire un programme capable de reconnaître un texte manuscrit à cause de l'absence d'un modèle mathématique pouvant modéliser ses caractères, et sur le même raisonnement reconnaître un gène dans une séquence d'ADN, reconnaître les objets dans une image, ...etc.

Aujourd'hui, L'apprentissage automatique est omniprésent. Lorsque nous interagissons avec les banques, magasinons en ligne ou utilisons les médias sociaux, les algorithmes d'apprentissage automatique entrent en jeu pour rendre notre expérience fluide, efficace et sécuritaire. L'apprentissage automatique et les technologies afférentes se développent rapidement, et nous commençons à peine à entrevoir leurs capacités.

I.2. L'apprentissage automatique

Au cours des deux dernières décennies, Apprentissage automatique est devenu l'un des principaux piliers des technologies de l'information et de l'analyse des données. Aujourd'hui, nous utilisons l'apprentissage automatique dans tous les domaines, lorsque nous interagissons avec les banques, achetons en ligne ou utilisons les médias sociaux. Des algorithmes

d'apprentissage automatique sont en place pour améliorer, rationaliser et sécuriser notre expérience.

I.2.1. Définition

Apprentissage automatique (en anglais : machine Learning) : est un sous-domaine de l'intelligence artificielle (IA) est l'étude scientifique des algorithmes et des modèles statistiques. Les systèmes informatiques apprentissage automatique sont utilisés pour effectuer une tâche spécifique efficacement sans utiliser d'instructions claires, en s'appuyant sur Types et raisonnement uniquement En général, l'objectif de l'apprentissage automatique est de comprendre la structure des données et de les intégrer dans des modèles que tout le monde peut comprendre et utiliser **[2].**

Bien que l'apprentissage automatique soit un domaine de l'informatique, il diffère des méthodes informatiques traditionnelles. En fait, les algorithmes d'apprentissage automatique préparent un modèle mathématique basé sur des données de modèle connues sous le nom de "données d'apprentissage" afin de faire des prédictions ou des décisions sans être explicitement programmées pour faire le travail.

I.2.2. Domaines d'application des systèmes d'apprentissage automatique

Les techniques d'apprentissage automatique sont utilisées dans de très nombreux domaines et même dans les domaines créatifs comme la peinture ou le cinéma. Parmi ces domaines figurent les suivants :

- **Du secteur de la santé :**

L'apprentissage automatique est de plus en plus utilisé dans le secteur de la santé, notamment grâce au développement d'objets connectés et d'autres capteurs qui permettent d'utiliser les données pour accéder aux données de santé des patients en temps réel. Cette technologie peut également aider les experts médicaux à analyser les données pour identifier les tendances inquiétantes afin d'améliorer le diagnostic et les traitements, par exemple, les machines nous aident à diagnostiquer le cancer**[3].**

- **Services financiers :**

Les banques et autres entreprises de l'industrie de la finance utilisent L'apprentissage automatique pour découvrir des informations importantes au sein des données, et pour empêcher la fraude. Les insights permettent d'identifier les opportunités d'investissement,

tandis que l'exploration de données permet d'identifier les clients à haut risque. La cyber surveillance permet quant à elle de repérer les signes de fraude **[4].**

- **Jeux :**

Logiciels de jeux a été une grande motivation pour les scientifiques. Aux échecs, aux dames ou au Go chinois, les systèmes d'apprentissage automatique se mesuraient contre des adversaires humains. Les développeurs de jeux informatiques utilisent également L'apprentissage automatique pour rendre leurs jeux plus intéressants. Les concepteurs de jeux peuvent utiliser L'apprentissage automatique pour créer le Game Play le plus équilibré possible et s'assurer que les adversaires de l'ordinateur s'adaptent de manière intelligente aux comportements des joueurs humains. **[20].**

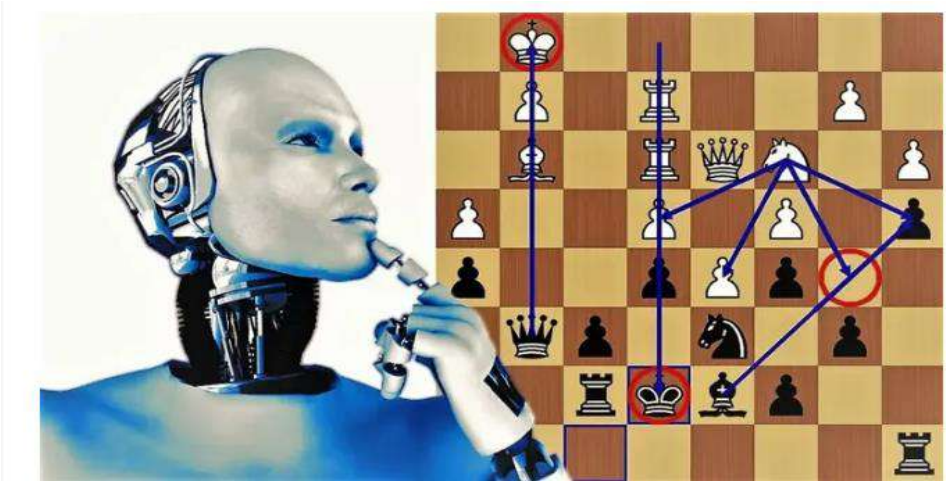


*Figure I 1 : l'apprentissage automatique par renforcement appliqué au jeu d'échec**[1]***

- **Robotique :**

Les robots sont maintenant omniprésents, surtout dans les usines. Ils aident, par exemple, dans la production de masse à automatiser des étapes de travail cohérentes. Si des systèmes d'apprentissage automatique sont utilisés en robotique, ces machines doivent également maîtriser de nouvelles tâches. Bien entendu, ces développements sont également très intéressants pour d'autres domaines, des robots dotés d'intelligence artificielle seront utilisés dans des domaines très variés **[20].**

I.2.3. Les différents types d'apprentissage automatique

Les algorithmes jouent un rôle fondamental dans l'apprentissage automatique , d'une part ils sont responsables de la reconnaissance des formes et, d'autre part, ils peuvent trouver des solutions. Les algorithmes peuvent être divisés en différentes catégories.

1. Apprentissage supervisé

Un algorithme supervisé nécessite un expert qui lui fournit un échantillon de formation avant utilisation. Les algorithmes supervisés sont formés à l'aide d'exemples étiquetés, ce qui signifie que les entrées et les sorties sont connues dans la première étape. Le système crée généralement un modèle qui correspond à la distribution des données utilisées dans l'apprentissage. Ce modèle peut facilement être utilisé pour prédire de nouveaux échantillons inconnus. L'apprentissage supervisé est généralement effectué dans le contexte de la classification et de la régression**[6].**

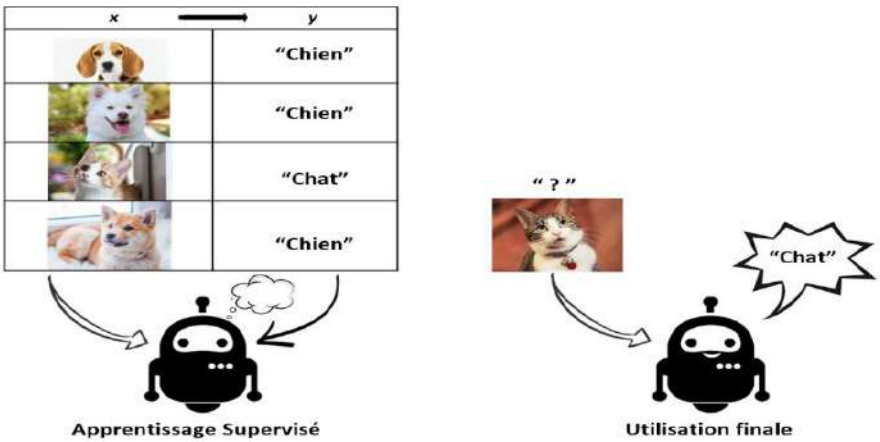


*Figure I 2: La machine apprend à partir de milliers d'exemples x, y**[6]***

2. Apprentissage non supervisé

Dans l'apprentissage non supervisé, les données sont non étiquetées, de sorte que l'algorithme d'apprentissage trouve tout seul des points communs parmi ses données d'entrée. Les données non étiquetées étant plus abondantes que les données étiquetées, les méthodes d'apprentissage automatique qui facilitent l'apprentissage non supervisé sont particulièrement utiles. L'objectif de l'apprentissage non supervisé peut être aussi simple que de découvrir des modèles cachés dans un ensemble de données, mais il peut aussi avoir un objectif

d'apprentissage des caractéristiques, qui permet à la machine intelligente de découvrir automatiquement les représentations nécessaires pour classer les données brutes **[7].**

*Figure I 3: L'apprentissage non supervisé **[7]***

3. Apprentissage semi-supervisé

L'apprentissage semi-supervisé qui prend en entrée certaines données annotées et d'autres non. Ce sont des méthodes très intéressantes qui tirent parti des deux mondes (supervisé et non supervisé), mais bien sûr réparti leur lot de difficultés **[7].**

4. Apprentissage par renforcement

Un algorithme d'apprentissage par renforcement, ou agent, apprend en interagissant avec son environnement. L'agent reçoit des récompenses en effectuant correctement et des pénalités pour mauvaise exécution. L'agent apprend sans intervention d'un humain en maximisant sa récompense et en minimisant sa pénalité**[7].**

I.2.4. Les problèmes auxquels s'applique l'apprentissage automatique

1. La classification

Un modèle de classification est un modèle de(AA) dont les sorties y ont été à un ensemble fini de valeurs (exemple : bon, moyen, mauvais) L'objectif de la classification supervisée est principalement de définir les règles permettant de classer les objets dans des classes à partir des variables qualitatives ou quantitatives caractérisant ces objets**[7].**

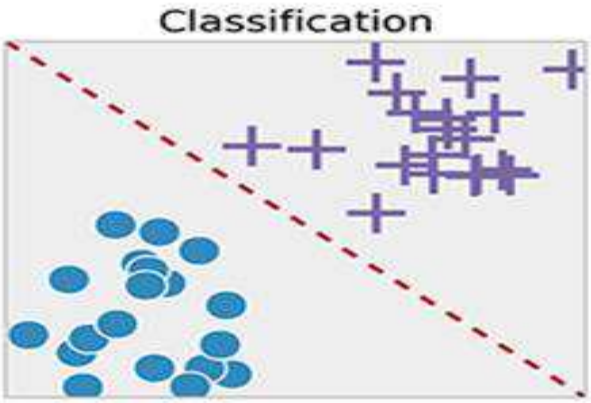


Figure I 4:La classification ***[7]***

2. La régression

Un modèle de régression est un modèle de (AA) dont les sorties y sont des nombres (exemple : la température de demain)**[7].**

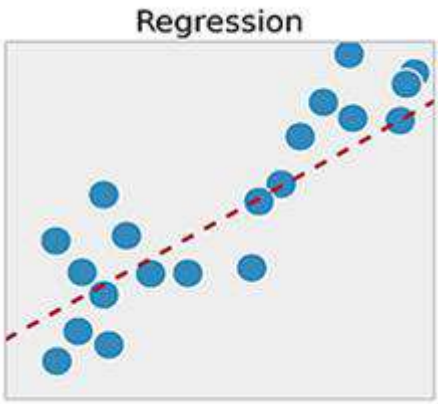


Figure I 5:Exemple de régression ***[7]***

3. Le regroupement (clustering)

Également connu sous le nom d'analyse de cluster, est une technique de regroupement d'ensembles d'objets similaires dans le même groupe qui est différent des objets d'un autre groupe**[7].**

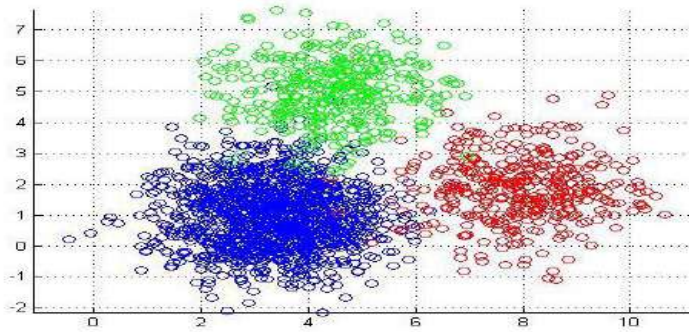


Figure I 6:Exemple de clustering ***[7]***

I.3. Les méthodes de classification

Les différents types d'algorithmes :

I.3.1. Les arbres de décision (AD)

Les arbres de décision(AD) sont des modèles d'apprentissage automatique supervisés, pouvant être utilisés pour la classification que pour la régression.

- **Arbres de classification :** la variable expliquée est de type nominale (facteur). A chaque étape du partitionnement, on cherche à réduire l'impureté totale des deux nœuds fils par rapport au nœud père.
- **Arbres de régression :** la variable expliquée est de type numérique et il s'agit de prédire une valeur la plus proche possible de la vraie valeur**[5].**

1. **Construction de l'arbre :**

   - **Nœuds internes** : sélection d'un attribut comme étiquette, les arcs sont étiquetés par les valeurs de l'attribut.
   - **Feuilles :** couper l'arbre avec une valeur de l'attribut cible

   Exemple : Un arbre de décision modélise une hiérarchie de tests sur les valeurs d'un ensemble de variables appelées attributs. À l'issue de ces tests, le prédicateur produit une valeur numérique ou choisit un élément dans un ensemble discret de conclusions. On parle de régression dans le premier cas et de classification dans le second. Par exemple,

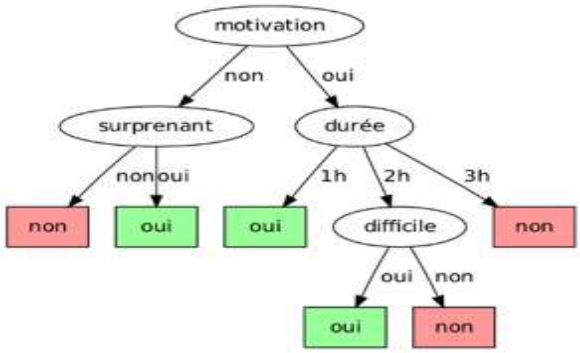


Figure I 7:Exemple d'arbre de décision [8]

l'arbre ci-dessus décide une réponse booléenne (classification dans l'ensemble {oui, non}) en fonction des valeurs discrètes des attributs {difficile, durée, motivation, surprenant}.

2. Inconvénient de AD :

- Détection difficile des interactions entre attributs.
- Certains problèmes sont difficiles à apprendre sous forme d'arbre.

I.3.2. K-Nearest Neighbors (KNN) :

On peut dire que l'algorithme d'apprentissage automatique le plus simple est la méthode d'apprentissage supervisé. Il peut être utilisé à la fois pour la régression et la classification. L'algorithme fonctionne sur les points de données les plus proches dans un ensemble de données d'apprentissage, c'est pourquoi il est appelé le plus proche voisin**[5].**

- Si **K-NN** est utilisé pour la régression, c'est la moyenne (ou la médiane) des variables y des **K**plus proches observations qui servira pour la prédiction.
- Si **K-NN** est utilisé pour la classification, c'est le mode des variables y des **K** plus proches observations qui servira pour la prédiction.

1. Le Choix de K (Nombre de voisins qui votent) :

Il faut inspecter les données.

- En général, K plus grand c'est mieux (mais pas toujours).
- Utiliser la cross-validation.
- Historiquement, pour la plupart des data sets (ensembles de données), choisir K entre 3 et 10 donne de bons résultats (beaucoup mieux que K=1).

- Remarque sur **K : Si** le nombre de classes = 2, il faut choisir **K** impair, **K** ne doit pas être un multiple du nombre de classes

Exemple :Dans l'exemple suivant, on a 3 classes et le but est de trouver la valeur de la classe de l'exemple inconnu x On prend la distance Euclidienne et **k**=5 voisins Des 5 plus proches voisins, 4appartiennent à ω1 et 1 appartient à ω3, donc x est affecté à ω1, la classe majoritaire**[9].**

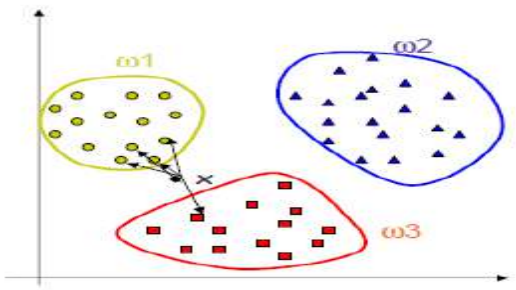


Figure I 8: Classification par KNN [9]

Tel que la distance Euclidienne qui calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points :

$$D_e(x, y) = \sqrt{\sum_{j=1}^{n}(x_j - y_j)^2}$$

Ce que nous pouvons observer à propos de cette méthode, c'est que le succès de cet algorithme dépend de la quantité de données d'apprentissage et de la qualité de l'échelle de similarité**[9].**

2. Inconvénient du KNN :

- Il doit conserver toutes les données d'entraînement en mémoire.
- Temps de recherche des **K** voisins pour chaque donnée.

I.3.3. Machines à vecteurs supports (SVM)

Est un algorithme d'apprentissage automatique supervisé qui peut être utilisé pour des défis de classification ou de régression. Cependant, il est principalement utilisé dans les problèmes de classification. Dans cet algorithme, nous traçons chaque élément de données

comme un point dans un espace à n dimensions où n est le nombre d'entités que vous avez, la valeur de chaque entité étant la valeur d'une coordonnée particulière. Ensuite, nous effectuons la classification en trouvant l'hyperplan qui différencie très bien les deux classes**[10].**

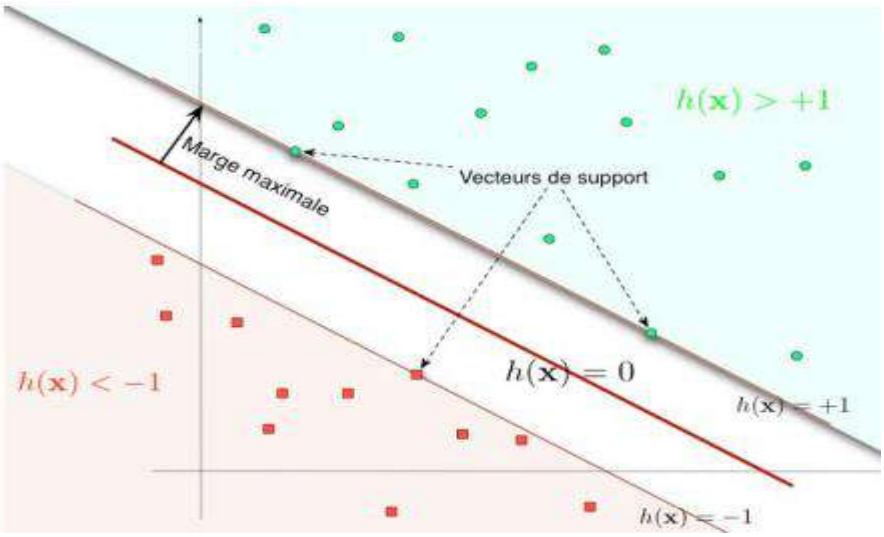


*Figure I 9 : Exemple sur le SVM **[10]***

En outre, il existe des techniques de regroupement essentielles comme :

- ✓ **K-means :** est un algorithme de partitionnement de données (clustering) relevant des statistiques et de l'apprentissage automatique. C'est une méthode dont le but est de diviser des observations en K partitions (clusters) dans lesquelles chaque observation appartient à la partition avec la moyenne la plus proche**[21].**
- ✓ **Analyse des composants principaux (**ACP**)**Il s'agit d'une classe de paradigme d'apprentissage non supervisé qui est utilisé pour réduire les dimensions des données.

I.4. Conclusion

Dans ce chapitre, nous avons introduit l'Apprentissage automatique qui est un sous-domaine de l'intelligence artificielle, son définition Les différents domaines d'application et leurs objectives.

Nous avons vu la différence entre la classification et La régression qui sont Représentées par les sorties (La classification : un ensemble fini de valeurs, La régression : des nombres), nous avons vu également les différents types d'apprentissage automatique (Apprentissage supervisé, Apprentissage non supervisé, L'apprentissage semi-supervisé,

L'apprentissage par renforcement) , ainsi que quelques méthodes de classification (arbres de décision, KNN ,…).

Dans le chapitre suivant, nous allons présenter la méthode des machines à vecteurs supports pour le problème de classification.

II. CHAPITRE II : Les Machines à Vecteurs Support (SVM)

II.1. Introduction

Parmi les méthodes à noyaux, inspirées de la théorie statistique de l'apprentissage de Vladimir Vapnik, les Machines à Vecteurs Support (**SVM**) constituent la forme la plus connue. SVM est une méthode de classification binaire par apprentissage supervisé, elle fut introduite par Vapnik en 1995**[9].** Cette méthode repose sur l'existence d'un classificateur linéaire dans un espace approprié.

Puisque c'est un problème de classification à deux classes, cette méthode fait appel à un jeu de données d'apprentissage pour apprendre les paramètres du modèle. Elle est basée sur l'utilisation de fonctions dites noyau (kernel) qui permettent une séparation optimale des données**[6].**

L'algorithme sous sa forme initiale revient à chercher une frontière de décision linéaire entre deux classes, mais ce modèle peut considérablement être enrichi en se projetant dans un autre espace permettant d'augmenter la séparabilité des données. On peut alors appliquer le même algorithme dans ce nouvel espace, ce qui se traduit par une frontière de décision non linéaire dans l'espace initial.

Dans la présentation des principes de fonctionnements, nous schématiserons les données par des « points » dans un plan.

II.2. Objectif de l'apprentissage statistique et SVM

La théorie d'apprentissage statistique étudie les propriétés mathématiques des machines d'apprentissage**[9]**. Ces propriétés représentent les propriétés de la classe de fonctions ou modèles que peut implémenter la machine. L'apprentissage statistique utilise un nombre limité d'entrées (appelées exemples) d'un système avec les valeurs de leurs sorties pour apprendre une fonction qui décrit la relation fonctionnelle existante, mais non connue, entre les entrées et les sorties du système**[7].**

On suppose premièrement que les exemples d'apprentissage, appelés aussi exemples d'entrainement, sont générés selon une certaine probabilité inconnue, c'est-à-dire indépendants et identiquement distribués. Ensuite, chaque exemple tracé comme un point dans un espace de dimension $\mathbf{n}(\in \boldsymbol{R^n})$pour le cas d'apprentissage supervisé, accompagnés

d'étiquettes caractérisant leurs types ou classes d'appartenance. Si ces étiquettes sont dénombrables, on parle de classification sinon on parle de régression. Dans le cas d'une classification binaire cette étiquette est soit +1 ou -1. L'ensemble des exemples et leurs étiquettes correspondantes est appelés ensemble d'apprentissage.

Le but est d'apprendre une fonction qui correspond aux exemples vus et qui prédit les sorties pour les entrées qui n'ont pas encore été vues. Les entrées peuvent être des descriptions d'objets et les sorties la classe des objets donnés en entrée

Soit : D=$\{(\boldsymbol{x_1}, \boldsymbol{y_1}), \dots, (\boldsymbol{x_n}, \boldsymbol{y_n})\}$l'ensemble des exemples d'entrainement,$\boldsymbol{x_i} \in \boldsymbol{R^n}$ et $\boldsymbol{y_i} = \pm \mathbf{1}$, f(x) est la fonction apprise par la machine d'apprentissage**[7].**

$$\boldsymbol{R_{emp}} = \frac{\mathbf{1}}{\boldsymbol{n}} \sum_{\mathbf{1}}^{\boldsymbol{n}} \boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{f}(\boldsymbol{x_i}))$$

Avec :$\boldsymbol{L} = \begin{cases} \mathbf{1} & \boldsymbol{si}\ \boldsymbol{y_i} = \boldsymbol{f}(\boldsymbol{x_i}) \\ \mathbf{0} & \boldsymbol{sinon} \end{cases}$, Où L désigne une fonction de coût.

Pour garantir la fiabilité de f sur des exemples ne participant pas à l'apprentissage, c'est-à-dire qu'ils n'ont jamais été vus auparavant, nous les mesurons sur différents exemples appelés exemples de test et il s'agit d'un test machine. Donc f minimise les erreurs de classification sur les deux ensembles : d'entrainement et de test**[7].**

Dans le cas des machines à vecteur support, la fonction recherchée est de forme linéaire. Les SVM sont, donc des systèmes d'apprentissage qui utilisent un espace d'hypothèses de fonctions linéaires dans un espace de caractéristique à haute dimension**[24].**

II.3. SVM à marge dure

II.3.1. Hyperplan séparateur

L'hyperplan séparateur est représenté par l'équation suivante : F(x) =w.x+b , Où w est un vecteur de m dimensions et b est un terme, La fonction de décision, pour un exemple x, peut être exprimée comme suit**[7]** :

$$\begin{cases} \text{Classe} = 1 & \text{Si}\ \ \mathrm{F(x)} > 0 \\ \text{Classe} = -1 & \text{Si}\ \ \mathrm{F(x)} < 0 \end{cases}$$

Sous l'hypothèse que les données sont linéairement séparables, trouver une règle pour les classer est très simple. En effet, il suffit de prendre un hyperplan qui sépare les classes,

puis de classer les données selon le côté de l'hyperplan où elles se trouvent. Plus formellement, soit :

$$w \cdot x + b = 0$$

Un hyperplan qui sépare les données. Alors, il suffit d'utiliser la fonction suivante pour effectuer la classification **[7]:**

$$\text{Classe}(x)=\text{signe}(w.x+b) \qquad \textbf{(2.1)}$$

Où :

$$\text{signe}(w \cdot x + b)=\begin{cases} -1 & \text{si } w \cdot x + b < 0 \\ 0 & \text{si } w \cdot x + b = 0 \\ 1 & \text{si } w \cdot x + b > 0 \end{cases} \textbf{(2.2)}$$

Cette fonction classe les données par rapport au côté de l'hyperplan où elles se trouvent.

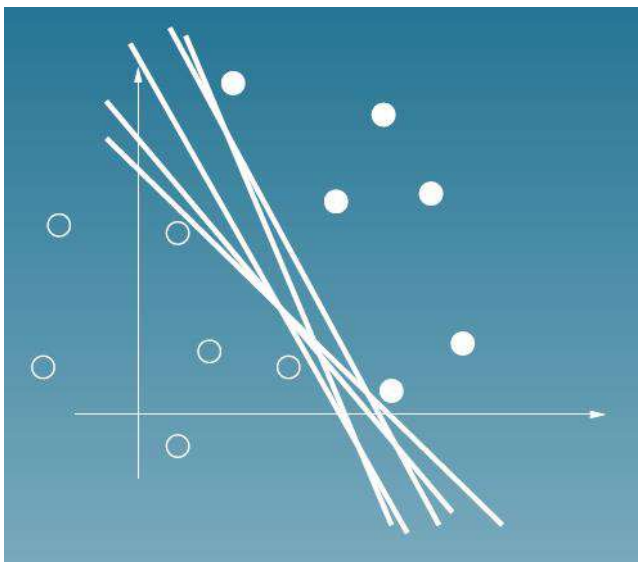


*Figure II 1: Il existe une infinité d'hyperplans pouvant séparer les données **[6]***

Si les données sont linéairement séparables, il existe une infinité d'hyperplans qui peuvent servir de séparateurs. L'idée des machines à vecteurs de support est de choisir le meilleur hyperplan, c'est-à-dire celui qui donnera la règle qui se généralisera le mieux à d'autres données que celles de l'ensemble d'apprentissage. Afin de déterminer ce qui caractérise le meilleur hyperplan, introduisons le concept de marge.

II.3.2. Maximisation de la Marge et hyperplan canonique

Définissons la marge d'un hyperplan comme étant la distance entre l'hyperplan et la donnée la plus proche. Plus formellement, si dist (x, w, b) représente la distance euclidienne entre le point x et l'hyperplan w · x + b = 0, alors la marge M est définie ainsi**[5][7]** :

$$M=\min\{dist(x_i, w, b): i = 1 \dots . . n\} \textbf{(2.3)}$$

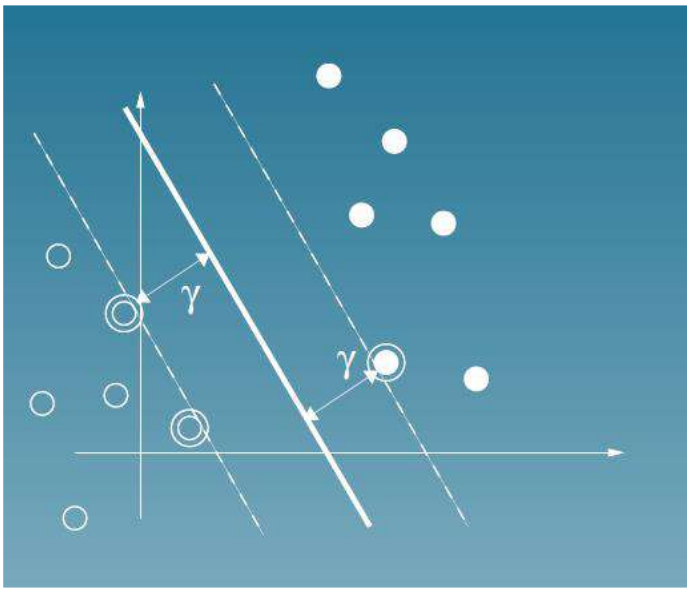


*Figure II 2 :la marge**[9]***

Où :x_i sont les données de l'ensemble d'apprentissage

- Dans le cas des données qui sont linéairement séparables, les **SVM** trouvent l'hyperplan qui sépare les données avec la plus vaste marge possible, puis utilisent cet hyperplan pour classer de nouvelles données à l'aide de la fonction indicatrice donnée plus haut(1.2).
- Considérons maintenant deux échantillons x1 et x2 de classes différentes telles qu'on ait ⟨ w, x1 ⟩ +b = +1 et ⟨ w, x2 ⟩ +b = -1. La marge γ correspond alors à la distance entre x1 et x2 mesurée perpendiculairement à l'hyperplan**[9][25]** :

$$\gamma= \langle w / ||w||, x1 - x2 \rangle = 2/||w|| \quad \textbf{(2.4)}$$

On peut donc en déduire que maximiser la marge revient à minimiser ||w|| sous certaines contraintes que nous verrons dans les paragraphes suivants.

- **Pourquoi maximiser la marge ?**

Le fait d'avoir une marge plus large procure plus de sécurité lorsqu'on classe un nouvel exemple. De plus, si l'on trouve le classificateur qui se comporte le mieux vis-à-vis données d'apprentissage, il est clair qu'il sera aussi celui qui permettra au mieux de classer les nouveaux exemples**[26].**

Dans le schéma **figure** *II.3* , la partie droite nous montre qu'avec un hyperplan optimal, un nouvel exemple reste bien classé alors qu'il tombe dans la marge. On constate sur la partie gauche qu'avec une plus petite marge, l'exemple se voit mal classé.

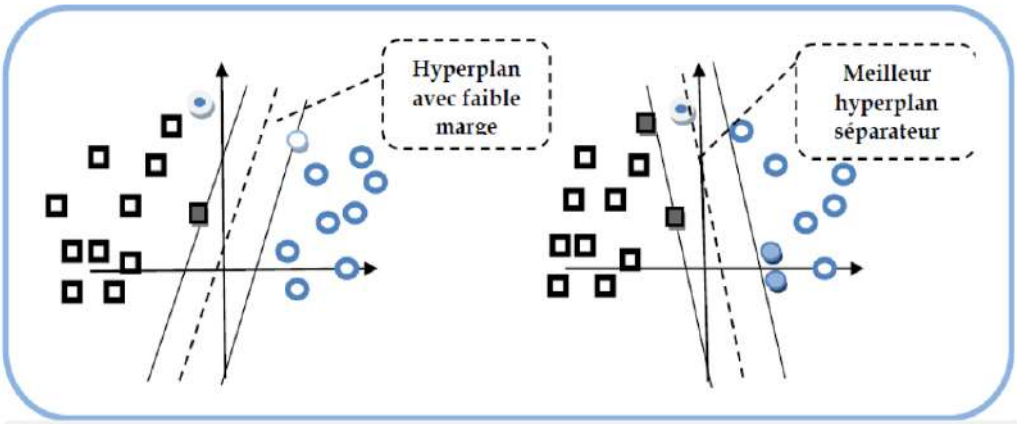


*Figure II 3: Meilleur hyperplan séparateur **[9]***

Formulation du problème

1. Problème primal :

Un point (x; y) est bien classé si et seulement si yF(x) > 0 Comme le couple (w, b) est défini à un coefficient multiplicatif près, on s'impose yF(x) ≥ 1,On en déduit le problème de minimisation sous contraintes suivantes**[23]** :

$$\begin{cases} \text{Minimiser } \frac{1}{2}||w||^2 \\ \text{sous contraintes} \\ y_i = (w^T x_i + b) \geq 1 \quad \forall i = 1..n \end{cases} \quad \text{n : la taille des échantillons d'apprentissage} \qquad \textbf{(2.5)}$$

C'est un problème d'optimisation quadratique à fonction objectif convexe et contraintes linéaires dont une solution unique est assurée. Il faut déterminer w^* et b^* minimisant $\frac{1}{2}||w||^2$tout en respectant les contraintes, dites de bon classement: les points de la base d'apprentissage soient bien classés sans dépasser les hyperplans canoniques**[23].**

Si la dimension des données (m) est assez petite, les méthodes classiques de programmation mathématique telles que la méthode stochastique de type Gauss-Seidel,

méthode d'ensemble actif, ou l'algorithme de point intérieur, de Newton avec région de confiance ou type gradient conjugué peuvent être utilisées pour résoudre ce (QP) sinon cela devient inenvisageable pour des valeurs de m dépassant quelques centaines. Il existe une transformation de ce problème dans une formulation duale que l'on peut utiliser en pratique. Notons que la complexité de ce problème d'optimisation est de l'ordre de O(n) et non pas de l'ordre de O(m)**[23].**

2. Problème dual :

Le problème dual est un programme quadratique de taille n (égal au nombre d'observations) qui peut s'avérer plus facile à résoudre que le problème primal, On passe du problème primal au problème dual en introduisant des multiplicateurs de Lagrange pour chaque contrainte :

$$L(w, b, \alpha) = \frac{1}{2} w.w - \sum_{i=1}^{n} \alpha_i (y_i (w.x_i + b) - 1) \qquad \textbf{(2.6)}$$

Il faut maintenant calculer la fonction objective du problème dual. Rappelons que cette fonction correspond à la valeur minimale du Lagrangien pour un α donné. Or, ce minimum correspond au point où la dérivée du Lagrangien par rapport aux variables du primal est nulle. On a donc ainsi**[7]** :

- ✓ Minimiser L par rapport à w :

$$\frac{\partial(w,b,\alpha)}{\partial w} = w - \sum_{i=1}^{n} \alpha_i \, y_i x_i = 0,$$

- ✓ Minimiser L par rapport à b :

$$\frac{\partial(w,b,\alpha)}{\partial b} = -\sum_{1}^{n} \alpha_i \, y_i = 0,$$

Alors ce qu'il est possible de réécrire de cette manière :

$$w = \sum_{i=1}^{n} \alpha_i \, y_i x_i \textbf{(2.7)}$$

$$\sum_{i=1}^{n} \alpha_i \, y_i = 0 \textbf{(2.8)}$$

Utilisons ces équations pour réécrire le Lagrangien minimal uniquement en fonction des variables duales :

$$min_w L(w, b, \alpha) = \frac{1}{2} w.w - \sum_{i=1}^{n} \alpha_i \, (y_i (w.x + b) - 1)$$

Ainsi, nous avons le problème dual suivant**[7]** :

$$\text{Maximizer} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i \alpha_j \, y_i y_j x_i . x_j)$$

$$\text{sujet à} \begin{cases} \sum_{i=1}^{n} \alpha_i \, y_i = 0 \\ \alpha_i \geq 0 \quad \text{i} = 1, \ldots, \text{n.} \end{cases} \textbf{(2.9)}$$

La solution de ce problème d'optimisation sera bien sûr un vecteur $\alpha^* = (\alpha_1^* , \ldots , \alpha_n^*$), alors que c'est l'équation d'un hyperplan qu'il faut pour classer les données à l'aide de la fonction indicatrice**(1)**puisque $W^* = \sum_{i=1}^{l} \alpha_i^* y_i x_i$

Alors :

$$\text{Classe (x)} = \text{signe}(\sum_{i=1}^{n} (\alpha_i y_i x_i . x) + b) \textbf{(2.10)}$$

Comme le paramètre b ne figure pas dans le problème dual, sa valeur optimale b^* peut être calculée à partir des contraintes primales, soit donc :

$$b^* = \frac{max_{y_i=-1}(\langle w^* . x_i \rangle) + min_{y_i=+1}(\langle w^* . x_i \rangle)}{2} \qquad \textbf{(2.11)}$$

Soit S = {i ∈ {1, 2, . . . , l} : $\alpha_i^* \neq 0$} l'ensemble des indices des vecteurs supports. Une fois les paramètres α^* et b^* calculés, la fonction de décision d'une nouvelle observation x basée sur l'hyperplan à marge maximale est donnée par :

$$\text{h (x)} = \text{sign}(\sum_{i \in S} y_i \, \alpha_i^* \langle x_i . x \rangle + b^*) \textbf{(2.12)}$$

Ainsi, la résolution du problème dual permet de construire l'hyperplan canonique séparant les données avec la plus grande marge et de l'utiliser pour classer des données, tout comme la résolution du problème primal.

II.3.3. Vecteurs supports

On a respecté la condition complémentaire de Karush-Kuhn-Tucker, c'est-à-dire que :

$$\alpha_i^* [y_i (< w^* \bullet x_i > + b^*) - 1] = 0, \text{ i=1,2,….,l} \qquad \textbf{(2.13)}$$

Où α^* représente la solution optimale du problème dual et (w^*, b^*) représente celle du problème primal.

Qui expriment le fait qu'à l'optimum le produit des variables duales et des contraintes associées doit être nul, nous donnent une information très utile sur la structure de la solution.

Les conditions (2.13) impliquent que les α_i^* sont nuls pour les contraintes non saturées. Les éléments α_i de l'échantillon d'apprentissage pour lesquels les coefficients α_i^* sont non nuls, sont appelés les **vecteurs supports**. Compte tenu des conditions de KKT, ces vecteurs définissent à eux seuls la **solution du problème** (2.5). Ils définissent la frontière de décision**[7][6].**

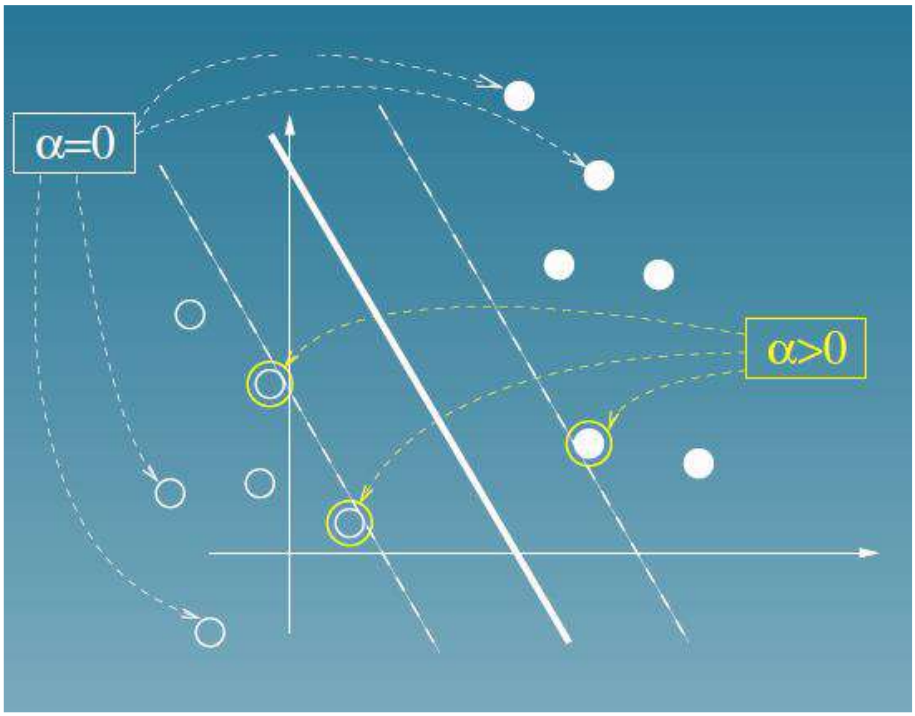


Figure II 4: Les vecteurs de support ***[6]***

Ainsi, tout point qui n'est pas sur la marge n'apporte aucune contribution, puisque αi est alors nul. Si tous les points sauf les vecteurs supports étaient retirés de l'ensemble d'apprentissage, on retrouverait le même hyperplan.

Les vecteurs supports peuvent donc être vus comme les points contenant toute l'information essentielle du problème.

II.4. SVM à marge souple

Souvent il arrive que même si le problème est linéaire, les données sont affectées par un bruit (par ex. de capteur) et les deux classes se retrouvent mélangées autour de l'hyperplan de séparation. Pour gérer ce type de problème on utilise une technique dite de **marge souple**, qui tolère les mauvais classements :

- ✓ Rajouter des variables de relâchement des contraintes ξ_i

- ✓ Pénaliser ces relâchements dans la fonction objective.

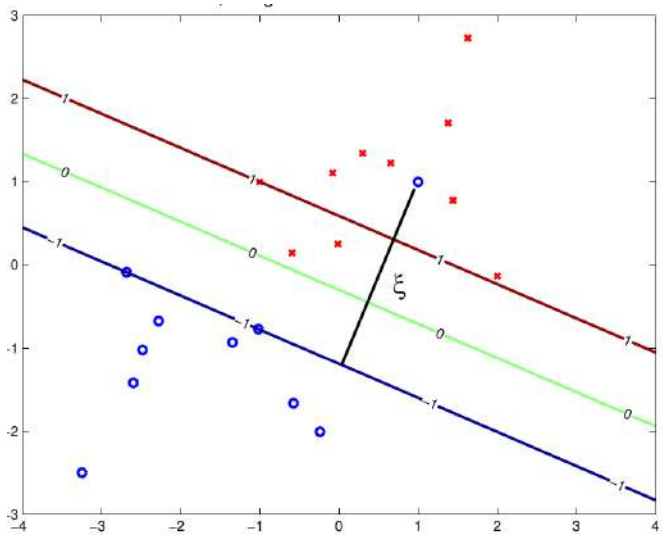


Figure II 5: Plus en exemple est éloigné du mauvais côté du séparateur (point bleu), plus la variable de relâchement ξ_i *a une valeur importante.**[8]***

On introduit alors sur les contraintes des variables $\xi_i$dites de relaxation pour obtenir la contrainte de l'équation :

$$y_i\,(<\mathbf{w}.\,x_i>+\mathbf{b}) \geq 1-\xi_i \quad \text{i=1}\ldots..\text{n} \qquad \textbf{(2.14)}$$

Si $\xi_i < 1$, x_i ne respecte pas la marge mais reste bien classé, sinon x_i est mal classé par l'hyperplan. Dans ce cas, au lieu de rechercher uniquement un hyperplan séparateur qui maximise la marge, on recherche un hyperplan qui minimise aussi la somme des erreurs permises c.-à-d. minimiser**[5].**

$$Q(w) = \sum_{i=1}^{n} \xi_i.$$

Le problème (2.5) devient alors :

$$\begin{cases} Minimiser \ \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{sous} \quad y_i\left(<w.\,x_i>+b\right) \geq 1-\xi_i \ \text{i} = 1..\text{n} \\ \xi_i \text{i} \ \geq \ 0 \end{cases} \textbf{(2.15)}$$

Si toutes les variables d'écart ξ_i=0, on retrouve le problème linéairement séparable traité plus tôt.

C est une variable de pénalisation des points mal classés faisant un compromis entre la largeur de la marge et les points mal classés. Les variables ξ_is'appellent aussi *variables ressort.*

Si C est important, moins d'erreurs sont autorisées. En suivant la même démarche du Lagrangien que précédemment, nous aboutissons à la forme duale :

$$\begin{cases} Maximiser_{\alpha} w(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i \, y_j \alpha_i \alpha_j x_i^T \, x_j \\ \text{Sous } \sum_{i=1}^{n} \alpha_i y_i = 0 \\ C \geq \alpha_i \geq 0, i = 1, \dots, n \end{cases} \quad \textbf{(2.16)}$$

La seule différence avec le SVM à marge dure est que les αi ne peuvent pas dépasser C, ils peuvent être dans l'un des trois cas suivants **[10]**:

- $\boldsymbol{\alpha_i = 0}$: l'exemple est bien classé , il n'est pas sur les hyperplans parallèles, on dira que l'exemple est un non support vecteur (sv).
- $\boldsymbol{0 < \alpha_i < C}$: l'exemple est bien classé , il se trouve sur l'un des deux hyperplans parallèles, l'exemple est appelé vecteur support (sv).
- $\boldsymbol{\alpha_i = C}$: l'exemple est mal classé , malgré tout il est considéré comme sv

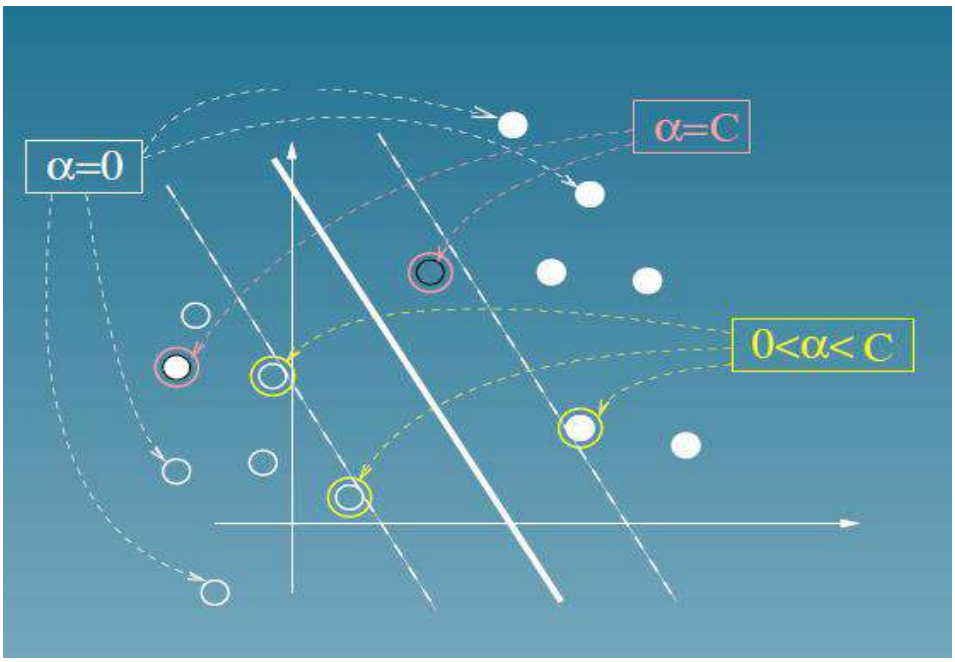


Figure II 6: les cas de $\boldsymbol{\alpha_i}$ ***[6]***

II.5. SVM non-linéaires

La notion de marge maximale et la procédure de recherche de l'hyperplan séparateur telles que présentées pour l'instant ne permettent de résoudre que des problèmes de discrimination linéairement séparables. C'est une limitation sévère qui condamne à ne pouvoir résoudre que des problèmes jouets, ou très particuliers. Afin de remédier au problème de l'absence de séparateur linéaire, l'idée de l'astuce du noyau (en anglais kernel trick) est de reconsidérer le problème dans un espace de dimension supérieure, éventuellement de dimension infinie. Dans ce nouvel espace, il est alors probable qu'il existe une séparation linéaire **[11].**

En remarquant que dans la résolution des problèmes (2.9) et (19), seuls les produits scalaires $< x_i , x_j >$ sont nécessaires, les SVM peuvent être étendues pour traiter le cas non-linéaire.

La prise en compte de non linéarités dans le modèle s'effectue par l'introduction de noyaux non linéaires. De manière inattendue, l'utilisation de noyaux ne modifie pas fondamentalement la nature des SVM (pourvu que l'on travaille dans le dual).

Cette transformation d'espace est réalisée souvent à l'aide d'une fonction

$$\varphi : R^p \rightarrow T$$

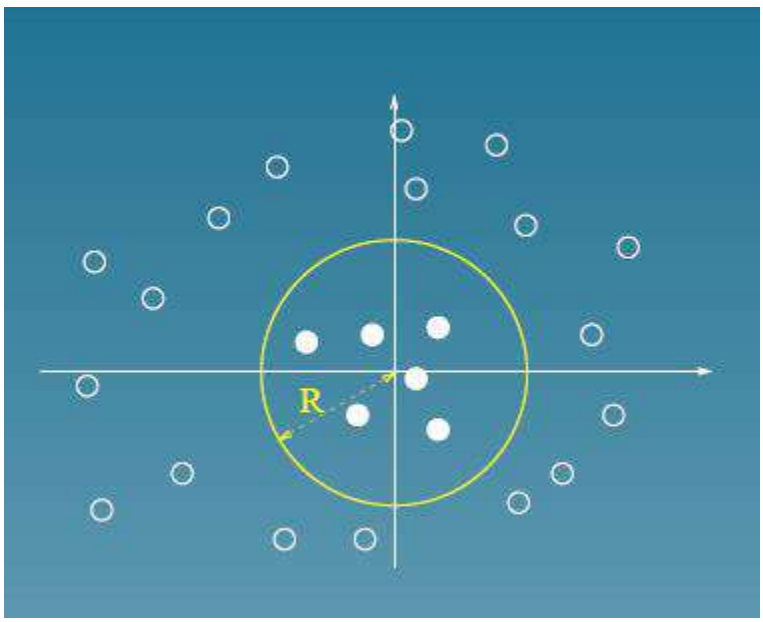


*Figure II 7: Exemple d'un problème de discrimination à deux classes, avec une séparatrice non-linéaire : le cercle unité. Le problème n'est pas linéairement séparable.**[6]***

En utilisant la même procédure que dans le cas sans transformation, on aboutit au problème d'optimisation suivant :

$$\begin{cases} max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \varphi(x_i).\varphi(x_j) \\ \forall i, C \geq \alpha_i \geq 0 \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases} \quad \textbf{(2.17)}$$

Et la solution de la forme :

$$F(x)=\sum_{i=1}^{n} \alpha_i y_i \varphi(x_i).\varphi(x_j) + b \textbf{(2.18)}$$

Sous certaines hypothèses sur φ, le produit scalaire $<\phi(x_i)\cdot\phi(x_j)>$ peut se calculer facilement à l'aide d'une fonction symétrique *K*, dite noyau, définie par :

$$K(x_i, x_j) = \varphi(x_i).\varphi(x_j)$$

La fonction noyau **K** est utilisée pour calculer la distance entre le vecteur à tester x et chaque vecteur support dans l'espace de caractéristique. Les résultats sont ensuite linéairement combinés en utilisant les multiplicateurs de Lagrange α_i et ajoutés au biais b. Le résultat final F permet de décider à propos du nouveau vecteur : si F(x) est positive, il s'agit du chiffre "1", sinon "-1 "**[11].**

Et la fonction de décision devient :

$$F(x)=\sum_{i=1}^{n} \alpha_i y_i k(x_i, x_j) + b \textbf{(2.19)}$$

Exemples de noyaux:

- **Noyau linéaire** : Si les données sont linéairement séparables, on n'a pas besoin de changer d'espace, et le produit scalaire suffit pour définir la fonction de décision :

$$K(x_i, x_j) = x_i.x_j \textbf{[6]}$$

- **Noyau polynomial :** Le noyau polynomial élève le produit scalaire à une puissance naturelle d :

$$K(x_i, x_j) = (c+< x_i.x_j >)^d, \text{ou } c \in R_+ \text{ et } d \in N \textbf{[11]}$$

- **Noyau RBF** : Les noyaux RBF (Radial Basis functions) sont des noyaux qui peuvent être écrits sous la forme :

$$K(x_i, x_j)=f(d(x_i, x_j)) \textbf{[11]}$$

Où d est une métrique sur X et f est une fonction dans R .Un exemple des noyaux RBF est le **noyau Gaussien** :

$$K(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$$, ou $\sigma \in R_+^*$ est la largeur de bande **[11]**

II.6. Résolution des problèmes d'optimisation issus des SVM

Dans la section précédente nous avons vu que l'apprentissage des **SVM** se ramène à la maximisation d'une forme quadratique convexe sous des contraintes linéaires. Dans ces cas il n'y a pas de problèmes de minimums locaux et la solution peut être trouvée en utilisant des algorithmes efficaces. Par contre les méthodes classiques de résolution sont inadaptées aux problèmes de grande taille.

Pour gérer les problèmes de grande taille il existe des méthodes dites de décomposition. Elles reviennent à décomposer le problème en plusieurs petits sous-problèmes tels que la résolution de chacun d'eux fournisse une approximation toujours meilleure de l'optimum.

L'algorithme d'optimisation minimale séquentielle (Séquentiel Minimal Optimisation, **SMO**) proposé par Platt est un cas extrême de ces méthodes. Il consiste à optimiser à chaque itération, deux $\alpha_i$conjointement.on trouve dans la littérature plusieurs raffinements de cet algorithme. L'algorithme **SMO** sera détaillé dans le chapitre suivant.

II.7. Evaluation du modèle

L'apprentissage supervisé effectué par la méthode **SVM** utilise une partie de l'ensemble d'exemples d'un espace, pour calculer un modèle de décision qui sera généralisé sur l'ensemble de tous les exemples de l'espace. Il est très important d'avoir des mesures permettant de qualifier le comportement du modèle appris sur les exemples qui ne sont pas utilisés lors de l'entrainement.

Ces métriques sont calculées soit sur les exemples d'entrainement eux-mêmes ou sur des exemples réservés à l'avance pour les tests**[7].**

II.8. Estimation basée sur un échantillon test

Si nous disposons d'un grand nombre d'observations, il est possible de construire un modèle **SVM** sur une partie (échantillon d'apprentissage) et estimer son erreur sur le reste (échantillon test). L'estimation que nous obtenons est non biaisée en plus sa variance est

d'autant plus réduite que la taille l0 de l'échantillon test est grande. Cette estimation est donnée par **[22]**:

$$T=\frac{1}{l'}\sum_{i=1}^{l'} sign(y_i' f(x_i'))$$

Où {($(x_i' y_i')_{1\le i\le l'}$)} est l'échantillon test.

II.9. SVM à plusieurs classes

Les machines à vecteur support sont dans leur origine binaires. Cependant, les problèmes du monde réel sont dans la plupart des cas multi classe. Dans ce cas, on ne cherche pas affecter un nouvel exemple à l'une de deux classes mais à l'une parmi plusieurs, donc un seul hyperplan ne suffit pas.

Les méthodes des machines à vecteur support multi classe, réduisent le problème multi classe à une composition de plusieurs hyperplans bien classés permettant de tracer les frontières de décision entre les différentes classes**[27][28]**. Ces méthodes décomposent l'ensemble d'exemples en plusieurs sous-ensembles représentant chacun un problème de classification binaire. Pour chaque problème un hyperplan de séparation est déterminé par la méthode **SVM** binaire. On construit lors de la classification une hiérarchie des hyperplans binaires qui est parcourue de la racine jusqu'à une feuille pour décider de la classe d'un nouvel exemple. On trouve dans la littérature plusieurs méthodes de décomposition :

II.9.1. Méthode Une-contre-reste (1vsR)

C'est la méthode la plus simple et la plus ancienne. Selon la formulation de Vapnik**[9]** ,elle consiste à déterminer pour chaque classe k un hyperplan $H_k(w_k, b_k)$ la séparant detoutes les autres classes. Cette classe k est considérée comme étant la classe positive (+1)et les autres classes comme étant la classe négative (−1), ce qui résulte, pour un problème à K classes, en K **SVM** binaires. Un hyperplan H_k est défini pour chaque classe k par lafonction de décision suivante**[7]** :

$$H_k(x) = signe(< w_k, x > + b_k)$$
$$= \begin{cases} +1 & si \quad f_k(x) > 0 \\ 0 & sinon \end{cases}$$

La valeur retournée de l'hyperplan permet de savoir si x appartient à la classe k ou non .Dans le cas où il n'appartient pas à k ($H_k(x)$=0) , nous n'avons aucune information sur l'appartenance de x aux autres classes. Pour le savoir, on présente x à tous les hyperplans ,ce qui donne la fonction de décision de l'équation suivante**[7]** :

$$\boldsymbol{k^* = Arg\ \ Max\ \left(H_k(x)\right)}(1 \leq k \leq K)$$

- Si une seule valeur $H_k(x)$ est egale a 1 , et toutes les autres sont égales à 0 , on déduit que x appartient a la classe k.
- Le problème est que l'équation peut être vérifiée pour plus d'une classe, ce qui produit des régions d'ambiguïté, et l'exemple x est dit non classifiable. On utilise dans ce cas un vote majoritaire pour attribuer l'exemple x à la classe k selon la formule suivante :

$$\boldsymbol{k^* = Arg\ \ Max\ (w_k(x) + b_k)}(1 \leq k \leq K)$$

La figure suivante représente un cas de séparation de 3 classes.

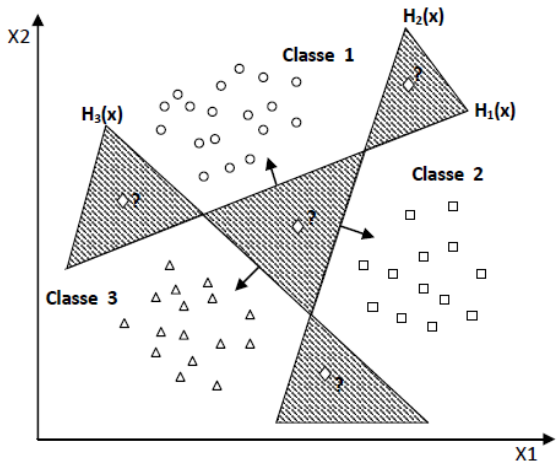


*Figure II 8 : approche Un contre Reste(1vsR)**[8]***

II.9.2.Méthode Une contre Une (1vs1)

Cette méthode, appelée aussi "pairwise", revient à Kner et ses co-auteurs qui l'ont proposée pour les réseaux de neurones**[29]**. Elle consiste à utiliser un classificateur pour chaque paire de classes. Au lieu d'apprendre K fonctions de décisions, la méthode 1vs1

discrimine chaque classe de chaque autre classe, ainsi K(K−1)/2 fonctions de décisions sont apprises.

Pour chaque paire de classes (k, s), la méthode 1vs1 définit une fonction de décision binaire $h_{ks} : R \rightarrow \{-1, +1\}$. L'affectation d'un nouvel exemple se fait par liste de vote.

On teste un exemple par le calcul de sa fonction de décision pour chaque hyperplan. Pour chaque test, on vote pour la classe à laquelle appartient l'exemple (classe gagnante). On définit pour le faire la fonction de décision binaire $h_{ks}(x)$ de l'équation suivante**[29]** :

$$H_{ks}(x) = signe(f_{ks}(x))$$
$$= \begin{cases} +1 & si \quad f_{ks}(x) > 0 \\ 0 & sinon \end{cases}$$

Sur la base de K(K-1)/2 fonctions de décision binaires , on définit K autres fonctions de décision :

$$H_k(x) = \sum_{s=1}^{m} H_{ks}(x)$$

Un nouvel exemple est affecte a la classe la plus votée. La règle de classification est donnée par l'équation suivante :

$$\boldsymbol{k^* = Arg \ \ Max \left(H_k(x)\right) \quad (1 \leq k \leq K)}$$

La figure suivante représente un cas de séparation de 3 classes.

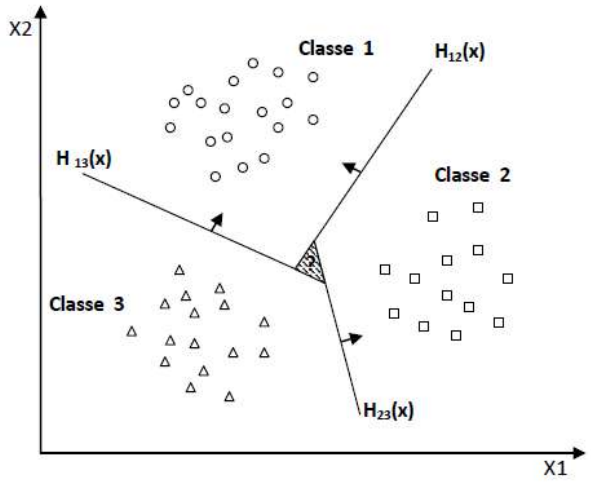


*Figure II 9 : approche Un contre Un (1vs1)**[8]***

II.10. Conclusion

Dans ce chapitre, nous avons tenté de présenter de manière simple et complète le concept de système d'apprentissage introduit par Vladimir Vapnik, les Machines à Vecteurs de Support. Nous avons donné une vision générale et une vision purement mathématique des **SVM**.

Cette méthode de classification est basée sur la recherche d'un hyperplan qui permet de séparer au mieux des ensembles de données. Nous avons exposé le cas linéairement séparable et les cas non linéairement séparables qui nécessitent l'utilisation de fonction noyau (kernel) pour changer d'espace. Cette méthode est applicable pour des tâches de classification à deux classes, mais il existe des extensions pour la classification multi classe dont nous avons présenté deux approches les plus connues.

III. Chapitre III : Implémentation de l'algorithme SMO et Expérimentation

III.1. Introduction

Un algorithme résolvant les **SVM** identifie parmi les exemples d'apprentissage quels sont les vecteurs supports et construit la frontière (ou fonction de décision) avec une combinaison linéaire de cette sélection. Résoudre ce problème équivaut à résoudre un programme quadratique.

A travers ce chapitre, on va détailler un algorithme très connu qui est dédié aux machines à vecteurs supports, l'algorithme nommé **SMO** (Séquentiel Minimisation Optimisation) de John Platt(1999) et pour l'implémentation de cet algorithme nous avons choisi le langage Python .

Afin de pouvoir comparer nos résultats expérimentaux de notre implémentation de l'algorithme **SMO** par rapport à une autre méthode de programmation quadratique classique, Nous avons choisi une bibliothèque intégrée à python consacrée aux problèmes de classification par les **SVMs** nommée **scikit-learn**. Pour lancer l'apprentissage il suffit d'utiliser le module **SVC** (support vecteur classification) (le package : **sklearn.SVM.SVC).**

Les tests des deux méthodes (**SMO et SVC**) sont effectués sur deux différents types de bases de données :

La première concerne une base de données artificielles contenant trois data set, la seconde concerne des données du monde réel, nos expérimentations seront interprétées.

III.2. La méthode d'optimisation séquentielle minimale(SMO)

John Platt a proposé en **1999** un algorithme pour l'entrainement des **SVM** qu'il a appelé algorithme d'Optimisation Séquentielle Minimale (**SMO**). Cet algorithme, décompose le problème de **PQ** mais choisi de résoudre le plus petit sous-problème possible à chaque étape d'optimisation de la fonction objectif.

La méthode d'optimisation par minimisation séquentielle peut être perçue comme le cas extrême des méthodes de décomposition successive.

Cela est réalisable en choisissant d'optimiser, non pas un ensemble de multiplicateurs de Lagrange à la fois, mais deux seulement. En effet, il nécessite plus d'itérations pour converger mais chaque itération effectue moins d'opérations.

III.2.1. Implémentation de l'algorithme SMO

L'algorithme **SMO** est basé sur trois éléments **[12]:**

1- Une méthode analytique pour résoudre les deux multiplicateurs de Lagrange.
2- Deux heuristiques pour choisir quels multiplicateurs à optimiser.
3- Une méthode pour le calcul du paramètre **b**.

III.2.2. Principe de l'algorithme

L'idée principale de cet algorithme est de décomposer le problème à l'extrême en optimisant uniquement deux points à chaque itération. L'avantage de ceci est qu'optimiser une équation à deux variables est un problème qui a une solution analytique.

-optimiser deux α_i [12]:

Il s'agit de l'itération principale de **SMO**, nous allons la détailler point par point en se basant sur des relations entre les variables.

-lien entre les α [12]:

Considérons deux coefficients quelconques choisis parmi l'ensemble des alphas .dans un but de simplification des calculs, nous supposerons par la suite que ces deux coefficients sont respectivement le premier et le second des α ce sont respectivement le premier et le second des α ce sont donc $\alpha_1$et α_2.

Ces deux coefficients ont chacun une ancienne valeur à optimiser : α_1^{old} et α_2^{old}

Pour la phase d'initialisation, ces deux anciennes valeurs pourront être fixées à 0.

Les autres coefficients α de l'équation sont considérés comme étant constants et fixés d'après la contrainte $\sum_i^l \alpha_i\, y_i = 0$,nous pouvons établir que :

$$\boldsymbol{y_1\alpha_1 + y_2\alpha_2 = y_1\alpha_1^{old} + y_2\alpha_2^{old}} = constante \textbf{(3.1)}$$

Il est donc possible d'établir une relation entre α_1 et α_2 :

$$\alpha_1 = \frac{1}{y_1}(y_i\alpha_1^{old} + y_2\alpha_2^{old} - y_2\alpha_2)$$

$$= \alpha_1^{old} + \frac{y_2}{y_1}\alpha_2^{old} - \frac{y_2}{y_1}\alpha_2$$

$=\alpha_1^{old} + y_1 y_2\ (\alpha_2^{old} - \alpha_2)$

Le calcul d'un α_2 optimal seul permet alors de résoudre le problème d'optimisation de ces deux points.

$$\alpha_1 = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2)$$

- Ajustement des α selon les contraintes de faisabilité

Afin de s'assurer d'avoir toujours une solution réalisable, il faut vérifier que les valeurs qui seront calculées soient conformes aux contraintes de faisabilité. En effet, les deux coefficients α sont bornés par le coefficient de marge d'erreur.

$0 \leq \alpha_1, \alpha_2 \leq C$

Appliqué à une valeur α_2^* résultant d'une itération d'optimisation de l'algorithme, cela donne une première contrainte :

$0 \leq \alpha_2^* \leq C$ **(3.2)**

Etant donné l'équation**(3.1)** et le fait que $y_1 y_2 \in \{-1, +1\}^2$,il est possible de prévoir une contrainte supplémentaire selon les cas le résultat de classification :

1. les deux valeurs d'appartenance aux classes sont égales, $\mathbf{y_1 = y_2}$

$$y_1 = y_2 => \alpha_1 + \alpha_2 = \alpha_1^{old} + \alpha_2^{old}$$

Donc $\alpha_1 = \alpha_1^{old} + \alpha_2^{old} - \alpha_2$

$0 \leq \alpha_1 \leq C$,donc $0 \leq \alpha_1^{old} + \alpha_2^{old} - \alpha_2 \leq C$

Soit $\alpha_1^{old} + \alpha_2^{old} - C \leq \alpha_2 \leq \alpha_1^{old} + \alpha_2^{old} 0 \leq \alpha_2 \leq C$**(3.3)**

Donc finalement, en prenant en compte les deux contraintes**(3.1)** et**(3.3)**,il advient que pour $y_1 = y_2$ il faut avoir :

$$\max(0, \alpha_2^{old} + \alpha_1^{old} - C) \leq \alpha_2 \leq \min(C, \alpha_1^{old}, \alpha_2^{old})$$

2. les deux valeurs d'appartenance aux classes différentes , $\mathbf{y_1 \neq y_2}$

$$y_1 \neq y_2 => \alpha_1 - \alpha_2 = \alpha_1^{old} - \alpha_2^{old}$$

Donc $\alpha_1 = \alpha_1^{old} - \alpha_2^{old} + \alpha_2$

$0 \leq \alpha_1 \leq \mathrm{C}$, donc$0 \leq \alpha_1^{old} - \alpha_2^{old} - \alpha_2 \leq \mathrm{C}$

Soit $\alpha_2^{old} - \alpha_1^{old} \leq \alpha_2 \leq \mathrm{C} - \alpha_1^{old} + \alpha_2^{old}$**(3.4)**

Donc finalement, en prenant en compte les deux contraintes **(3.1)**et **(3.4)** , il advient que pour $y_1 \neq y_2$,il faut avoir :

$$\max\left(0, \alpha_2^{old} - \alpha_1^{old}\right) \leq \alpha_2 \leq \min\left(\mathrm{C}, \mathrm{C} - \alpha_1^{\mathrm{old}} + \alpha_2^{\mathrm{old}}\right)$$

Une fois le coefficient est calculé, sa valeur est ramenée à la borne la plus proche de l'intervalle si celle-ci la dépassait.

Par la suite, ces deux bornes inférieure et supérieure seront respectivement dénommées L et H.

$\mathrm{L} = \max\left(0, \alpha_2^{old} + \alpha_1^{old} - \mathrm{C}\right) \quad si \quad y_1 = y_2$

$$= \max\left(0, \alpha_2^{old} - \alpha_1^{old}\right) \qquad si \quad y_1 \neq y_2$$

$\mathrm{H} = \min(\mathrm{C}, \alpha_1^{\mathrm{old}} + \alpha_2^{\mathrm{old}})$ si $\mathrm{y}_1 = \mathrm{y}_2$

$= \min\left(\mathrm{C}, \mathrm{C} - \alpha_1^{\mathrm{old}} + \alpha_2^{\mathrm{old}}\right)$ si $y_1 \neq y_2$

-calcul d'un α_2 optimal [11]:

Considérons maintenant la formulation de la fonction objective du problème dual :

$$\omega(\alpha) = \sum_{i=1}^{\mathrm{n}} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\mathrm{n}} \alpha_i \, \alpha_j y_i y_j k\left(x_i, x_j\right)$$

Celle-ci peut être réécrite pour faire apparaitre α_1 et α_2 en considérant tous les autres α comme étant connus et fixés.

$$\omega(\alpha) = \alpha_1 + \alpha_2 - \frac{1}{2}\left(\alpha_1 \alpha_1 y_1 y_1 k(x_1, x_1) + \alpha_2 \alpha_2 y_2 y_2 k(x_2, x_2)\right) + \sum_{i=3}^{\mathrm{n}} \alpha_i$$
$$- \frac{1}{2}\left[2\alpha_1 y_1 \sum_{i=3}^{\mathrm{n}} \alpha_i \, y_i k(x_1, x_i) + 2\alpha_2 y_2 \sum_{i=3}^{\mathrm{n}} \alpha_i \, y_i k(x_2, x_i)\right]$$

Cette écriture peut être simplifiée en posant quelques nouvelles variables :

$$k_{11} = k(x_1, x_1)$$

$$k_{12} = k(x_1, x_2)$$

$$k_{22} = k(x_2, x_2)$$

$$v_i = \sum_{j=3}^{n} \alpha_j \, y_j k(x_i, x_j)$$

Ce qui donne :

$$\omega(\alpha_1, \alpha_2) = \alpha_1 + \alpha_2 - \frac{1}{2}(\alpha_1^2 k_{11} + \alpha_2^2 k_{22}) - \alpha_1 \alpha_2 y_1 y_2 k_{12} - \alpha_1 y_1 v_1 - \alpha_2 y_2 v_2 + const$$

Etant donné que l'on cherche à optimiser la valeur de α_2 , il s'agit maintenant d'éliminer α_1 de la formulation de la fonction objective. Or d'après l'équation et en posant

$$s = y_1 y_2 = \frac{y_1}{y_2} = \frac{y_2}{y_1}$$

Il est possible d'écrire que $\alpha_1 + s\alpha_2 = \alpha_1^{old} + s\alpha_2^{old} = \gamma$ **(3.5)**

Donc $\alpha_1 = \gamma - s\alpha_2$

Et par conséquent (remplaçant α_1 par $\gamma - s\alpha_2$ dans la fonction objective) :

$\omega(\alpha_2) = \gamma - s\alpha_2 + \alpha_2 - \frac{1}{2}((\gamma - s\alpha_2)^2 k_{11} + \alpha_2^2 k_{22})$**(3.6)**

$$-\alpha_2 s(\gamma - s\alpha_2)k_{12} - (\gamma - s\alpha_2)y_1 v_1 - \alpha_2 y_2 v_2 + const$$

Une valeur optimale de α_2 est atteinte quand la dérivée de cette fonction objective s'annule.

$\frac{\partial \omega(\alpha_2)}{\partial \alpha_2}$ =0

$\frac{\partial \omega(\alpha_2)}{\partial \alpha_2} = -s + 1 - \frac{1}{2}[2(\gamma - s\alpha_2)(-s)k_{11} + 2\alpha_2 k_{22}]$

$$-s(\gamma - s\alpha_2)k_{12} - \alpha_2 s(-s)k_{12} - (-s)y_1 v_1 - y_2 v_2$$

$=1 - s + sk_{11} - \alpha_2 k_{11} - \alpha_2 k_{22}$

$$-s\gamma k_{12} + 2\alpha_2 k_{12} + y_2 v_1 - y_2 v_2$$

En égalisant cette dernière équation avec 0 , on obtient une nouvelle équation permettant d'obtenir le nouveau coefficient α_2 optimal :α_2^*

$$\alpha_2^*(k_{11} + k_{22} - 2k_{12}) = 1 - s + s\gamma(k_{11} - k_{12}) + y_2(v_1 - v_2)$$

$$= y_2(y_2 - y_1 + y_1\gamma(k_{11} - k_{12}) + v_1 - v_2)$$

En posant k=$k_{11} + k_{22} - 2k_{12}$ et en multipliant par y_2, il est possible de simplifier cette écriture pour obtenir :

$$\alpha_2^* k y_2 = y_2 - y_1 + y_1\gamma(k_{11} - k_{12}) + v_1 - v_2$$

Il s'agit maintenant d'éliminer les variables simplificatrices γ et v_1,v_2.pour ce faire la formulation de $f(x_i)$ est réécrite en fonction des v_i :

$$f(x_i) = \sum_{j=1}^{n} \alpha_j\, y_j k(x_j, x_i) + b$$

$$= \sum_{j=3}^{n} \alpha_j\, y_j k(x_j, x_i) + \sum_{j=1}^{2} \alpha_j\, y_j k(x_j, x_i) + b$$

$$= v_i + \sum_{j=1}^{2} \alpha_j\, y_j k(x_j, x_i) + b$$

Donc $v_i = f(x_i) - \sum_{j=1}^{2} \alpha_j\, y_j k(x_j, x_i) - b$**(3.7)**

Par conséquent

$$\alpha_2^* k y_2 = y_2 - y_1 + (y_1\alpha_1 + y_2\alpha_2)(k_{11} - k_{12}) + f(x_1) - \sum_{i=1}^{2} \alpha_i\, y_i k(x_i, x_1)$$

$$- \quad f(x_2) + \sum_{i=1}^{2} \alpha_i\, y_i k(x_i, x_2)$$

$$= y_2 - y_1 + f(x_1) - f(x_2) + y_2\alpha_2 k_{11} - 2y_2\alpha_2 k_{12} + y_2\alpha_2 k_{22}$$

$$= f(x_1) - y_1) - (f(x_2) - y_2) + y_2\alpha_2 k$$

$$\alpha_2^* = \frac{y_2\alpha_2 k}{k y_2} + \frac{(f(x_1) - y_1) - (f(x_2) - y_2)}{k y_2}$$

Ce qui donne finalement

$$\alpha_2^* = \alpha_2^{old} + \frac{(f(x_1) - y_1) - (f(x_2) - y_2)}{k y_2}$$

Ceci n'est évidemment valable que si k>0.dans le cas ou k=0,la division est impossible et il faut effectuer les calculs différemment.

En posant $E_i^{old} = f(x_i) - y_i$ la solution optimale pour le coefficient α_2 est

$$\alpha_2^* = \alpha_2^{old} + \frac{y_2(E_1^{old} - E_2^{old})}{k}$$

Valeur qui est ensuite tronquée pour satisfaire les contraintes de faisabilité :

Si k>0 , $\alpha_2 = \begin{cases} \alpha_2^* \ \ si\ L < \alpha_2^* < H \\ H \qquad\qquad si\ \alpha_2^* \geq H \\ L\ \ si\ \alpha_2^* \leq L \end{cases}$

Dans le cas ou k=0,cette valeur tend vers l'infini mais celle-ci devra toujours être bornée pour que $L \leq \alpha_2^* \leq H$. Étant donné qu'il s'agit d'une phase d'optimisation, la valeur qui sera retenue serra celle qui maximise la fonction objective.

Reprenons le calcul de la fonction objectif **(3.6)** afin de la réécrire en fonction de $E_1^{old}, E_2^{old}, \alpha_2^{old}\ et\ \alpha_2$

$$\omega(\alpha_2) = \gamma - s\alpha_2 + \alpha_2 - \frac{1}{2}((\gamma - s\alpha_2)^2 k_{11} + \alpha_2^2 k_{22}) - \alpha_2 s(\gamma - s\alpha_2)k_{12} - (\gamma - s\alpha_2)y_1 v_1 - \alpha_2 y_2 v_2 + const$$

$$= \alpha_2 - s\alpha_2 + \gamma - \frac{\gamma^2 k_{11}}{2} + cons - \frac{1}{2}(\alpha_2^2 k_{11} - 2\gamma s\alpha_2 k_{11} + \alpha_2^2 k_{22}) + \alpha_2^2 k_{12} - k_{12}\alpha_2 s\gamma - \gamma y_1 v_1 + s\alpha_2 y_1 v_1 - \alpha_2 y_2 v_2$$

Or $\gamma - \frac{\gamma^2 k_{11}}{2}$est une constante qui sera donc incluse dans la variable constante (const), donc

$w(\alpha_2) = \frac{1}{2}k\alpha_2^2 + \alpha_2(1 - s + s\gamma k_{11} - s\gamma k_{12} + y_2 v_1 - y_2 v_2)$+ const

Le calcul du second membre de $w(\alpha_2)$ est décomposable en plusieurs parties , d'après les équations **(3.5)**et **(3.7)** :

$$\gamma = \alpha_1^{old} + s\alpha_2^{old}$$

$$v_1 = f(x_1) - \alpha_1^{old} y_1 k_{11} - \alpha_2^{old} y_2 k_{21} - b^{old}$$

$$v_2 = f(x_2) - \alpha_1^{old} y_1 k_{12} - \alpha_2^{old} y_2 k_{22} - b^{old}$$

Ce qui permet de calculer d'une part :

$$s\gamma k_{11} - s\gamma k_{12} = s\gamma(k_{11} - k_{12})$$

$$= s\left(\alpha_1^{old} + s\alpha_2^{old}\right)(k_{11} - k_{12})$$

$$= \left(s\alpha_1^{old} + \alpha_2^{old}\right)(k_{11} - k_{12})$$

$$= s\alpha_1^{old}k_{11} + \alpha_2^{old}k_{11} - s\alpha_1^{old}k_{12} - \alpha_2^{old}k_{12}$$

Et d'autre part :

$$y_2v_1 - y_2v_2 = y_2(f(x_1) - \alpha_1^{old}y_1k_{11} - \alpha_2^{old}y_2k_{21} - b^{old}) - y_2(f(x_2) - \alpha_1^{old}y_1k_{12} - \alpha_2^{old}y_2k_{22} - b^{old})$$

$$= y_2f(x_1) - \alpha_1^{old}sk_{11} - \alpha_2^{old}k_{21} - y_2b^{old} - y_2f(x_2) + \alpha_1^{old}sk_{12} + \alpha_2^{old}k_{22} + y_2b^{old})$$

$$= s\alpha_1^{old}k_{12} + \alpha_2^{old}k_{22} - \alpha_1^{old}sk_{11} - \alpha_2^{old}k_{21} + y_2(f(x_1) - f(x_2))$$

Donc

$$1 - s + s\gamma k_{11} - s\gamma k_{12} + y_2v_1 - y_2v_2$$
$$= 1 - s + y_2\left(f(x_1) - f(x_2)\right) + \alpha_1^{old}sk_{11} + \alpha_2^{old}k_{11} + s\alpha_1^{old}k_{12} - \alpha_2^{old}k_{12} + s\alpha_1^{old}k_{12} + \alpha_2^{old}k_{22} - \alpha_1^{old}sk_{11} - \alpha_2^{old}k_{21}$$

$$= 1 - s + y_2\left(f(x_1) - f(x_2)\right) + \alpha_2^{old}k_{11} + \alpha_2^{old}k_{22} - 2\alpha_2^{old}k_{12}$$

Or $1 - s = y_1^2 - y_1y_2 = y_2^2 - y_1y_2$

Donc $1 - s + s\gamma k_{11} - s\gamma k_{12} + y_2v_1 - y_2v_2 = y_2\left(y_2 - y_1 + f(x_1) - f(x_2)\right) + k\alpha_2^{old}$

$$= y_2\left(E_1^{old} - E_2^{old}\right) + k\alpha_2^{old}$$

La fonction objectif est donc finalement

$$w\left(\alpha_2, E_1^{old}, E_2^{old}, \alpha_2^{old}\right) = \frac{1}{2}k\alpha_2 + \alpha_2\left(y_2\left(E_1^{old} - E_2^{old}\right) + k\alpha_2^{old}\right) + const$$

Comme annoncé antérieurement à ces calculs , cette équation va être utilisée pour deux valeurs extrêmes de α_2.

Ces deux valeurs porteront la dénomination de objectif L et objectif H

objectif $L = w(\alpha_2 = L, E_1^{old}, E_2^{old}, \alpha_2^{old})$

objectif $H = w(\alpha_2 = H, E_1^{old}, E_2^{old}, \alpha_2^{old})$

soit objectif $L = \frac{1}{2}kL + L\left(y_2\left(E_1^{old} - E_2^{old}\right) + k\alpha_2^{old}\right) + const$

objectif $H = \frac{1}{2}kH + H\left(y_2\left(E_1^{old} - E_2^{old}\right) + k\alpha_2^{old}\right) + const$

la valeur retenue pour α_2^* est alors choisie afin de maximiser la valeur de la fonction objectif.

Si k=0 , $\alpha_2^* \begin{cases} L \ si \quad \text{objectif } L > \text{ objectif } H \\ H \ si \ \text{objectif } L < \text{ objectif } H \\ \alpha_2^{old} \qquad si non \end{cases}$

- Calcul du coefficient d'ajustement b

Le lecture aura certainement remarqué la mention d'un coefficients b^{old} au cours des calculs précédents. Celui-ci est utilisé dans la formulation de $f(x)$:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \, k(x_i, x) + b$$

Soit $b = f(x) - \alpha_1 y_1 k(x_1, x) - \alpha_2 y_2 k(x_2, x) - \sum_{i=3}^{n} \alpha_i y_i \, k(x_i, x)$

Cette valeur dépendant des deux coefficients α_1 et α_2 qui sont ajustés au cours d'une itération de **SMO**, il est nécessaire de la mettre à jour elle aussi pour prendre en compte les changements effectués dans les coefficients de Lagrange.

Rappelons la définition de la mesure d'erreur appliquée aux anciennes et nouvelles valeurs desα

$$E_i^{old} = f^{old}(x_i) - y_i = \sum_{j=1}^{n} \alpha_j^{old} \, y_j k\left(x_j, x_i\right) + b^{old} - y_i$$

$$E_i = f(x_i) - y_i = \sum_{j=1}^{n} \alpha_j \, y_j k\left(x_j, x_i\right) + b - y_i$$

Ceci permet de calculer l'écart $\Delta E = E_i - E_i^{old}$

$$\Delta E = \sum_{j=1}^{n} \alpha_j \, y_j k\left(x_j, x\right) + b - y_i - \sum_{j=1}^{n} \alpha_j^{old} y_j \, k\left(x_j, x\right) - b^{old} + y_i$$

Notons de manière similaire $\Delta\alpha_1, \Delta\alpha_2$et Δb les changements respectifs des valeurs de $\alpha_1, \alpha_2 \, et \, b$.

Si pour une valeur donnée de l'ensemble d'apprentissage, on a $0 < \alpha_1 < C$,alors il s'agit d'un vecteur support. Les vecteurs supports sont les éléments délimitant les classes, ils sont

donc par nature bien classés. Donc $0 < \alpha_1 < \mathrm{C} => E_1 = 0$ dans ce cas il est possible de calculer :

$$\Delta b = -E_1^{old} - \Delta\alpha_1 y_1 k_{11} - \Delta\alpha_2 y_2 k_{21}$$

Il en va de même pour α_2 : $0 < \alpha_2 < \mathrm{C} => E_2 = 0$ donc

$$\Delta b = -E_2^{old} - \Delta\alpha_1 y_1 k_{12} - \Delta\alpha_2 y_2 k_{22}$$

Dans le cas ou les deux coefficients α ne se conforment pas à ces conditions , i.e. $(\alpha_1, \alpha_2) \in \{0, \mathrm{C}\}^2$,il est impossible de connaitre la nouvelle erreur commise aussi bien pour l'un que pour l'autre . pour néanmoins calculer la nouvelle valeur de b, la solution proposée dans la version originale de l'algorithme **SMO** et de calculer les deux valeurs en supposant que la nouvelle erreur est nulle et d'en prendre la moyenne.

$$b_1 = b^{old} - E_1^{old} - \Delta\alpha_1 y_1 k_{11} - \Delta\alpha_2 y_2 k_{21}$$

$$b_2 = b^{old} - E_2^{old} - \Delta\alpha_1 y_1 k_{12} - \Delta\alpha_2 y_2 k_{22}\mathrm{b}$$

$$b = \frac{b_1 + b_2}{2}$$

Finalement b est ajusté de façon suivante :

$$b = \begin{cases} b_1 & si\ 0 < \alpha_1 < \mathrm{C} \\ b_2 & si\ 0 < \alpha_2 < \mathrm{C} \\ \dfrac{b_1 + b_2}{2} & si\ (\alpha_1, \alpha_2) \in \{0, \mathrm{C}\}^2 \end{cases}$$

III.3. Environnement de travail

III.3.1. Présentation de PyQt

PyQt est la contraction de deux mots :

1. **Python** :le langage de programmation utilisé réputé fort simple d'apprentissage
2. **QT** :Un cadriciel extrêmement complet (principalement pour des interfaces graphiques)

a. Python

Python est un langage de programmation de haut niveau orienté objet, facile à apprendre, open source, extensible. Python est un langage interprété, polyvalent et largement utilisé dans de nombreux domaines, tels que la création de programmes autonomes à l'aide d'interfaces

graphiques et dans des applications Web, en plus d'être utilisé comme langage de script pour contrôler les performances de nombreux programmes tels que Blender. En général, Python peut être utilisé pour créer des programmes simples pour les débutants et pour réaliser de grands projets en même temps. Il est souvent recommandé aux débutants dans le domaine de la programmation d'apprendre ce langage car il fait partie des langages de programmation d'apprentissage les plus rapides.

Python a été développé à l'Institut néerlandais de mathématiques et d'informatique (CWI) à Amsterdam par Guido van Rossum à la fin des années quatre-vingt du XXe siècle, et sa première annonce remonte à 1991**[12]** . Le noyau du langage est écrit en langage C. Ross a appelé son langage "Python" pour exprimer son admiration pour une célèbre troupe de sketchs britannique qui s'appelait Monty Python.

b. **Qt Designer**

Qt Designer est l'outil Qt pour la conception et la construction d'interfaces utilisateur graphiques (GUI) avec les widgets Qt. Vous pouvez composer et personnaliser vos fenêtres ou boîtes de dialogues selon ce que vous voyez est ce que vous obtenez (WYSIWYG) et les tester en utilisant différents styles et résolutions**. [13].**

III.3.2. Les bases de données utilisées

Plusieurs bases de données contenant des informations qui permettent de classifier entre les classes que ce soit binaire ou non alors dans notre projet , nous nous occupons que des problèmes de classification binaire (2 classes) , nous avons choisi de tester nos programmes sur deux différents types de données , le premier type s'agit des données artificielles et le second concernant des données réelles.

a. **Les données Artificielles :**Ce sont des données établies par le Professeur Andrew NG dans le cadre du Cours de machine Learning en 2011
   - **DATA1 :**Ce sont des données bidimensionnelles qui peuvent être séparées par une frontière linéaire, Contient 51 échantillons.
   - **DATA2 :**Contient 863 échantillons. Nous allons créer un classificateur de machine vectorielle de support à l'aide du noyaux gaussiens (RBF) intégré et examiner sa précision sur les données d'entraînement**.**

- **DATA3 :** Contient 211 échantillons. Vous pouvez éviter qu'aucune limite de décision linéaire ne sépare les exemples positifs et négatifs pour cet ensemble de données. Cependant, en utilisant le noyau gaussien ou polynomial avec le SVM, vous pourrez apprendre une limite de décision non linéaire qui peut fonctionner raisonnablement bien pour l'ensemble de données. Si vous avez correctement implémenté la fonction noyau gaussienne.

b. **Les donnée Réelles**

- **Les iris de Fisher(150, 4,3) :**Les chiffres entre parenthèses indiquent respectivement, le nombre d'objets, le nombre d'attributs numériques et le nombre de classe, sont des données proposées en 1933 par le statisticien Ronald Aylmer Fisher comme données de référence pour l'analyse discriminante et la classification. Les données correspondent à 3 espèces de fleurs (Iris setosa, Iris virginica, Iris versicolor).Il est représenté par la longueur des sépales, largeur des sépales, longueur des pétales, largeur des pétales et sont mesurées en millimètres sur cinquante données de chacune des trois espèces, Pour notre travail, nous avons sélectionné que les données concernant deux classes (classe SETOSA et classe Versicolor) et nous avons choisi quarante échantillons de chacune des deux classes pour la phase d'apprentissage et dix échantillons de chacune des classes pour la phase du test**[14**].
- **Diabètes (768 ,8,2) :**Cet ensemble de données est à l'origine de l'Institut national du diabète et des maladies digestives et rénales. L'objectif de l'ensemble de données est de prédire de manière diagnostique si un patient est diabétique ou non, sur la base de certaines mesures diagnostiques incluses dans l'ensemble de données.Les ensembles de données se composent de plusieurs variables prédictives médicales et d'une variable cible(nommée Outcome) Les colonnes sont les suivantes :
  Prégnancies,Glucose,BloodPressure,SkinThickness,Insulin,BMI,DiabetesPedigreeFunction,Age **[15].**
  Pour notre travail, nous avons assignés 500 échantillons pour l'apprentissage et 200 pour le teste.
- L'ensemble de données **d'authentification des billets** comprend des caractéristiques de 1372 images de billets de banque en niveaux de gris, chacune de taille 400x400 pixels, qui ont été capturées à l'aide d'une caméra industrielle. Les caractéristiques ont été extraites des images après les avoir soumises à une transformation en ondelettes et

les caractéristiques extraites sont la variance (VAR), l'asymétrie (SKE), le kurtosis (KUR)et l'entropie (ENT) **[16].**Pour notre travail, nous avons assignés 300 échantillons pour l'apprentissage et 400 pour le teste.

III.3.3. La bibliothèque scikit-learn

On a utilisé une librairie pour Python spécialisée dans l'apprentissage automatique pour faire une comparaison avec la méthode **SMO** s'appeler **scikit-learn[17].** Le module qui implémente l'algorithme de classification du SVM est **sklearn.SVM.SVC.**

Le classificateur **SVC** est un classificateur binaire à la base (i.e. Il ne peut classifier qu'entre 2 classes à la fois), son principe de résolution du problème passe par la construction d'un hyperplan qui permet de séparer les deux classes

III.4. Résultats Expérimentaux

III.4.1. Précision et rappel

Les performances en termes de classification sont généralement mesurées à partir de deux indicateurs traditionnellement utilisés c'est les mesures du « rappel » et de « précision ».

Formellement, pour chaque classe Ci, on calcule deux probabilités qui peuvent être estimées à partir de la matrice de confusion correspondante ainsi ces deux mesures peuvent être définies de la manière suivante :

Le rappel : est la capacité à détecter les ”vrais positifs(VP)".

Rappel = VP/ VP + FN

La précision : est la capacité à éliminer les "vrai négatif (VN)".

Précision = VN/ VN + FP

Dans notre cas :

- VP ou vrai positif représente les exemples classés positifs et qui le sont vraiment.
- FP ou faux positif représente les exemples classés positifs et qui sont en fait négatifs.
- VN ou vrai négatif représente les exemples classés négatifs et qui le sont vraiment.

- FN ou faux négatif représente les exemples classés négatifs et qui sont en fait positifs.

A partir de ces définitions on peut faire le calcul de la performance :

Performance = (VP+VN)/(VP+FN+FP+VN)

III.4.2. Matrice de Confusion

Une matrice de Confusion est un résumé des résultats de prédictions sur un problème de classification. Les prédictions correctes et incorrectes sont mises en lumière et réparties par classe. Les résultats sont ainsi comparés avec les valeurs réelles**[18].**

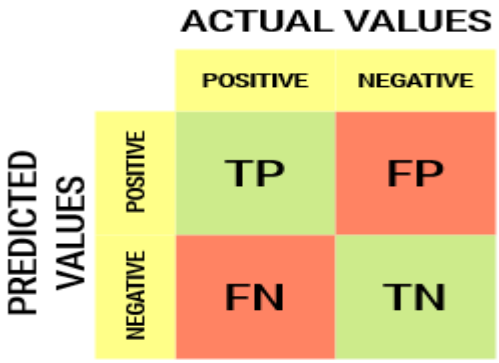


Figure III 1 : Exemple de matrice de Confusion[19].

III.4.3.Rapport de classification

Un rapport de classification est utilisé pour mesurer la qualité des prédictions à partir d'un algorithme de classification. Combien de prédictions sont vraies et combien sont fausses. Plus spécifiquement, les vrais positifs, les faux positifs, les vrais négatifs et les faux négatifs sont utilisés pour prédire les métriques d'un rapport de classification, comme indiqué ci-dessous.

Exemple 1 - Les résultats suivants concernant la base de données Diabètes:

Testing classification report

	precision	recall	f1-score	support
0	0.76	0.91	0.83	122
1	0.76	0.50	0.60	70
accuracy		0.76		192
macro avg	0.76	0.70	0.72	192

weighted avg	0.76	0.76	0.75	192

Exemple 2 - Les résultats suivants concernant la base de données Iris:

Training classification report

precision	recall	f1-score	support
-1	1.00	0.90	0.95	50
1	0.91	1.00	0.95	50
accuracy			0.95	100
macro avg	0.95	0.95	0.95	100
weighted avg	0.95	0.95	0.95	100

III.4.4.L'interface de l'application

Nous exposerons l'interface de notre application, L'Application se compose de **03 interfaces**, la première (01) interface comporte le **menu principal**, et la deuxième (02) interface pour l'**Apprentissage** et la troisième (03) interface pour le **Test.**

III.4.4.1.Première interface

Figure III 2: l'interface pour lancée l'application

Pour passer à la fenêtre suivante il suffit de cliquer dans la première interface sur le bouton Lancée.

III.4.4.2.Deuxième interface

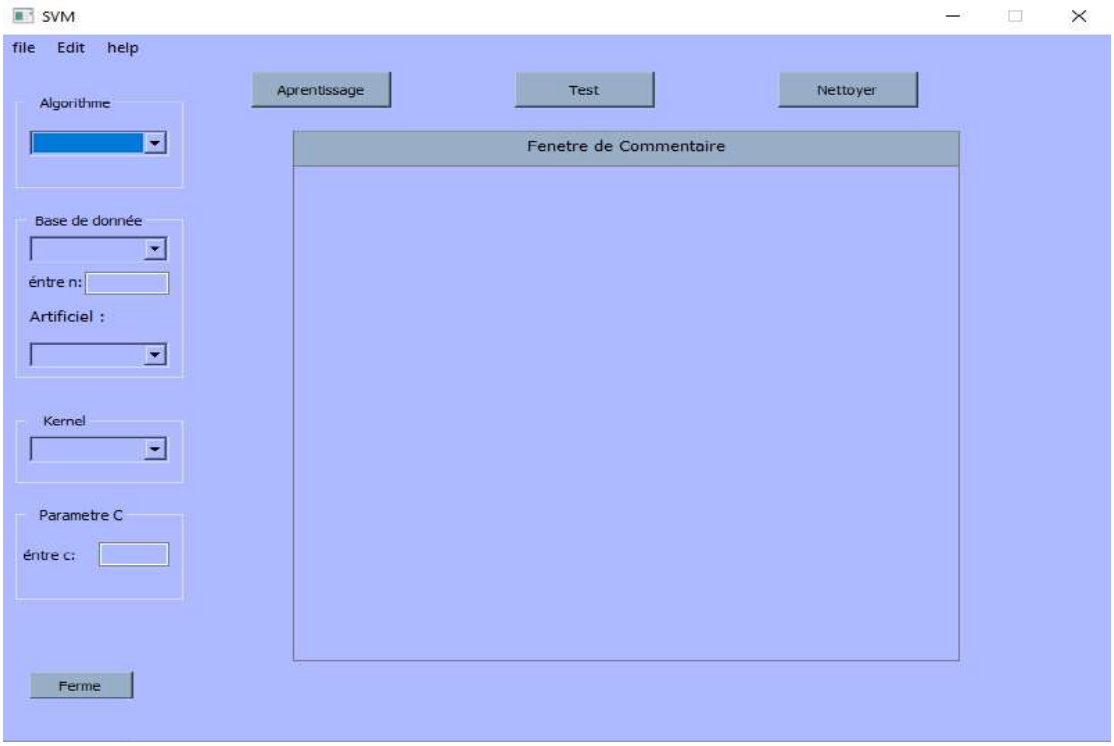


Figure III 3 :l'interface principale de l'application

Cette interface contient un menu et des boutons même aussi **4**boites de sélections pour choisir les algorithmes **(SMO** ou **SVC**) et une autre boite de sélection pour choisir une base de donnée, et la troisième pour choisir en cas les donnée artificiel et la fin pour le Kernel (linear,poly,RBF) ,puis les 2 boites de texte pour le paramètre **n** c'est le nombre d'attributs ,et l' autre pour le paramètres constant-C

Le bouton **Test** nous permet d'afficher une nouvelle fenêtre à travers laquelle l'utilisateur puisse entrer les valeurs des attributs après avoir choisi la base de donnée.

Le bouton **Nettoyer permet de** supprimer les résultats affichés lors du dernier apprentissage.

Cette fenêtre contient aussi une petite **fenêtre de commentaires** à travers laquelle des statistiques seront affichés après chaque application d'apprentissage sur les données d'entrée pour chaque méthode sélectionnée.

Le bouton **Ferme** permet de quitter l'application.

III.4.4.3. Troisième interface

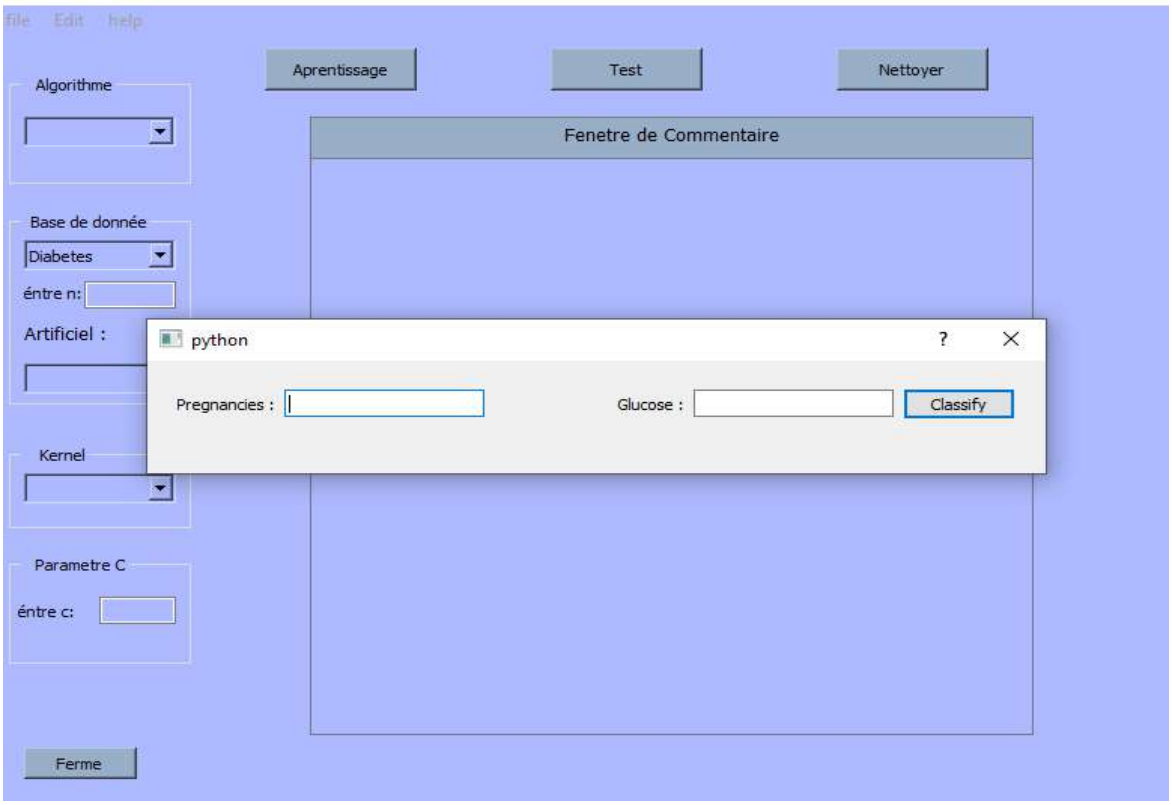


Figure III 4 : l'interface de dialogue

Pour tester , on choisit d'abord le base de donnée puis on clique sur le bouton **test** afin que ce dernier nous affiche une fenêtre de dialogue , pour pouvoir entrer nos attributs et finalement on clique sur le bouton **classify** pour afficher la classe correspondante de l'élément que nous venons de tester.

III.4.5. Les résultats pour les données Artificielles

Dans cette première partie nous allons présenter nos résultats expérimentaux sur divers exemples de jeux de données 2D. Expérimenter avec ces ensembles de données nous aidera à acquérir une intuition sur le fonctionnement des **SVM** et comment utiliser un noyau gaussien et un noyau RBF avec des **SVM**.

III.4.5.1.DATA1

La première chose que nous allons faire est d'examiner un simple ensemble de données en 2 dimensions et de voir comment un **SVM** linéaire fonctionne sur l'ensemble de données pour des valeurs variables de **C**

En tracera les données d'entraînement (Figure III 5). Dans cet ensemble de données, les positions des exemples positifs (indiqués par + rouge) et des exemples négatifs (indiqués par **o** bleu)

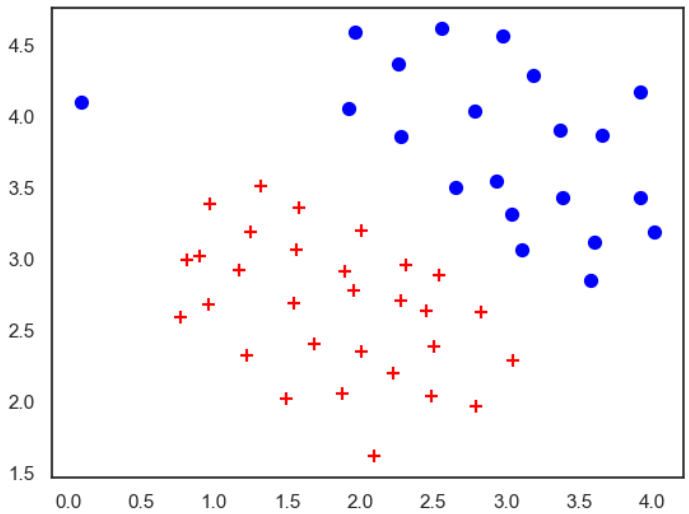


Figure III 5 :DATA1

Dans cette partie, Nous allons essayer d'utiliser différentes valeurs du paramètre **C** avec des **SVM**.le paramètre **C** est une valeur positive qui contrôle la pénalité pour les exemples de formation mal classés.

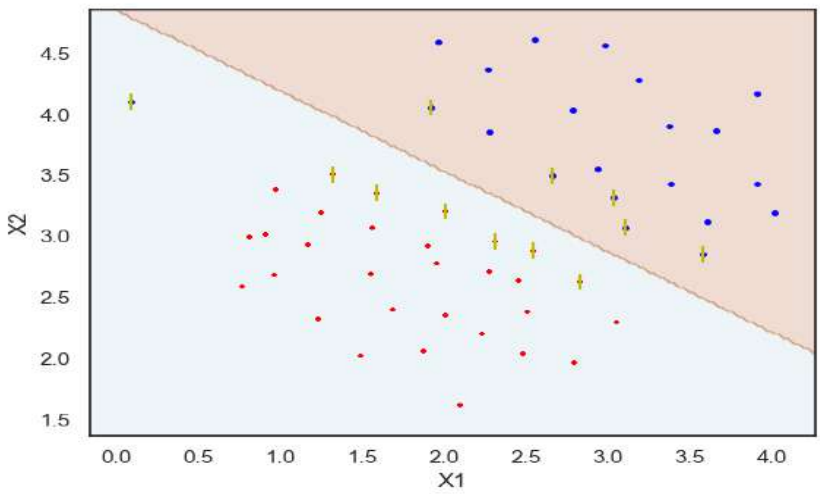
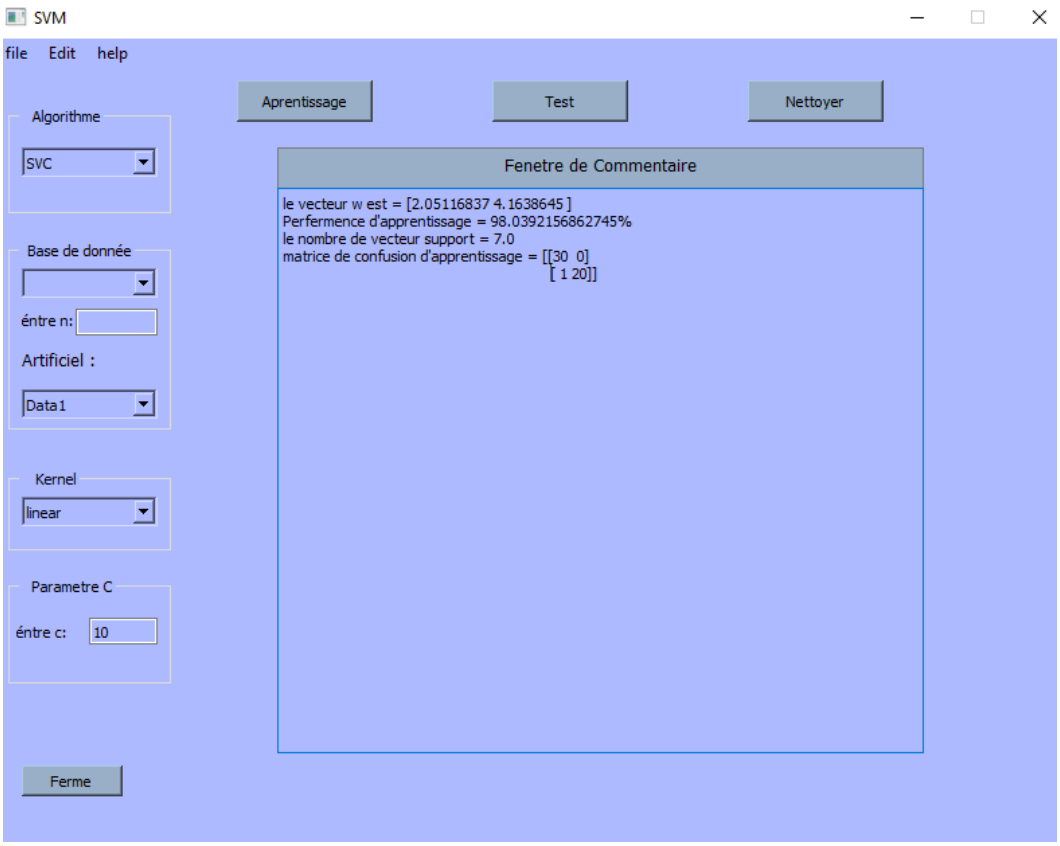


Figure III 6 : Limite de décision SVM avec C=10



Remarque : au niveau de la matrice de confusion, nous avons **FN**=1 ,c'est le seul point bleu qui est placé du côté des points rouges.

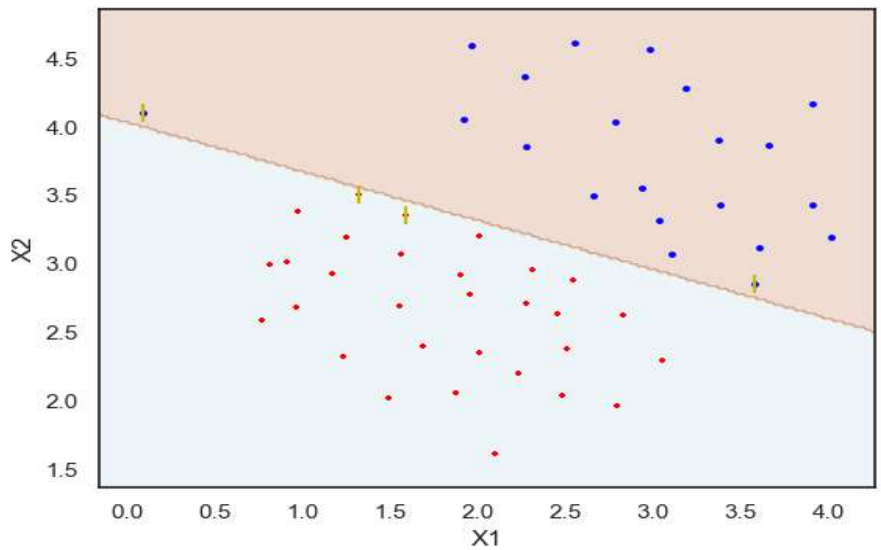
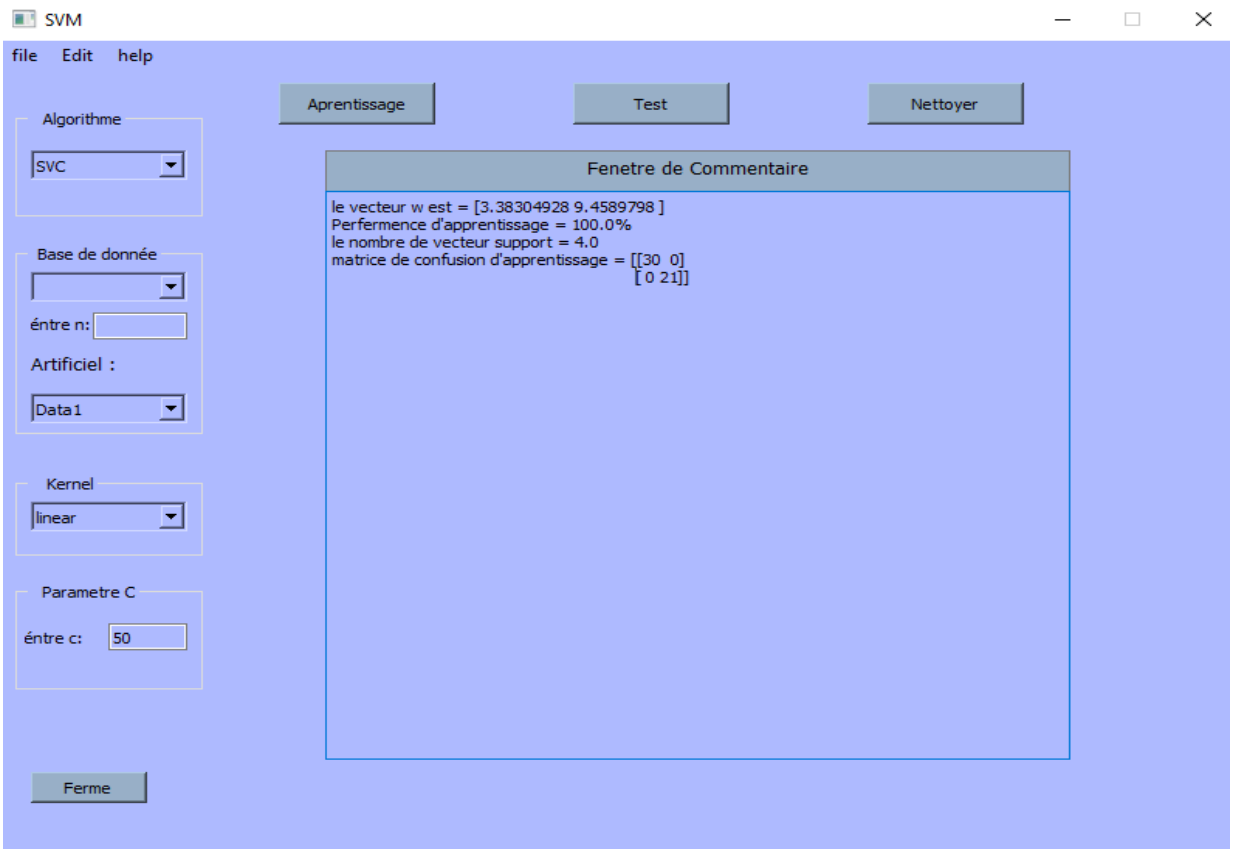


Figure III 7 : Limite de décision SVM avec C=50



Remarque : au niveau de la matrice de confusion, nous avons maintenant **FN**=0, le seul point bleu qui est du coté des points rouges cette fois ci il est placé avec les points bleus, donc ya aucun point qui est mal classé et la performance est égale à 100%

Lorsque **C** = 10, on trouve que le **SVM** place la limite de décision dans l'espace entre les deux ensembles de données et classe mal le point de données à l'extrême gauche (***Figure III 7***)

Alors le paramètre **C** affecte la classification des données, plus sa valeur est élevée, plus on aura un apprentissage par cœur.

III.4.5.2.DATA2

Dans cet ensemble de données nous utilisons un **SVM** pour effectuer une classification non linéaire. En particulier, On choisit un noyau gaussien sur des ensembles de données qui ne sont pas linéairement séparables**. (*Figure III 8*)**.

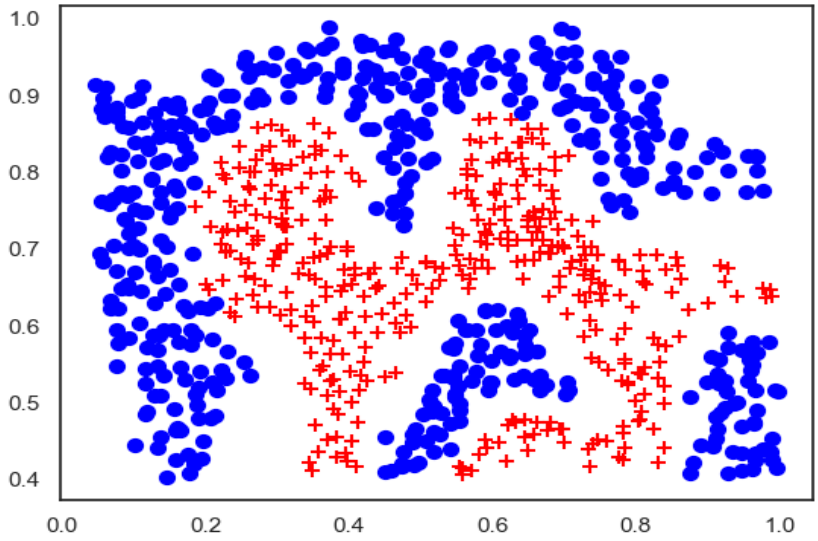


Figure III 8 :DATA2

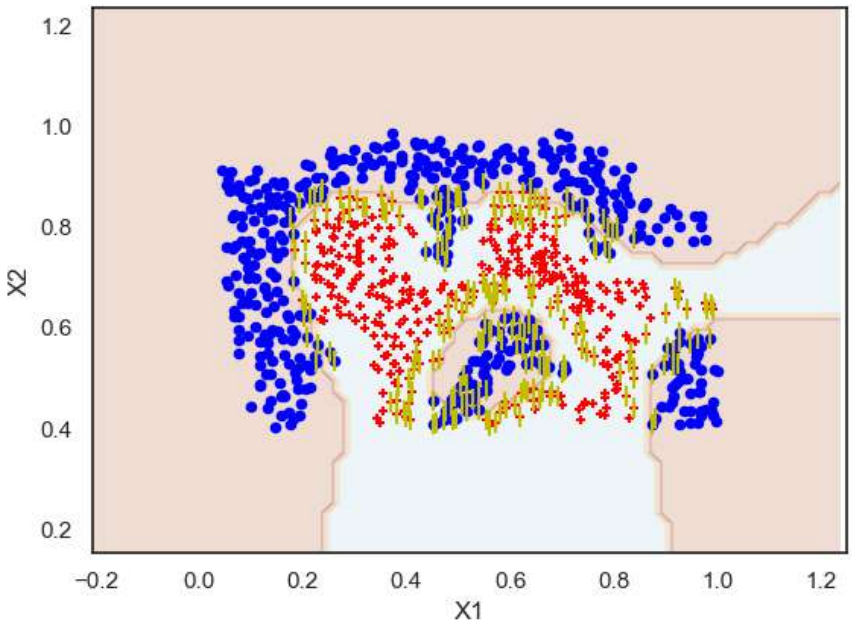


Figure III 9 : .Limite de décision SVM (noyau gaussien)

(***Figure III 9***) montre la frontière de décision trouvée par le **SVM** avec un noyau gaussien. La frontière de décision est capable de séparer correctement la plupart des exemples positifs et négatifs et suit bien les contours de l'ensemble de données.

III.4.5.3.DATA3

Dans cet ensemble de données ,un noyau linéaire(une droite) ne peut pas convenir , donc on peut choisir entre un noyau Gaussien ou polynomial.

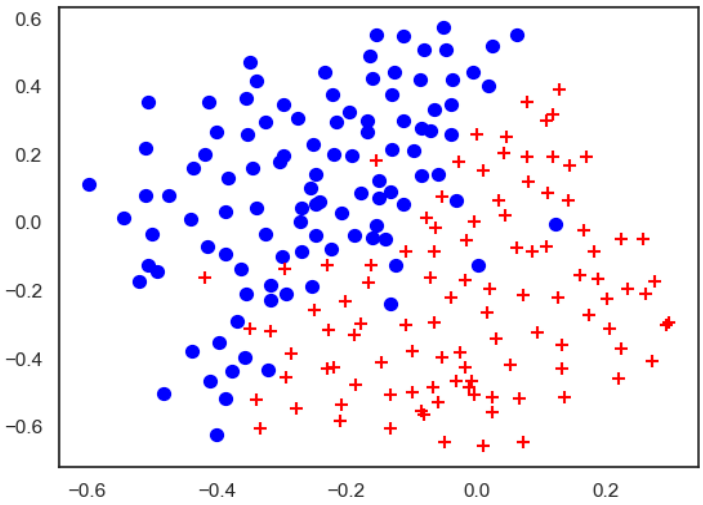


Figure III 10 :DATA3

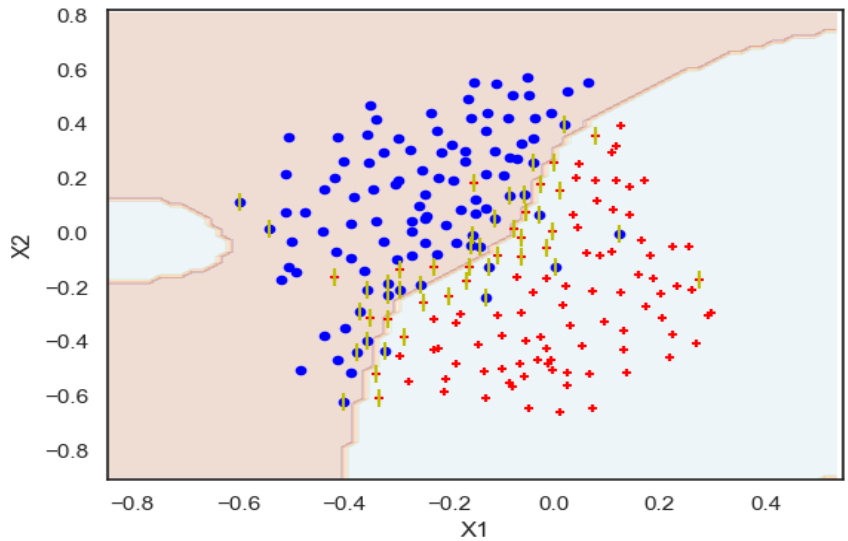
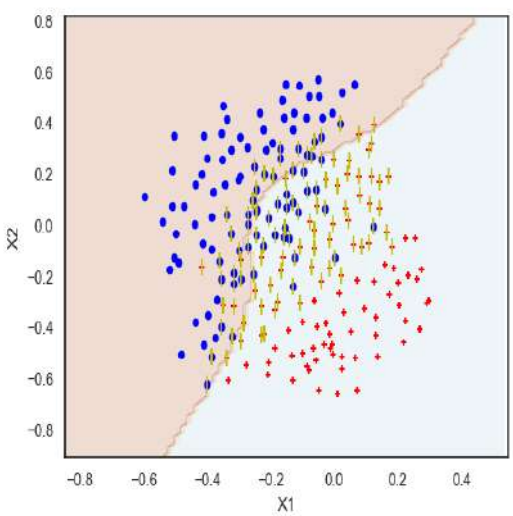
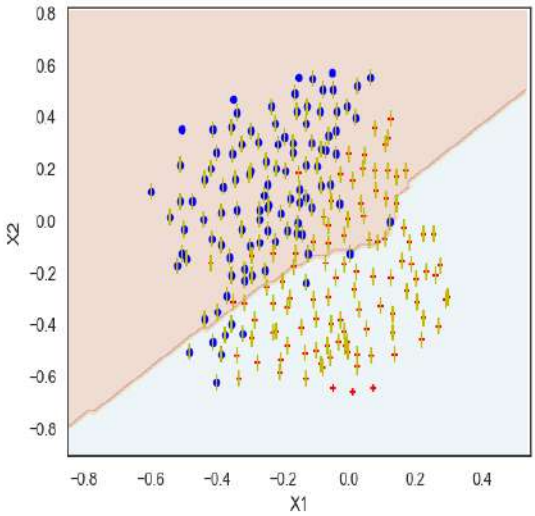


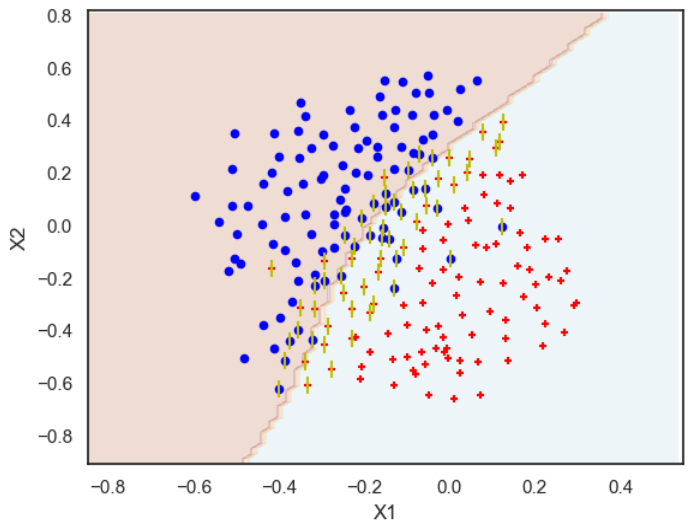
Figure III 11 : Limite de décision SVM (noyau gaussien)



C=0.1



C=0.001



C= 10

Figure III 12 : : Limite de décision SVM (noyau poly)

Remarque :

Lorsque **C** = 0.1 et au niveau de la matrice de confusion, nous avons maintenant **FP**=7 ,**FN**=25 il y a32 points qui sont mal classés et la performance est égale à 84%***(Figure III 12)***

Lorsque **C**= 0.001,et au niveau de la matrice de confusion, nous avons maintenant **FP**=33, FN=9il y a42 points qui sont mal classés et la performance est égale à 80% (***Figure III 12***)

Lorsque **C** = 10,et au niveau de la matrice de confusion, nous avons maintenant **FP**=5 ,**FN**=14 il y a19 points qui sont mal classés et la performance est égale à 90% (***Figure III 12***) , c'est le meilleur Cas .

Pour cela, nous remarquons que lorsque nous prenons une valeur plus grande pour le paramètre C, nous constatons que le nombre des points mal classés diminue, et donc la performance d'apprentissage s'améliore. Par contre si le paramètre C prend une très petite valeur (Exemple C=0.001) c'est-à-dire nous avons toléré beaucoup d'erreurs (points mal classés en tenant compte uniquement de la maximisation de la marge , on voit ici que la performance en généralisation se dégrade(80%)

III.4.6.Les résultats pour les données Réelles :

III.4.6.1.La base iris :

L'apprentissage de la base de donnée « **iris** » pour la méthode **SMO** avec 2 attributs et C=0.1 est comme suit :

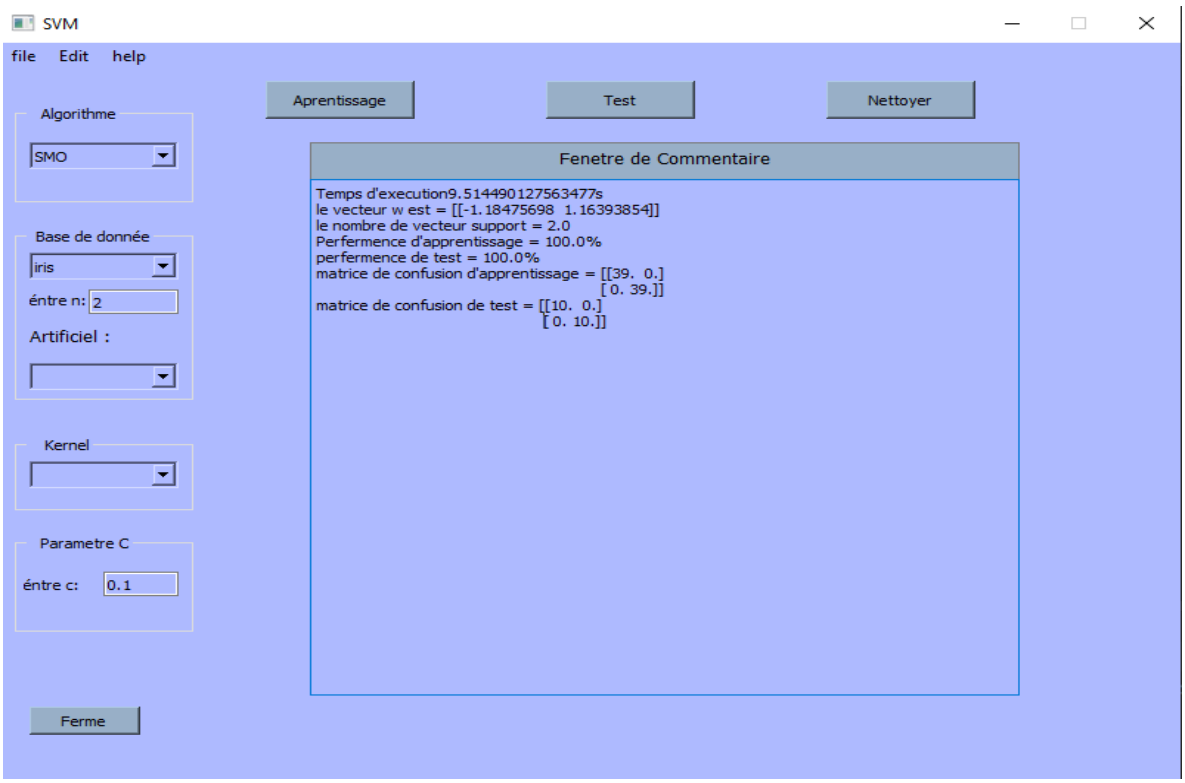
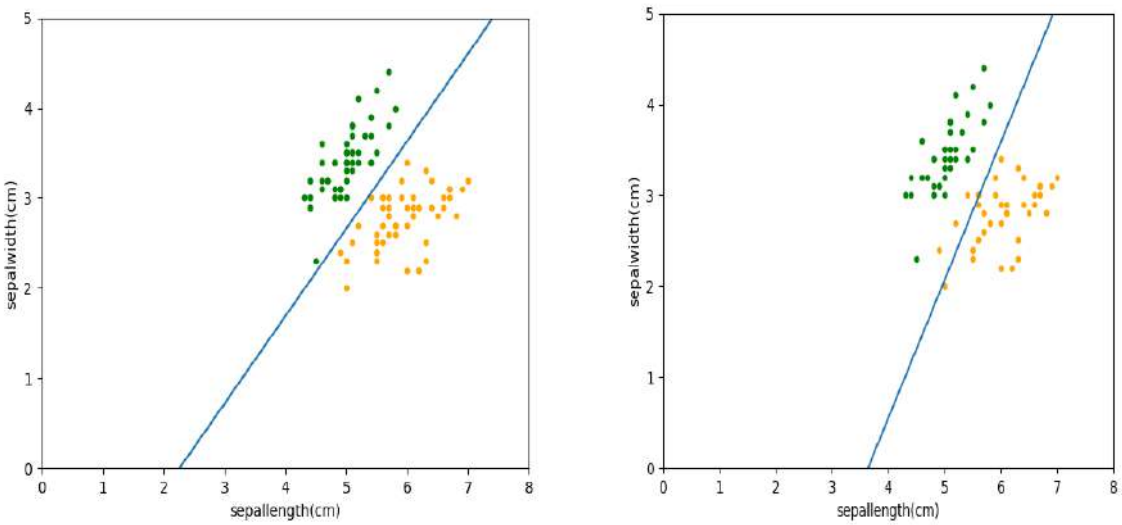


Figure III 13 : Résultat d'apprentissage pour le data « iris »



C=0.1 **C=0.001**

Figure III 14: le plot de le data « iris »

Remarque :

Lorsque **C** = 0.1 ,et au niveau de la matrice de confusion, nous avons maintenant **FP**=0,il ya aucun point qui est mal classé et la performance est égale à 100%(***Figure III 13 ,Figure III 14***)

Lorsque **C** = 0.001, on trouve que le **SVM** place la limite de décision dans l'espace entre les deux ensembles de données et classe mal les points de données à l'extrême droit la performance est égale à 93% (***Figure III 14***)

L'apprentissage de la base de donnée « iris » pour la méthode **SVC** avec 2 attributs et **C**=0.1 est comme suit :

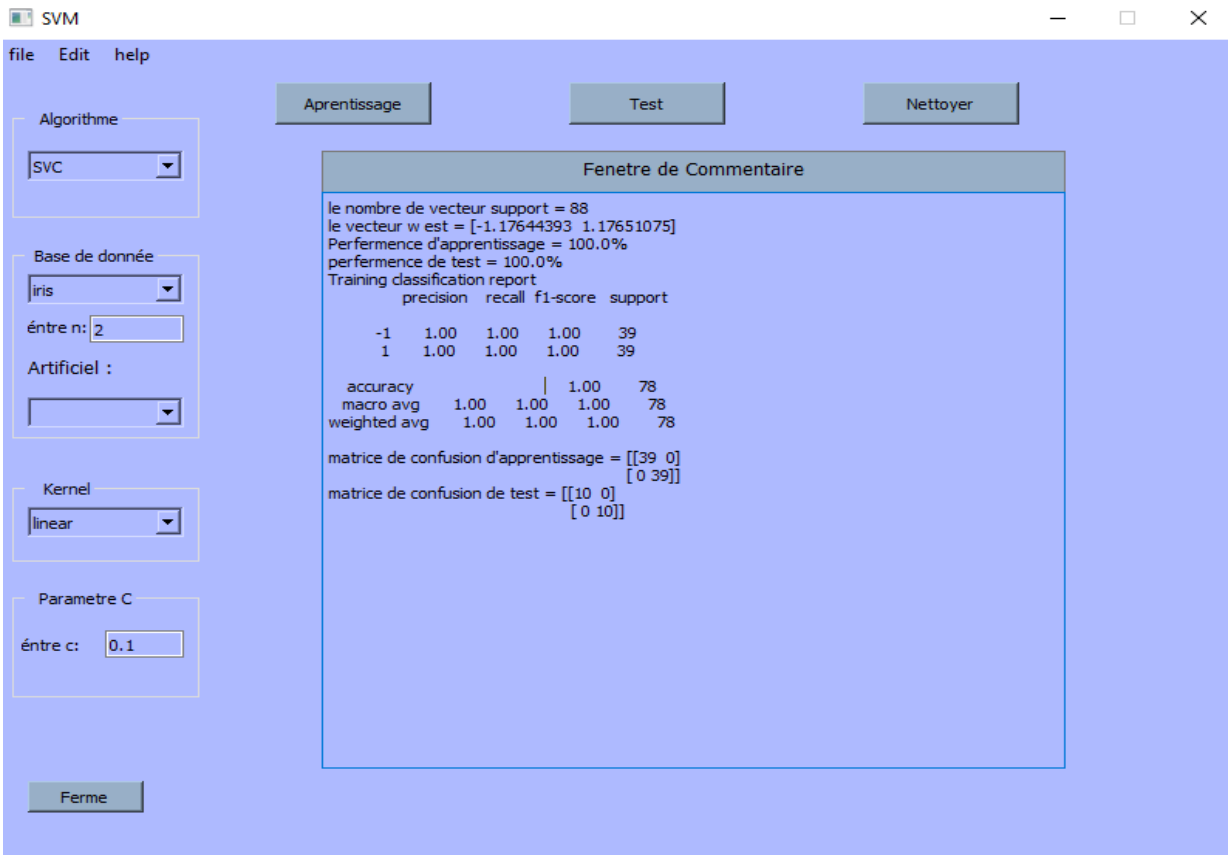


Figure III 15: Résultat d'apprentissage pour le data « iris » par la méthode SVC

Tableau III 1 : la comparaison du résultat de la base iris

	Nombre de vecteur support	Performance d'apprentissage	Performance de Test	Le vecteur W
SMO	2	100%	100%	2
SVC	88	100%	100%	2

III.4.6.2.La base Billet authentication

Le Résultat de l'apprentissage de la base de donnée « billet » pour la méthode **SMO** avec **4** attributs et **C=0.1** est le suivant :

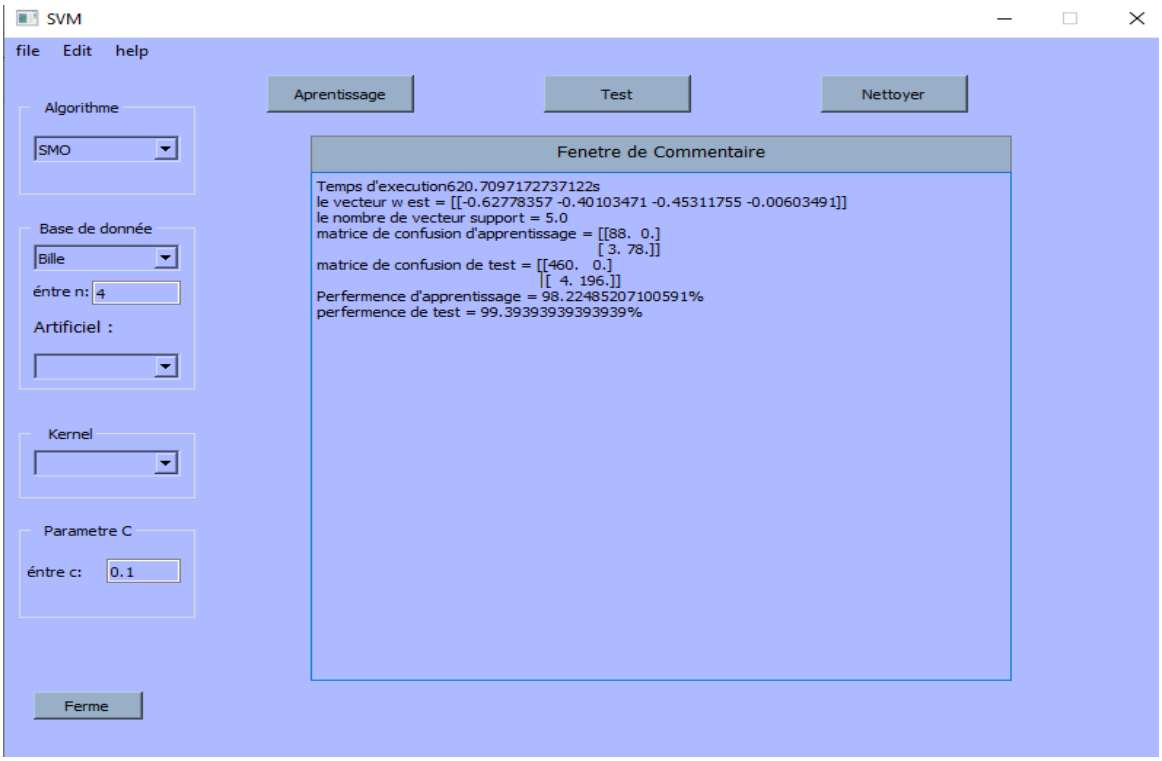


Figure III 16: Résultat d'apprentissage pour les données « billet » par SMO(n=4 c=0.1)

Le Résultat de l'apprentissage de la base de donnée « billet » pour la méthode **SMO** avec **2**attributs et C=**0.1** est le suivant :

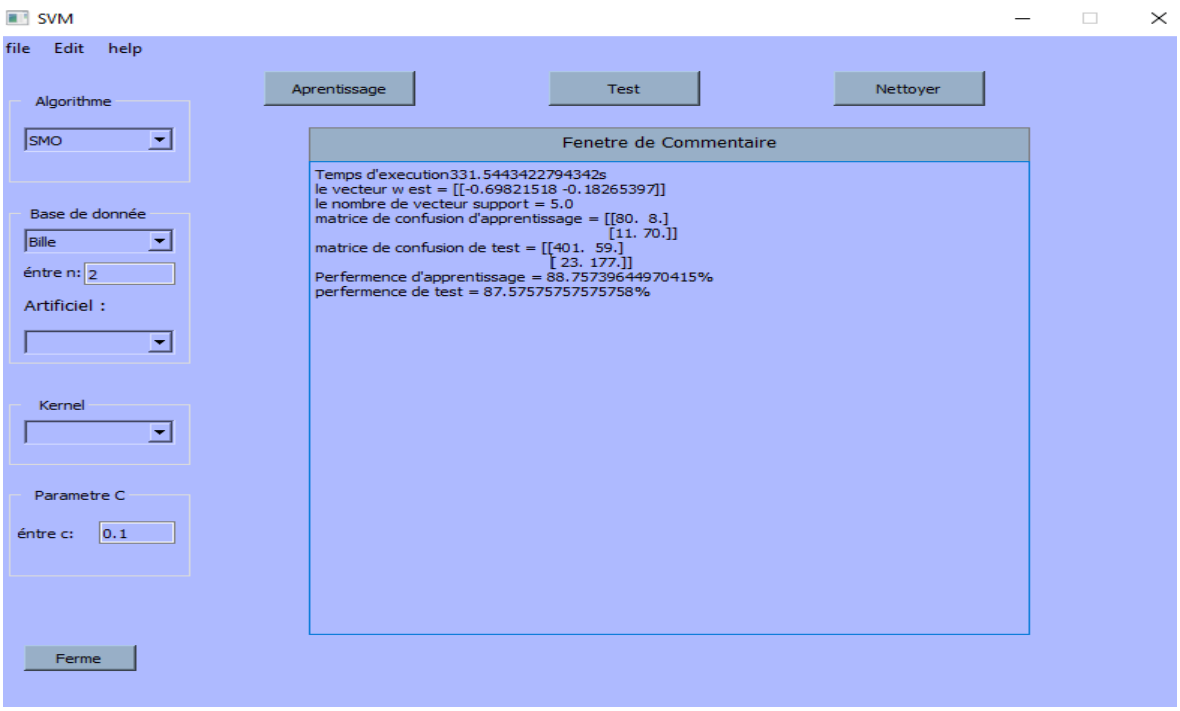


Figure III 17: Résultat d'apprentissage pour les données « billet » par SMO(n=2 c=0.1)

Interprétation des Résultats du data set billet authentication :

Selon les Résultats affichés dans les deux figures ci-dessus, on voit que en augmentant le nombre d'attributs la performance est beaucoup plus meilleure que ce soit en apprentissage ou en phase de **Test .**

n = 4, c =0.1 **la performance d'apprentissage** = 98.22% et celle du test = 99.24% (Figure III 16), alors dans la (Figure III 17) n=2 c = 0.1 **la performance d'apprentissage** =88.75% et celle du test = 87.57%

Tableau III 2 : la comparaison du résultat de la base « billet » par SMO

	Nombre de vecteur support	Performance d'apprentissage	Performance de Test	Le vecteur W
N=2	4	88.77%	87.57%	2
N=4	5	98.22%	99.24%	4

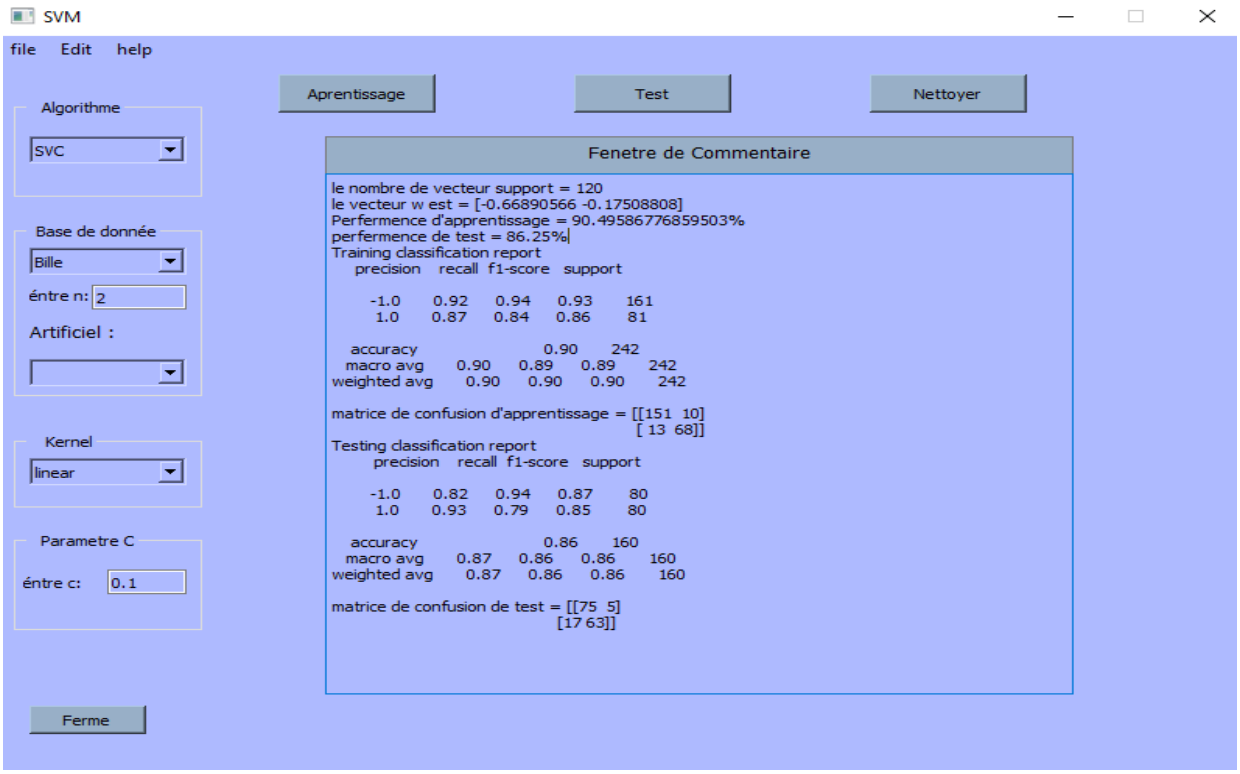


Figure III 18: Résultat d'apprentissage pour la donnée « billet » par SVC

Grâce au résultat obtenu en modifiant la valeur de **n**, nous concluons que le nombre d'attribut d'entrée affecte la performance d'apprentissage.

Tableau III 3 : la comparaison du résultat de la base « billet »par SMO et SVC

	Nombre de vecteur support	Performance d'apprentissage	Performance de Test	Le vecteur W
SMO	4	88.77%	87.57%	2
SVC	84	90.47%	86.25%	2

III.4.6.3.La base diabètes :

L'apprentissage de la base de donnée « Diabètes » pour la méthode **SMO** avec **2** attributs et **C**=0.1 est comme suit :

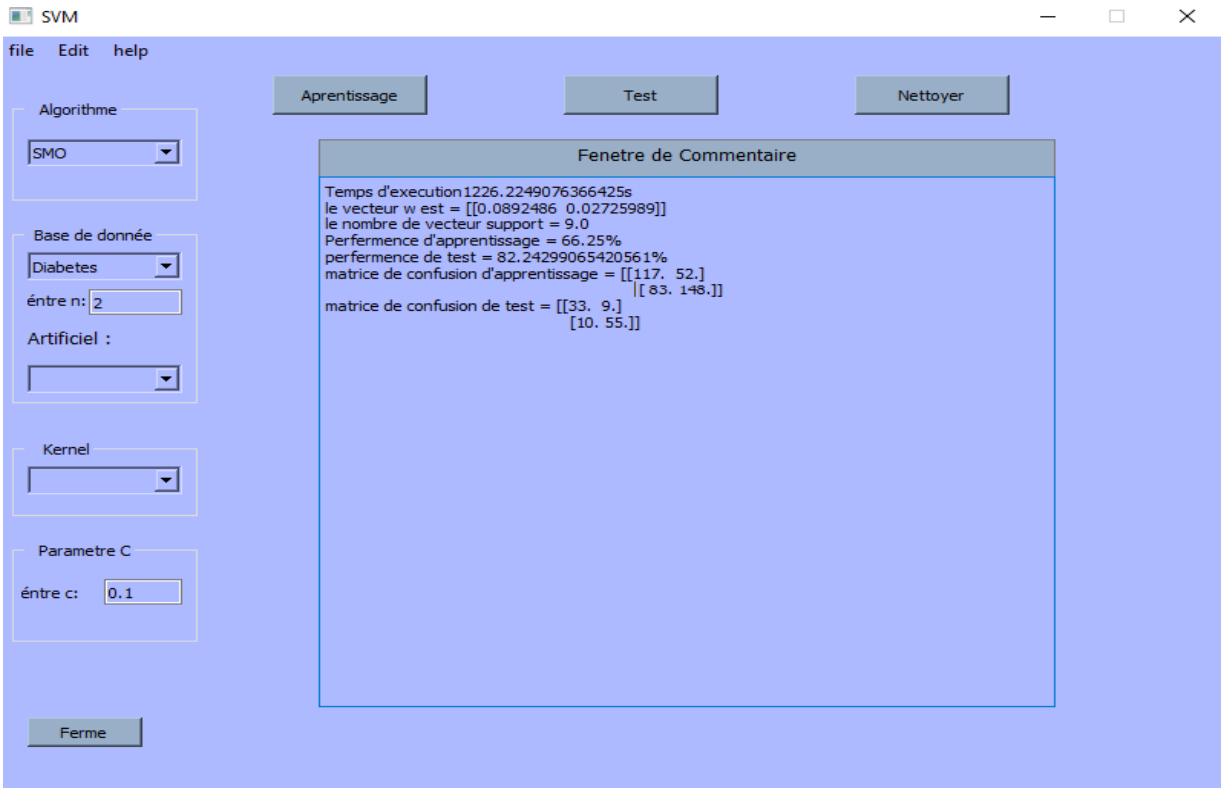


Figure III 19: Résultat d'apprentissage pour la donnée « diabètes » avec SMO

L'apprentissage de la base de donnée « Diabètes » pour la méthode **SVC**avec 2 attributs et **C**=0.1est comme suit :

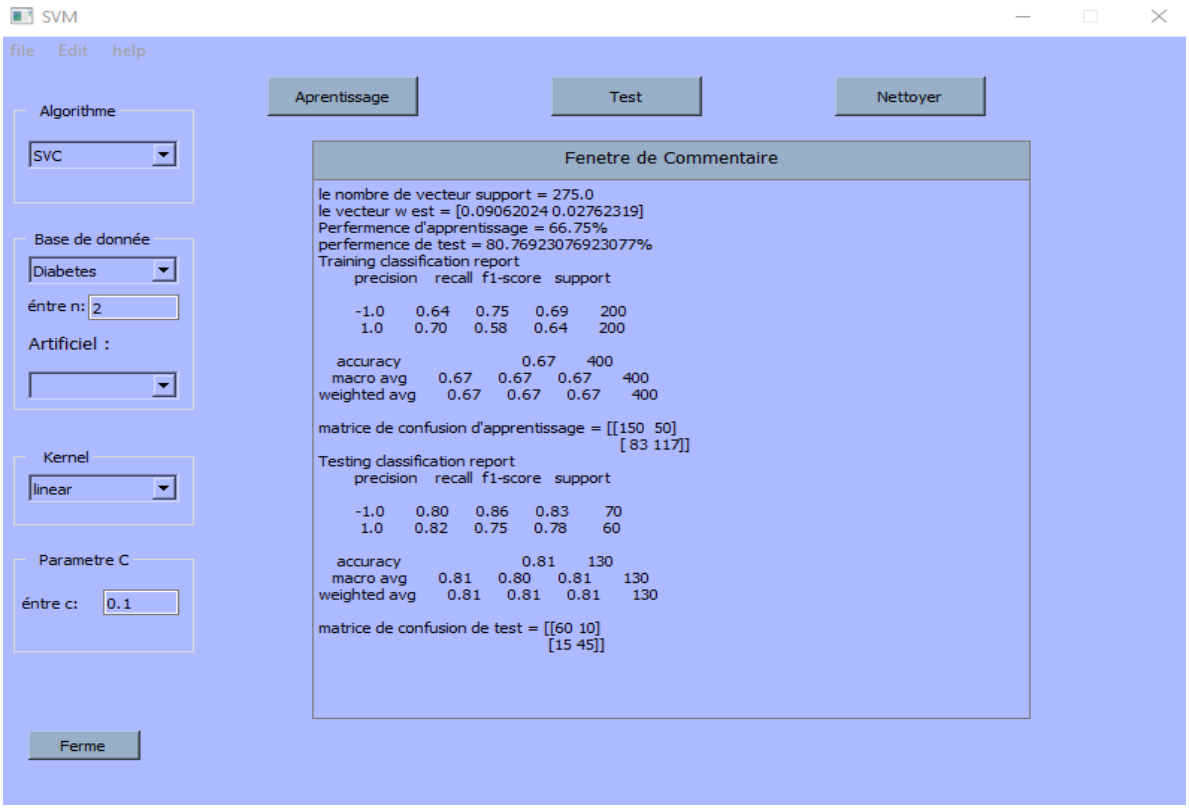


Figure III 20: Résultat d'apprentissage pour la donnée « diabètes » avec SVC

Interprétation des résultats

D'après les résultats présentés dans le(**Figure III 19)**concernant le classifieur basé sur notre implémentation de l'algorithme **SMO** la performance de test est de l'ordre de 82.24% , elle est meilleure que celle du classifieur obtenu par la méthode **SVC** (**Figure III 20**) qui est de l'ordre de 80.76% , mais le temps écoulé lors de l'apprentissage de la méthode **SMO** est beaucoup plus long que celui de la méthode **SVC** à cause du nombre élevé d'itérations puisque à chaque itération ya que deux multiplicateurs qui seront optimisés.

III.4.7.Résultat du test du classifieur :

Dans cette fenêtre on a sélectionné la base « diabètes »en cliquant sur le bouton test pour pouvoir entrer les valeurs des deux attributs « Prégnances » et « Glucose » en fin en cliquant sur le bouton classifier pour afficher la classe(diabétique ou non diabétique)

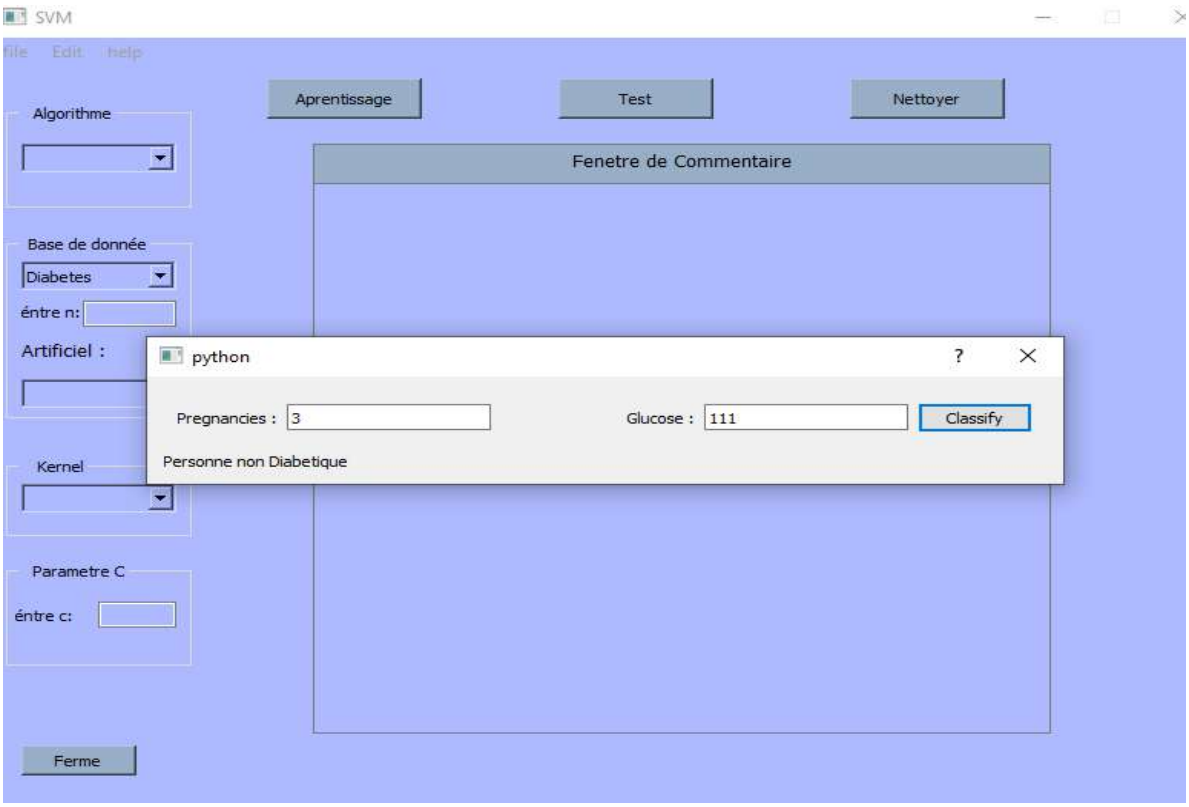


Figure III 21: l'interface pour teste la base de donnée « Personne non diabétique »

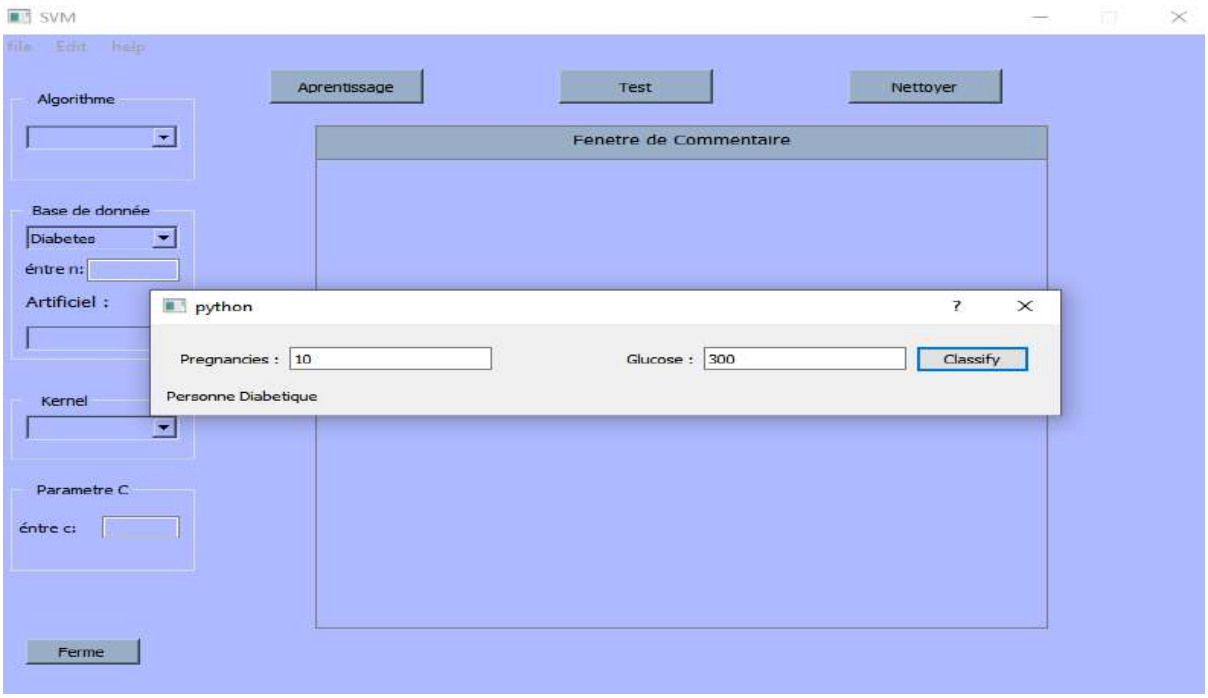


Figure III 22: l'interface pour le test de la base de donnée « Personne diabétique »

Si on choisi le nombre d' attributs =3 en cliquant sur le bouton test pour pouvoir entrer les valeurs des trois attributs « Prégnances » et « Glucose »et « Blood Pressure »on affichier la suivant :

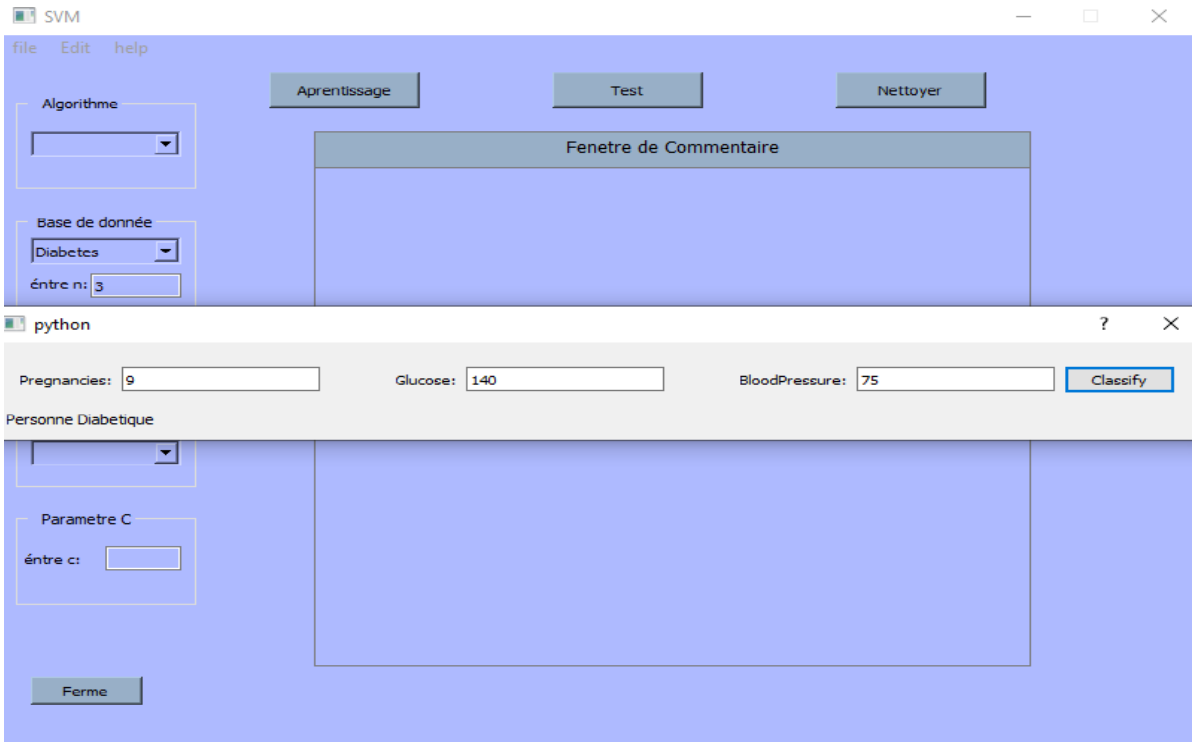


Figure III 23l'interface pour le test de la base de donnée par n=3« Personne diabétique »

Figure III 24 l'interface pour le test de la base de donnée par n=3«Personne non diabétique »

Quand on choisi la base « iris »on va introduire les valeurs des attributs comme suit :

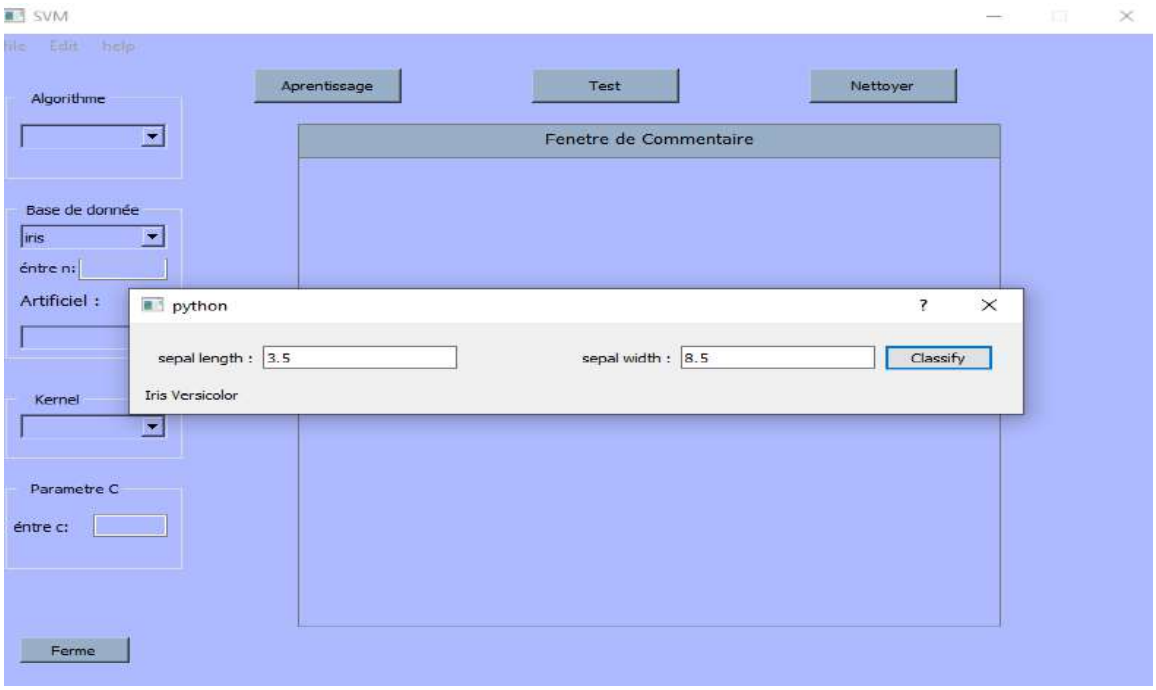


Figure III 25l'interface pour le test de la base de donnée «iris Versicolor »

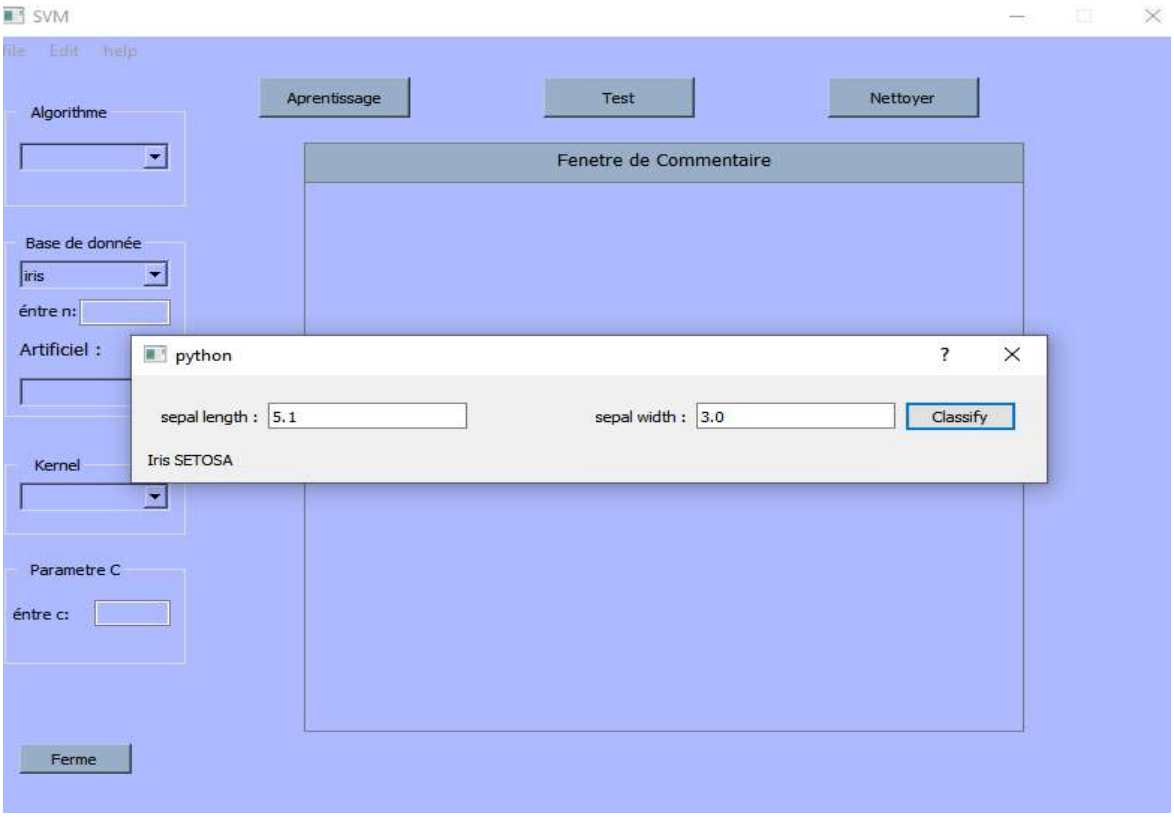


Figure III 26 l'interface pour le test de la base de donnée «iris SETOSA »

III.5. Conclusion

Ce chapitre met en avant la prudence nécessaire à la comparaison de méthodes de résolution. En effet l'ordre de grandeur des hyper paramètres est un facteur influant sur la rapidité de résolution et sur sa performance en généralisation. Nous pouvons dire que les deux algorithmes ont pu montrer leurs performances en un temps d'exécution négligeable pour les données de cardinalités moyennes, mais dès que la cardinalité augmente ils montrent leurs limites du point de vu temps d'exécution. Ce qui est très logique puisque la résolution du problème d'optimisation se fait sur tous les exemples de la base d'apprentissage. Enfin, nous pouvons dire que tous les fondements théoriques sur lesquels les **SVMs** s'appuient ont été prouvés à travers cette expérimentation en prenant comme exemple le paramètre de régularisation **C** qui est un compromis entre la maximisation de la marge et la minimisation des erreurs (points mal classés) c'est-à-dire si **C** prend des valeurs très petites on tient compte de la maximisation de la marge et on peut trouver plusieurs exemples mal classifiés (outliers) à l'intérieur de la marge et ça c'est le phénomène du surapprentissage et dans le cas ou **C** prend des valeurs assez grandes c'est-à-dire on donne une grande importance à la minimisation des erreurs (on accepte pas les outliers) et ça c'est le phénomène de l'apprentissage par cœur , donc plusieurs tests ont été effectués afin de trouver la bonne valeur du paramètre **C** dont l'objectif c'est d'avoir une bonne performance en généralisation .

Conclusion Générale et Perspectives

La réalisation d'un programme d'apprentissage par SVM se ramène à résoudre un problème d'optimisation impliquant un système de résolution dans un espace de dimension conséquente. L'utilisation de ces programmes revient surtout à sélectionner une bonne famille de fonctions noyau et à régler les paramètres de ces fonctions. Ces choix sont le plus souvent faits par une technique de validation croisée, dans laquelle on estime la performance du Système en la mesurant sur des exemples n'ayant pas été utilisés en cours d'apprentissage.

Dans notre travail, nous avons traité de la classification automatique de nombreuses bases de données, y compris normales et autres. Nous avons également vu normal, comme Diabètes Qui contient 768 personnes, y compris les blessés et les non infectés, et nous avons alloué 400 personnes pour la phase d'apprentissage, et pour la phase de test 200 personnes. D'autre part, nous avons la base iris , qui en contient 50 de chaque classe, nous en avons donc attribué 20 à la phase de test et le reste à l'entraînement. Enfin nous avons la base billet comprend des caractéristiques de 1372 images de billets de banque, nous avons assignés 300 échantillons pour l'apprentissage et 400 pour le teste.

Et nous avons testé les 3 bases de données artificielles dont les données sont linéairement séparables et d'autres non linéairement séparables et pour ces derniers nous avons choisi les noyaux(polynomial et Gaussien(RBF)).

On a utilisé L'algorithme **SMO** a une bonne complexité calculatoire, que nous la voyons logique puisque tous les échantillons de l'apprentissage sont totalement chargés en mémoire et nous a donné une performance en généralisation de plus de 90%. Nous avons utilisé un classificateur **SVC**(module en python qui implémente l'algorithme de classification du **SVM) dont l'objectif est de** faire une comparaison avec notre implémentation de l'algorithme SMO.

A travers nos résultats, nous constatons que la valeur du paramètre de régularisation **C** et le **nombre de caractéristiques** ou attributs ont une influence très importante pour l'obtention d'une bonne performance en généralisation, nous avons consacré beaucoup de temps pour le choix de ces paramètres vus que nous avons choisi une sélection manuelle c'est-à-dire on fixe dans notre implémentation la valeur du **C** et celle du **nombre d'attributs** et selon nos résultats on refait d'autres sélections.

et concernant le temps d'exécution écoulé pendant la phase d'apprentissage nous trouvons que l'algorithme **SMO** prend beaucoup plus de temps que celui de la méthode **SVC surtout pour les bases de taille assez grande et de dimension aussi importante** parce que la méthode SMO fait plusieurs itérations à cause de la décomposition du problème d'optimisation en dessous problèmes de deux inconnues(multiplicateurs de Lagrange).

Et comme perspectives on propose :

- La sélection automatique du paramètre C, on peut choisir une structure de données qui nous permet d'enregistrer les performances en généralisation pour différentes valeurs de C, ensuite on détermine la valeur de C qui nous retourne la performance en généralisation la plus grande.
- Concernant les bases de données de grande dimension, afin de minimiser le temps réservé à l'apprentissage, on peut penser à appliquer la méthode d'analyse en composantes principales (ACP)afin de réduire cette dimension.
  Ou bien on peut aussi sélectionner les attributs pertinents (plus représentatifs) parmi la totalité des attributs d'une manière automatique.

Résumé

Les machines à vecteurs supports (Support Vector Machines) sont des techniques d'apprentissage statistique proposées par V. Vapnik en 1995. Elles permettent d'aborder des problèmes très divers comme la classification, la régression. L'idée de cette technique consiste à projeter les données de l'espace d'entrée (appartenant à deux classes différentes) non-linéairement séparables dans un espace de plus grande dimension appelé espace de caractéristiques de façon à ce que les données deviennent linéairement séparables.

L'objectif de notre travail est la résolution des problèmes de classification par les machines à vecteurs supports. l'entrainement d'un SVM consiste à résoudre le problème d'optimisation quadratique convexe. Des techniques standards de programmation quadratiques telle que la méthode du gradient conjugué ou la méthode des points intérieurs, … peuvent résoudre le problème du SVM mais pour les problèmes de grandes taille (nombre des échantillons élevé ou une grande dimension) ces méthodes deviennent inenvisageables. Il ya cependant d'autres techniques dédiées aux SVM qui servent à décomposer le problème d'optimisation en dessous problèmes de petites tailles , on peut citer l'algorithme de Joachims (Implémentation SVMLight) et l'algorithme d'optimisation séquentielle minimale (SMO) de John Platt , nous avons choisi d'implémenter l'algorithme SMO et de comparer nos résultats avec une méthode standard de programmation quadratique sur deux différents types de données :données artificielles et données réelles.

Mots clés : Apprentissage automatique, Machines à vecteurs supports, Classification

ملخص

آلات المتجهات الداعمة هي تقنيات تعليمية إحصائية اقترحها في. فابنيك في عام 1995. وهي تجعل من الممكن معالجة مشاكل متنوعة للغاية مثل التصنيف والانحدار. تتمثل فكرة هذه التقنية في إسقاط بيانات مساحة الإدخال (التي تنتمي إلى فئتين مختلفتين) قابلة للفصل بشكل غير خطي إلى مساحة أكبر الأبعاد تسمى الفضاء المميز بحيث تصبح البيانات قابلة للفصل خطيًا....

الهدف من عملنا هو حل مشاكل التصنيف عن طريق دعم آلات المتجهات. تدريبSVM هو حل مشكلة التحسين التربيعية المحدبة. تقنيات البرمجة التربيعية القياسية مثل طريقة التدرج المترافق أو طريقة النقاط الداخلية ... يمكن أن تحل مشكلةSVMولكن بالنسبة لمشاكل الحجم الكبير (عدد كبير من العينات أو بعد كبير) تصبح هذه الطرق غير واردة. ومع ذلك، هناك تقنيات أخرى مخصصة لSVMs تعمل على حل مشكلة التحسين في ظل مشاكل الحجم الصغير ، يمكننا الاستشهادبخوارزميةjoachims(لتنفيذSVMLight) وخوارزمية التحسين التسلسلي الأدنى لجون بلات (SMO) ، اخترنا تنفيذ الخوارزمية SMOومقارنة نتائجنا مع طريقة البرمجة التربيعية القياسية على نوعين مختلفين من البيانات: البيانات الاصطناعية والبيانات الحقيقية.

الكلمات الرئيسية: تعلم الآلة ، دعم آلات المتجهات ، التصنيف

Abstract

Support vector machines are statistical educational techniques suggested by V. vapnik in 1995. It makes it possible to address very diverse problems such as classification and regression. The idea of this technique is to project the input space data (which belong to two different classes) non-linearly separable into a larger dimensional space called the distinct space so that the data becomes linearly separable ....

The aim of our work is to solve classification problems by supporting vector machines. SVM training is to solve the convex quadratic optimization problem. Standard quadratic programming techniques such as conjugated gradient method or internal point method ... can solve the SVM problem but for problems of large size (large number of samples or large dimension) these methods become out of the question. However, there are other techniques dedicated to SVMs that solve the optimization problem under problems of small size, we can cite joachims algorithm (to implement SVMLight) and John Platt's Minimal Sequential Optimization algorithm (SMO), we chose to implement the SMO algorithm and compare our results with the standard quadratic programming method on two different types From data: synthetic data and real data.

Keywords: machine learning, vector machine support, classification

Reference

[1] ghanjati, c. (17/02/2020). Pas d'Intelligence sans Apprentissage. Nouvelle aquitaine science et culture

[2] metomo joseph bert, r. r. (2017, 10 10). supinfo international university. Apprentissage Automatique

[3] daniel, g. (2018, 3 6). HIMSS 2018: Perspectives on Health Industry use of AI and Machine Learning. Récupéré sur inside big data

[4] 2019). Dans B. G. JEAN-HERVÉ LORENZI, Livre Blanc IA et Technologies Quantiques, FINANCE. RB

[5] Hawarah, L. (22/10/2008). Une approche probabiliste pour le classement d'objets.

[6] vert, J. p. (2002, 7 17). support vector machines in bioinformatics. Human Genome Center University, tokyo JAPAN.

[7] DJEFFAL, a. (2012). cours support vector machine. Récupéré sur site personnel abdelhamid

[8] RAKOTOMALALA, R. (s.d.). Discrimination linéaire. université lumière lyon 2.

[9] V.N. Vapnik. Statistical Learning Theory. Edition Wiley, 1998.

[10] J. C. Platt, 1998, Fast Training of Support Vector Machines using Sequential Minimal Optimization, SMO-Book, chap. 12, p. 41-65, In B.

[11] Schölkopf, C. J. C. Burges, A. J. Smola, editors, Advance in Kernel Methods - Support Vector Learning, Cambridge, MA, 1998, MIT Press.

[12] Hasan, M. (16.01.2006). *Machines à vecteurs de support.* FRANCE.

[13] Rossum, G. V. (2009, 1 20). The history of python . *Abrief timeline of python*.

[14] Nebra, M. (2020, 6 3). *Programmez avec le langage c++:modélisez ses fenetre avec QT designe.*

[15] Abdenour, S. (2011). *Détection des défauts dans les moteurs asynchrones par*

[16] Anand, S. (s.d.). *Exploratory Data Analysis on Diabetes dataset.* Récupéré sur DATA DRIVEN

[17] BEIESP. (s.d.). *Innovative technology and exploring engineering(TM)*.

[18] Rakotomalala, R. (s.d.). *Intallation et gestion des package sous R*

[19] BHANDARI, A. (2020, 4 17). *Everything you Should Know about Confusion Matrix for Machine Learning.*Récupéré sur Analytics Vidhya

[20] https://www.ionos.fr/digitalguide/web-marketing/analyse-web/quest-ce-que-lapprentissage-automatique/

[21] https://mrmint.fr/algorithme-k-means

[22] *A.Ben ishak , « Selection de variables par les machines a vecteurs supports pour la discrimination binaire et multiclasse en grande dimension ».These de doctorat de l'universite de Tunis , 2007*

[23] *S.Khellat Kihel, « Les séparateurs à vaste marge Bi- classe »,Université des sciences et de technologie d'Oran, exposé de Master2, 2012*

[24] *V.N. Vapnik. The nature of statistical learning theory. Springer, 2000*

[25] *O. Bousquet :" Introduction aux ' Support Vector Machines '(SVM) ". Centre de Mathématiques Appliquées Ecole Polytechnique, Palaiseau, Orsay, Novembre 2001.*

[26] *Mohamadally Hasan,Fomani Boris :" SVM machine à vecteurs de support ou séparateur à vaste marge ". BD Web, ISTY3,Versailles St Quentin, France, janvier 2006.*

[27] *A. Shigeo. Support Vector Machines for Pattern Classification. Springer-Verlag London Limited, 2005*

[28] N. Cristianini and J. Shawe-Taylor. An Introduction to Support Vector Machines and ther Kernel-Based Learning Methods. Cambridge University Press, 2000.

[29] S. Knerr, L. Personnaz, J. Dreyfus, et al. Single-layer ll learning revisited : A stepwise procedure for building and training a neural network. Optimization Methods and Software, 1 :23–34, 1990.