## Université of Science et Technologies HOUARI BOUMEDIENE Département d'informatique



APPRENTISSAGE AUTOMATIQUE

ET RESEAUX DE NEURONNES

# **« RAPPORT TP 7 »**

|  |  |  |  |
| --- | --- | --- | --- |
| *NOM* | *PRENOM* | *MATRICULLE* | *GROUPE* |
| ***BELIMI*** | ***Ibrahim Sabri*** | ***161631074255*** | ***A1 TP1*** |
| ***ZAIT*** | ***Fouad*** | ***181831072145*** | ***A1 TP1*** |
| ***ASMA*** | ***SRAOUIA*** | ***161631102642*** | ***A1 TP1*** |

***OBJECTIFS***

Dans ce TP, nous aimerions explorer le classifieur machine à vecteurs de support. Nous essayerons d'utiliser plusieurs noyaux et voir la différence entre leur capacité de séparation.

Pour ceci, nous utiliserons plusieurs ensembles de données.

On essayera de séparer nos données avec un SVM linéaire et un SVM avec un noyau non linéaire.

# **Définition**:

# Les machines à vecteurs de support (SVM) sont un ensemble de méthodes d'apprentissage supervisé utilisées pour la classification, la régression et la détection d’anomalies.

# SVM est utilisé comme classificateur linéaire ou non linéaire basé sur le noyau utilisé. Si nous utilisons un noyau linéaire, alors le classificateur et donc la limite de prédiction sont linéaires. Ici, pour séparer deux classes, nous devons tracer une ligne. La ligne est telle qu’il y a une **marge maximale**. Cette ligne est tracée à égale distance des deux ensembles. Nous dessinons deux autres lignes de chaque côté, appelées vecteurs de support. Les SVM apprennent des vecteurs de support, contrairement aux autres modèles d’machine learning qui apprennent à partir des données correctes et incorrectes. **Les avantages des machines à vecteurs de support**: Performant, non biaisé par les valeurs aberrantes, insensible au surajustement.

# **Les désavantages des machines à vecteurs de support :**

# Ne convient pas aux problèmes non linéaires, pas le meilleur choix pour un grand nombre de caractéristiques.

**Importation des librairies nécessaires pour le travail**



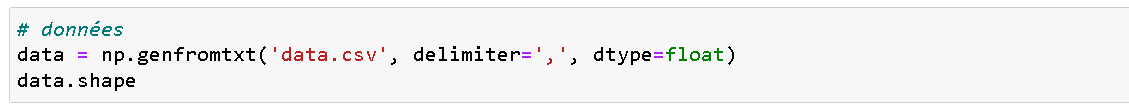
***numpy :*** pour la manipulation des tableaux (array)

***matplotlib :*** pour la manipulation des graphs (2d, 3d)

***SKLEARN :*** pour utiliser les outils de la machine learning pour utilliser la fonction prédefinni SVM

**Lecture des fichiers de données pour les classifier**

**Code :**



**Résultat :**

===> 

***863:*** signifie le nombre de données (ou exemples) dans le fichier (***data.csv***)

***3:*** représente 3 colonnes dans le fichier

**(les 2 premières colonnes c’est des caractéristiques X 1 X2 et la 3eme c’est nos étiquettes ).**

**Remarque :**

**Dans les SVM on n’a pas besoin de rajouter la 1ere colonne X0 =1 pour tous les exemples .**

Dans ces données (data), les 2 premières colonnes représentent les caractéristiques, la dernière colonne représente la classe de l'exemple (positive ou négative).

Chaque ligne représente un exemple de notre ensemble de données.

Mettons ces données dans leurs vecteurs correspondants.

# 

# **Entrainement**

Nous utiliserons ici une MVS avec different noyaux:

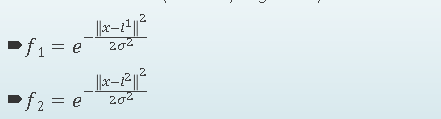
* Linéaire
* Gaussien (RBF)
* Polyomial

On va utiliser dans ce qui suit la fonction prédéfini de svm en lui donnant en paramétres C ==1 qui est égale à 1/lambda . Le choix de la valeur de C est très important pour éviter un sur apprentissage ou sous apprentissage Si on donne lambda petit on risque de tomber sur un sur apprentissage par contre si on donne lambda trop grand on risque de tomber sur un sous apprentissage . Donc pour C sa sera le contraire car C=1/lambda lorsque C est grand on tombe sur un sur apprentissage et pour C trop petit on tombe sur un sous apprentissage .

rbf\_svc = svm.SVC(kernel='rbf', gamma=0.7, C=C).fit(X, y).

Rbf\_svc est un noyau linéaire utilisé dans les machines à vecteurs de support .(transformation de x1 et x2 avec le noyau gaussien).

Elle aura pour rôle d’appliquer la fonction du noyau gaussien vu en cours :



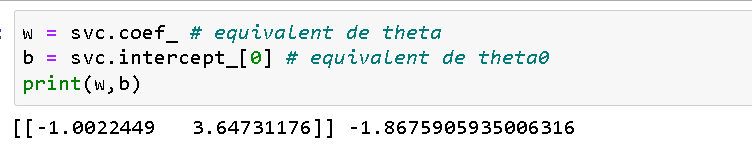
Gamma =0,7 est l’ecart type de notre gaussiénne σ .

poly\_svc = svm.SVC(kernel='poly', degree=3, C=C).fit(X, y).

Poly\_svc pour élever les degrés de nos caractéristique (en un polynome).

Ensuite on appelle la fonction fit sur les 2 noyau pour entrainer nos données .

Mettons un lien avec ce que nous avons appris en régression linéaire

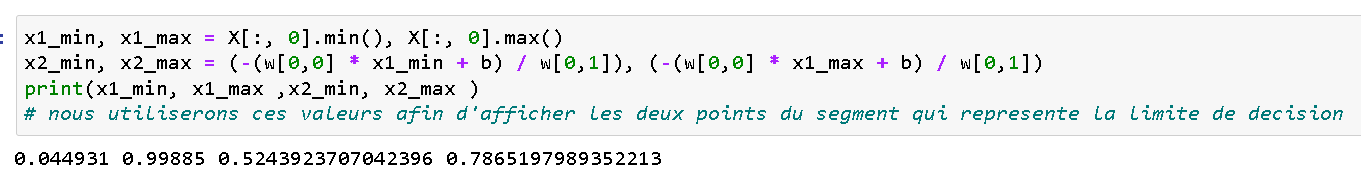


w représente nos theta 1 , tehta 2 et b représente theta 0.

**Decision Boundary :**

Pour afficher la decision boundary on va devoir écrire x2 en fonction de x1 en mettant l’equation theta0+theta1x1+theta2x2 =0 et la l’afficher comme vu dans les tps précédents .

X2=(-theta1X1 + theta0 )/theta 2



on va choisir deux points pour dessiner notre ligne x1\_min x1\_max qui sont le minimum et le maximum de x1 et x2\_min x2\_max qui sont les minimum et les maximum de x2. On aura donc 2 points pour dessiner la ligne séparatrice .

1er point :

X1\_min=0,04

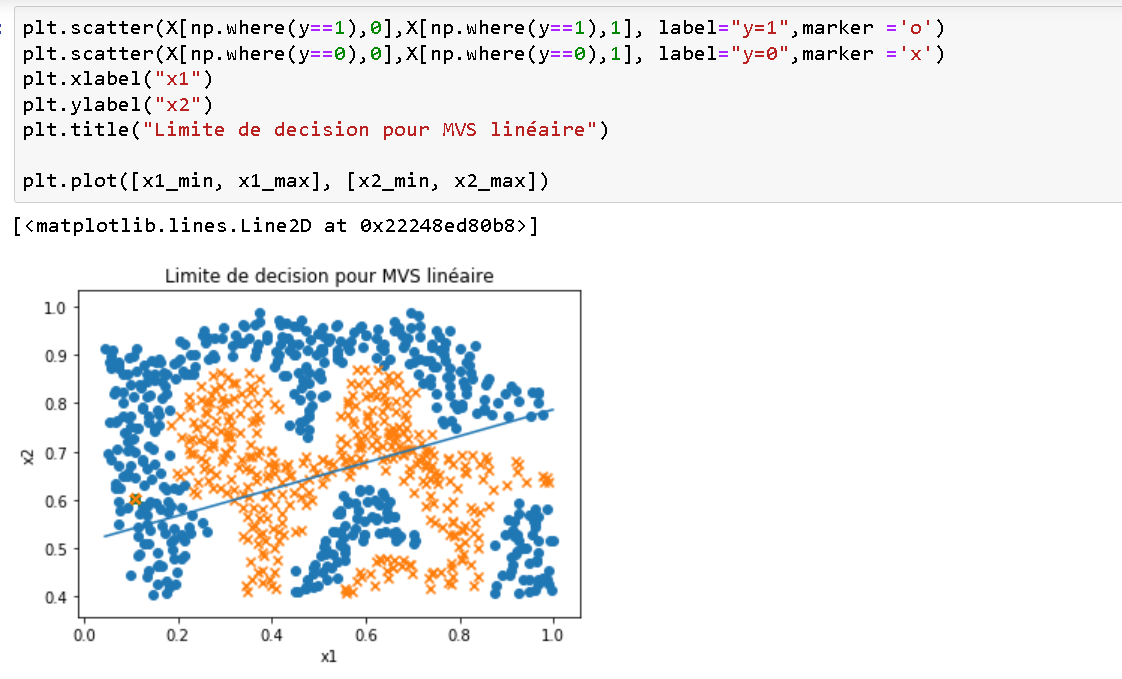
X2\_min=0,52

2eme point :

X1\_max=0,99

X2\_max=0,78

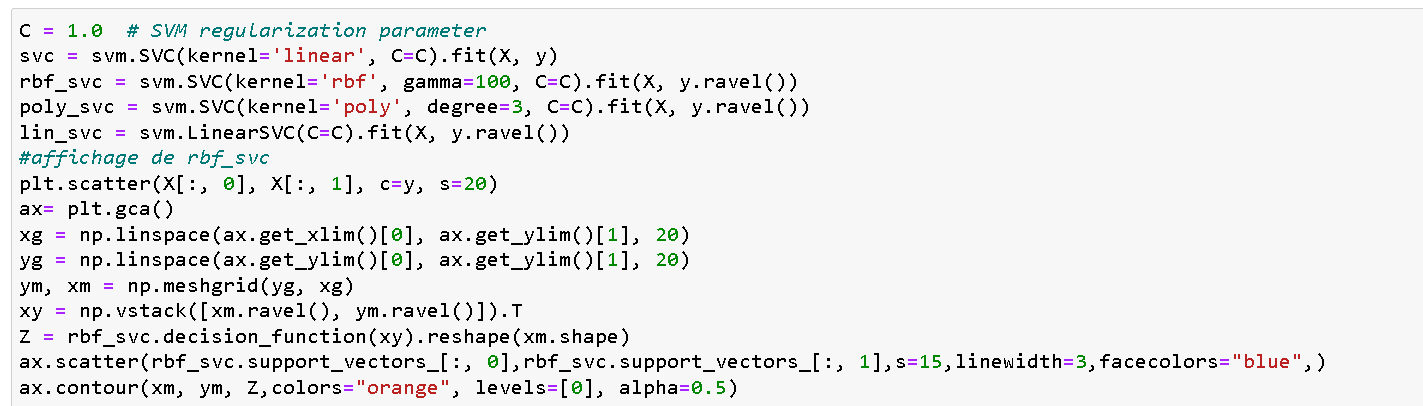
**Affichage de la limite de décision :**



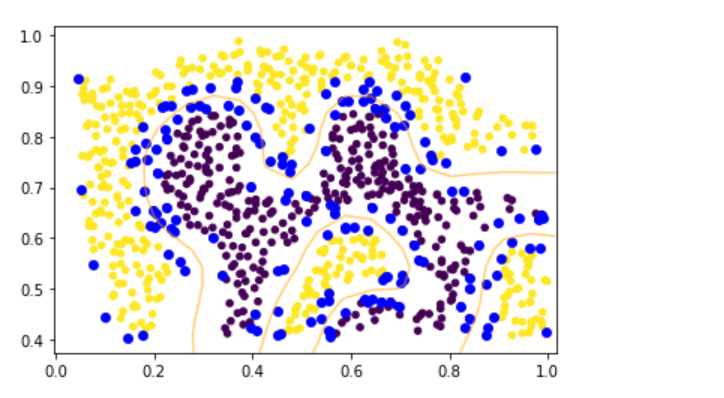
**Remarque** :

on voit que nos données ne sont pas linéairement séparables .

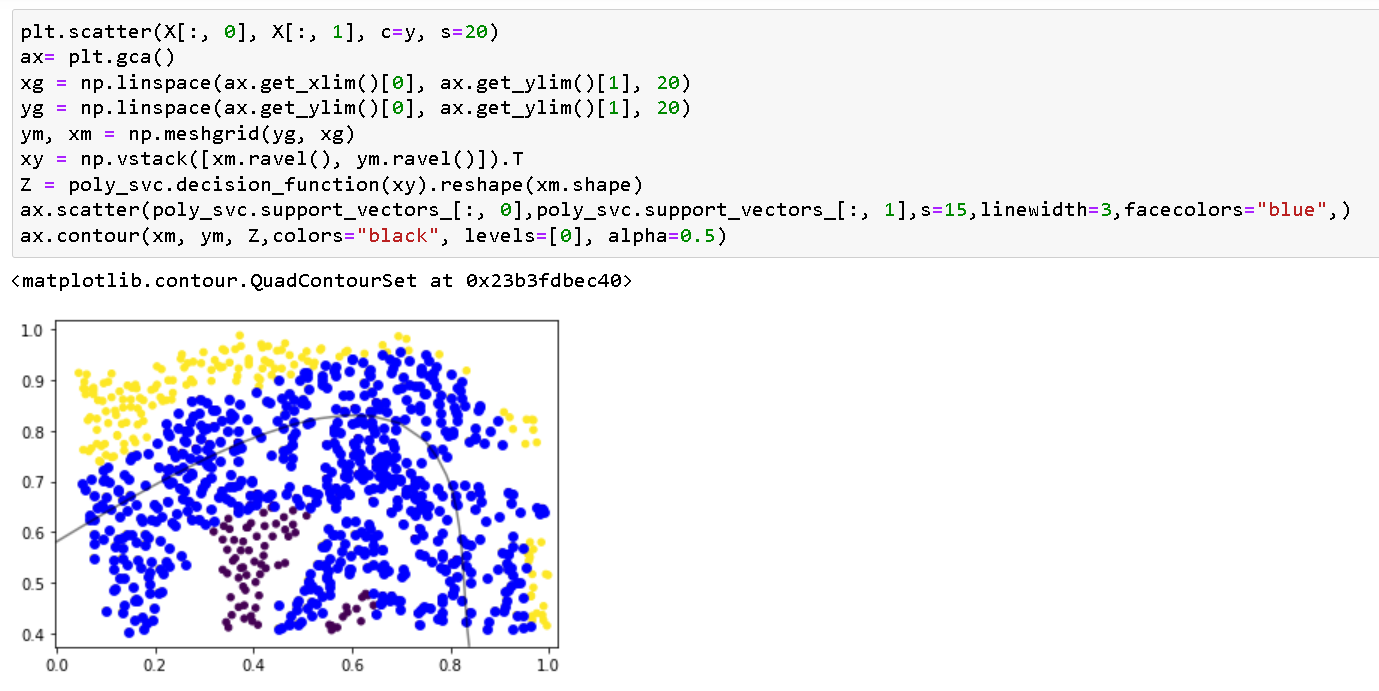
-Modification du code en fixant l’écart type gamma à 100 :



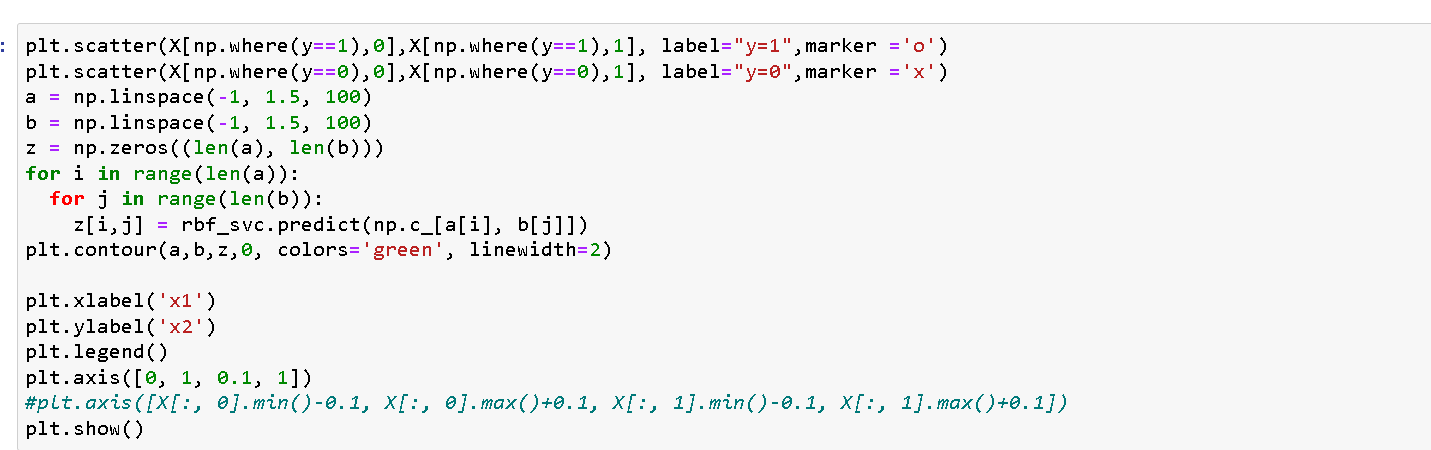
Affichage du noyau rbf\_svc:



Affichage du noyau poly\_svc:



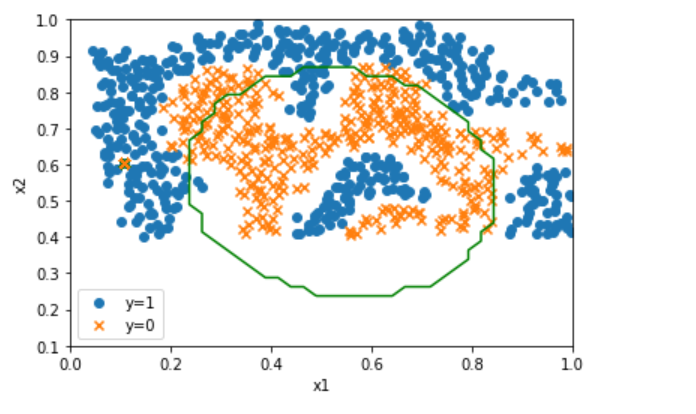
**Affichage de la decision Boundary pour les noyaux gaussiens** :



on va utiliser notre fonction pour avoir nos valeurs de retours pour la fonction hypothese pour ce faire on va utiliser

le retour de notre svm .on va générer quelques points séparés par un intervalle fixe et essayer d'avoir notre fonction hypothese.pour chaque element ca va etre une matrice on va appeller la fonction rbf\_svc pour avoir nos nos z .une fois qu’on a nos z on va pouvoir dessiner le contour en fonction des x , y et z.

**Affichage** :

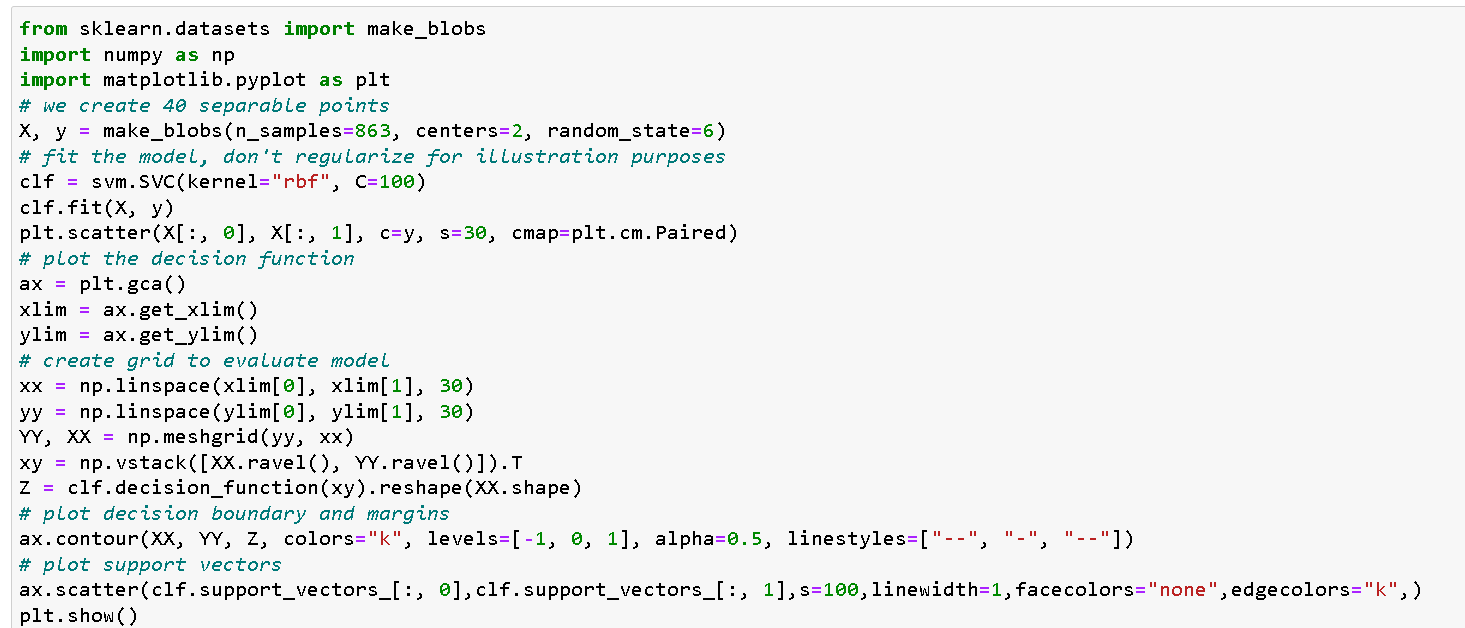


**Un autre ensemble de données pour la visualisation des vecteurs de support :**

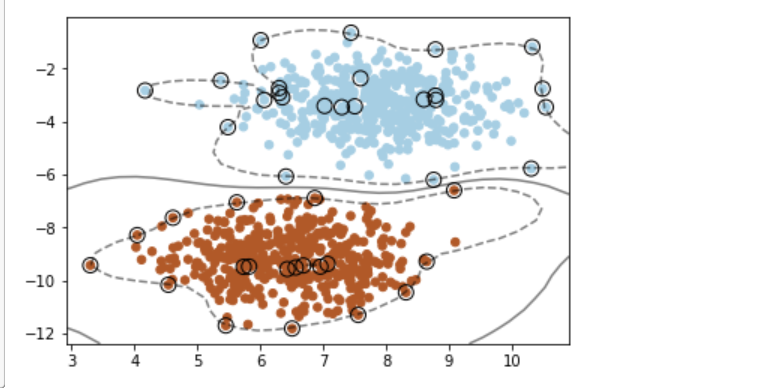
Les SVM vont choisir certains points X de tous l'ensemble pour essayer de maximiser la marge de séparation .

La projection de X sur nos theta pi || θ ||devrait etre maximal . on va générer un ensemble de vecteurs et on essaye de voir ce que donne notre hypothése par rapport à cet ensemble .

Code pour visualiser la Decision Boundary et les vecteurs de supports:



**Affichage** :



**On varie c=10 pour bien visualiser les vecteurs de supports:**

