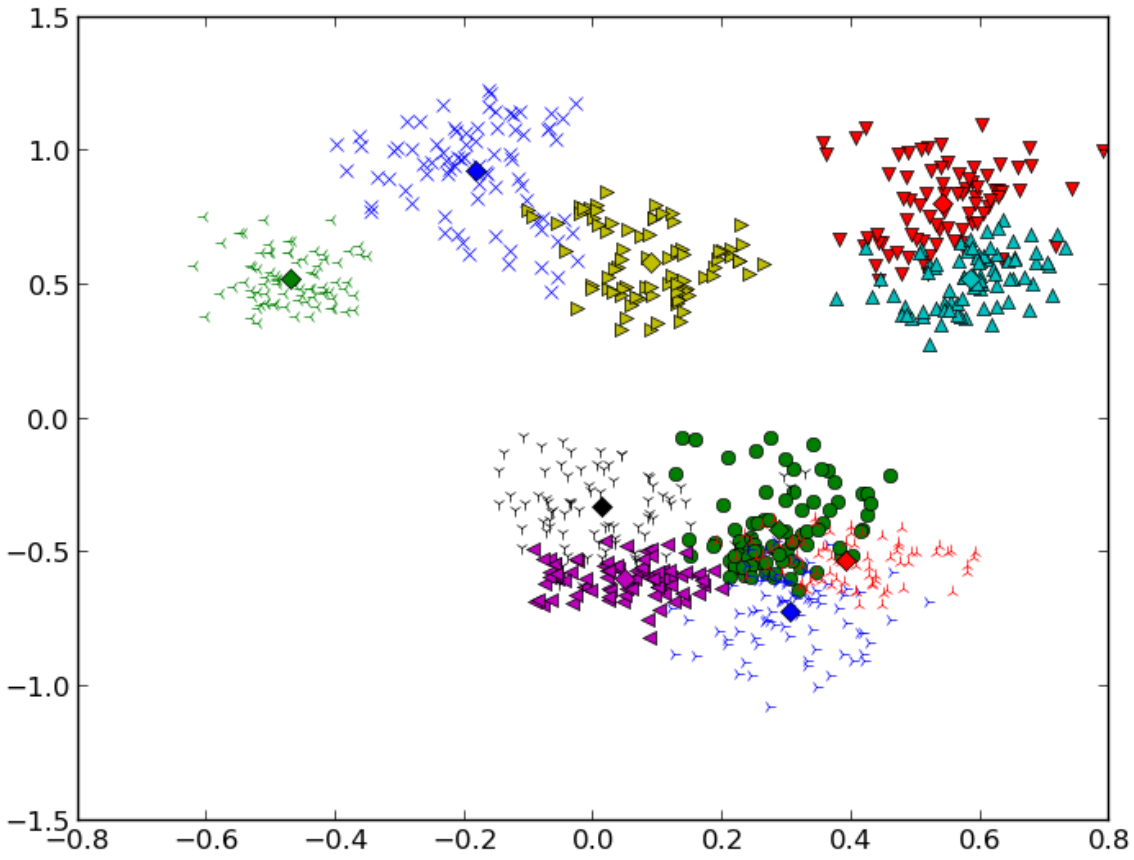
TP5

IAA

Classification et évaluation

1.Classification supervisée

Voici, pour le fichier data2.py, l'affichage des points d'apprentissage et leurs centres.



Classe1 : croix bleus
Classe2 : ronds vert
Classe3 : triangles rouges
Classe4 : triangles cyan
Classe5 : triangles magenta
Classe6 : triangles jaunes
Classe7 : Y noirs
Classe8 : Y bleus
Classe9 : Y verts
Classe10 : Y rouges

Voici les résultats trouvés avec K = 5, on a toujours 80 % de données d'apprentissage.

```
In [25]: run ./tp5.py
Pour les Kppv je prends K=5


La matrice de confusion pour les kppv est:
[[ 20.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   6.   0.   0.   0.   0.   0.   3.   0.  11.]
 [  0.   0.  19.   1.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.  20.   0.   0.   0.   0.   0.   0.]
 [  0.   2.   0.   0.   4.   0.  12.   1.   0.   1.]
 [  0.   0.   2.   0.   0.  18.   0.   0.   0.   0.]
 [  0.   2.   0.   0.   9.   0.   9.   0.   0.   0.]
 [  0.   5.   0.   0.   0.   0.   0.   9.   0.   6.]
 [  0.   0.   0.   0.   0.   0.   0.   0.  20.   0.]
 [  0.  14.   0.   0.   0.   0.   0.   0.   0.   6.]]

La matrice de confusion pour les loi normales est:
[[ 20.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   4.   0.   0.   0.   0.   0.   2.   0.  14.]
 [  0.   0.  19.   1.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.  20.   0.   0.   0.   0.   0.   0.]
 [  0.   3.   0.   0.   3.   0.  12.   2.   0.   0.]
 [  0.   0.   3.   0.   0.  17.   0.   0.   0.   0.]
 [  0.   2.   0.   0.   9.   0.   9.   0.   0.   0.]
 [  0.   4.   0.   0.   0.   0.   0.   7.   0.   9.]
 [  0.   0.   0.   0.   0.   0.   0.   0.  20.   0.]
 [  0.  14.   0.   0.   0.   0.   0.   0.   0.   6.]]
taux de reconnaissance avec kppv:
0.655
taux de reconnaissance avec apprentissage loi normales:
0.625
intervalle de confiance kppv:
0.0658826820948
intervalle de confiance apprentissage loi normales:
0.0670960132944
```

On remarque que les points des classes 2, 5, 7, 8 et 10 sont très mal classés, avec les méthodes kppv comme avec les lois normales.

En revanche les points des classes 1, 4 et 9 sont toujours parfaitement identifiées dans les deux cas.

La méthode des kppv offre le meilleur taux de reconnaissance dans ce cas présent.

Avec les données en dim 3 de data3.py , on obtient les résultats suivants:

```
In [27]: run ./tp5.py
Pour les Kppv je prends K=5


La matrice de confusion pour les kppv est:
[[ 20.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.  20.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.  20.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.  20.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.  14.   0.   2.   4.   0.   0.]
 [  0.   0.   0.   0.   0.  20.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.  20.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.   0.  19.   0.   1.]
 [  0.   0.   0.   0.   0.   0.   0.   0.  20.   0.]
 [  0.   0.   0.   0.   0.   0.   0.   0.   0.  20.]]

La matrice de confusion pour les loi normales est:
[[ 20.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.  20.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.  20.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   3.  17.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   8.   0.   5.   6.   0.   1.]
 [  1.   0.   0.   0.   0.  19.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.  20.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.   0.  19.   0.   1.]
 [  0.   0.   0.   0.   0.   0.   0.   0.  20.   0.]
 [  0.   0.   0.   0.   0.   0.   0.   0.   0.  20.]]
taux de reconnaissance avec kppv:
0.965
taux de reconnaissance avec apprentissage loi normales:
0.915
intervalle de confiance kppv:
0.02547057518
intervalle de confiance apprentissage loi normales:
0.0386510310341
```

On observe que les résultats sont beaucoup plus fiables. La méthode des kppv identifie catégoriquement les différentes classes sauf pour la 5eme classe où 14 points sur 20 ont été bien classés. Avec les lois normales, on a beaucoup d'erreurs sur la 5eme classe, seuls 8 points sur 20 ont été bien classé.

La méthode des kppv est encore celle offrant le meilleur taux de reconnaissances.

Avec le fichier dim12, on observe :

```
In [30]: run ./tp5.py
Pour les Kppv je prends K=5


La matrice de confusion pour les kppv est:
[[ 20.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.  20.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.  20.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.  20.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.  20.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   0.  20.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.  20.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.   0.  20.   0.   0.]
 [  0.   0.   0.   0.   0.   0.   0.   0.  20.   0.]
 [  0.   0.   0.   0.   0.   0.   0.   0.   0.  20.]]

La matrice de confusion pour les loi normales est:
[[ 20.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.  20.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.  20.   0.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.  20.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.  20.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   0.  20.   0.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.  20.   0.   0.   0.]
 [  0.   0.   0.   0.   0.   0.   0.  20.   0.   0.]
 [  0.   0.   0.   0.   0.   0.   0.   0.  20.   0.]
 [  0.   0.   0.   0.   0.   0.   0.   0.   0.  20.]]
taux de reconnaissance avec kppv:
1.0
taux de reconnaissance avec apprentissage loi normales:
1.0
intervalle de confiance kppv:
0.0
intervalle de confiance apprentissage loi normales:
0.0
```

On a cette fois ci une reconnaissance parfaite de chacun des points, pour les deux méthodes. En effet en augmentant les dimensions on réduit le risque d'erreurs.

En temps de calculs, la méthode des kppv recalcule les distances à chacun des points d'apprentissage pour chaque point de test.

La méthode des lois normales calcule le score de chaque point pour chaque loi normale de chaque classe d'apprentissage.

En temps de calcul la méthode des loi normales est donc plus performante.

Pour K=5, avec data12.py et 80 % d'apprentissage, on a 4.34 secondes de temps d'exécution pour la méthode des kppv et 0.23 secondes pour le temps d'exécution des lois normales, sachant que le temps de calcul de l'apprentissage des paramètres des lois normales n'est pas compris dedans.

4.34/0.23~18.87

La méthode des kppv est donc environ 18 fois plus lente que la méthode des loi normales.

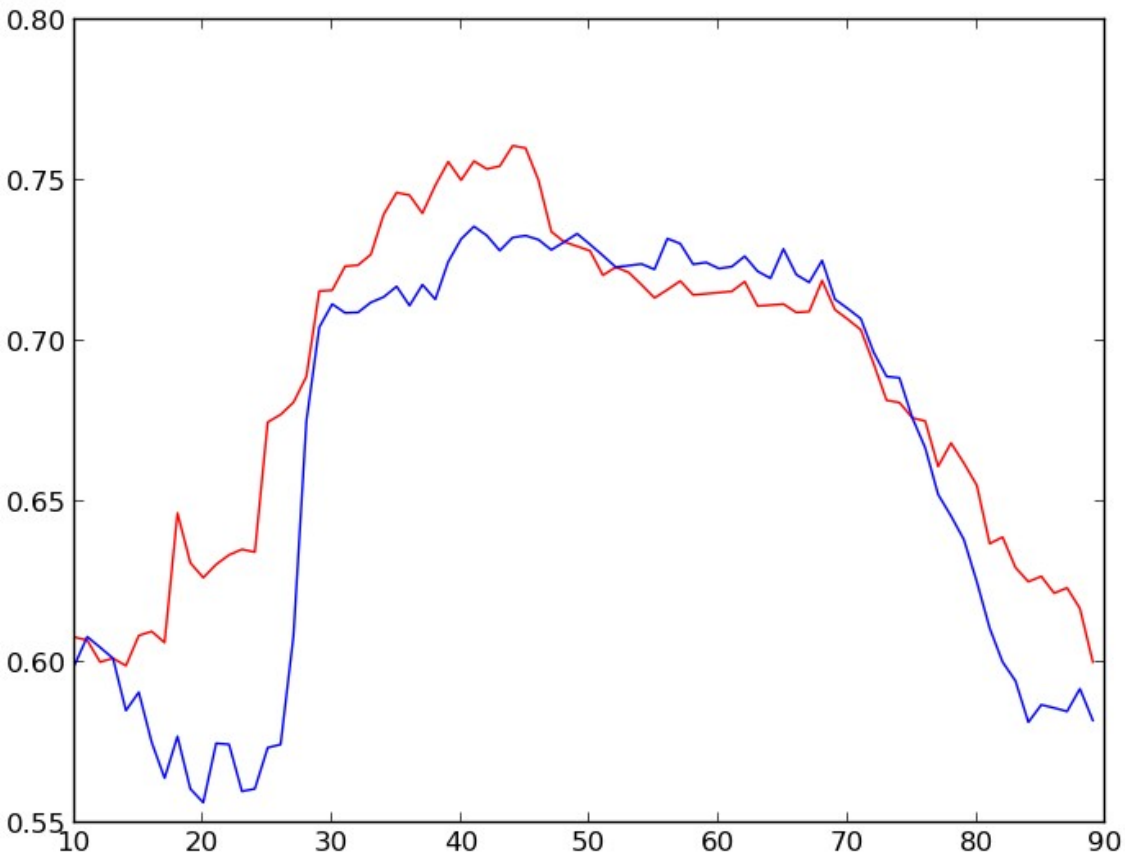
Pour les prérequis sur les données, la méthode des kppv a juste besoin des données d'apprentissages.

Pour la méthode des lois normales, il faut que les données décrivent une loi normale et que les données d'apprentissage soient assez importantes pour avoir une bonne approximation des paramètres.

En termes de reconnaissances, les deux méthodes se valent au niveau de la performance. (voir le graphique ci dessous page suivante)

Voici ci-dessous les courbes représentant les taux de reconnaissances pour la méthode des kppv et celle des lois normales sur data2 en fonction du pourcentage des données d'apprentissage.

La courbe rouge représente le taux de reconnaissance de la méthode des kppv et la courbe bleu représente le taux de reconnaissance de la méthode des lois normales.



Cette courbe est très instructive. On observe que dans certains cas une méthode est plus efficace par rapport à l'autre, et que contrairement à ce que l'on pourrait penser, quand les données d'apprentissages sont très importantes les données de tests ne sont pas forcément mieux reconnue. Les meilleurs reconnaissances se situent vers 50 %.

Validation croisée

La validation croisée repose sur l'idée d'un découpage des données en plusieurs sous-échantillons. On utilise la première partie des données pour construire une estimation de la courbe avec chacun des modèles en compétition. Ensuite, avec les données restantes, on évalue la qualité de chacune des estimations en les comparant aux valeurs observées.

Cette méthode est appelé « *cross-validation* ». Cependant il existe plusieurs techniques de validation croisée : « *testset validation* » ou « *holdout method* »,« *k-fold cross-validation* » et « *leave-*

one-out cross validation » (LOOCV).

C'est la méthode que l'on vient d'utiliser.

Une autre méthode possible consiste à diviser k fois l'échantillon originale, puis de choisir un de ces sous échantillon comme ensemble de test, et les échantillons restant comme apprentissage. Puis on refait la même chose pour les autres échantillons qui n'ont pas été utilisé comme ensemble de test. On fait ensuite une moyenne sur les erreurs.

Contact

Si il y a des erreurs, des remarques, des ajouts à faire, etc.

Veuillez m'en faire part à cette adresse :

wedg@hotmail.fr