|  |  |
| --- | --- |
| |  | | --- | |  | |

**Devoir #2 : Les réseaux de Hopfield**

Session : Eté 2015

*Réseaux de neurones*

*Département d’informatique et de mathématiques*

*Présenté à : Professeur: H.Ezzaidi*

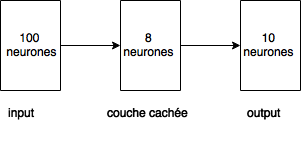
*Travail de : AMAMOU Houssem*

Code permanent : AMAH10029004

# Classification par réseau multicouches : toutes les données

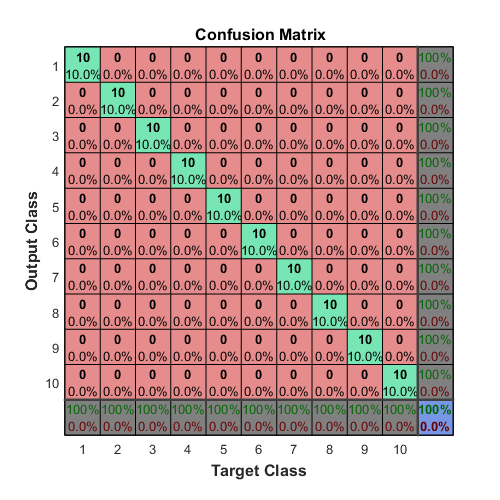
1. L’architecture choisie pour résoudre ce problème est un perceptron qui a cent neurones en entrée. Cela représente la taille de chaque échantillon pour chaque classe. Comme chaque donnée est représenté sous forme d’un vecteur-ligne de taille cent. Nous n’avons plus qu’à mettre en entrée pour le perceptron la transposée de chaque échantillon.

En sortie, comme nous allons effectuer une classification avec dix classes, la couche de sortie est de taille 10. Pour un échantillon donné, la classe de cet échantillon est représenté en mettant à un le bit de sortie correspondant. Par exemple, si une donnée est classé appartenant à la classe 2 c’est le troisième bit de note vecteur de sortie qui va être à 1 et tous les autres bits vont être à zéro. Concernant le nombre de neurone dans la couche cachée, d’après le théorème de Kolmogorov qui dit que tous problèmes de variable peuvent être résolus avec neurones. Donc théoriquement cents neurones sont suffisants pour résoudre notre problème mais nous pouvons procéder de la façon à estimer le nombre de frontière nécessaire pour résoudre notre problème et procéder de façon empirique pour estimer un nombre de neurones suffisants. Après avoir effectué plusieurs essais, nous avons constaté que 8 neurones dans la couche cachée étaient largement suffisants à ce problème de classification.



Architecture du réseau de neurones

1. Nous avons joint au document les commandes exécutées sur Matlab
2. La matrice de confusion suivante permet de vérifier les résultats de l’entraînement.

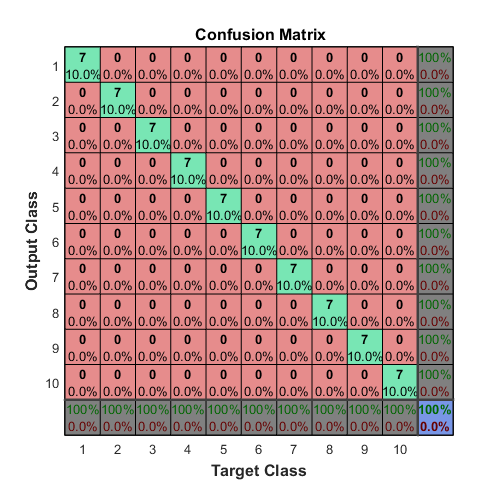


Comme nous pouvons le voir le taux de reconnaissance est de 100% pour toutes les classes donc le réseau classe bien les données d’entraînement.

1. Toutes les frontières ont été bien estimées.

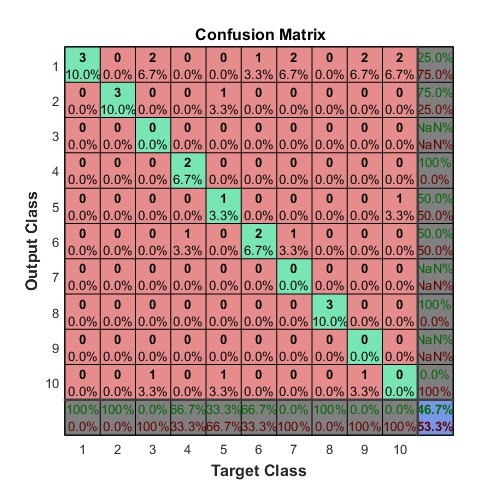
# Classification par réseau multicouches : données incomplètes

1. Pour faire l’entraînement de cette partie nous avons sélectionné les 7 premiers échantillons de chaque classe pour entraîner notre réseau. Donc nous allons avoir 70% des données pour entraîner le réseau et nous allons effectuer les tests sur les 30% des données restantes.
2. Après avoir entraîné le réseau, nous avons effectué la reconnaissance sur les premiers 70% des données nous avons obtenus un taux de reconnaissance de 100% pour toutes les classes comme la montre la matrice de confusion suivante



Matrice de confusion pour les données d’entraînement

Ensuite, nous avons effectué la reconnaissance sur les données de tests (30% des données restantes) et nous avons obtenus les résultats suivants



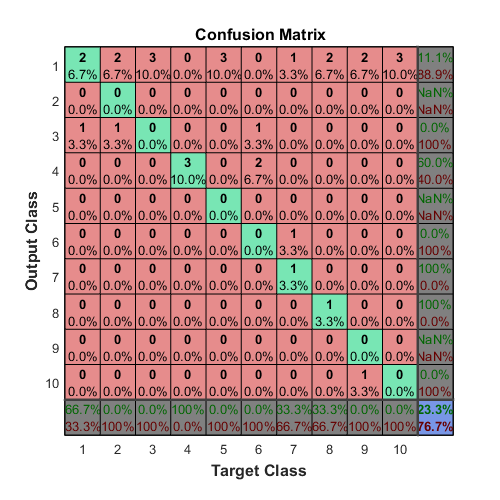
Matrice de confusion pour les données de tests

Nous remarquons que le taux de reconnaissance chute à 46.7%. Le réseau n’a pas bien reconnu certaines classes notamment il a confondu les 8 en 0. Il a aussi confondu les 6 en 0. On constate que le réseau n’est plus très performant. Nous avons essayé de modifier quelques caractéristiques du réseau ainsi que de modifier les données d’entrainement pour choisir d’autres 70% des données autres que les premiers données et tester sur les données restantes mais nous n’avons pas constaté d’amélioration de résultats.

1. Le taux de reconnaissance chute considérablement par rapport à la question 2. On s’attendait à une baisse du taux de reconnaissance mais la diminution a été plus importante que prévu. Nous pouvons dire que le perceptron multicouche n’est pas très efficace pour la reconnaissance de ce type de problème.
2. Les matrices de confusions des classes ont été générées et montré précédemment dans le document

# Analyse et conclusion

Les résultats que nous avons obtenus montrent que le réseau est performant pour les données d’entrainement mais que son taux de reconnaissance baisse pour atteindre les 46.7% sur les données de tests mais il reste plus performant que le perceptron sans couche cachée car nous avons testé que ce modèle tombe à 23.3% comme le montre la matrice de confusion suivante



Matrice de confusion avec le perceptron sans couche cachée sur données de test