# Apprentissage supervisé et non supervisé
# TP2 : Réseaux de neuronnes
M2 SISE - Université Lyon 2- 2017/2018

Responsable : Julien Ah-Pine

## 1 Présentation de l'étude de cas

Les données que vous devez modéliser correspondent à des (bouts de) verres. Les données recuillies sont la composition chimique des verres et le problème consiste à prédire de quel type il s'agit (bris de glace de véhicule, de fenêtre, verre de table...). Ce problème a été motivé par des besoins d'investigation dans le cadre de scènes de crime. En effet, les verres pouvaient être utilisés comme preuves à condition qu'ils aient été bien identifiés !

Nous avons à notre disposition 214 observations et pour chacune différentes mesures dont les compositions selon les composants chimiques. Les différents type de verre sont au nombre de 6 : (1) building windows float processed, (2) building windows non float processed, (3) vehicule windows float processed, (5) containers, (6) tableware, (7) headlamps.

## 2 Lecture et decription des données

1. Chargez la librairie `mlbench` (installez la si besoin).
2. Chargez le jeu de données ``Glass'' à l'aide de la commande `data`.
3. Stockez dans la variable `nom_var` le nom des variables à l'aide de la commande `names`.
4. Identifiez les variables explicatives et la variable à expliquer. Stockez dans `p` le nombre de variables explicatives et dans `n` le nombre d'observations.
5. Stockez dans `nom_var_exp` le nom des variables explicatives et stockez dans `nom_clas_cib` le nom des classes cibles comme ci-dessous :

```
nom_clas_cib=c("building_fp","building_nfp","vehicule_fp","containers
    ","tableware","headlamps")
```

6. Concernant les variables explicatives, utilisez la commande `summary` pour avoir un aperçu de leurs statistiques descriptives. Les variables ont-elles des grandeurs qui sont comparables ?

## 3 Pré-traitement des données

7. Entrez, exécutez et commentez les commandes suivantes :

```
X=scale(Glass[,1:p])
Y=cbind(ifelse(Glass$Type==1,1,0),ifelse(Glass$Type==2,1,0),ifelse(
    Glass$Type==3,1,0),ifelse(Glass$Type==5,1,0),ifelse(Glass$Type
    ==6,1,0),ifelse(Glass$Type==7,1,0))
Y=as.data.frame(Y)
Glass=cbind(X,Y)
names(Glass)=c(nom_var_exp,nom_clas_cib)
```

# 4 Réseaux de neurones avec 1 couche cachée

Nous allons utiliser la librairie R `neuralnet` pour mettre en oeuvre le modèle de réseaux de neurones pour ce problème de catégorisation. Vous trouverez la page du CRAN dédiée à la librairie ici : `https://cran.r-project.org/package=neuralnet`.

8. Installez puis chargez la librairie `neuralnet`.

La commande principale pour l'estimation des paramètres est `neuralnet`. Pour instancier un modèle et définir variable à expliquer et variables explicatives, on utilise ici également les `formulas` de R (comme pour la commande `lm` par exemple). En revanche, il n'est pas possible d'utiliser les racourcis de type `y~.,data=X`. Il nous faut "écrire" la formule "à la main".

9. Entrez, exécutez et commentez les commandes suivantes :

```
var_exp=paste(nom_var_exp,collapse = "+")
clas_cib=paste(nom_clas_cib,collapse = "+")
mod=paste(clas_cib,"~",var_exp,sep="")
mod=as.formula(mod)
```

Attention ! la formule est symbolique et en aucun cas, le problème se limite à un modèle linéaire. Rappelons qu'un des intérêts des réseaux de neurones consiste à modéliser non linéairement les dépendances entre la variable cible et les variables explicatives via les fonctions d'activation.

10. Entrez, exécutez et commentez la commande suivante :

```
nnet_fit_1hl=neuralnet(formula = mod,data = Glass, hidden = p, err.fct
   = "ce", linear.output = FALSE)
```

11. Etudier en détail les résultats stockés dans `nnet_fit_1hl`. Quelle est la fonction d'activation utilisée par défaut ici ? Quelle sous-liste contient les coefficients synaptiques estimés ?
12. Entrez, exécutez et commentez la commande suivante :

```
plot(nnet_fit_1hl)
```

13. Entrez, exécutez et commentez la commande suivante :

```
pred_sc=compute(nnet_fit_1hl,Glass[1:10,1:p])
```

# 5 Validation croisée

Pour une meilleure estimation de l'erreur en généralisation, il est indispensable de faire de la validation croisée. Il nous faut ici implémenter "à la main" la validation croisée. Voici une façon de faire qu'il vous faut compléter. En particulier, déterminez le taux d'erreur de chaque fold et le taux d'erreur moyen.

14. Entrez, commentez et complétez les commandes suivantes en vue de mettre en oeuvre la validation croisée du modèle précédent :

```
nb_folds=5
folds_obs=sample(rep(1:nb_folds,length.out=n))
for (k in 1:nb_folds)
{
  print(paste("===== Fold :",k))
  test=which(folds_obs==k)
  train=setdiff(1:n,test)
...
}
```

## 6 Réseaux de neuronnes avec plusieurs couches cachées et autres types de fonction d'activation

15. Sur le même jeu de données, utilisez les commandes précédentes pour estimer les coefficients et analyser les résultats d'un réseau de neuronne avec 2 couches cachées. Comparez les résultats avec ceux du réseau de neuronnes à 1 couche cachée.
16. Sur le même jeu de données, utilisez les commandes précédentes et tester un autre type de fonction d'activation. Comparez les résultats avec ceux des modèles précédents.