

Nom:			
Prénom:			
Parcours	ISI gr1	ISI gr2	SAR

## Examen de TP sur 25 pts

## Exercice 1 (7pts)

On exécute le programme :

```
[X, y, name]=np.load("TP1.npy")  
print(X.shape)  
print(y.shape)  
qui nous donne les sorties
```

Out[]: (1000, 100)

Out[]: (1000,)

1. (1pt) Quelle est le nombre d'exemples N et la dimension du problème n?

N=1000  
n=100

On réalise ensuite une division des données avec :

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)
```

2. (1pt) Quelle sera la taille des variables X\_train, X\_test, y\_train et y\_test ?

X\_train 750x100  
X\_test 250x100  
y\_train 750x1  
y\_test 250x1

On réalise ensuite une standardisation des données avec :

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)  
scaler = StandardScaler()  
X_train = scaler.fit_transform(X_train)  
X_test = scaler.fit_transform(X_test)
```

3. (2pts) Ce code est-il correct ? Si non, apporter les modifications nécessaires. Quelle sera la taille de X\_train et X\_test ?

Non, il n'est pas correct  
`scaler = StandardScaler()`  
`scaler.fit(X_train)`  
`X_train = scaler.transform(X_train)`  
`X_test = scaler.transform(X_test)`  
La taille des variables ne change pas

On souhaite maintenant réduire la dimension des données avec :

```
pca=PCA(n_components=40)
pca.fit(X_train)
X_train=pca.transform(X_train)
X_test=pca.transform(X_test)
```

4. (1pt) Le code est-il correct ? Si non, apporter les modifications nécessaires. Quelle sera la taille de `X_train` et `X_test` ? Faut-il aussi appliquer `pca.transform` sur `y_train` et `y_test` ?

Oui, le programme est correct  
`X_train` 750x40  
`X_test` 250x40

Non, pas de PCA sur les labels !

On lance enfin la classification et affiche la matrice de confusion :

$$\begin{bmatrix} 5 & 1 & 0 \\ 7 & 15 & 5 \\ 2 & 5 & 10 \end{bmatrix}$$

5. (2pts) Combien y a-t-il de classes ? Combien y a-t-il d'exemples par classe ? Quel est le taux de reconnaissance ?

Il y a 3 classes  
C1 – 6 exemples  
C2 – 27 exemples  
C3 – 17 exemples  
Taux de reco =  $30/50 * 100 = 60\%$

## Exercice 2 (5pts)

On souhaite réaliser une classification binaire en modélisant une seule classe. Pour cela, on a calculé les probabilités  $p\_test[i]$  que chaque exemple de test  $i$  appartienne à la classe 1. Ainsi, si  $p\_test[i] > \text{seuil}$ , l'exemple sera attribué à la classe C1. Sinon, il sera classifié C0. La vérité de terrain est stockée dans  $y\_test[i]$ . Pour un seuil donné, on souhaite estimer le nombre de vrais positifs (TP), faux positifs (FP), vrais négatifs (TN) et faux négatifs (FN).

### 1. (3pts) Compléter le programme suivant :

```
TP , FP , TN , FN = 0, 0, 0, 0
for i in range(len(y_test)):
    if p_test[i] > seuil:
        y_pred[i] = 1
    else:
        y_pred[i] = 0
    if cl==1 and y_test[i]==1:
        TP = TP+1
    if cl==1 and y_test[i]==0:
        FP = FP +1
    if cl==0 and y_test[i]==0:
        TN = TN+1
    if cl==0 and y_test[i]==1:
        FN = FN+1
```

Le programme donne :

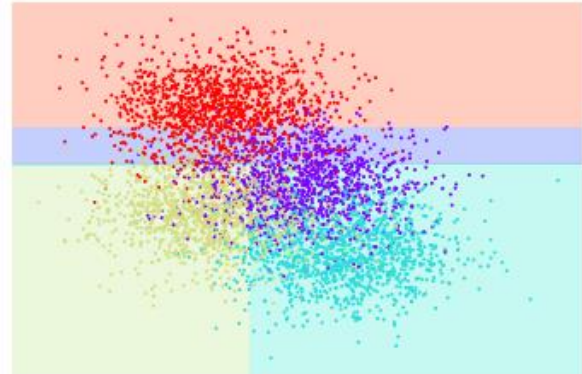
TP= 100 TN= 150 FP= 50 FN= 10

### 2. (2pt) Combien y a-t-il d'exemple dans chaque classe ? Que valent la sensibilité et la spécificité ?

110 exemples positifs et 200 exemples négatifs  
 $\text{Sensibilité} = TP / (TP + FN) = 100 / 110$   
 $\text{Spécificité} = TN / (TN + FP) = 150 / 200$

### Exercice 3 (3 pts)

On souhaite réaliser une classification en utilisant un arbre de décision :  
`tree1 = DecisionTreeClassifier(criterion='entropy', max_depth = M)`  
 La partition de l'espace obtenu, sur ces données 2D, est représentée ci-contre.



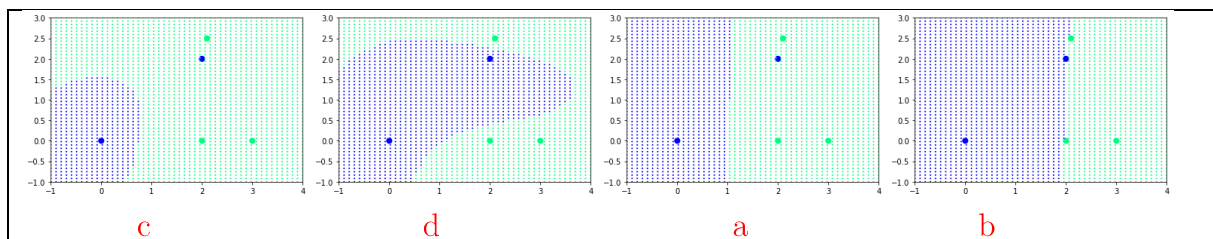
#### 1. (1.5pts) Quelle valeur de M a été utilisée ? Justifier votre raisonnement

$M = 2$  : lors de la première découpe, on obtient 2 cases. Chacune est divisée lors de la seconde découpe pour arriver à 4 partitions.

On souhaite réaliser une classification avec des SVM. En utilisant toujours les mêmes données d'apprentissage, on obtient les frontières de décision avec 4 paramétrages :

- (a) noyau lineaire et  $C=1$
- (b) noyau lineaire et  $C=1000$
- (c) noyau rbf et  $C=1$
- (d) noyau rbf et  $C=1000$

#### 2. (1.5pts) Réattribuer le paramétrage utilisé pour chaque courbe :



### Exercice 4 (10 pts)

On souhaite lancer un apprentissage avec un réseau de neurone. Pour cela, on télécharge les données :  $(X\_train, y\_train), (X\_test, y\_test) = load\_data()$ . La base de test est composée de 3 exemples tels que  $y\_test=[0\ 2\ 2]$ . Afin de mettre en forme les étiquettes, on réalise :

```
Y_test = tf.keras.utils.to_categorical(y_test, 4)
```

#### 1. (2pts) Donner les valeurs de la variable $Y\_test$

$$Y\_test = \begin{bmatrix} 1000 \\ 0010 \\ 0010 \end{bmatrix}$$

On définit un réseau de neurones avec :

```
inputs = Input(shape=(10,10,1))  
x= inputs  
x=Flatten()(x)  
x=Dense(20, activation='relu')(x)  
x=Dropout(0.5)(x)  
outputs=Dense(10, activation='softmax')(x)  
model = Model(inputs, outputs)
```

#### 2. (2pts) Combien y a-t-il de paramètres à estimer ?

$$(100*20+20) + (20*10+10) = 2,230$$

On ajoute entre les deux premières lignes précédentes la ligne :

```
x=Convolution2D(10,(3,3), activation='relu',padding='same')(x)
```

#### 3. (2pts) Quelle sera la taille du tenseur en sortie de cette couche ?

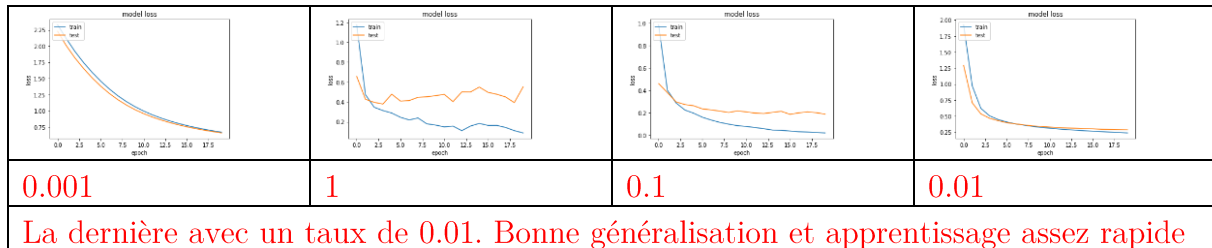
Combien de paramètres y a-t-il à estimer avec cette nouvelle architecture (reprendre tous le réseau) ?

Le tenseur aura pour dimension 10x10x10

$$(10*3*3+10) + (1000*20+20) + (20*10+10) = 20,330$$

On réalise l'apprentissage d'un réseau MLP avec 4 taux d'apprentissage différents (1 ; 0.1 ; 0.01 ; 0.001) et on obtient les 4 courbes d'apprentissage suivantes.

4. (2pts) Réattribuer le taux utilisé dans chaque cas. Quelle courbe vous paraît la plus adaptée ?



On a réalisé l'apprentissage d'un réseau MLP (1 couche cachée) en introduisant une couche de dropout de taux rate (0 ; 0.5 ; 0.7) entre les deux dernières couches.

$x = \text{Dropout}(\text{rate})(x)$ . Pour mémoire, la documentation donne :

**rate:** Float between 0 and 1. Fraction of the input units to drop

5. (2pts) Réattribuer le taux utilisé dans chaque cas. Quelle courbe vous paraît la plus adaptée ?

