

SY32 – Vision et apprentissage

TD 01 : Apprentissage automatique

On souhaite distinguer deux types (A et B) de chat en utilisant le poids comme facteur discriminant. Les fichiers SY32_P20_TD01_data_X.csv et SY32_P20_TD01_data_y.csv contiennent les poids mesurés sur un ensemble de 300 chats ainsi que leur type, respectivement. Le premier fichier représente le poids et le second le type : -1=Type A et +1=Type B. La Figure 1 représente la distribution des données observées.

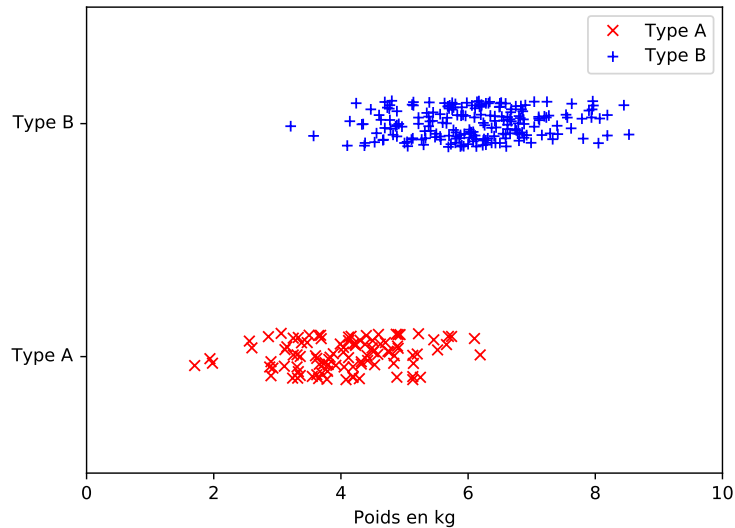


FIGURE 1 – Poids mesurés sur un ensemble de 300 chats.

On considère un classifieur f paramétré par une variable h tel que

$$f_h(x) = \begin{cases} -1 & \text{si } x \leq h \\ +1 & \text{sinon} \end{cases},$$

où x représente un poids.

1. Charger les données contenues dans les fichiers SY32_P20_TD01_data_*.csv en utilisant la fonction loadtxt de la librairie numpy.

```
import numpy as np
X_train = np.loadtxt('SY32_P20_TD01_data_X.csv', ndmin=2)
y_train = np.loadtxt('SY32_P20_TD01_data_y.csv')
```

2. Le fichier catClassifier.py définit une classe Python CatClassifier pour le classifieur de chat. Écrire la fonction predict qui prend en entrée un tableau de poids X , une valeur de h et prédit pour chacun des poids le type de chat correspondant sous la forme d'un tableau y .
3. Prédire le type des chats X_{train} avec $h = 5$.

```
from catClassifier import CatClassifier
clf = CatClassifier()
print(clf.predict(X_train, 5))
```

4. Écrire la fonction err_emp de CatClassifier qui prend en entrée un ensemble de données (X, y) et une valeur de h puis calcule le taux d'erreur empirique du classifieur f_h sur l'ensemble (X, y) .
5. Calculer le taux d'erreur empirique sur $(X_{\text{train}}, y_{\text{train}})$ avec $h = 5$.

```
print(clf.err_emp(X_train, y_train, 5))
```

6. Écrire la fonction fit de CatClassifier qui prend en entrée un ensemble de données (X, y) et calcule une valeur optimale \hat{h} qui minimise le taux d'erreur sur (X, y) .
7. Donner une valeur de h qui minimise le taux d'erreur sur $(X_{\text{train}}, y_{\text{train}})$.

```
clf.fit(X_train, y_train)
print(clf.h_hat)
```

8. Calculer le taux d'erreur empirique de $f_{\hat{h}}$ sur l'ensemble $(X_{\text{train}}, y_{\text{train}})$.

```
print(clf.err_emp(X_train, y_train))
```

9. Comparer le taux d'erreur calculé à la question précédente avec le taux d'erreur réel consultable à l'adresse suivante : (<http://robotics.gi.utc/SY32/TD01/>).
10. Implémenter une validation croisée à K échantillons et calculer un nouvel estimateur du taux d'erreur associé à la valeur de \hat{h} calculée à la question 7. Comparer le résultat pour différentes valeurs de K au taux d'erreur réel et conclure.
11. La librairie **scikit-learn** propose différents échantillonnages pour la validation croisée :
- K-Fold
 - Shuffle Split
 - Stratified K-Fold
 - Stratified Shuffle Split

La description des différents échantillonnages est disponible dans la documentation : https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation-iterators
Discuter de l'intérêt de chacun de ces échantillonnages.

12. Reproduire les résultats de la question 10 en utilisant les échantillonnages proposés dans **scikit-learn**.