TP Noté ACID 4TIN702S - GROUPE 2

Philippe Pérez de San Roman

6 Décember 2018

1 Introduction

Ce TP noté a pour but de valider vos connaissances pratiques sur les points suivants:

- Affichage de données multidimensionnelles.
- Classification Bayesienne naïve.
- Réduction de dimension avec l'analyse de composantes principales.
- Réduction de dimension avec l'analyse de composantes indépendantes.
- Classification par droite séparante optimisée avec un Perceptron.

Au cours de ce TP noté il vous faudra implémenter un ensemble de fonctions à trous où les parties essentielles du code ont été effacées. Ces fonctions sont appelées dans un ordre pré-établi par un script principal (appelé "Main.m") qui définie les paramètres d'appels de ces fonctions. Ce script et les paramètres qu'il définit ne doivent en aucun cas être modifiés. Pour calculer votre note, il sera écrasé par sa version d'origine.

Dans un premier temps, ce sujet présente le jeu de données utilisé pour les expériences que vous aller mener et implémenter. Vous devrez ensuite implémenter des fonctions d'affichage (plot) qui ne sont pas prises en compte dans la note calculée automatiquement. Cette partie peux vous rapporter jusqu'à 5 points et dépendra de la qualité de votre implémentation. Enfin les méthodes évoquées ci-dessus sont à implémenter et une note automatique sera calculée en fonction de votre capacité à les implémenter. Chaque méthode implémentée rapporte 3 points, pour un sous-total de 15 points. La note final est donc au maximum de 20.

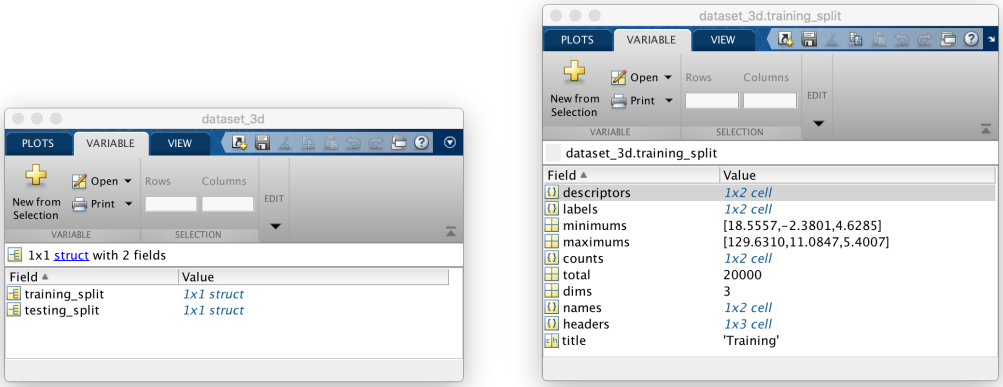
2 Jeu de donnée

Le jeu de données (ou "dataset") contient des mesures effectuées sur deux espèces de serpents: des vipères (Viper) et des couleuvre (Grass Snake). Trois caractéristiques sont mesurées: la longueur (Length), la largeur (Width) et enfin l'écartement entre les yeux (Eye distance).

Il est séparé en deux ensembles (splits):

- **training_split**: ensemble réservé à l'entraînement.
- **testing_split**: ensemble réservé à l'évaluation des résultats.

Chaque ensemble comporte les champs suivants:

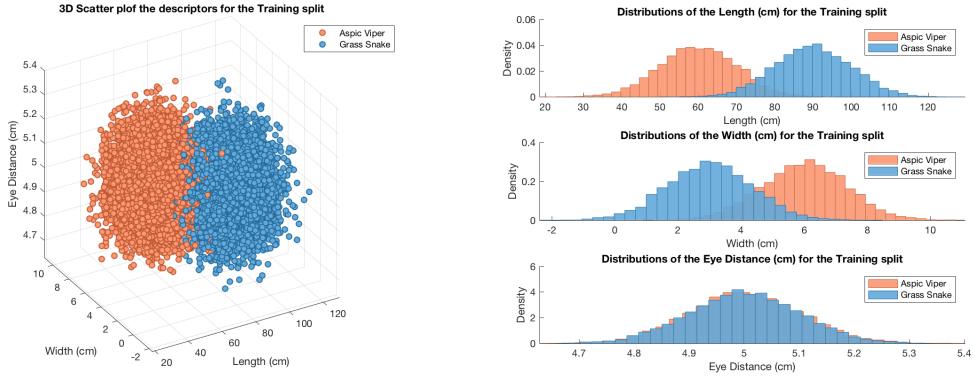


(a) Dataset (b) Split

Figure 1: **(a)** structure d'une variable de type dataset et **(b)** d'une variable de type split

- **descriptors**: Liste de deux matrices de dimensions $N \times D$ contenant les D caractéristiques de N serpents. descriptors $\{1\}$ contient les caractéristiques des vipères et descriptors $\{2\}$ ceux des couleuvres.
- **labels**: Liste de deux vecteurs contenants les labels de N Vipères et N Couleuvres. labels $\{1\}$ contient les labels des vipères (des 1) et labels $\{2\}$ contient les labels des couleuvres (des 2).
- **minimums**: Liste de deux vecteurs contenant les D valeurs minimum de chaque caractéristique. minimums$\{1\}$ contient les valeurs minimum des Vipères tandis que minimums$\{2\}$ contient les valeurs minimum des Couleuvres.
- **maximums**: Liste de deux vecteurs contenant les D valeurs maximum de chaque caractéristiques. maximums$\{1\}$ contient les valeurs maximum des vipères et maximums$\{2\}$ contient les valeurs maximum des couleuvres.
- **counts**: Liste de deux nombres correspondant aux nombres de vipères et de couleuvres. counts$\{1\}$ est le nombre de Vipères et counts$\{2\}$ le nombre de Couleuvres.
- **total**: Nombre total de serpents (égale à counts$\{1\}$+counts$\{2\}$).
- **dims**: Nombre de caractéristiques, par défaut il vaut 3 mais après une projection il peut être plus petit.
- **names**: Liste contenant les noms des deux espèces de serpents: names$\{1\}$ contient 'Aspic Viper' et names$\{2\}$ contient 'Grass Snake'.
- **headers**: Liste contenant les noms des caractéristiques des serpents. Par défaut headers$\{1\}$ contient 'Length', headers$\{2\}$ contient 'Width' et enfin headers$\{3\}$ contient 'Eye Distance'. Après une projection, cela change.
- **title**: Titre de l'ensemble, qui est soit 'Training', soit 'Testing'.

Pour plus de détails, exécutez dans MatLab le début du script "Main.m" et observez la variable "dataset_3d" comme présenté dans la Figure 1.



(a) Nuage 3D

(b) Distributions

Figure 2: **(a)** Résultat attendu pour l'affichage d'un nuage de points 3D. **(b)** Résultat attendu pour l'affichage des distributions.

3 Interprétation

Dans cette partie, il vous est demandé d'implémenter 3 fonctions situées dans le dossier "**Plots**". Les données qui vous sont passées en paramètre et dont vous avez besoins sont déjà explicitées dans le corps de chaque fonction et les paramètres d'affichages sont déjà définis (couleurs, largeurs de traits...). Il vous est demandé d'appeler les bonnes fonctions MatLab pour afficher les données, d'ajouter les légendes, le titre, les noms des axes et de les limites des axes.

3.1 Nuages 3D

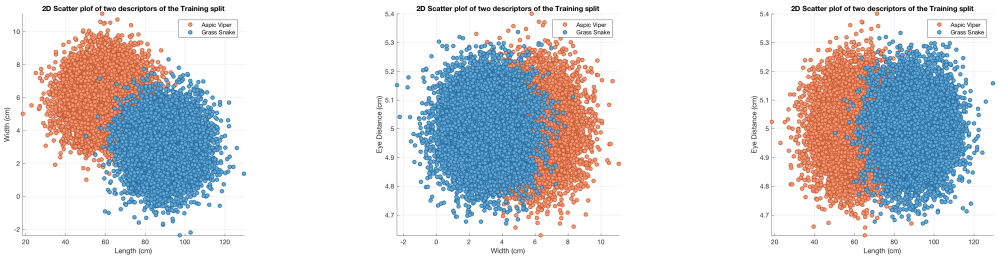
Implémentez la fonction PlotDescriptors3D qui prend en paramètre un split et affiche le nuage de points 3D des caractéristiques des deux catégories de serpent. Le point de vue du plot est fixé à $[-30°, 30°]$. Si votre implémentation est correcte et que vous lancez la première partie du script "Main.m", vous devez obtenir la Figure 1 comme celle présentée dans Figure 2 **(a)**.

3.2 Nuages 2D

Implémentez la fonction PlotDescriptors2D qui prend en paramètre un split , et les indices des axes des deux caractéristiques à afficher l'une contre l'autre dims. Par exemple PlotDescriptors2D (split , [1,3]) affichera les points dont les abscisses seront les valeurs de la première caractéristique et dont les ordonnées seront les valeurs de la troisième caractéristique. Si votre implémentation est correcte et que vous lancez la première partie du script "Main.m", vous devez obtenir les Figures 2, 3 et 4 comme celles présentées dans la Figure 3.

3.3 Distributions 1D

Implémentez la fonction PlotDistributions qui prend en paramètre un split et affiche D sous-figures correspondantes aux distributions des D caractéristiques de chaque espèce de serpents. Les distributions seront affichées à l'aide de la fonction histogram de MatLab dont il faudra spécifier correctement l'option 'Normalization' pour afficher des lois de probabilités entre 0 et 1. Si votre implémentation est



(a) Caractéristique 1 v/s 2 (b) Caractéristique 2 v/s 3 (c) Caractéristique 1 v/s 3

Figure 3: Résultats attendus pour l'affichage d'un nuage de point 2D.

correcte et que vous lancez la première partie du script "Main.m" vous devez obtenir la Figure 5 comme présentée dans la Figure 2 **(b)**.

4 Classification

Dans cette partie il vous est demandé d'implémenter deux méthodes de classification (Bayésienne Naïve et Perceptron) et 3 méthodes de réductions de données (Manuelle, ACP et ACI).

4.1 Naïve Bayésienne

Les fonctions relatives à la classification Bayésienne sont situées dans le dossier "**Bayesian**".

Train: Dans un premier temps, implémentez la fonction TrainMAPClassifier qui prend en paramètre un split contenant les caractéristiques des deux espèces de serpents et qui retourne une liste contenant les deux modèles Bayésiens. Chaque modèle comporte les champs μ, σ et *prior* qui sont calculés comme:

$$\mu_i = mean(descriptors_i) \tag{1}$$

$$\sigma_i = \sqrt{var(descriptors_i)} \tag{2}$$

$$prior_i = \frac{count_i}{\sum_{j=1}^{2} count_j} \tag{3}$$

Classify: Dans un second temps, implémentez la fonction ClassifyWithMAP qui prend en paramètre la liste des deux modèles Bayésiens créés précédemment ("models") et une matrice de caractéristiques *NxD* "descriptors" et retourne un vecteur de prédictions "predictions". Les Vipères auront comme valeurs prédites des 1 et les Couleuvres des 2. La probabilité qu'un échantillon appartienne à une classe sera calculée grâce à la fonction mnvpdf multipliée par la probabilité a-priori (prior) pour obtenir la probabilité à posteriori. Un échantillon sera ensuite considéré comme de la classe 1 et non de la classe 2 si la probabilité à postériori de la classe 1 est supérieure ou égale à celle de la classe 2.

$$proba_i = mvnpdf(d_i, \mu_i, \sigma_i) \times prior_i category = proba_1 >= proba_2 \tag{4}$$

Reste enfin à affecter les labels puisque *category* est un vecteur d'indices logiques: 1 pour la classe 1 et 0 pour la classe 2, or on veut des 2 pour la classe 2.

4.2 Réduction manuelle

Les fonctions de projection sont situées dans le dossier "**Projection**".

Project Descriptors : Implémentez la fonction ProjectDescriptors qui prend en paramètre une matrice de caractéristiques $N \times D$ et une matrice de projection $D \times D-k$ et qui retourne une matrice de caractéristiques projetées de dimensions $N \times D-k$ où k est le facteur de réduction.

Project Split and Dataset : Les fonctions ProjectSplit et ProjectDataset sont fournies.

Test manuel : A la ligne 60 du script "Main.m", modifiez la matrice de projection W pour ne garder que 2 des 3 caractéristiques du dataset. Pour faire votre choix, regardez la distribution des caractéristiques dans la Figure 5, puis supprimez celle qui vous semble la moins pertinente.

On rappel qu'une matrice de projection pour passer de 3 à 2 caractéristiques est de dimension 3×2. Par exemple la matrice suivante garde les caractéristiques 2 et 3 et supprime la première:

$$\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5}$$

4.3 Réduction avec ACP

Nous allons maintenant implémenter la réduction de dimension avec l'analyse de composantes principales. Implémentez la fonction ComputePCA du dossier "**Reduction**". Cette fonction prend en paramètre un split utilisé pour calculer la matrice de projection et le nombre dims qui permet de régler le nombre de dimensions à garder.

La matrice de projection est composée de "dims" plus grand vecteurs propres de la "scatter matrix" (plus grand signifie que la valeur propre associé à ce vecteur est plus grande). La "scatter matrix" est calculée comme le produit de $N-1$ et de la matrice de covariance des caractéristiques considérées. Elle retourne la matrice de projection.

4.4 Réduction avec ACI

Nous allons maintenant implémenter la réduction de dimension avec l'analyse de composantes indépendantes. Implémentez la fonction ComputeICA du dossier "**Reduction**". Cette fonction prend en paramètre un split utilisé pour calculer la matrice de projection et le nombre dims qui permet de régler le nombre de dimensions à garder. Elle retourne la matrice de projection.

La matrice S_b est calculée comme:

$$\begin{aligned} S_b = & \, counts_1 * (\mu_1 - \mu)' * (\mu_1 - \mu) \\ & + counts_2 * (\mu_2 - \mu)' * (\mu_2 - \mu); \end{aligned} \tag{6}$$

Où $count_i$ est le nombre d'échantillon de la classe i, μ_i est la moyenne des échantillons de la classe i et μ est la moyenne des échantillons sur les deux classes.

4.5 Perceptron

FindIncorrects Implémentez la fonction FindIncorrects qui prend en paramètre les coefficients de la ligne line et les caractéristiques descriptors . Elle retourne les caractéristiques qui se situent du mauvais côté de la ligne compte tenue de leur catégorie.

Train Implémentez la fonction TrainPerceptron qui permet d'optimiser la ligne séparatrice du jeu de données.

Les coefficients de la ligne sont mis à jours selon la formule

$$line_{k+1} = linek + \eta \times s \tag{7}$$

Où s est défini comme la somme des distances des points mal placés par rapport à la ligne séparatrice.

On arrêtera la boucle d'optimisation lorsque la norme L2 de la différence de deux vecteurs de coefficient successif est supérieure à epsilon:

$$||line_k - line_{k+1}||_2 > \varepsilon \tag{8}$$