Examen Introduction aux Sciences des Données – Session 1

Durée 2h – Barême indicatif (note sur 20, avec possibilité de 4.5 points en bonus)
Matériel autorisé : une feuille A4 recto-verso de synthèse personnelle du cours, manuscrite (pas de photocopie) + calculatrice non-programmable (téléphone interdit même pour sa calculatrice)
Tous les programmes donnés et demandés sont en Python3

Ce sujet d'examen est constitué de trois exercices indépendants, énoncés sur 3 pages.

1 Régression linéaire (4 pts + bonus 0.5 pt)

(Exercice inspiré de l'Université de Nice, 2014)

On observe 5 spécimens fossiles du *crapaud de Mercure*, un animal disparu (s_1, s_2, s_3, s_4 et s_5). Pour chaque spécimen s_i, on possède la mesure x_i de la longueur de son humérus, et la longueur y_i de son fémur (les longueurs sont en cm).

	s_1	s_2	s_3	s_4	s_5
x	44	62	71	73	87
y	40	57	59	65	77

1. Placer les couples (x, y) sur un graphique adéquat. Observe-t-on une corrélation entre longueurs de l'humérus et du fémur ?
2. Donner une équation de la droite f de régression $y = f(x)$ pour tenter de trouver le plus justement possible la longueur du fémur étant donnée celle de l'humérus : on utilisera l'estimateur des moindres carrés. Placer cette droite sur le graphique précédent.
3. Quelle erreur est faite par ce modèle pour le specimen s_4 ?
4. (**Bonus 0.5 pt**) D'après ce modèle, quelle serait la longueur du fémur d'un crapaud de Mercure dont l'humérus mesure 53 cm ?

2 Clustering par k-moyennes (5 pts + bonus 1 pt)

Soit l'échantillon de données S, composé de 8 exemples (points) décrits sous deux dimensions : $S = \{A(2,10); B(2,8); C(8,4); D(5,8); E(7,5); F(6,4); G(1,2); H(4,9)\}$.

1. Dessiner le graphique faisant apparaître ces points. Quels sont les 3 clusters que vous pouvez identifier visuellement ?
2. En prenant comme centroïdes initiaux les points A, B et C, appliquer l'algorithme des k-moyennes pour regrouper ces points en trois clusters, en utilisant la distance de Manhattan.
3. Dans quel cluster se placera le point $P(2,9)$? Justifier graphiquement et par le calcul.
4. **(Bonus 1 pt)** Est-il possible de minimiser le nombre d'itérations par un autre choix initial des centroïdes ? Justifier la réponse.

3 Sélection de modèle de classification (11 pts + bonus 3 pts)

Pour le jeu de données accessible sous `sklearn` via la fonction `load_mystere`, possédant n exemples, nous cherchons à savoir quelle solution, entre les arbres de décision (C_1) et les k-plus proches voisins (C_2), serait la meilleure pour un problème de classification supervisée binaire – meilleure au sens du taux de bonne classification.

3.1 Une fonction à comprendre (2 pts)

Pour cela, la fonction suivante est définie :

```
# C1_pred est le vecteur de predictions faites par le classifieur C1
# C2_pred est le vecteur de predictions faites par le classifieur C2
# truth est le vecteur des vraies classes
# Les trois vecteurs sont supposés de meme taille (nombre d'exemples)
# Les trois vecteurs sont à valeurs booleennes

def carmmen(C1_pred, C2_pred, truth):
    n01 = n10 = 0
    nex = truth.shape[0]
    for i in range(nex):
        if C1_pred[i] != C2_pred[i]:
            if C1_pred[i] == truth[i]:
                n10 += 1
            else:
                n01 += 1
    return [n01, n10]
```

1. A quoi correspondent les valeurs retournées `n01` et `n10` au regard des paramètres de cette fonction ?
2. Ces deux valeurs sont considérées dans un test statistique vu en TDM et TP. A quelle question permet de répondre ce test ?

3.2 Cas concret d'utilisation de cette fonction (7 pts + bonus 1 pt)

Le code ci-après suppose que tous les `import` de bibliothèques et classes nécessaires ont été faits. Les numéros de lignes sont indiqués pour faciliter l'écriture des questions et réponses de cet exercice.

1. (3 pts) Indiquer un commentaire pertinent (environ une ligne) pour chaque ligne de code numéro : 2, 5, 11, 12, 14, 15, 16. Par exemple, le commentaire de la ligne 3 pourrait être : `on stocke le nombre d'exemples du jeu de données, et le nombre de variables qui les décrivent`.
2. (2.5 pts + bonus) Les lignes 7 à 18, semblent indiquer que l'on reproduit une expérience un certain nombre de fois, puis qu'on moyenne leurs résultats.
   (a) Combien M d'expériences sont menées ? (question facile, pas piège)
   (b) Que fait chaque expérience ? Quelles sont les mesures calculées dans chacune ?
   (c) **(bonus 1 pt)** Pour quelle raison semble-t-il pertinent de mener ces M expériences plutôt qu'une seule ? (indice : pour la même raison qu'il est pertinent de reproduire M fois un hold-out pour estimer un score / une erreur).

(d) Que calcule la ligne 19 ?

3. (1.5 pts) Le résultat de l'interprétation du code ci-avant donne :

```
299 12
7.343059490084987 0.7955555555555556 0.6055555555555555
```

(a) Quelle est la dimension du jeu de données mystère (en termes d'exemples et de variables les décrivant) ?

(b) Quelle solution parmi les deux classifieurs est la meilleure avec 95% de confiance ? Justifier.

```
1. mydataset = load_mystere()
2. X, Y = mydataset.data, mydataset.target
3. nbex, nbvar = X.shape[0],  X.shape[1]
4. print(nbex,nbvar)

5. dt = DecisionTreeClassifier()
6. kppv = KNeighborsClassifier()

7. N01, N10 = np.zeros((10), dtype = int), np.zeros((10), dtype = int)
8. SCORE_dt, SCORE_kppv = np.zeros((10), dtype = float), np.zeros((10), dtype = float)

9. for iexp in range(10):
11.     Xtrain, Xtest, ytrain, ytest = train_test_split(X,y,test_size=0.30, random_state=iexp)
12.     dt.fit(Xtrain, ytrain)
13.     kppv.fit(Xtrain, ytrain)
14.     SCORE_dt[iexp], SCORE_kppv[iexp] = dt.score(Xtest, ytest), kppv.score(Xtest, ytest)
15.     dt_pred, kppv_pred = dt.predict(Xtest), kppv.predict(Xtest)
16.     N01[iexp], N10[iexp] = carmmen(dt_pred, kppv_pred, ytest)

17. meanN01, meanN10 = N01.mean(), N10.mean()
18. score_dt, score_kppv = SCORE_dt.mean(), SCORE_kppv.mean()

19. statest = (math.pow((abs(meanN01-meanN10)-1),2))/(meanN01+meanN10)
20. print(statest, score_dt, score_kppv)
```

3.3 Pour parfaire la sélection de modèle (2 pts + bonus 2 pts)

1. Qu'appelle-t-on *hyper-paramètre* d'un algorithme d'apprentissage ?
2. Quel est un hyper-paramètre fondamental des arbres de décisions d'une part, et des k-plus proches voisins d'autre part ? Pour chacun, quel est son rôle dans l'algorithme considéré ?
3. (**bonus 2 pts**) Remplacer les lignes 5 et 6 par du code (même approximatif au regard de la syntaxe et des noms de fonctions `sklearn`), pour qu'au final les lignes suivantes permettent d'affiner cette comparaison de deux solutions aux regards de leurs meilleurs hyper-paramètres ?