EPU GB5 **Introduction à l'IA pour la biologie** 25 octobre 2023

**TD n$^o$ 4 : Classifieurs Bayesiens**

**Exercice 1 :** (**Jouer au tennis**)

| Jour | Ciel | Température | | Humidité | | Vent | Jouer au tennis ? |
|---|---|---|---|---|---|---|---|
| 1 | Ensoleillé | 27,5 | Chaude | 85 | Élevée | Faible | Non |
| 2 | Ensoleillé | 25 | Chaude | 90 | Élevée | Fort | Non |
| 3 | Couvert | 26,5 | Chaude | 86 | Élevée | Faible | Oui |
| 4 | Pluie | 20 | Tiède | 96 | Élevée | Faible | Oui |
| 5 | Pluie | 19 | Fraiche | 80 | Normale | Faible | Oui |
| 6 | Pluie | 17,5 | Fraiche | 70 | Normale | Fort | Non |
| 7 | Couvert | 17 | Fraiche | 65 | Normale | Fort | Oui |
| 8 | Ensoleillé | 21 | Tiède | 95 | Élevée | Faible | Non |
| 9 | Ensoleillé | 19,5 | Fraiche | 70 | Normale | Faible | Oui |
| 10 | Pluie | 22,5 | Tiède | 80 | Normale | Faible | Oui |
| 11 | Ensoleillé | 22,5 | Tiède | 70 | Normale | Fort | Oui |
| 12 | Couvert | 21 | Tiède | 90 | Élevée | Fort | Oui |
| 13 | Couvert | 25,5 | Chaude | 75 | Normale | Faible | Oui |
| 14 | Pluie | 20,5 | Tiède | 91 | Élevée | Fort | Non |

1. Dans un premier temps, on s'intéresse aux caractéristiques discrètes (les valeurs de température et d'humidité sont qualitatives). Calculez le tableau des effectifs par caractéristiques et par valeur de ces dernières.
2. Pour cette question encore, on s'intéresse aux caractéristiques discrètes (les valeurs de température et d'humidité sont qualitatives). On souhaite prédire la classe de la donnée "x = (Ciel = Ensoleillé, Température = Fraiche, Humidité = élevée, Vent = Fort)".
   (a) Pour cela, écrivez la formule de Bayes qui permet d'estimer $Pr[jouer = oui \mid x, \mathcal{X}]$.
   (b) Estimez les différents termes apparaissant dans la formule précédente.
   (c) Quelle sera la classe prédite pour l'événement "x = (Ciel = Ensoleillé, Température = Fraiche, Humidité = élevée, Vent = Fort)" ?
3. On s'intéresse maintenant aux caractéristiques continues (les valeurs des températures et d'humidité sont quantitatives). Dans ce cas, et en l'absence d'autres informations, on suppose que la distribution de la valeur de l'attribut est normale. Pour cela, on calcule la moyenne et l'écart-type de chaque attribut numérique et pour chaque valeur de l'attribut cible ($y$).
   Rappelons que la fonction de densité de probabilité d'une distribution normale est :

$$Pr[X = x \mid \mathcal{X}, \tilde{\mathcal{N}}(\overline{x}, \sigma^2)] = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\overline{x})^2}{2\sigma^2}}$$

   (a) Calculez les moyennes et écarts-types de la température et de l'humidité dans le cas où la classe est "jouer=oui" et dans le cas où la classe est "jouer=non".
   (b) Calculez $Pr[\text{Température} = 18 \mid jouer = oui, \mathcal{X}]$
   (c) Calculez $Pr[\text{Température} = 18 \mid jouer = non, \mathcal{X}]$
   (d) Calculez $Pr[\text{Humidité} = 90 \mid jouer = oui, \mathcal{X}]$
   (e) Calculez $Pr[\text{Humidité} = 90 \mid jouer = non, \mathcal{X}]$
   (f) Calculez la classe prédite pour journée J =(Ciel=Ensoleillé, Température=18, Humidité=90, Vent=Fort).

**Exercice 2 :** (**Tennis : avec ou sans données manquantes dans l'ensemble d'apprentissage**)

On cherche maintenant à estimer la classe pour la journée J =(Ciel=Couvert, Température=18, Humidité=90, Vent=Fort).

1. Expliquez pourquoi la méthode précédente ne peut pas donner de bon résultat.
2. Utilisez la méthode de Laplace pour corriger le problème soulevé à la question précédente.

**Exercice 3 :** (**Tennis : Données manquantes pour la prédiction**)

1. Quelle est la classe prédite en utilisant la règle de Bayes d'une journée ensoleillée avec vent faible ?
2. Quelle est la classe d'une journée ensoleillée, température de 23°C, humidité de 70 % et vent faible ?
3. Quelle est la probabilité de jouer un jour où la température est de 23°C ?
4. Quelle est la probabilité de jouer un jour où l'humidité est comprise entre 60 et 75 % ?

**Exercice 4 :** (**Utilisation de `GaussianNB` avec `scikit-learn`**)

Nous allons réutiliser le jeu de données `iris`.

1. Importation des données :
```
from sklearn import datasets
iris = datasets.load_iris()
data = iris.data # Pour un accès plus rapide
target = iris.target # Les labels associés à chaque enregistrement
```
2. Construction du classifieur et prédiction :
```
from sklearn.naive_bayes import GaussianNB
clf = GaussianNB()
clf.fit(data, target) # On aurait aussi pu utiliser le dataframe df
result = clf.predict(data)
```
3. Évaluez le nombre d'erreur faites sur l'ensemble d'apprentissage.
```
from sklearn.metrics import accuracy_score
accuracy_score(result, target) # 96% de réussite
```
4. Construction de la matrice de confusion :
```
from sklearn.metrics import confusion_matrix
conf = confusion_matrix(target, result)
```
5. Représentation graphique de la matrice de confusion :
```
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib as mpl
sns.heatmap(conf, square=True, annot=True, cbar=False,
            xticklabels=list(iris.target_names),
            yticklabels=list(iris.target_names))
plt.xlabel('valeurs prédites')
plt.ylabel('valeurs réelles');
plt.show()
```
S'il n'y a pas seaborn installé, on peut utiliser :
```
import matplotlib.pyplot as plt
import matplotlib as mpl
plt.matshow(conf, cmap='PiYG');
plt.show()
```
6. Il faut maintenant envisager une validation externe. Pour cela on va couper l'ensemble des données en deux ensembles (entrainement, test) :
```
from sklearn.model_selection import train_test_split # version 0.18.1
data_test = train_test_split(data, target, random_state=0, train_size=0.5)
(data_train, data_test, target_train, target_test) = data_test
```
7. Il faut maintenant, ré-entraîner le modèle sur l'ensemble d'apprentissage, recalculer le taux d'erreur sur l'ensemble d'apprentissage, contruire la matrice de confusion...
8. On se place maintenant en 2D en ne gardant que les longueurs/largeurs des sépales :
```
data = iris.data[:, :2]
target = iris.target
```
Reconstruire un modèle qui se base uniquement sur ces deux mesures.
9. Construction des zones de décision :
   a. après avoir calculer les valeurs minimales et maximales, on construit un maillage ($h$ est le pas du maillage) :
```
x = np.arange(x_min, x_max, h)
y = np.arange(y_min, y_max, h)
xx, yy = np.meshgrid(x,y)
data_samples = list(zip(xx.ravel(), yy.ravel()) )
```
   b. On prédit sur chacun des points du maillage :
```
Z = clf.predict(data_samples)
Z1 = Z.reshape(xx.shape)
plt.figure(1)
plt.pcolormesh(xx, yy, Z1)
```
   c. on affiche le tout :
```
plt.xlim(xx.min() - .1, xx.max() + .1)
plt.ylim(yy.min() - .1, yy.max() + .1)
plt.xlabel('Longueur du sepal (cm)')
plt.ylabel('Largeur du sepal (cm)');
colors = ['violet', 'orange', 'red']
C = [colors[x] for x in target]
plt.scatter(data[:, 0], data[:, 1], c=C)
plt.show()
```