femto-st SCIENCES & TECHNOLOGIES

M2 info. I2A : Apprentissages Machine

Jean-François COUCHOT
Université de Franche-Comté

cnrs UBFC UNIVERSITÉ BOURGOGNE-FRANCHE-COMTÉ UNIVERSITÉ DE FRANCHE-COMTÉ SUP MICRO TECH ENSMM utbm université de technologie Belfort-Montbéliard

Plan

Plan

Plan

Apprentissage suppervisé probabiliste

Exemple [1] de classification

Preg.	Gluc.	BloodP.	SkinThick.	Insul.	BMI	DPF	Age	Outcome
6	148	72	35	0	33.6	0.627	50	YES
1	85	66	29	0	26.6	0.351	31	NO
8	183	64	0	0	23.3	0.672	32	YES
1	89	66	23	94	28.1	0.167	21	NO
				...				

- Connaissant une valeur pour chaque attribut de mesure : predire Outcome
- Comparer les probabilités suivantes et conclure :

$$\Pr[\text{OutCome} =' \text{YES}' | \text{Preg} = p, \text{Gluc} = g, \ldots, \text{Age} = a]$$
$$\Pr[\text{OutCome} =' \text{NO}' | \text{Preg} = p, \text{Gluc} = g, \ldots, \text{Age} = a]$$

- A évaluer : $\Pr[y_1 | X_1, X_2, \ldots, X_d]$ et $\Pr[y_2 | X_1, X_2, \ldots, X_d]$
- Remarque : attributs discrets (Preg., Gluc, ...) ou réels (BMI, DPF)

1. https ://www.kaggle.com/uciml/pima-indians-diabetes-database

Bayes : théorème et conséquences

Théorème (Bayes)

$$\Pr[Y|X_1,\ldots,X_d] = \frac{\Pr[Y] \times \Pr[X_1,\ldots,X_d|Y]}{\Pr[X_1,\ldots,X_d]}$$

Simplifications immédiates

$$\left.\begin{array}{l} \Pr[y_1|X_1,\ldots,X_d] = \frac{\Pr[y_1]\times\Pr[X_1,\ldots,X_d|y_1]}{\Pr[X_1,\ldots,X_d]} \\ \Pr[y_2|X_1,\ldots,X_d] = \frac{\Pr[y_2]\times\Pr[X_1,\ldots,X_d|y_2]}{\Pr[X_1,\ldots,X_d]} \end{array}\right\} \begin{array}{l} \text{m̂ dénom. } \Pr[X_1,\ldots,X_d] \\ \rightsquigarrow \text{ son calcul : inutile} \end{array}$$

A évaluer : $\Pr[y_j]$, et $\Pr[X_1,\ldots,X_d|y_j]$

- $\Pr[y_j]$: fréquence d'apparition de la valeur y_j pour l'attribut y
- $\Pr[X_1,\ldots,X_d|y_j] = \Pr[X_1|y_j] \times \cdots \times \Pr[X_d|y_j] = \prod_{i=1}^{d} \Pr[X_i|y_j]$:
  - Naïveté : indépendance de chaque X_i p.r. aux autres $X_{i'}$, $i' \neq i$, conditionnell[t] à y_j

Attention

- Publication des proba. $\Pr[X_i|y_j] \leftrightarrow$ fuite d'information sur X

Exemple avec des données discrètes-1

Données[2], question et premières probabilités

Age	Income	Gender	Missed Payment
Young	Low	Male	Yes
Young	High	Female	Yes
Medium	High	Male	No
Old	Medium	Male	No
Old	High	Male	No
Old	Low	Female	Yes
Medium	Low	Female	No
Medium	Medium	Male	Yes
Young	Low	Male	No
Old	High	Female	No

- Q : défaut de payement pour une jeune femme avec un revenu moyen ?
- $y_1 \equiv'$ Missed Payment$' =' $ YES$'$, $y_2 \equiv'$ Missed Payment$' =' $ NO$'$
- Comparer $\Pr[y_1|\text{Age} = \text{Young}, \text{Income} = \text{Medium}, \text{Gender} = \text{Female}]$ et $\Pr[y_1|\text{Age} = \text{Young}, \text{Income} = \text{Medium}, \text{Gender} = \text{Female}]$
- $\Pr[y_1] = \frac{4}{10}$ et $\Pr[y_2] = \frac{6}{10}$

Probabilités pour chaque attribut conditionnellement à y_j

- $\Pr[\text{Age} = \text{Young}|y_1] = \frac{2}{4}$, $\Pr[\text{Age} = \text{Young}|y_2] = \frac{1}{6}$ ...
- $\Pr[\text{Income} = \text{Medium}|y_1] = \frac{1}{4}$, $\Pr[\text{Income} = \text{Medium}|y_2] = \frac{1}{6}$ ...
- $\Pr[\text{Gender} = \text{Female}|y_1] = \frac{2}{4}$, $\Pr[\text{Gender} = \text{Female}|y_2] = \frac{2}{6}$ ...

2. Yilmaz, E., Al-Rubaie, M., & Chang, J. M. (2019). Locally differentially private naive bayes classification. arXiv preprint arXiv:1905.01039.

Exemple avec des données discrètes-2

Evaluation de $\Pr[y_j|\text{Age} = \text{Young}, \text{Income} = \text{Medium}, \text{Gender} = \text{Female}]$

- $Pr[y_1] \times \Pr[\text{Age} = \text{Young}|y_1] \times \Pr[\text{Income} = \text{Medium}|y_1] \times \Pr[\text{Gender} = \text{Female}|y_1] = \frac{4}{10} \times \frac{2}{4} \times \frac{1}{4} \times \frac{2}{4} = \frac{1}{40} \approx 0.025$
- $Pr[y_2] \times \Pr[\text{Age} = \text{Young}|y_2] \times \Pr[\text{Income} = \text{Medium}|y_2] \times \Pr[\text{Gender} = \text{Female}|y_2] = \frac{6}{10} \times \frac{1}{6} \times \frac{1}{6} \times \frac{2}{6} = \frac{1}{180} \approx 0.0056$

Réponse

La probabilité qu'elle ratte son paiement est beaucoup plus importante que celle opposée.

Exemple avec des données continues-1

Données [3], question et premières probabilités

sexe	taille (cm)	masse (kg)	point. (cm)
masc.	182	81.6	30
masc.	180	86.2	28
masc.	170	77.1	30
masc.	180	74.8	25
fém.	152	45.4	15
fém.	168	68.0	20
fém.	165	59.0	18
fém.	175	68.0	23

- Q : sexe d'une personne mesurant 183cm, pesant 59kg et dont les pieds mesurent 20cm ?
- $y_1 \equiv'$ Sexe$' ='$ masc.$'$, $y_2 \equiv'$ Sexe$' ='$ fém.$'$
- Comparer $\Pr[y_1|\text{Taille} = 183, \text{Masse} = 59, \text{Point.} = 20]$ et $\Pr[y_2|\text{Taille} = 183, \text{Masse} = 59, \text{Point.} = 20]$
- $\Pr[y_1] = \Pr[y_2] = \frac{1}{2}$

Probabilités pour chaque attribut X_i conditionnellement à y_j : $\mathcal{N}(\mu_{i,j}, \sigma^2_{i,j})$

1. Estimation des parmètres $\mu_{i,j}$ et $\sigma_{i,j}$:

Sexe	μ_{taille}	σ^2_{taille}	μ_{masse}	σ^2_{masse}	$\mu_{point.}$	$\sigma^2_{point.}$
masc.	178	29.3	79.9	25.5	28.25	5.58
fém.	165	92.7	60.1	114	19	11.3

2. Densité de probabilité de $\mathcal{N}(\mu_{i,j}, \sigma^2_{i,j})$: $\frac{1}{\sqrt{2\pi\sigma^2_{i,j}}} \exp\left(-\frac{1}{2\sigma^2_{i,j}}(x - \mu_{i,j})^2\right)$,
3. $\Pr[\text{taille} = 183|y_1]$ remplacé par $\frac{1}{\sqrt{2\pi \times 29.3}} \exp\left(\frac{-1}{2\times 29.3}(183 - 178)^2\right) \approx 0.0481$

3. Classification naïve bayésienne, Wikipedia

Exemple avec des données continues-2

Valeurs numérique des probabilités pour chaque attribut X_i conditionnellement à y_j

- $\Pr(taille = 183|y_1) \equiv 0.0481$, $\Pr(poids = 59|y_1) \equiv 0.0000146$ et $\Pr(point. = 20|y_1) \equiv 0.000381$
- $\Pr(taille = 183|y_2) \equiv 0.00721$, $\Pr(poids = 59|y_2) \equiv 0.0372$ et $\Pr(point. = 20|y_2) \equiv 0.114$

Evaluation de $\Pr[y_j|taille = 183, poids = 59, point. = 20]$

- $Pr[y_1] \times \Pr(taille = 183|y_1) \times \Pr(poids = 59|y_1) \times \Pr(point. = 20|y_1) \equiv 1.3404 \times 10^{-10}$
- $Pr[y_2] \times \Pr(taille = 183|y_2) \times \Pr(poids = 59|y_2) \times \Pr(point. = 20|y_2) \equiv 1.52 \times 10^{-5}$

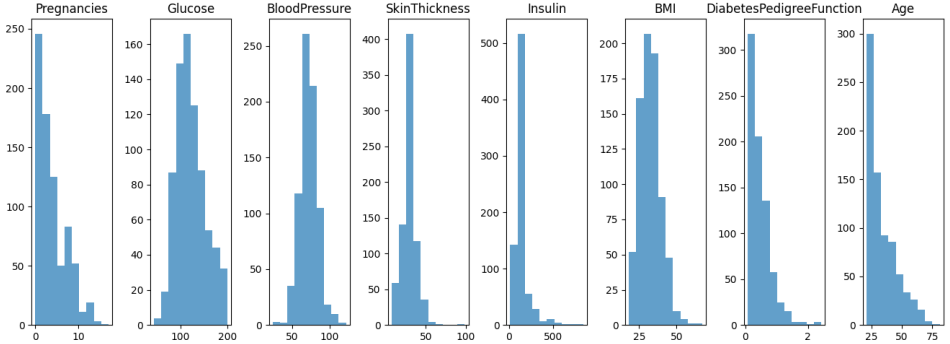
Réponse

La personne est probablement une femme.

Prétraitement du jeu de données

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np



df = pd.read_csv('diabetes.csv')
df['Outcome'].replace({'NO': 0, 'YES': 1},inplace=True)
X = df.iloc[:, 0:8]
y = df.iloc[:, 8]

for column in ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin']:
    X[column] = X[column].replace(0, np.NaN)
    mean = int(X[column].mean(skipna=True))
    X[column] = X[column].replace(np.NaN, mean)
for column in ['BMI','DiabetesPedigreeFunction']:
    X[column] = X[column].replace(0, np.NaN)
    mean = X[column].mean(skipna=True)
    X[column] = X[column].replace(np.NaN, mean)




"""
fig, axes = plt.subplots(nrows=1, ncols=len(X.columns), figsize=(15, 5))
for i, col in enumerate(X.columns):
    axes[i].hist(X[col], bins=10, alpha=0.7)
    axes[i].set_title(col)
plt.tight_layout()
plt.show()
```

Distribution des données



Distributions

- Normales : Glucose, BloodPressure, SkinThickness, Insulin,BMI
- Autres : Pregnancies, DiabetesPedigreeFunction, Age

Apprentissage naïf bayésien (code)

```python
import pretraitement

from sklearn.model_selection import cross_val_score, StratifiedShuffleSplit, train_test_split
from sklearn.naive_bayes import GaussianNB
import numpy as np

X = pretraitement.X
y = pretraitement.y

gnb = GaussianNB()

cv = StratifiedShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
scores = cross_val_score(gnb, X, y, cv=cv, scoring='accuracy')

mean_score = np.mean(scores)
std_deviation = np.std(scores)

print(f"Scores de validation croisée: {scores}")
print(f"Précision moyenne: {mean_score:.4f}")
print(f"Écart type des précisions: {std_deviation:.4f}")

gnb = GaussianNB()
gnb.fit(X.values, y)
print(np.array(gnb.predict([[1, 184, 84, 33, 82, 35.5, 0.355, 41]])))
```

Plan

Introduction puis formalisation

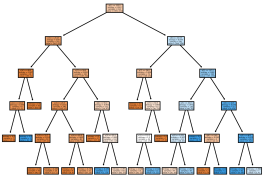
Même exemple que pour Bayes naïf

Preg.	Gluc.	BloodP.	SkinThick.	Insul.	BMI	DPF	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
			...					

- Connaissant une valeur pour chaque attribut numérique $X_1, \ldots, X_d$, de mesure : predire $Y = Y_i$

Formalisation

- Base $D = \{(X_{11}, \ldots, X_{1d}, Y_1), \ldots, (X_{n1}, \ldots, X_{nd}, Y_n)\}$ de n tuples $t_1, \ldots t_n$
- Hypothèse : données réprésentées sous la forme d'arbre de décision
  - Nœud feuille : distribution des valeurs de l'attribut cible Y
  - Nœud interne : prédicat sur un attribut entrée X_i
  - Arête sortante d'un nœud : validité ou non du prédicat



Attention

- Publication de l'arbre $\leftrightarrow$ fuite d'information sur X et y

Idées générales

Type d'arbres de décision

- Arbres de classification : prédiction de la classe de l'attribut cible
- Arbres de régression : prédire une valeur numérique pour l'attribut cible (durée de séjour d'un patient dans un hôpital...)

Processus récursif glouton de contruction d'un arbres

- Choix de l'attribut d'entrée X_i et d'un ensemble δ de valeurs qu'elle peut prendre (discret ou continu) réalisant la "meilleure scission du jeu de données"
- Partitionnement de l'ensemble des données selon X_i et δ
- Jusqu'à un critère de terminaison :
  - même valeur de la caractéristique-cible pour tous les sous-ensembles
  - prédiction plus améliorée par une séparation
  - profondeur maximale atteinte

Algo. général de consructº de l'arbre [4]

```python
class DecisionTree():

    def _create_tree(self, data, current_depth):
        # 1 First stopping criteria
        if current_depth > self.max_depth: return None

        # 2 Find best split
        spl1_data, spl2_data, spl_feat_idx, spl_feat_val, spl_entropy
                    = self._find_best_spl(data)

        # 3 Create current node
        label_probs = self._find_label_probs(data)
        info_gain = self._entropy(label_probs) - spl_entropy
        node = TreeNode(data, spl_feat_idx, spl_feat_val, label_probs, info_gain)

        # 4 Other stopping criterion acording to split data
        if stopping_criterion(..): return node

        # 5 recursive building child nodes
        node.left = self._create_tree(spl1_data, current_depth+1)
        node.right = self._create_tree(spl2_data, current_depth+1)
        return node
```

#1 Profondeur maximale atteinte : arrêt

#2 Attribut (`spl_feat_idx`) et val. assoc. (`spl_feat_val`) dont le partitionnt en `spl1_data` et `spl2_data` minimise l'hétérogénéité (ici l'entropie `spl_entropy`)

#3 Nœud courant créé

#5 Construction des nœuds enfant avant envoi

4. inspiré de https://github.com/enesozeren/machine_learning_from_scratch/blob/main/decision_trees/decision_tree.py

Nœud : partitionnement

Objectif

- Choix d'un attribut et d'une valeur associée dont le partitionnement en 2 sous-ensembles diminue l'hétérogénéité des nœuds
- Hétérogénéité : mesurée selon Gini index[5], Information Gain

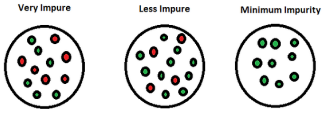
Information Gain pour un ensemble d'apprentissage $T = \{(X_1^1, X_2^1, X_3^1, \ldots, X_k^1, y^1) \ldots\}$

- v_a : une partie du domaine de l'attribut X_a (valeur précise, ensemble de val., intervalle...)
- $S_a(v_a) = \{(X_1, X_2, X_3, ..., X_k, y) \in T |\ X_a \in v_a\}$: sous ens. de T dont la val. de X_a appartient à v_a
- Entropie $H(T) = -\sum_{y \in Y} p(y) \log_2 p(y)$ (élevée $\equiv$ hétérogène)
- Entropie conditionnelle à (X_a, v_a) $H(T|(X_a, v_a)) = \frac{|S_a(v_a)|}{|T|}.H(S_a(v_a)) + \frac{|S_a(\overline{v_a})|}{|T|}.H(S_a(\overline{v_a}))$: moyenne des entropies après partionnement de T selon (X_a, v_a).
- $IG(T, (X_a, v_a)) = H(T) - H(T|(X_a, v_a))$

5. https://en.wikipedia.org/wiki/Diversity_index#Gini%E2%80%93Simpson_index

Partitionnement : exemples-1

Illustrations de l'entropie [6]



- Très hétérogène : $H(T_V) = -\left(\frac{6}{12}\log_2(\frac{6}{12}) + \frac{6}{12}\log_2(\frac{6}{12})\right) = 1$
- Moins hétérogène : $H(T_L) = -\left(\frac{4}{14}\log_2(\frac{4}{14}) + \frac{10}{14}\log_2(\frac{10}{14})\right) = 0.86$
- Uniforme : $H(T_M) = -\left(\frac{9}{9}\log_2(\frac{9}{9})\right) = -\log_2(1) = 0$

Exemple : mutations génétiques causant le cancer [7]

- Attributs : ID (numéro de ligne), X_i (présence de mutat° génétique i), Y (cancer ou non)
- $1 \equiv$ présence, $C \equiv$ Cancer, $NC \equiv$ No Cancer,
- Quel Y pour une valeur de (X_0, X_1, X_2, X_3) ?

ID	X0	X1	X2	X3	Y
1	1	1	1	0	C
2	1	1	0	1	C
3	1	0	1	1	C
4	0	1	1	0	C
5	0	0	0	0	NC
6	0	1	0	0	NC
7	1	1	0	0	NC

6. https://www.numpyninja.com/post/what-is-entropy-and-information-gain-how-are-they-used-to-construct-decision-trees
7. https://en.wikipedia.org/wiki/Information_gain_(decision_tree)

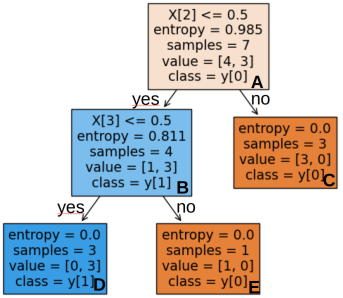
Partitionnement : exemples-2

- Entropie dataset complet T : 4 pers. sur 7 tq. $Y = C$: $H(T) = -\left(\frac{4}{7}\log_2(\frac{4}{7}) + \frac{3}{7}\log_2(\frac{3}{7})\right) = 0.985$
- Tentative de partitionnement de T avec X_0 et $v_0 = \{0\}$ (et donc $\overline{v_0} = \{1\}$) :
  - $S_0(\{0\}) = \{4, 5, 6\}$ et $S_0(\{1\}) = \{1, 2, 3, 7\}$
  - $H(S_0(\{0\})) = -\left(\frac{1}{3}\log_2(\frac{1}{3}) + \frac{2}{3}\log_2(\frac{2}{3})\right) = 0.918$,
    $H(S_0(\{1\})) = -\left(\frac{3}{4}\log_2(\frac{3}{4}) + \frac{1}{4}\log_2(\frac{1}{4})\right) = 0.811$
  - $H(T|(X_0, \{0\})) = \frac{3}{7}.H(S_0(\{0\})) + \frac{4}{7}.H(S_0(\{1\})) = 0.857 \rightsquigarrow$
    $IG(T, (X_0, \{1\})) = 0.985 - 0.857 = 0.128$
- Tentative de partitionnement de T avec X_2 et $v_2 = \{0\}$ (et donc $\overline{v_0} = \{1\}$) :
  - $S_2(\{0\}) = \{2, 5, 6, 7\}$, $S_2(\{1\}) = \{1, 3, 4\}$
  - $H(S_2(\{0\})) = -\left(\frac{1}{4}\log_2(\frac{1}{4}) + \frac{3}{4}\log_2(\frac{3}{4})\right) = 0.811$,
    $H(S_2(\{1\})) = -\left(\frac{3}{3}\log_2(\frac{3}{3}) + \frac{0}{3}\log_2(\frac{0}{3})\right) = 0$
  - $H(T|(X_2, \{0\})) = \frac{4}{7}.H(S_2\{0\})) + \frac{3}{7}.H(S_2(\{1\})) = 0.463 \rightsquigarrow$
    $IG(T, (X_2, \{1\})) = 0.985 - 0.463 = 0.522$, le plus élevé parmi X_0, X_1, X_2 et X_3
- Partitionnement selon $X_2 = 0$, avec à fille gauche $T = S_2(\{0\})$ et fille droite $T = S_2(\{1\})$

Partitionnement : exemples-3

Code

```
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier, plot_tree
import pandas as pd

df = pd.read_csv('cancermuutation.csv')
df['Y'].replace({'C': 0, 'NC': 1},inplace=True)
X,y = df.iloc[:, 1:5], df.iloc[:, 5]
clf = DecisionTreeClassifier(criterion="entropy",max_depth=4)
clf.fit(X,y)

plt.figure() ; plot_tree(clf,class_names=True,filled=True)
plt.show()

print(clf.predict([[0, 1, 0, 0]])) # 1 <=> NC
```



Analyse

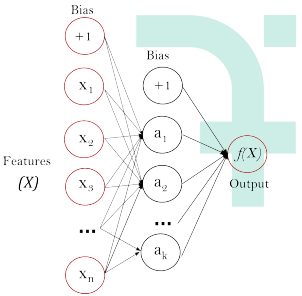
- Noeud A : partition^{mnt} sera fait selon $X_2 = 0$, 7 enregistr^{mnt}. (4 valent 0, 3 valent 1)
- Noeud B : res. du partition^{mnt} de A selon $X_2 = 0$, 4 enregistr^{mnt}. (1 vaut 0 et 3 valent 1)
- Noeud C : issu du partition^{mnt} de A selon $X_2 \neq 0$, 3 enregistr^{mnt}. (tous valent 0)
- Dans $(0, 1, 0, 0)$, $X_2 = 0$, puis $X_3 = 0 \rightsquigarrow$ classifié en "1"

Arbre de décision (code)

```
import pretraitement
import matplotlib.pyplot as plt
from sklearn.model_selection import cross_val_score, StratifiedShuffleSplit, train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
import numpy as np

X,y = pretraitement.X,pretraitement.y

clf = DecisionTreeClassifier(criterion="entropy",max_depth=5)
cv = StratifiedShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
scores = cross_val_score(clf, X, y, cv=cv, scoring='accuracy')

mean_score = np.mean(scores)
std_deviation = np.std(scores)
print(f"Scores de validation croisée: {scores}")
print(f"Précision moyenne: {mean_score:.4f}")
print(f"Écart type des précisions: {std_deviation:.4f}")

clf = DecisionTreeClassifier(criterion="entropy",max_depth=5)
clf.fit(X.values, y)

print(np.array(clf.predict([[1, 184, 84, 33, 82, 35.5, 0.355, 41]])))

plt.figure()
plot_tree(clf,filled=True)
plt.title("Decision tree trained on all the pima features")
plt.savefig("dt.svg")
```

Plan

Apperçu du modèle [8]



Le modèle de Perceptron multi couches (MLP)

- **Couche d'entrée** : reçoivent les caractéristiques d'entrée
- **Chaque couches cachée** j : calcule $y_j = g(w_1 x_1 + \ldots w_n x_n + b_j)$ résultat de l'application de la fonction d'activation non linéaire $g : R^{n+1} \rightarrow R$ sur une somme pondérée avec b_j le biais de cette couche
- **Couche de sortie** : produisent les prédictions

Exemples de fonctions d'Activation

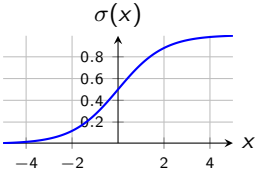


Figure – *Sigmoïde* : $\sigma(x) = \frac{1}{1+e^{-x}}$

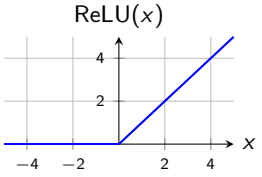


Figure – *ReLU* (Rectified Linear Unit) : $f(x) = \max(0, x)$

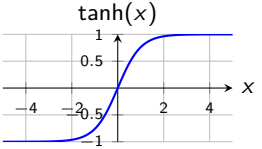


Figure – *Tanh* : $\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$

8. https://scikit-learn.org/stable/modules/neural_networks_supervised.html

Apprentissage, entraînement : motivations

Qu'attend-t-on d'un réseau de neurones ?

- Qu'il apprenne de "stimulations" provenant de son environnement

Que fait la phase d'apprentissage ?

- Optimise les paramètres ajustables (poids w_{ij} et biais b_j) pour minimiser une erreur de sortie
- Utilise des règles pour les mettre à jour itérativement

Que signifie "stimuler" ou entraîner un réseau de neurons ?

- Utiliser des données pour guider la mise à jour des paramètres

Un problème classique d'optimisation

En pratique : méthode de descente de gradient

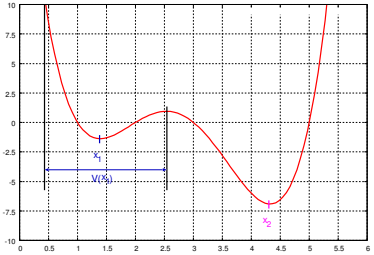
- Une méthode d'optimisation locale qui converge vers un minimum local
- Variantes (non présentées ici) de la descente de gradient
  - Descente batch ou lot (batch gradient descent)
  - Descente stochastique (stochastic gradient descent - SGD)
  - Descente par mini-batch (mini-batch gradient descent)
- Méthodes d'optimisation avancées à base du gradient
  - Momentum, Adagrad, etc.

Optimisation - terminologie et méthodes étudiées

Description générale d'un problème d'optimisation

- Étant donné un ensemble Ω de configurations (solutions admissibles) du probléme à résoudre et une fonction objectif f
- Trouver une configuration x' de Ω qui minimise f *i.e.* $f(x') = \min_{x \in \Omega} f(x)$

Exemple avec $f(x) = (x-1) \cdot (x-2) \cdot (x-3) \cdot (x-5)$ sur $\Omega = [0; 6]$



- x_1 est un minimum local sur le voisinage $V(x_1)$;
- x_2 est le minimum global

Descente de gradient pour recherche un minimum local

Rappel pour une fonction dérivable

- Chaque minimum x' vérifie $f'(x') = \frac{\partial f}{\partial x}(x') = 0$

Exemple avec $f(x) = (x-1)\cdot(x-2)\cdot(x-3)\cdot(x-5)$ sur $\Omega = [0;6]$

- $f(x) = (x-1)(x-2)(x-3)(x-5) = x^4 - 11x^3 + 41x^2 - 61x + 30$
- $f'(x) = \frac{\partial f}{\partial x}(x) = 4x^3 - 33x^2 + 82x - 61$
- Résoudre l'équation $f'(x) = 0$: trouver les racines d'un polynôme de degré 3 (assez difficile)

Descente de gradient : méthode itérative

- À partir d'une valeur initiale x^0 (plus ou moins bien choisie),
- on construit une suite de valeurs $x^1, x^2, \ldots$ qui converge vers un minimum de la fonction objectif f

Descente de gradient : méthode itérative

Théorie de la descente

- En un point a la fonction f décroît dans la direction opposée au signe de $f'(a)$.
- Preuve :
  - Si $f'(a)$ est postif, alors f est croissante, *i.e.* $x^{k+1} \leq x^k \Leftrightarrow f(x^{k+1}) \leq f(x^k)$. Pour réduire $f(x^k)$, il faut réduire x^k.
  - Si $f'(a)$ est négatif, alors f est décroissante, *i.e.* $x^k \leq x^{k+1} \Leftrightarrow f(x^k) \geq f(x^{k+1})$. Pour réduire $f(x^k)$, il faut augmenter x^k.
- Ainsi, en définissant x^{k+1} par

$$x^{k+1} = x^k - \gamma \cdot f'\left(x^k\right) = x^k - \gamma \cdot \frac{\partial f}{\partial x}\left(x^k\right)$$

pour $\gamma > 0$ suffisamment petit on a $f\left(x^{k+1}\right) \leq f\left(x^k\right)$

Descente de gradient : convergence

Conditions d'arrêt

- Valeurs successives de x^k suffisamment voisines : $\left|x^{k+1} - x^k\right| \leq \epsilon$
- Nombre maxium d'itérations atteint

Le pas γ

- influe sur la vitesse de convergence
- influe également sur le minimum trouvé : plus ou moins proche du minimum exact
- valeur de γ trop grand : divergence possible

```
def f(x): return (x-1)*(x-2)*(x-3)*(x-5)

def fp(x): return 4*x**3 -33*x**2 + 82*x - 61

def descgrad(f,fp,x0,gamma=1E-2,eps=1E-5,nbitermax=1000):
    xk,l,cpt,test = x0, [],0, False
    while (not test):
        cpt += 1
        l.append(xk)
        xkp1 = xk - gamma * fp(xk)
        test = abs(xkp1-xk) <= eps or cpt > nbitermax
        xk = xkp1
    return (xk,l)

xp,l = descgrad(f,fp, 0.6,0.01)
print (xp) # affiche  1.3926845950730142
xp,l = descgrad(f,fp, 0.6,0.1)
print (xp) # affiche  4.104990789406446
```

Réseau de neurones (code)

```python
import pretraitement
import matplotlib.pyplot as plt
from sklearn.model_selection import cross_val_score, StratifiedShuffleSplit, train_test_split
from sklearn.neural_network import MLPClassifier
import numpy as np

X,y = pretraitement.X,pretraitement.y
clf = MLPClassifier(solver='lbfgs', max_iter=500, hidden_layer_sizes=(12, 8), random_state=1)
scores = cross_val_score(clf, X, y, scoring='accuracy')

mean_score = np.mean(scores)
std_deviation = np.std(scores)
print(f"Scores de validation croisée: {scores}")
print(f"Précision moyenne: {mean_score:.4f}")
print(f"Écart type des précisions: {std_deviation:.4f}")


clf = MLPClassifier(solver='lbfgs', max_iter=500, hidden_layer_sizes=(12, 8), random_state=1)
clf.fit(X.values, y)
print(np.array(clf.predict([[1, 184, 84, 33, 82, 35.5, 0.355, 41]])))
```

Plan

Plan

Rappels

Regroupement par k-moyenne

- Soit $D = \{x^1, x^2, \ldots, x^N\}$ un ensemble de données avec d attributs, *i.e.* chaque $x^j = (x^j_1, \ldots, x^j_d)$, $1 \leq j \leq N$ est de dimension d.
- Objectif : partitioner D en k groupes O^1, ..., O^k, de centroids o^1, ..., o^k minimisant $\sum_{j=1}^{k} \sum_{x \in O^j} ||x - o^j||^2$
- Publication : centroids $\{o^1, \ldots, o^k\}$ et cardinalités $|O^1|$, $|O^2|$, ..., $|O^k|$

Algorithme simplifié

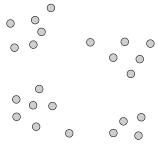
1. choisir des centroïds initiaux o^1, ..., o^k (au hasard ?)
2. répéter jusqu'à convergence, au plus t fois :
   - 2.1 associer chaque x au groupe O^j de centroïd o^j le plus proche :

$$O^j = \left\{ x : \left\| x - o^j \right\| \leq \left\| x - o^{j'} \right\| \ \forall\, j' = 1, \ldots, k, j' \neq j \right\}$$

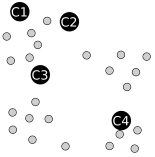
   - 2.2 m.-à-j. du centroïd de chaque groupe : $o^j = \frac{1}{|O^j|} \sum_{x \in O^j} x$

Exemple, synthèse

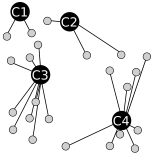
Exemple visuel [9]



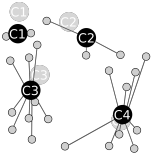
0a. Données d'entrée



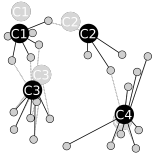
0b. intialisation



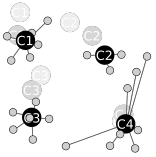
1a. assignation



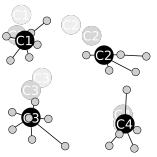
1b. calcul des points moyens



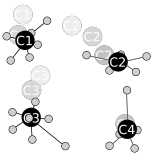
2a. assignation



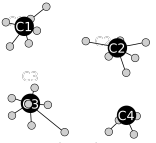
2b. calcul des points moyens



3a. assignation



3b. calcul des points moyens



4a. assignation

clusters stables (fin)

Points clefs

1. choix des centroïdes initiaux au hasard
2. association des x au groupe O^j