Intelligence Artificielle
TD 5 : classification linéaire et classification des vins

1 Classification linéaire

Nous allons implémenter une méthode standard pour la classification linéaire appelée *analyse discriminante linéaire* (ADL). Cette méthode implémente la même fonction de décision que le classifieur optimal,

$$f(x) = \begin{cases} 1, & \text{si } P(Y=1|X=x) \geq P(Y=-1|X=x) \\ -1, & \text{sinon,} \end{cases}$$

mais avec des probabilités estimées à partir des données.

En fait, les probabilités conditionnelles du type $P(Y=1|X=x)$ sont difficiles à estimer, alors que $P(X=x|Y=y)$ (ou $p(x|Y=y)$ dans le cas continu) peut être estimée assez simplement en faisant une hypothèse sur le modèle qui génère les données. Par exemple, pour la classification dans $\mathbb{R}^d$, l'ADL utilise l'hypothèse que, dans une classe y, les données sont générées selon une loi gaussienne de densité :

$$p(\boldsymbol{x} \mid Y=y) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{C}_y|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_y)^T \boldsymbol{C}_y^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_y)\right),$$

où les moyennes $\boldsymbol{\mu}_y$ peuvent être estimées simplement par

$$\boldsymbol{\mu}_y = \frac{1}{m_y} \sum_{y_i=y} \boldsymbol{x}_i$$

avec m_y le nombre d'exemples de la classe y dans la base d'apprentissage $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, et les matrices de covariance intra-classe par

$$\boldsymbol{C}_y = \frac{1}{m_y - 1} \sum_{y_i=y} (\boldsymbol{x}_i - \boldsymbol{\mu}_y)(\boldsymbol{x}_i - \boldsymbol{\mu}_y)^T. \tag{1}$$

En ADL, on ajoute l'hypothèse simplificatrice stipulant que la matrice de covariance est constante sur l'ensemble des classes : $\forall y$, $\boldsymbol{C}_y = \boldsymbol{C}$. Ainsi celle-ci s'estime par

$$\boldsymbol{C} = \frac{1}{m - |\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \sum_{y_i=y} (\boldsymbol{x}_i - \boldsymbol{\mu}_y)(\boldsymbol{x}_i - \boldsymbol{\mu}_y)^T.$$

En remarquant que $m = \sum_{y \in \mathcal{Y}} m_y$, cette formule correspond à la moyenne pondérée,

$$\boldsymbol{C} = \frac{\sum_{y \in \mathcal{Y}} (m_y - 1)\boldsymbol{C}_y}{\sum_{y \in \mathcal{Y}} (m_y - 1)},$$

des matrices de covariance intra-classe estimées par la formule (1).

Par ailleurs, les probabilités a priori $P(Y=y)$ sont aussi très faciles à estimer par comptage :

$$P(Y=y) \approx \frac{m_y}{m}.$$

1. En utilisant la règle de Bayes, réécrivez $f(\boldsymbol{x})$ en fonction uniquement des densités $p(\boldsymbol{x}|Y=y)$ et des probabilités $P(Y=y)$ qui peuvent être estimées aisément.
   Nous considérons ici un couple de variables aléatoires (X,Y) mixte où X est continue et Y est discrète. La "densité jointe" de ce couple peut s'exprimer de manière factorisée en considérant des probabilités pour Y et des densités pour X :
   $$p(x,y) = p(x|Y=y)P(Y=y) = P(Y=y|X=x)p(x)$$

2. Montrez que l'ADL est bien une méthode de classification linéaire, c'est-à-dire que f peut s'écrire comme
   $$f(\boldsymbol{x}) = \text{signe}(\boldsymbol{w}^T\boldsymbol{x} + b)$$
   Indice : considérez le log du rapport $P(Y=1|X=x)/P(Y=-1|X=x)$.

3. Dans le fichier `tp2adl.py`, implémentez en Python la fonction `(w,b) = adl(X,Y)` pour apprendre les paramètres $\boldsymbol{w}$ et b ci-dessus ainsi que la fonction `y = adlpredict(X, w, b)` pour classer un ensemble d'exemples donnés dans une matrice `X`.
   Si `X_y` contient tous les vecteurs $\boldsymbol{x}_i$ appartenant à la classe y, alors `np.cov(X_y.T)` permet de calculer la matrice de covariance $\boldsymbol{C}_y$.

4. Modifiez le code pour éviter d'utiliser l'opérateur `np.linalg.inv()`.

5. Testez ces fonctions sur des données générées à la souris avec des moyennes $\boldsymbol{\mu}_+ = [3,3]^T$, $\boldsymbol{\mu}_- = [-3,-3]^T$ et une matrice de covariance $\boldsymbol{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ (rappelle : une matrice de covariance diagonale indique que les variables sont indépendantes ; la diagonale contient les variances des différentes variables).

6. Visualisez la classification du plan obtenue en appuyant sur ESPACE et testez ensuite différentes distributions des points.

2 Prédiction de la couleur du vin

Récupérez le fichier `vins.py` sur ARCHE ainsi que les données sur les vins dans `vinsX.dat` et `vinsY.dat`[1].

Vous avez maintenant accès à une matrice `X` contenant la description de 6497 vins selon 11 variables correspondant à des mesures chimiques (fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol) et une 12ème mesurant la "qualité" du vin sur une échelle de 1 à 10. De plus, le vecteur `y` indique pour chaque ligne de `X` la couleur du vin (+1 rouge, -1 blanc).

1. Créer un classifieur capable de prédire à partir de la mesure des 12 variables s'il s'agit d'un vin rouge ou blanc et donnez une estimation de sa performance.

2. Idem pour classer les vins rouges en deux catégories : les bons avec une qualité ≥ 5 et les mauvais.

3 Analyse discriminante quadratique

L'analyse discriminante quadratique est similaire à l'ADL à ceci près que la matrice de covariance n'est plus supposée constante : chaque classe possède une matrice $\boldsymbol{C}_y$ différente.

1. Montrez que l'analyse discriminante quadratique est bien quadratique, c'est-à-dire que la fonction de prédiction peut s'écrire comme
   $$f(\boldsymbol{x}) = \text{signe}\left(\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{w}^T\boldsymbol{x} + b\right)$$

[1] Données issues de P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, 47(4):547-553, 2009.

et donnez l'expression des paramètres $\boldsymbol{Q}, \boldsymbol{w}$ et b.

2. Implémentez ce classifieur et testez-le dans le plan avec `tp2adl.py` en utilisant des variances différentes pour chaque classe de points.

3. Appliquez cette méthode pour prédire la couleur des vins et comparez avec les résultats de l'ADL.