

# Fiche ML

Charles Vin

S2-2023

## 1 Généralité

- Fonction de perte : quantifie l'erreur associé à une décision. Erreur simple : A chaque fois qu'on se trompe, on compte 1 : 0-1 loss
- Risque : Proba de se tromper,  $R(y_i|x) = \sum_j l(y_i, y_j)P(y_j|x)$  = Moyenne de la Loss pondéré par les probas à post
- Risque continue? :  $R(f) = \int_{x \in \mathcal{X}} R(f(x)|x)p(x)dx$  ( $p(x) = ???$ ) = Esperance du X sur notre domaine continue
- iso-contours == courbe de niveau
- Une epoque = on a vu une fois tous les exemples dans le gradient
- Hinge-loss =  $\max(0, 1 - yf_w(x))$ 
  - the Hinge loss penalizes predictions  $y < 1$ , corresponding to the notion of a margin in a support vector machine.
  - When  $y$  and  $f_w(x)$  have the same sign (meaning  $y$  predicts the right class) and  $|f_w(x)| \geq 1$ , the hinge loss = 0 When they have opposite signs, the hinge loss increases linearly with  $f_w(x)$  and similarly if  $|f_w(x)| \geq 1$ , even if it has the same sign (correct prediction, but not by enough margin).

## 2 Arbre de décision

Algo général :

1. Déterminer la meilleure caractéristique dans l'ensemble de données d'entraînement.
2. Diviser les données d'entraînement en sous-ensembles contenant les valeurs possibles de la meilleure caractéristique.
3. Générez de manière récursive de nouveaux arbres de décision en utilisant les sous-ensembles de données créés.
4. Lorsqu'on ne peut plus classifier les données, on s'arrête.

Méthode de division des données : On vas utiliser l'entropie

**Définition 2.1** (Entropie). [Origine de la formule de l'entropie](#) Soit  $X$  une variable aléatoire pouvant prendre  $n$  valeurs  $x_i$

$$H(X) = - \sum_{i=1}^n P(X = x_i) \log(P(X = x_i)).$$

Mesure l'homogénéité d'un dataset. C'est également la moyenne de la suprise (voir la vidéo)

**Définition 2.2** (Gain d'information). Mesure la réduction expected de l'entropie causé par le partitioning des exemples.

En faisant un test  $T$  sur un des attributs, on obtient deux partitions d'exemples de  $X$  :  $X_1$  qui vérifie le test et  $X_2$  qui ne vérifie pas le test (resp.  $Y_1$  et  $Y_2$ ).

$$H(Y|T) = \frac{|X_1|}{|X|} H(Y_1) + \frac{|X_2|}{|X|} H(Y_2).$$

Gain d'information :

$$I(T, Y) = H(Y) - H(Y|T).$$

**On veut maximiser le gain d'information par le split**  $\Leftrightarrow$  minimiser  $H(Y|T)$

### 3 Classifieur bayésien

On a :

- $P(y)$  fréquence des classe dans le dataset
- $P(x|y)$  les points de notre jeux de donnée. Graphiquement : les points coloriés

On cherche :

$$\arg \max_y P(y|x) = \arg \max_y \frac{P(x|y)P(y)}{P(x)}.$$

Par indépendance des dimensions de  $x$ , on peut parfois développer le  $P(x|y) = P(x_1|y) \dots P(x_d|y)$ .

Puis rapport de vraisemblance **en utilisant le risque** pour prendre la décision.

Remarque :

- Classifier bayésien = le classifier qui minimise le risque = le meilleurs classifieur possible
- Classifier optimal car minimise l'erreur car en choisissant la plus grande proba, on peut pas réduire  $1 - P(y|x)$  qui est déjà le plus grand possible
- $P(x)$  difficile à calculer = répartition des points dans l'espace, dans le graph 2d non colorié. En général très petit, uniquement utile pour générer des données, pas pour faire l'argmax (aka classifieur).

Autre truc important :

- On utilise classiquement une 0-1 loss
- Frontière de décision :  $\frac{R(+|x)}{R(-|x)} > 1 \rightarrow$  Permet de prendre en compte les coûts asymétriques des classes

### 4 Estimation de densité

#### 4.1 Par histogramme

**Définition 4.1** (Estimation par histogramme). — Cas discret : Comptage dans chaque classe puis normalisation par le nombre d'exemple  $N$   
— Cas continue : Discrétisation des valeurs puis comptage et normalisation

Importance de la discrétisation :

- Petit  $\rightarrow$  sur-apprentissage, trou dans l'histogramme
- Trop grand  $\rightarrow$  sous-apprentissage

Limite :

- Grande dimension  $\rightarrow$  Perte de sens exponentiel (3 ou 4 max)
- Effet de bord : petit changement dans les bins, gros changement d'estimation.

$\rightarrow$  Solution : Estimation par noyaux

#### 4.2 Estimation de densité par noyaux

Intuition figure 1 : Plutôt que de décider d'une discrétisation a priori, l'estimation est faite en centrant une fenêtre autour du point d'intérêt  $x_0$  (dans un espace de dimension  $d$ ) à posteriori.  $\rightarrow$  Problème : pas continue (si on bouge la boîte et qu'un point rentre dedans, ça fait faire un saut à la fonction)

##### 4.2.1 Fenêtre de Parzen

On combine la solution précédente avec une densité/noyaux. Classiquement Gaussien. pour obtenir un truc lisse et continue

**Définition 4.2** (Fenêtre de Parzen). Soit  $(x_1, \dots, x_N) \sim f$  iid

$$\hat{f}_h(x) = \frac{1}{N * h} \sum_{i=1}^N K\left(\frac{x - x_i}{h}\right).$$

Avec  $K$  le noyaux **centrée et réduit sur**  $x$ , souvent une fonction gaussienne. Si c'est une fonction rectangle ça fonctionne aussi. Puis y'a plein d'autre noyaux possible.

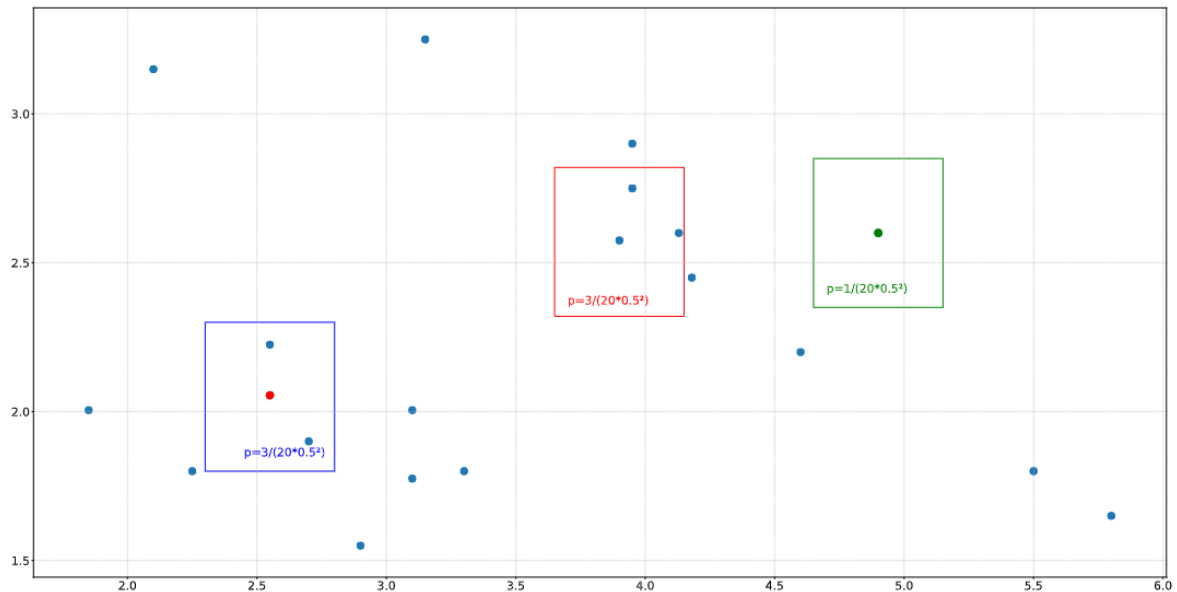


Figure 1 - Intuition de l'estimation par noyaux

## 5 Régression Linéaire

—  $MSE : (XW - Y)^T(XW - Y) = W^T X^T X W - (Y^T X W)^T - Y X W + Y^T Y$

$$\begin{aligned} \nabla_W MSE &= 2X^T X W - X^T Y - Y^T X \\ &= 2X^T X W - X^T Y - X^T Y \text{ car } \lambda \in \mathbb{R}, \lambda^T = \lambda \\ &= 2X^T (XW - Y) = 0 \\ &\Leftrightarrow W = (X^T X)^{-1} X^T Y \end{aligned}$$

— Sinon descente de gradient

## 6 Régression Logistique

- On peut pas utiliser la MSE car distance à la frontière de décision peut être très grande pour un point qui est très très très certainement dans une classe
- On vas plutot essayer de modéliser la confiance qu'on a dans la classif d'un point  $\rightarrow$  Proba :  $p(y = 1|x) = \mu(x)$
- Modélisation de cette proba par un truc linéaire qu'on projette entre 0 et 1 avec la sigmoïde ou tanh
- On remarque que le log ratio :  $\log \frac{\mu(x)}{1-\mu(x)} = f_w(x)$  pour la sigmoïde
- Pas de solution analytique à la log vrais : descente de gradient

## 7 Perceptron

- $f_w(x) = x \bullet w$
- Hinge-loss =  $\max(0, -y f_w(x))$ , vaut 0 quand bonne prédiction
- gradient Hinge loss

$$\nabla H_w = \begin{cases} 0 & \text{si } -y x w < 0 \\ -yx & \text{sinon} \end{cases}.$$

— into descente de gradient

Théorème de convergence : si

- $\exists R, \forall x \|x\| \leq R$
- Les données peuvent être séparées avec une marge  $p$
- L'ensemble d'apprentissage est présenté au perceptron un nombre suffisant de fois

Alors : après au plus  $\frac{R^2}{p^2}$  correction, l'algo converge

## 8 SVM

- Donnée non linéaire  $\rightarrow$  Projection, dim ++  $\rightarrow$  Attention sur apprentissage + quel dim choisir  $\rightarrow$  SMV do this auto
- Maximiser la marge  $\gamma \Leftrightarrow$  minimiser  $\|w\|$  sous la contrainte  $\forall i, (wx^i + b)y^i \geq 1$  par des calculs obscures ( $\geq 1$  car on veut que la distance entre la droite de régression et ces deux marges soit supérieur 1)
- Prise en compte des erreurs :
  - $\xi$  variable de débordement par rapport à sa marge pour chaque point mal classé  $\rightarrow$  Raison obscure  $\rightarrow \xi = \max(0, 1 - (wx^i + b)y^i)$  Hinge loss
  - On avait  $\min \|w\|^2$  maintenant  $\min \|w\|^2 + K \sum \xi$  avec  $K$  hyper param nombre d'erreur
- Optimisation avec lagrangien cas simple

$$\begin{cases} \min_{w,b} \frac{1}{2} \|w\|^2 \\ \text{s.c } y^i (wx^i + b) \geq 1 \end{cases} \Leftrightarrow L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_i \alpha_i (y^i (wx^i + b) - 1).$$