Projet MALAP : Classification par diffusion

Quentin Duchemin, Tong Zhao, Pierre Boyeau

Introduction et Problématiques

1) Contexte

Dans de nombreuses approches du machine learning, nous obtenons un prédicteur à partir de données étiquetées. Cependant, le travail qui consiste à annoter les données est long et fastidieux. Le problème qui consiste à s'intéresser à combiner des données étiquetées en très petite quantité et d'autres pas est donc d'une importance capitale en machine learning. C'est ce qu'on appelle un problème d'**apprentissage semi-supervisé**. Dans notre projet, nous utilisons **le champs de Markov** et **la fonction harmonique** pour résoudre ce problème.

2) Objectif de l'étude

- **Cadre** : Classification multi-classes
- **Input** : Des données $(x_i)_{i\in[\![1,u+l]\!]}$ dont les l premières sont étiquetées ($l \ll u$)
- **Output** : Réussir à prédire les étiquettes des données sans label $(x_i)_{i\in[\![l,u+l]\!]}$

3) Modélisation

- On considère un graphe $G(V, E)$:
  - Chaque sommet du graphe est une donnée
  - Graphe complet et le le poids de l'arc $i \to j$ est : $w_{ij} = \exp\left(-\sum_{d=1}^{m}\frac{(x_{id}-x_{jd})^2}{\sigma_d^2}\right)$
- But : Trouver $f^* = \underset{s.c\ f_{|L}=f_l}{\arg\min}\ E(f) = \frac{1}{2}\sum_{i,j} w_{ij}(f(i)-f(j))^2$ (∗)

4) Algorithmes réalisés

- Classification avec un seuil fixé
- Classification avec les proportions des classes
- Classification avec un classifieur externe
- Apprentissage des hyperparamètres définissant les poids

Méthode

Principe général

f^* est **harmonique** et satisfait donc :

- $\nabla f^* = 0$ sur l'ensemble des données non étiquetées
- est égale à f_l sur l'ensemble des données étiquetées

$\nabla = D - W$ = laplacien combinatoire où $D = diag(d_i)$ est une matrice diagonale avec $d_i = \sum_j w_{ij}$.

$$f^* = Pf^* \text{ où } P = D^{-1}W \quad \begin{cases} f^* \text{ est unique} \\ 0 < f^*(j) < 1\ \forall\ j \in U \end{cases}$$

Notant $W = \begin{bmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{bmatrix}$ et $f = \begin{bmatrix} f_l \\ f_u \end{bmatrix}$, on a :

$$f_u^* = (D_{uu} - W_{uu})^{-1}W_{ul}f_l = (I - P_{uu})^{-1}P_{ul}f_l$$

Méthodologie avancée

Fixer le seuil

Le plus naturel

$$Classe\ (i) = \begin{cases} 1 \text{ si } f(i) > \frac{1}{2} \\ 0 \text{ sinon.} \end{cases}$$

Le problème
Données non idéalement séparées ⇒ classification très déséquilibrée.

Proportions des classes

On pose q la proportion souhaitée d'exemples d'étiquette 1, donc on a:

$$Classe\ (i) = 1 \Leftrightarrow q\frac{f_u(i)}{\sum_j f_u(j)} > (1-q)\frac{1-f_u(i)}{\sum_j(1-f_u(j))}$$

Ajout d'un classifieur externe

On introduit un classifieur externe et on pose h_u la prédiction du classifieur sur les données non étiquetées. $\forall i \in U$, on ajoute au graphe $\mathcal{G}$ un noeud avec l'étiquette h_i relié à i par une arête de poids η, donc on a:

$$f_u = (I - (1-\eta)P_{uu})^{-1}((1-\eta)P_{ul}f_l + \eta h_u)$$

Dans notre projet, on utilise la machine à vecteurs de support et on extrait les étiquettés doux.

Apprendre la matrice de poids

Dans cette partie, on souhaite apprendre les hyperparamètres σ_d.La méthode classique consiste à maximiser une vraisemblance sur les données étiquetées. Mais dans notre cas, la vraisemblance n'a pas de sens sur l'ensemble U parce que nous n'avons pas de modèle génératif.

Principe Minimiser l'entropie moyenne sur les données non étiquetées.

$$H(f) = \frac{1}{u}\sum_{i=l+1}^{l+u} H_i(f(i))$$

$$H_i(f(i)) = -\sum_{i=1}^{n} f(i)\log(f(i))$$

Cependant, H tend vers 0 quand σ tend vers 0. Dans ce cas, on ajoute un paramètre ϵ pour glisser la matrix P.

$$\tilde{P} = \epsilon\mathcal{U} + (1-\epsilon)P$$

$$\mathcal{U}_{ij} = \frac{1}{l+u}$$

Le papier étudié propose une descente de gradient pour trouver les σ qui minimise H.

Approche critique de la méthode

- le calcul du gradient de l'entropie est très coûteux
- minimiser l'entropie nécessite l'introduction d'un paramètre de régularisation qui perturbe le problème de départ

Solutions apportées

1. Descente de gradient sur l'énergie.
   Nous revenons à l'expression initiale du problème étudié. Nous calculons le gradient par rapport aux hyperparamètres σ_d de l'énergie $E(f)$, et nous effectuons quelques pas de descente dans la direction opposée au gradient.
   Regard critique :
   - \+ la descente de gradient est effectuée sur la véritable fonction objectif
   - − le calcul du gradient de $E(f)$ est toujours aussi coûteux en temps
2. Utiliser un unique hyperparamètre σ.
   On suppose que tous les dimensions ont la même σ. Dans ce cas, on remplace la descente de gradient par le grid search.
   Pour accélérer la vitesse, on fait sortir le terme σ pour qu'on n'ait plus besoin de calculer la distance d'euclidean chaque fois qu'on change le paramètre σ
   Regard critique :
   - \+ la vitesse de calcul est supportable
   - − le performance final peut-être diminuer.
3. *Grid Search*
   Nous nous donnons un ensemble fini de paramètres σ. Pour chacun d'eux, nous calculons la solution harmonique f^* au problème de minimisation (∗). Nous retenons finalement celui ayant permis d'obtenir la plus petite énergie possible.

Résultats

Les bases de données choisies

Nom	Nombre de données	Nombre des features	Nombre de classes
USPS	7291	256	10
MNIST	60000	784	10
spambase	4601	57	2

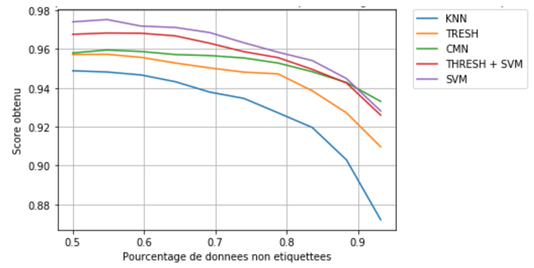
Les algorithmes comparatifs

- K plus proche voisin
- Machine à veteurs de support

La performance sur USPS

On choisit aléatoirment 5000 données dans la base de données. Les paramètres de données sont comme suit:

- K plus proche voisin: $K = 5$
- Machine à veteurs de support: $C = 5$, $gamma = 0.01$,$kernel = rbf$
- Notre algortihme: $\sigma = 2.5$, $\eta = 0.4$



Comparaison des algorithmes

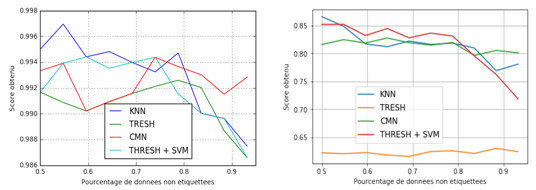


Figure 1: Test sur les données Mnist

Figure 2: Test sur les données spambase

Performance et nombre de features

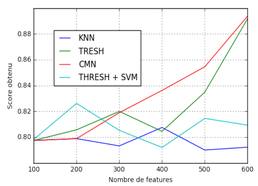


Figure 3: Etude de l'influence du nombre de features sur les performances de l'algorithme sur la base de données Mnist avec 0.02% de données étiquetées. On constate que nos prédictions se dégradent sensiblement lorsque le nombre de features devient trop faible.

Performance et nombre de classes

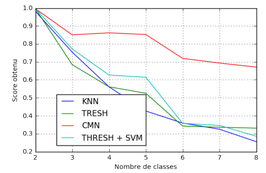


Figure 4: Etude de l'influence du nombre de classes sur les performances de l'algorithme sur la base de données Mnist avec 0.02% de données étiquetées. On constate que le classifieur CMN est le plus robuste sur les données Mnist à une augmentation du nombre de classes.

Apprentissage de la matrice de poids

Minimisation de l'entropie

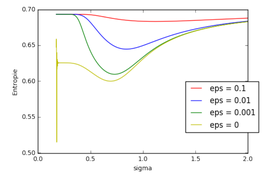


Figure 5: La régularisation permet d'éviter la situation où $H \underset{\sigma\to 0}{\to} 0$

Descente de gradient sur l'énergie

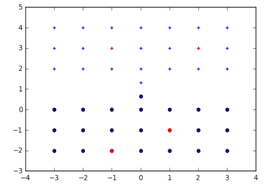
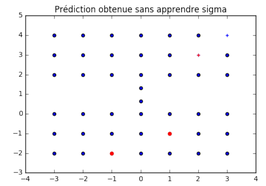
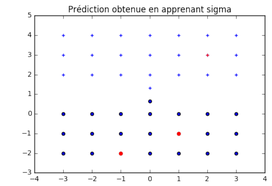


Figure 6: Données initiales

Test effectué sur ce jeu de données jouet $2D$. Les données rouges sont celles considérées comme étiquetées. Le score obtenu sans apprendre les poids est de 0.51 alors que l'apprentissage des poids permet une prédiciton exacte.





Sources

[1] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. *In The 20th International Conference on Machine Learning (ICML), 2003.*,

[2] UCI Machine Learning Repository

[3] Yair Weiss, William T. Freeman, Correctness of belief propagation in Gaussian graphical models of arbitrary topology.