IFT-7002 Fondements de l'apprentissage machine

Multi-classes et problèmes de prédiction complexes

Shai Shalev-Shwartz
The Hebrew University of Jerusalem

Traduit et adapté par Mario Marchand
Université Laval

Hiver 2024

Sommaire

Classification multi-classes

- Jusqu'ici, nous avons étudié des algorithmes d'apprentissage pour construire :
  - des classificateurs binaires $h : \mathcal{X} \to \{\pm 1\}$
  - des régresseurs $h : \mathcal{X} \to \mathbb{R}$
- Examinons maintenant les algorithmes pour construire des classificateurs multi-classes : $h : \mathcal{X} \to \{1, \ldots, k\}$.
  - *e.g.*, $x \in \mathcal{X}$ est une image et $\{1, \ldots, k\}$ représente k objets possibles.
- Nous examinerons ensuite le cas où k est très élevé.
  - *e.g.*, la traduction : $x \in \mathcal{X}$ est phrase écrite en français et $\{1, \ldots, k\}$ est l'ensemble de toutes les phrases écrites en anglais.

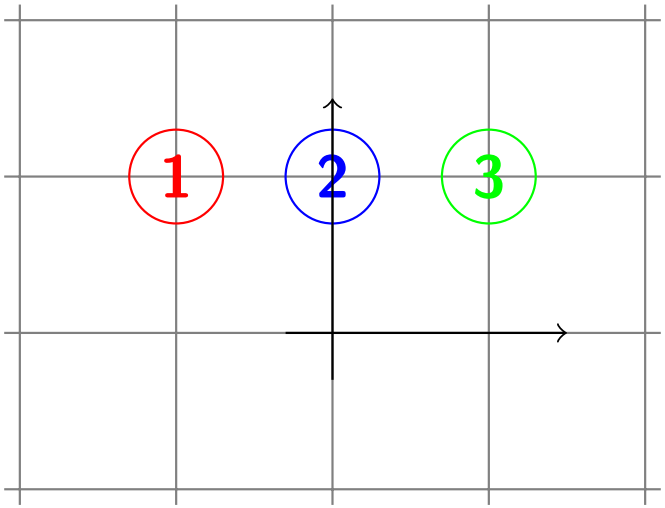
La réduction un-contre-le-reste

- Utilisons un algorithme A construisant un classificateur binaire pour obtenir un algorithme construisant un classificateur multi-classes.
- Soit $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ avec $y_i \in \{1, \ldots, k\}$.
- Un-contre-tous-le-reste : apprendre k classificateurs binaires, où le classificateur h_i tente de discriminer la classe i du reste des classes :
  - Pour tout $i \in \{1, \ldots, k\}$, utiliser l'échantillon $S_i = \{(\mathbf{x}_1, (-1)^{\mathbb{1}[y_1 \neq i]}), \ldots, (\mathbf{x}_m, (-1)^{\mathbb{1}[y_m \neq i]})\}$ avec A pour obtenir h_i.
  - Générer le classificateur multi-classes h t.q. pour tout $\mathbf{x} \in \mathcal{X}$ on a

$$h(\mathbf{x}) = \operatorname*{argmax}_{i \in [k]} h_i(\mathbf{x})$$

- Lorsque plus d'un h_i prédit 1, et lorsque $h_i(\mathbf{x}) = \operatorname{sign}[\langle \mathbf{w}_i, \boldsymbol{\psi}(\mathbf{x}) \rangle]$, remplacer chaque h_i par leur "confiance de prédiction" $\langle \mathbf{w}_i, \boldsymbol{\psi}(\mathbf{x}) \rangle$.

L'utilité de la confiance de prédiction $\text{argmax}_i \langle \mathbf{w}_i, \mathbf{x} \rangle$



- Si la distribution des classes est 40%, 20%, 40%, le meilleur classificateur 2 v.s. le reste prédira toujours négatif.
- L'erreur du classificateur multi-classes obtenu par la réduction un-contre-le-reste sera donc de 20%.
- Par contre, l'erreur est nulle pour $h(\mathbf{x}) = \text{argmax}_i \langle \mathbf{w}_i, \mathbf{x} \rangle$ avec

$$\mathbf{w}_1 = (-1/\sqrt{2}, 1/\sqrt{2}), \ \mathbf{w}_2 = (0, 1), \ \mathbf{w}_3 = (1/\sqrt{2}, 1/\sqrt{2}), \ .$$

La réduction toutes-les-paires

- Pour chaque paire (i, j) de classes, on utilise un algorithme d'apprentissage A pour classification binaire afin discriminer entre la classe i et la classe j.
- Donc, pour tout $1 \leq i < j \leq k$, on construit l'échantillon $S_{i,j}$ contenant uniquement les exemples de S appartenant à la classe i et la classe j avec $(\mathbf{x}_k, y_k) = (\mathbf{x}_k, +1)$ si $y_k = i$ et $(\mathbf{x}_k, y_k) = (\mathbf{x}_k, -1)$ si $y_k = j$.
- Soit $h_{i,j} = A(S_{i,j})$ avec $h_{i,j}(\mathbf{x}) = 1$ signifiant que la classe prédite pour $\mathbf{x}$ est i et $h_{i,j}(\mathbf{x}) = -1$ signifiant que la classe prédite pour $\mathbf{x}$ est j.
- Le classificateur multi-classes h prédit la classe majoritaire parmi toutes celles prédites par ces $h_{i,j}$. *i.e.*, pour tout $\mathbf{x} \in \mathcal{X}$, on a

$$h(\mathbf{x}) = \operatorname*{argmax}_{i \in \mathcal{Y}} \left(\sum_{j>i} \mathbb{1}_{[h_{i,j}(\mathbf{x})=+1]} + \sum_{j<i} \mathbb{1}_{[h_{j,i}(\mathbf{x})=-1]} \right) .$$

Classificateurs linéaires multi-classes

- La sortie $h_{\mathbf{w}}(\mathbf{x})$ d'un classificateur linéaire $h_{\mathbf{w}}$ sur $\mathbf{x}$ est donnée par

$$h_{\mathbf{w}}(\mathbf{x}) = \operatorname{sign}(\langle \mathbf{w}, \mathbf{x} \rangle) \,,$$

- ce qui peut se ré-écrire comme

$$h_{\mathbf{w}}(\mathbf{x}) \;=\; \underset{y \in \{\pm 1\}}{\operatorname{argmax}} \langle \mathbf{w}, y\mathbf{x} \rangle \,.$$

- Généralisation naturelle pour le cas multi-classes :

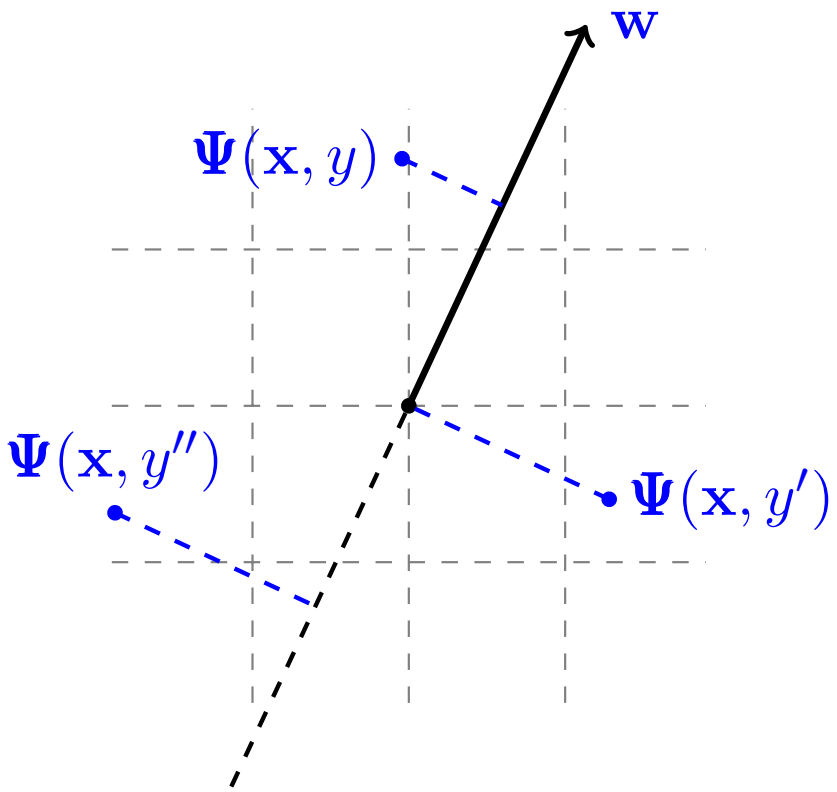
$$h_{\mathbf{w}}(\mathbf{x}) \;=\; \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y) \rangle,$$

où $\mathbf{\Psi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ est une fonction de projection dépendante de la classe.

- Chaque composante $\Psi_i(\mathbf{x}, y)$ s'interprète comme le score (ou le degré d'atteinte) de la propriété i sur l'exemple $(\mathbf{x}, y)$.
- Alors comment construire $\mathbf{\Psi}$?

Classificateurs linéaires multi-classes

$$h_{\mathbf{w}}(\mathbf{x}) \;=\; \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \, \langle \mathbf{w}, \boldsymbol{\Psi}(\mathbf{x}, y) \rangle,$$



Fonction de projection dépendante de la classe

- Le problème de choisir $\mathbf{\Psi}$ est similaire au problème de choisir le noyau pour le SVM : ça demande généralement une connaissance a priori.
- **Exemple : Approche TF-IDF pour classification de documents :**
  - $\mathcal{X}$ = documents, $\mathcal{Y}$ = sujets, d = nombre de mots du dictionnaire.
  - Term Frequency : $TF(j, \mathbf{x})$ = nombre de fois que le mot j apparaît dans le document $\mathbf{x}$.
  - Document Frequency : $DF(j, y)$ = nombre de documents qui ne sont pas du sujet y et contenant le mot j.
    - Mesure de l'utilisation du mot j dans les documents d'autres sujets.
  - Term-Frequency-Inverse-Document-Frequency :

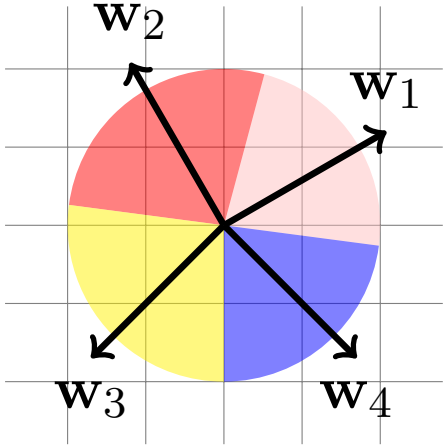
$$\Psi_j(\mathbf{x}, y) = TF(j, \mathbf{x}) \log\left(\frac{m}{1+DF(j,y)}\right) .$$

  - $\Psi_j(\mathbf{x}, y)$ devrait être élevé si le mot j est fréquent dans $\mathbf{x}$ mais est rare dans les documents dont le sujet n'est pas y.
    - Dans ce cas, il est probable que le document $\mathbf{x}$ soit du sujet y.
  - Notez que $\dim(\mathbf{\Psi}(\mathbf{x}, y))$ ne dépends pas de $|\mathcal{Y}|$.

La construction à vecteurs multiples

Si l'on désire prédire *indépendamment* chacune des classes, un classificateur multi-classe peut s'écrire comme

$$h_{\mathbf{w}}(\mathbf{x}) = \underset{y \in [k]}{\operatorname{argmax}} (W\mathbf{x})_y = \underset{y \in [k]}{\operatorname{argmax}} \langle \mathbf{w}_y, \mathbf{x} \rangle \,.$$



De manière équivalente, on a $h_{\mathbf{w}}(\mathbf{x}) = \operatorname{argmax}_y \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y) \rangle$ pour

$$\mathbf{\Psi}(\mathbf{x}, y) = [\, \underbrace{0, \ldots, 0}_{\in \mathbb{R}^{(y-1)n}}, \underbrace{x_1, \ldots, x_n}_{\in \mathbb{R}^n}, \underbrace{0, \ldots, 0}_{\in \mathbb{R}^{(k-y)n}} \,] \,.$$

Ici $\dim(\mathbf{\Psi}(\mathbf{x}, y)) = n|\mathcal{Y}|$.

Fonction de perte sensible aux coûts

Certaines erreurs sont pires que d'autres !

$h_1($  $) = \text{cat}$

$h_2($  $) = \text{whale}$

- Fonction de perte (sensible aux coûts) : $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$.
- La perte zéro-un est un cas particulier : $\Delta(y', y) = \mathbb{1}_{[y' \neq y]}$.

Minimisation du risque empirique

- Algorithme $\mathrm{ERM}_{\mathcal{H}}$: trouver $\mathbf{w}$ minimisant

$$L_S(h_{\mathbf{w}}) = \frac{1}{m} \sum_{i=1}^{m} \Delta(h_{\mathbf{w}}(\mathbf{x}_i), y_i) \ ,$$

  avec $h_{\mathbf{w}}(\mathbf{x}) = \mathrm{argmax}_{y' \in \mathcal{Y}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y') \rangle$.
- Dans le cas réalisable, ce problème est équivalent au problème de programmation linéaire :

$$\forall i \in [m], \ \forall y \in \mathcal{Y} \setminus \{y_i\}, \quad \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}_i, y_i) \rangle > \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}_i, y) \rangle \, .$$

- Dans le cas non réalisable, $\mathrm{ERM}_{\mathcal{H}}$ est $\mathcal{NP}$-difficile.
- Utilisons une fonction de perte substitut (convexe et majorant Δ).

Hinge Loss généralisé

Considérons le classificateur linéaire multi-classes :

$$h_{\mathbf{w}}(\mathbf{x}) = \underset{y' \in \mathcal{Y}}{\operatorname{argmax}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y') \rangle \ .$$

Pour tout $y \in \mathcal{Y}$, nous avons donc

$$\langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y) \rangle \leq \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, h_{\mathbf{w}}(\mathbf{x})) \rangle \ .$$

Alors,

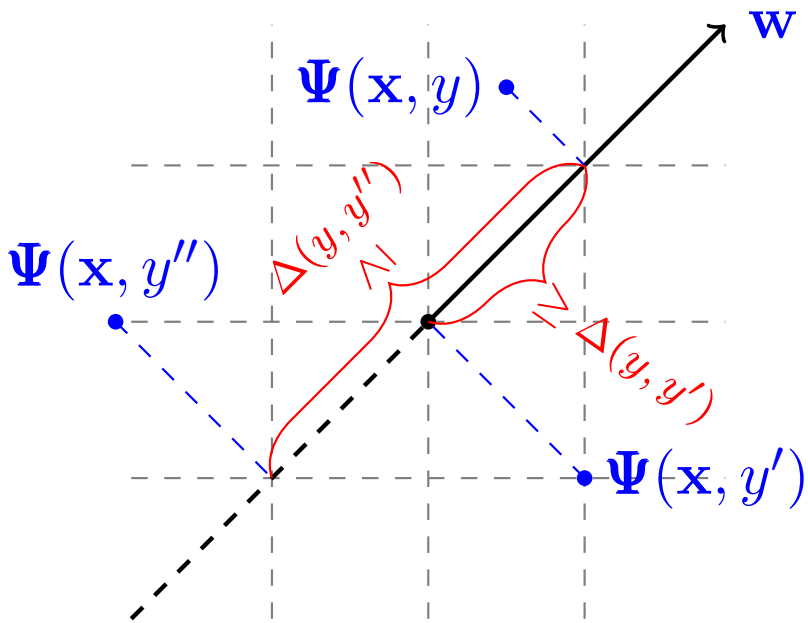
$$\begin{aligned} \Delta(h_{\mathbf{w}}(\mathbf{x}), y) &\leq \Delta(h_{\mathbf{w}}(\mathbf{x}), y) + \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, h_{\mathbf{w}}(\mathbf{x})) - \mathbf{\Psi}(\mathbf{x}, y) \rangle \\ &\leq \underbrace{\max_{y' \in \mathcal{Y}} \left(\Delta(y', y) + \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y') - \mathbf{\Psi}(\mathbf{x}, y) \rangle \right)}_{\stackrel{\text{def}}{=} \ell(\mathbf{w}, (\mathbf{x}, y))} \end{aligned}$$

- La fonction de perte de hinge généralisée $\ell(\cdot, z)$ est convexe et ρ-Lipschitzienne, pour $\rho = \max_{y' \in \mathcal{Y}} \|\mathbf{\Psi}(\mathbf{x}, y') - \mathbf{\Psi}(\mathbf{x}, y)\|$.

Hinge Loss généralisé

La fonction de perte de hinge généralisée devient égale à zéro lorsque

$$\forall y' \in \mathcal{Y} \setminus \{y\}, \quad \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y) \rangle \geq \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y') \rangle + \Delta(y', y) \ .$$



Le SVM multi-classes

Paramètres :

- fonction de projection dépendante de la classe $\mathbf{\Psi}$
- fonction de perte (sensible aux coûts) $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
- paramètre de régularisation $\lambda > 0$

Résoudre :

$$\underset{\mathbf{w}\in\mathbb{R}^d}{\operatorname{argmin}} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \max_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}_i, y') - \mathbf{\Psi}(\mathbf{x}_i, y_i) \rangle \right) \right)$$

Sortie :

$$h_{\mathbf{w}}(\mathbf{x}) = \underset{y\in\mathcal{Y}}{\operatorname{argmax}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y) \rangle$$

Garantie PAC sur le risque du SVM multi-classes

- Le SVM multi-classes étant un cas particulier de la régularisation de Tikhonov, la garantie PAC de cette cette dernière s'applique.
- Si nous supposons que $||\mathbf{\Psi}(\mathbf{x}, y)|| \leq \rho/2$, $\forall(\mathbf{x}, y)$, on a que $\ell(\cdot, z)$ est ρ-Lipschitzienne. Nous avons alors :

Corollaire (Garantie PAC pour le SVM multi-classes)

Soit $\mathbf{\Psi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ *tel que* $||\mathbf{\Psi}(\mathbf{x}, y)|| \leq \rho/2$, $\forall(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$. *Pour tout entier* $m > 0$ *et pour tout réel* $B > 0$, *soit* $\lambda = (\rho/B)\sqrt{2/m}$. *Pour tout* $\mathcal{D}$ *sur* $\mathcal{X} \times \mathcal{Y}$, *la sortie* $h_{\mathbf{w}}$ *du SVM multi-classes satisfait*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} L_{\mathcal{D}}^{\Delta}(h_{\mathbf{w}}) \leq \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w}) \leq \min_{\mathbf{u}:||\mathbf{u}||\leq B} L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{u}) + \sqrt{\frac{8\rho^2 B^2}{m}}\,,$$

où $L_{\mathcal{D}}^{\Delta}(h) \stackrel{\text{def}}{=} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \Delta(h(\mathbf{x}), y)$ *et* $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w}) \stackrel{\text{def}}{=} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \ell(\mathbf{w}, (\mathbf{x}, y))$.

DGS pour le SVM multi-classes

- La fonction de perte utilisée est la perte généralisée de hinge :

$$\ell(\mathbf{w}, (\mathbf{x}, y)) = \max_{y' \in \mathcal{Y}} \left(\Delta(y', y) + \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y') - \mathbf{\Psi}(\mathbf{x}, y) \rangle \right) .$$

- C'est le maximum parmi un ensemble de fonctions linéaires en $\mathbf{w}$.
- Le sous gradient de $\ell(\mathbf{w}^{(t)}, (\mathbf{x}, y))$ est donc obtenu en effectuant :
  - trouver $\hat{y} \in \operatorname{argmax}_{y' \in \mathcal{Y}} \left(\Delta(y', y) + \langle \mathbf{w}^{(t)}, \mathbf{\Psi}(\mathbf{x}, y') - \mathbf{\Psi}(\mathbf{x}, y) \rangle \right)$
  - alors $\mathbf{\Psi}(\mathbf{x}, \hat{y}) - \mathbf{\Psi}(\mathbf{x}, y) \in \partial \ell(\mathbf{w}^{(t)}, (\mathbf{x}, y))$.

Rappel : DGS pour la régularisation de Tikhonov

DGS pour la régularisation de Tikhonov (rappel) :

- **Objectif :** Minimiser $L_S(\mathbf{w}) + \lambda||\mathbf{w}||^2$.
- **initialiser :** $\mathbf{w}^{(1)} = \mathbf{0}$
- **pour** $t = 1, 2, \ldots, T$
  - tirer $i \sim U(m)$.
  - soit $\mathbf{u}_i^{(t)} \in \partial\ell(\mathbf{w}^{(t)}, z_i)$.
  - mise à jour : $\mathbf{w}^{(t+1)} = \left(1 - \frac{1}{t}\right)\mathbf{w}^{(t)} - \frac{1}{2\lambda t}\mathbf{u}_i^{(t)}$.
- **sortie :** $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}$
- Pour le SVM multi-classes, il suffit donc d'utiliser
  - $\hat{y} \in \operatorname{argmax}_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}^{(t)}, \mathbf{\Psi}(\mathbf{x}_i, y') - \mathbf{\Psi}(\mathbf{x}_i, y_i)\rangle\right)$
  - $\mathbf{u}_i^{(t)} = \mathbf{\Psi}(\mathbf{x}_i, \hat{y}) - \mathbf{\Psi}(\mathbf{x}_i, y_i)$.

DGS pour le SVM multi-classes

DGS pour le SVM multi-classes :

- **Objectif :** Minimiser
  $\lambda\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}_i, y') - \mathbf{\Psi}(\mathbf{x}_i, y_i)\rangle\right)$.
- **initialiser :** $\mathbf{w}^{(1)} = \mathbf{0}$
- **pour** $t = 1, 2, \ldots, T$
  - tirer $i \sim U(m)$.
  - soit $\hat{y} \in \operatorname{argmax}_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}^{(t)}, \mathbf{\Psi}(\mathbf{x}_i, y') - \mathbf{\Psi}(\mathbf{x}_i, y_i)\rangle\right)$.
  - mise à jour : $\mathbf{w}^{(t+1)} = \left(1 - \frac{1}{t}\right)\mathbf{w}^{(t)} - \frac{1}{2\lambda t}[\mathbf{\Psi}(\mathbf{x}_i, \hat{y}) - \mathbf{\Psi}(\mathbf{x}_i, y_i)]$.
- **sortie :** $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}^{(t)}$

DGS pour minimiser $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w})$

- Il est également possible d'utiliser la DGS pour minimiser directement $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w})$ (sans considérer $||\mathbf{w}||^2$ et $L_S^{\text{g}-\text{hinge}}(\mathbf{w})$).

DGS pour l'apprentissage multi-classes :

- **Objectif :** Minimiser $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w})$.
- **initialiser :** $\mathbf{w}^{(1)} = \mathbf{0}$
- **pour** $t = 1, 2, \ldots, T$
  - tirer $(\mathbf{x}, y) \sim \mathcal{D}$.
  - soit $\hat{y} \in \text{argmax}_{y' \in \mathcal{Y}} \left(\Delta(y', y) + \langle \mathbf{w}^{(t)}, \mathbf{\Psi}(\mathbf{x}, y') - \mathbf{\Psi}(\mathbf{x}, y) \rangle\right)$.
  - mise à jour : $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta[\mathbf{\Psi}(\mathbf{x}, \hat{y}) - \mathbf{\Psi}(\mathbf{x}, y)]$.
- **sortie :** $\bar{\mathbf{w}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{w}^{(t)}$

Garantie PAC pour la DGS minimisant $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w})$

- Puisque la fonction de perte utilisée pour la DGS précédente est convexe et ρ-Lipschitzienne lorsque $||\mathbf{\Psi}(\mathbf{x}, y)|| \leq \rho/2$, $\forall(\mathbf{x}, y)$, la garantie PAC obtenue (corollaire 14.12 du manuel) pour la DGS pour problèmes convexes-Lipschitziens-bornés s'applique directement.
- Nous obtenons alors la garantie suivante.

Corollaire (Garantie PAC pour la DGS minimisant $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w})$)

Soit $\mathbf{\Psi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ tel que $||\mathbf{\Psi}(\mathbf{x}, y)|| \leq \rho/2$, $\forall(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$. Pour tout $B > 0$ et $\epsilon > 0$, si nous exécutons la DGS pour l'apprentissage multi-classes durant un nombre T d'itérations tel que $T \geq (B\rho/\epsilon)^2$, et avec $\eta = B/(\rho\sqrt{T})$, alors le prédicteur $\bar{\mathbf{w}}$ satisfait

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^T} L_{\mathcal{D}}^{\Delta}(h_{\bar{\mathbf{w}}}) \leq \mathop{\mathbb{E}}_{S\sim\mathcal{D}^T} L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\bar{\mathbf{w}}) \leq \min_{\mathbf{u}:||\mathbf{u}||\leq B} L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{u}) + \epsilon\,,$$

où $L_{\mathcal{D}}^{\Delta}(h) \stackrel{\text{def}}{=} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \Delta(h(\mathbf{x}), y)$ et $L_{\mathcal{D}}^{\text{g}-\text{hinge}}(\mathbf{w}) \stackrel{\text{def}}{=} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \ell(\mathbf{w}, (\mathbf{x}, y))$.

Remarques sur les garanties PAC

- Les deux dernières bornes supérieures sur $\mathbb{E}\, L^{\Delta}_{\mathcal{D}}(h_{\bar{\mathbf{w}}})$ ne dépendent pas explicitement du nombre de classes $|\mathcal{Y}|$.
- Est-il alors possible d'apprendre même lorsque $|\mathcal{Y}|$ devient énorme ?
- Oui en autant que :
  - la norme de $\mathbf{\Psi}(\mathbf{x}, y)$ ne soit pas trop grande,
  - il existe un prédicteur $\mathbf{u}$ de faible norme tel que $L^{\mathrm{g-hinge}}_{\mathcal{D}}(\mathbf{u})$ soit faible,
  - nous puissions effectuer efficacement le calcul de

$$\operatorname*{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, y) \rangle \,,$$

  - nous puissions effectuer efficacement le calcul de

$$\operatorname*{argmax}_{y' \in \mathcal{Y}} \left(\Delta(y', y) + \langle \mathbf{w}^{(t)}, \mathbf{\Psi}(\mathbf{x}, y') - \mathbf{\Psi}(\mathbf{x}, y) \rangle \right) .$$

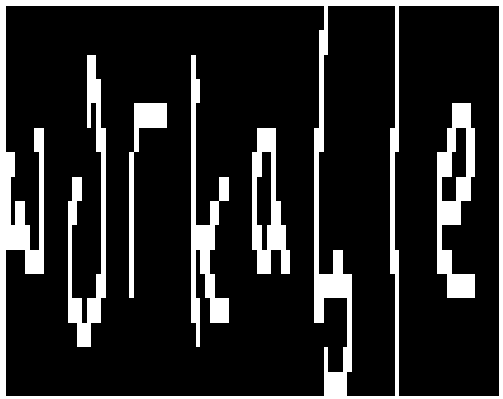
- Ceci nous amène à considérer la prédiction de sorties structurées.

Sommaire

Prédiction de sorties structurées

Prédiction de sorties structurées = problème multi-classes tel que $\mathcal{Y}$ est vaste, mais possède une certaine structure prédéfinie.
Example — reconnaissance de mots manuscrits

- $\mathcal{X}$ = l'ensemble des images de mots manuscrits
- $\mathcal{Y}$ = tous les mots anglais possibles



⟶ workable

Prédiction de sorties structurées

- L'espoir : la complexité d'échantillon du SVM multi-classes ne dépend pas de $|\mathcal{Y}|$,
  - mais seulement de $\|\mathbf{\Psi}(\mathbf{x}, \mathbf{y})\|$ et $\|\mathbf{w}\|$.

Par contre la taille de $\mathcal{Y}$ nous donne des défis computationnels. En effet, il faut pouvoir résoudre efficacement :

$$\operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, \mathbf{y}) \rangle \,,$$

$$\operatorname*{argmax}_{\mathbf{y}' \in \mathcal{Y}} \left(\Delta(\mathbf{y}', \mathbf{y}) + \langle \mathbf{w}^{(t)}, \mathbf{\Psi}(\mathbf{x}, \mathbf{y}') - \mathbf{\Psi}(\mathbf{x}, \mathbf{y}) \rangle \right) .$$

- Il faut donc choisir $\mathbf{\Psi}$ (et possiblement Δ) de manière à ce qu'il soit possible de maximiser efficacement ces fonctions sur $\mathcal{Y}$.

Modélisation pour reconnaissance des mots manuscrits

- Construisons un $\mathbf{\Psi}$ suffisamment riche mais nous permettant d'effectuer efficacement $\operatorname{argmax}_{\mathbf{y}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, \mathbf{y}) \rangle$.
- Chaque mot manuscrit $\mathbf{x} \in \mathcal{X}$ est une matrice $n \times r$:
  - Chacun des r caractères du mot $\mathbf{x}$ est un vecteur de n pixels représentant l'image du caractère.
- $\mathbf{y} = (y_1, \ldots, y_r)$, avec $y_i \in \Sigma$, est une séquence de lettres.
- Caractéristiques de type 1 : mesurent le niveau de gris moyen du pixel i dans une image de la lettre j :

$$\Psi_{i,j,1}(\mathbf{x}, \mathbf{y}) = \frac{1}{r} \sum_{t=1}^{r} x_{i,t} \, \mathbb{1}_{[y_t = j]} \ .$$

- Caractéristiques de type 2 : mesurent la fréquence avec laquelle la lettre i suit la lettre j :

$$\Psi_{i,j,2}(\mathbf{x}, \mathbf{y}) = \frac{1}{r} \sum_{t=2}^{r} \mathbb{1}_{[y_t = i]} \, \mathbb{1}_{[y_{t-1} = j]} \ .$$

Modélisation pour reconnaissance des mots manuscrits

Définissons :

$$\phi_{i,j,1}(\mathbf{x}, y_t, y_{t-1}) \stackrel{\text{def}}{=} \frac{1}{r} \mathbb{1}_{[t \geq 1]} x_{i,t} \mathbb{1}_{[y_t = j]}$$

$$\phi_{i,j,2}(\mathbf{x}, y_t, y_{t-1}) \stackrel{\text{def}}{=} \frac{1}{r} \mathbb{1}_{[t \geq 2]} \mathbb{1}_{[y_t = i]} \mathbb{1}_{[y_{t-1} = j]} \,.$$

Alors

$$\Psi_{i,j,1}(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{r} \phi_{i,j,1}(\mathbf{x}, y_t, y_{t-1})$$

$$\Psi_{i,j,2}(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{r} \phi_{i,j,2}(\mathbf{x}, y_t, y_{t-1}) \,.$$

Soit $h_{\mathbf{w}}(\mathbf{x})$, l'étiquette prédite pour l'instance $\mathbf{x}$. Nous avons alors

$$h_{\mathbf{w}}(\mathbf{x}) \stackrel{\text{def}}{=} \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \boldsymbol{\Psi}(\mathbf{x}, \mathbf{y}) \rangle = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \sum_{t=1}^{r} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, y_t, y_{t-1}) \rangle \,,$$

où $\boldsymbol{\phi}(\mathbf{x}, y_1, y_0) \stackrel{\text{def}}{=} \tilde{\boldsymbol{\phi}}(\mathbf{x}, y_1)$ est un vecteur dont les caractéristiques de type 2 sont nulles.

Solution par programmation dynamique

- Démontrons qu'il est possible de résoudre efficacement $h_{\mathbf{w}}(\mathbf{x}) = \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}}\langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, \mathbf{y})\rangle$ par programmation dynamique.
- Soit $q \stackrel{\text{def}}{=} |\Sigma|$ le nombre de lettres de notre alphabet. Maintenons une matrice $M \in \mathbb{R}^{q,r}$ telle que

$$M_{s,\tau} \stackrel{\text{def}}{=} \max_{(y_1,\ldots,y_\tau):y_\tau=s} \sum_{t=1}^{\tau} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, y_t, y_{t-1})\rangle$$

- Notez que $\max_{\mathbf{y}\in\mathcal{Y}}\langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, \mathbf{y})\rangle = \max_{s\in\Sigma} M_{s,r}$, et que

$$M_{s,1} = \langle \mathbf{w}, \tilde{\boldsymbol{\phi}}(\mathbf{x}, s)\rangle = \frac{1}{r}\sum_{i,j} w_{i,j,1}x_{i,1}\mathbb{1}_{[s=j]} = \frac{1}{r}\sum_{i=1}^{n} w_{i,s,1}x_{i,1}\,.$$

- Trouvons donc une récurrence entre $M_{s,\tau}$ et $M_{s,\tau-1}$ pour tout $\tau \geq 2$.

Solution par programmation dynamique

Pour y arriver, notons que pour tout $\tau \geq 2$, on a

$$
\begin{aligned}
M_{s,\tau} &\stackrel{\text{def}}{=} \max_{(y_1,\ldots,y_\tau):y_\tau=s} \sum_{t=1}^{\tau} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, y_t, y_{t-1}) \rangle \\
&= \max_{(y_1,\ldots,y_{\tau-1})} \sum_{t=1}^{\tau-1} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, y_t, y_{t-1}) \rangle + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, y_{\tau-1}) \rangle \\
&= \max_{s'} \max_{(y_1,\ldots,y_{\tau-1}):y_{\tau-1}=s'} \sum_{t=1}^{\tau-1} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, y_t, y_{t-1}) \rangle + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, s') \rangle \\
&= \max_{s'} \left(M_{s',\tau-1} + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, s') \rangle \right) .
\end{aligned}
$$

Pour tout $\tau \geq 2$, notre relation de récurrence est donc donnée par

$$
M_{s,\tau} = \max_{s'} \left(M_{s',\tau-1} + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, s') \rangle \right) .
$$

Solution par programmation dynamique

- Pour trouver la séquence de lettres $\mathbf{y} = (y_1, \ldots, y_r)$ prédite pour l'instance $\mathbf{x}$, étant donné que $\max_{s \in \Sigma} M_{s,r} = \max_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \boldsymbol{\Psi}(\mathbf{x}, \mathbf{y}) \rangle$, on a que

$$y_r = \operatorname*{argmax}_{s \in \Sigma} M_{s,r} \, .$$

- Sachant y_r, la valeur de y_{r-1} est alors donnée par

$$y_{r-1} = \operatorname*{argmax}_{s' \in \Sigma} \left(M_{s',r-1} + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, y_r, s') \rangle \right) \, .$$

- Donc, plus généralement, pour $\tau \in \{2, \ldots, r\}$, définissons

$$I_{s,\tau} \stackrel{\text{def}}{=} \operatorname*{argmax}_{s' \in \Sigma} \left(M_{s',\tau-1} + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, s') \rangle \right) \, .$$

- Alors $y_{r-1} = I_{y_r,r}$, et pour tout $\tau \in \{2, \ldots, r\}$, on a

$$y_{\tau-1} = I_{y_\tau,\tau} \, .$$

Solution par programmation dynamique

On obtient donc l'algorithme suivant :

Algorithme de programmation dynamique pour obtenir $h_{\mathbf{w}}(\mathbf{x})$:

- **Entrée :** une instance $\mathbf{x} \in \mathbb{R}^{n,r}$

  $n =$ nombre de pixels par image de caractère

  $r =$ nombre de caractères manuscrits de l'instance $\mathbf{x}$
- **Initialisation :** $M_{s,1} = \langle \mathbf{w}, \tilde{\boldsymbol{\phi}}(\mathbf{x}, s) \rangle \ \forall s \in \Sigma$
- Pour $\tau = 2, \ldots, r$:
  - Pour tout $s \in \Sigma$:

$$M_{s,\tau} = \max_{s'} \ (M_{s',\tau-1} + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, s') \rangle)$$

$$I_{s,\tau} = \operatorname*{argmax}_{s' \in \Sigma} \ (M_{s',\tau-1} + \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, s, s') \rangle)$$

- $y_r = \operatorname{argmax}_s M_{s,r}$
- Pour $\tau = r, \ldots, 2$:

$$y_{\tau-1} = I_{y_\tau,\tau}$$

- **Sortie :** $\mathbf{y} = (y_1, \ldots, y_r)$

Résumé

- Comment réduire un problème multi-classes en plusieurs problèmes de classification binaire.
- Utilisation de classificateurs linéaires pour l'apprentissage multi-classes.
- Le SVM multi-classes.
- Apprentissage d'un très grand nombre de classes mais avec une structure : prédiction de sorties structurées.
- Calcul efficace de $h_{\mathbf{w}}(\mathbf{x}) = \text{argmax}_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \mathbf{\Psi}(\mathbf{x}, \mathbf{y}) \rangle$ en utilisant la programmation dynamique.