Théorie et Algorithmes de l'Apprentissage Automatique

6 - Évaluation et sélection de modèles

Simon BERNARD
simon.bernard@univ-rouen.fr

Erreur en généralisation

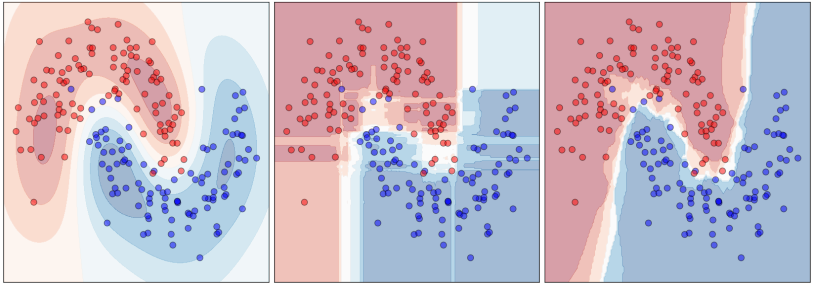


Figure 1 – SVM (RBF kernel), Forêts Aléatoires (100 arbres), K-Plus Proches Voisins (K=5)

- SVM, avec quel noyau ? si RBF quel γ et quel C ?
- Forêts Aléatoires, avec combien d'arbres, quelle profondeur des arbres ?
- *kNN*, avec combien de voisins ?

$\Rightarrow$ Comment choisir pour garantir de bonnes performances en généralisation ?

- La capacité à généraliser est définie via la fonction de perte :

$$\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$$

tel que $\ell(y, h(\mathbf{x})) \geq 0$ mesure à quel point la prédiction $h(\mathbf{x})$ est proche de la vraie valeur y

- La "qualité " globale d'un h est définie par le risque réel :

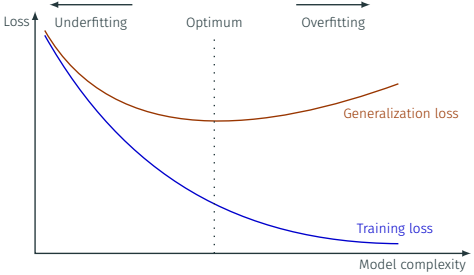
$$R(h) = E_{(\mathbf{x},y)\sim p_{X,Y}}[\ell(y, h(\mathbf{x}))]$$

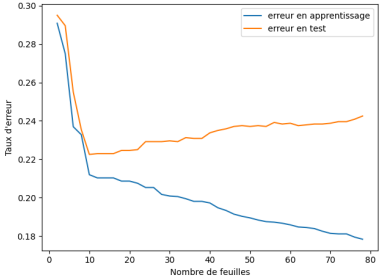
- $p_{X,Y}$ étant inconnue, l'apprentissage se base sur le risque empirique :

$$R_n(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(\mathbf{x}_i), y)$$

  à partir des observations $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$
- $R_n(h)$ est une approximation de $R(h)$
- La qualité de cette approximation dépend de n et de la "complexité" du modèle

- Si le modèle est trop complexe, $R_n(h)$ peut être faible mais $R(h)$ élevé (*sur-apprentissage*)





- La différence entre le risque empirique et le risque réel est appelée *generalization gap*
- On souhaite pouvoir garantir que ce *generalization gap* est faible :

$$P\left(\sup_{h\in\mathcal{H}} |R(h) - R_n(h)| > \epsilon\right) < \delta$$

  pour $\epsilon > 0$ et $\delta > 0$ les plus petits possibles[1]
- ϵ permet de formuler une borne sur l'erreur en généralisation
- δ est le degré de confiance en cette borne

1. Appelé théorie de l'apprenabilité *Probably Approximatively Correct* (PAC)

Borne sur l'erreur en généralisation

Pour tout $h \in \mathcal{H}$, avec une probabilité $1 - \delta$ on a

$$R(h) \leq R_n(h) + O\left(\sqrt{\frac{8d_{VC}\left(\ln\left(\frac{2n}{d_{VC}}\right)+1\right)+8\ln\left(\frac{4}{\delta}\right)}{n}}\right)$$

- $d_{VC} > 0$ est la dimension Vapnik-Chervonenkis[2] qui mesure la capacité de $\mathcal{H}$
- C'est une mesure de la "complexité" de $\mathcal{H}$

2. Correspond au cardinal du plus grand ensemble de points pour lequel $\exists h \in \mathcal{H}$ pouvant prédire n'importe quel combinaison de label pour ces points

Ce que nous dit ce résultat :

- L'erreur en généralisation est borné par $R_n(h)$ + un terme qui dépend notamment de la complexité de $\mathcal{H}$:

$$R(h) \leq \frac{1}{n}\sum_{i=1}^{n} \ell(h(\mathbf{x}_i, y_i)) + O\left(d_{VC}(\mathcal{H}), \frac{1}{n}, \log\left(\frac{1}{\delta}\right)\right)$$

- Minimiser la *loss* ne suffit pas, il faut aussi minimiser la complexité de $\mathcal{H}$
- De façon générale, $\mathcal{H}$ avec d_{VC} faibles préférables (linéaire > quadratique)
- Mais il faut aussi que $\mathcal{H}$ soit suffisamment complexe pour avoir un $R_n(h)$ faible

- Pour un hyperplan séparateur, la marge d'un point $\mathbf{x}_i$ est :

$$\gamma_i = \frac{|\mathbf{w}^\top \mathbf{x} + b|}{\|\mathbf{w}\|}$$

- La marge de l'hyperplan est la marge du point le plus proche :

$$\gamma = \min_{i=1,\ldots,n} \gamma_i$$

- Soit $\mathcal{H}_\gamma$ l'ensemble des hyperplans séparateurs avec une marge d'au moins γ. alors :

$$d_{VC}(\mathcal{H}_\gamma) \leq \min\left(\frac{R^2}{\gamma^2}, n\right) + 1$$

  où R est le rayon de la plus petite hypersphère contenant les données

- Alors, on peut montrer :

$$R(h) \leq R_n(h) + \sqrt{\frac{O\left(\frac{R^2}{\gamma^2}\right) + \ln\left(\frac{4}{\delta}\right)}{n}}$$

- Plus γ est grand, plus on peut obtenir un *generalization gap* petit

- Borne sur l'erreur en généralisation : indications solides sur les qualités souhaités de $\mathcal{H}$
- Mais...
  - d_{VC} est difficile à calculer
  - La borne est souvent pessimiste
  - Elle ne donne pas d'indication sur comment choisir les hyper-paramètres
- D'autres décompositions de l'erreur existent (e.g. biais-variance) qui donnent des indications plus pratiques
- Mais on ne peut pas garantir de bonnes performances en généralisation sans évaluer sur des données inconnues

Évaluation de modèles

Objectifs :

- Évaluation des modèles : quelle(s) mesure(s) de performance ?
- Estimation de la capacité de généralisation du modèle
- Procédures pratiques de sélection de modèles

Remarques :

- On se limitera dans ce cours aux problèmes de classification binaire
- Mais ce qui va être dit s'appliquent à d'autres types de problèmes d'apprentissage

Valeurs prédites ($\hat{y}$)	Vérité terrain (y) : positifs	Vérité terrain (y) : négatifs
positifs	True Positive (TP)	False Positive (FP)
négatifs	False Negative (FN)	True Negative (TN)

- TP : nombre de positifs classés positifs (bonnes prédictions)
- FP : nombre de négatifs classés positifs (erreurs)
- FN : nombre de positifs classés négatifs (erreurs)
- TN : nombre de négatifs classés négatifs (bonnes prédictions)

Plusieurs critères peuvent être construits à partir de cette matrice...

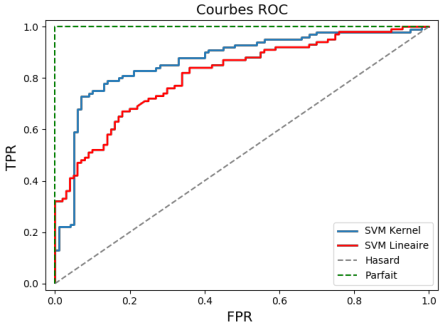
- Précision [3] $= \frac{TP}{TP + FP}$ ($\nearrow$)
- Rappel [4] $= \frac{TP}{TP + FN}$ ($\nearrow$)
- Spécificité [5] $= \frac{TN}{TN + FP}$ ($\nearrow$)
- Erreur Type-I [6] $= \frac{FP}{TN + FP}$ ($\searrow$)
- Erreur Type-II [7] $= \frac{FN}{TP + FN}$ ($\searrow$)

3. Mesure à quel point les prédictions positives du modèle sont fiables
4. Mesure à quel point le modèle réussi à détecter les vrais positifs
5. Mesure à quel point le modèle réussi à détecter les vrais négatifs
6. Mesure à quel point le modèle à échoué à prédire les positifs
7. Mesure à quel point le modèle à échoué à prédire les négatifs

- Taux d'erreur $= \dfrac{FP + FN}{TP + TN + FP + FN}$ ($\searrow$)
- Taux de bonne classification $= \dfrac{TP + TN}{TP + TN + FP + FN}$ ($\nearrow$)
- Taux de bonne classification balancé $= \dfrac{1}{2}\left(\dfrac{TP}{TP + FN} + \dfrac{TN}{TN + FP}\right)$ ($\nearrow$)
- $F_1 - score$ [8] $= 2\dfrac{Precision \times Rappel}{Precision + Rappel} = \dfrac{2TP}{2TP + FP + FN}$ ($\nearrow$)
- MCC (Mathews Correlation Coefficient) [9] $= \dfrac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$ ($\nearrow$)

8. équivalent au taux de bonne classification si les classes sont équilibrées
9. Mesure de corrélation entre les prédictions et les vraies valeurs. Particulièrement adapté pour les problèmes déséquilibrés

- $TPR =$ taux de vrais positifs (rappel)
- $FPR =$ taux de faux positifs (Erreur Type-I)
- Courbe ROC : $TPR = f(FPR)$ (pour tout point de fonctionnement/seuil de décision)



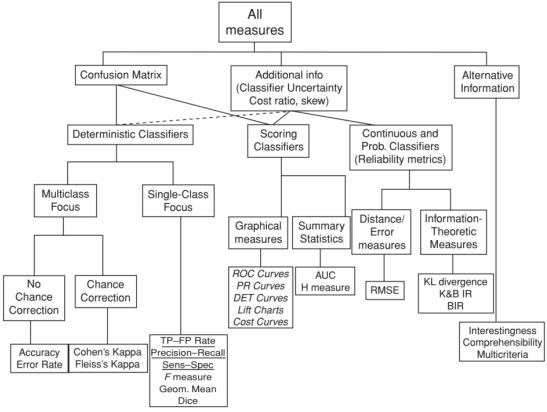
Permet de comparer visuellement plusieurs modèles, pour plusieurs compromis TPR/FPR

- Soit $\mathcal{D} = \{(\mathbf{x}_i, y_i = 1) \cup (\mathbf{x}_j, y_j = -1), i = 1, \ldots, n_+, j = 1, \ldots, n_-\}$
- L'AUC (*Area Under the ROC Curve*) est définie par :

$$AUC = \frac{\sum_{i=1}^{n_+} \sum_{j=1}^{n_-} \mathbf{1}_{h(\mathbf{x}_i) > h(\mathbf{x}_j)}}{n_+ n_-}$$

  avec $\mathbf{1}$ la fonction indicatrice
- l'AUC est comprise entre 0 et 1 ($\nearrow$)
- Privilégie la fonction de décision telle que $h(\mathbf{x}_i) > h(\mathbf{x}_j), \forall\ (y_i = 1, y_j = -1)$

- Il en existe beaucoup d'autres, chacune avec des spécificités
- Il faut choisir la bonne mesure de performances en fonction de la tâche d'apprentissage, des modèles étudiés, de la problèmatique spécifique, etc.



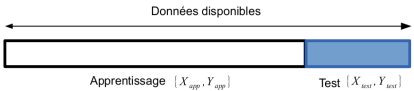
- L'enjeu est de pouvoir mesurer ces performances sur des données inconnues
- On "construit" h sur des données d'apprentissage $\mathcal{D}_n = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$
- On s'intéresse à $S(\mathcal{D}, h)$, la performance réelle de h sur des données inconnues

Capacité de généralisation

La capacité de h à donner de bonnes performances (mesurées avec une des métriques précédentes) lorsqu'il est testé sur des données autres que celles qui ont servi à l'apprentissage

- Comment estimer $S(\mathcal{D}, h)$ en pratique ?

Découper aléatoirement $\mathcal{D}_n$ en deux sous-ensembles disjoints $\mathcal{D}_{app}$ et $\mathcal{D}_{test}$



- $\mathcal{D}_{app} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n_{app}\}$: données pour l'apprentissage de h
- $\mathcal{D}_{test} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n_{test}\}$: données pour évaluer la capacité de généralisation de h

Remarques :

- Plus n_{app} est grand, meilleur est l'apprentissage
- Plus n_{test} est grand, meilleure est l'estimation de la performance en généralisation de h
- $\mathcal{D}_{test}$ n'est utilisé qu'une seule fois !

- On peut calculer un intervalle de confiance en fonction des résultats sur $\mathcal{D}_{test}$
- Soit $\hat{p}$, la probabilité de succès, alors [10] :

$$\hat{p}_\alpha = \hat{p} \pm z_\alpha \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} = \frac{n_S}{n} \pm \frac{z_\alpha}{n} \sqrt{\frac{n_S n_F}{n}}$$

  avec α le niveau de confiance, n_S/n_F le nombre de succès/échecs, et z_α le quantile de la loi normale
- Exemple : pour $n = 100$, $\hat{p} = 0.9$, $\alpha = 1 - 0,95 = 0.05$, $z_\alpha = 1.96$, on a :

$$0.9 \pm 1.96 \times \sqrt{\frac{0.9 \times 0.1}{100}} = 0.9 \pm 0.06$$

- Donc, on peut dire que 95% du temps : $0,84 < \hat{p} < 0,96$

10. Intervalle de confiance binomial, ici avec l'intervalle de Wald, mais il en existe d'autres

- En augmentant le nombre de tests, on augmente la confiance
- Séparation aléatoires répétées : plusieurs découpages aléatoires $\mathcal{D}_{app}/\mathcal{D}_{test}$
- Validation croisée (*K-fold cross-validation*) (détails plus loins)
- *Leave-one-out* : app. de $n-1$ données et test sur la donnée restante, répété n fois.
- *Bootstrap* : échantillonnage avec remise, répété B fois

- Répétitions : permet de calculer des statistiques sur les performances
- Soit N le nombre de répétitions et ϵ_i la performance de h sur le découpage i
- Statistiques :
  - Moyenne : $\bar{\epsilon} = \frac{1}{N} \sum_{i=1}^{N} \epsilon_i$
  - Variance (non-biaisée) : $\sigma^2 = \frac{1}{N-1} \sum_{i=1}^{N} (\epsilon_i - \bar{\epsilon})^2$
  - Intervalle de confiance :

$$\bar{\epsilon} \pm t_{N-1,\alpha/2} \frac{\sigma}{\sqrt{N}}$$

où $t_{N-1,\alpha/2}$ est le quantile de la loi de Student [11]

11. Ces quantiles sont données dans des tables de statistiques

Bonnes pratiques :

- Simuler des conditions réelles (distribution de classes, bruit, etc.)
- Utiliser des métriques adaptées à la tâche
- Fiabiliser les estimations (plusieurs découpages, validation croisée)
- Privilégier la stabilité plutôt que la performance (écart-type faible)
- Ne pas oublier de comparer avec des modèles simples (baseline)

Sélection de modèles

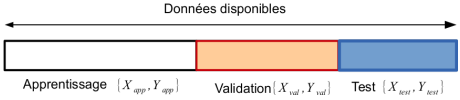
Sélection de modèle

Consiste à choisir le modèle le plus pertinent pour une tâche, parmi un ensemble de modèles candidats et compte-tenu des données

Par exemple :

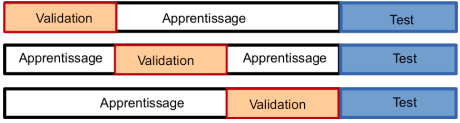
- Quelle valeur d'hyper-paramètres pour obtenir les meilleures performances en généralisation ?
- Quel algorithme d'apprentissage pour un problème d'apprentissage donné ?
- Quelle architecture (noyaux pour les SVM, nombres de neurones et de couches pour les réseaux de neurones, nombre de classifieurs pour les ensembles, etc.) ?
- ...

Comment sélectionner le meilleur modèle sans toucher à $\mathcal{D}_{test}$?



1. Découper aléatoirement $\mathcal{D}_n = \mathcal{D}_{app} \cup \mathcal{D}_{val} \cup \mathcal{D}_{test}$
2. Apprendre chaque modèle possible sur $\mathcal{D}_{app}$
3. Évaluer leur performances sur $\mathcal{D}_{val}$
4. Sélectionner le modèle qui donne la meilleure performance sur $\mathcal{D}_{val}$

Si le nombre de données dans $\mathcal{D}_n$ est faible ? (ou pour fiabiliser)



1. Découper aléatoirement $\mathcal{D}_n = \mathcal{D}_{app} \cup \mathcal{D}_{test}$
2. Découper aléatoirement $\mathcal{D}_{app} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_K$ en K sous-ensembles disjoints (*fold*)
3. Pour $k = 1, \ldots, K$
   3.1 Mettre de coté $\mathcal{D}_k$
   3.2 Apprendre h sur les $K - 1$ *fold* restants
   3.3 Évaluer sa performance R_k sur $\mathcal{D}_k$
4. Moyenner les K mesures de performances R_k

Pour un jeu d'hyper-paramètres $\mathcal{P} = \{p_1, p_2, \dots\}$ donné...

Procédure :

Entrées : $\mathcal{P} = \{h_{p_1}, h_{p_2}, \dots\}$, $\mathcal{D}_n = \{(\mathbf{x}_i, y_i), i = 1, \dots, n\}$,

1. Découper les données : $(\mathcal{D}_{app}, \mathcal{D}_{test}) \leftarrow \mathit{SplitData}(\mathcal{D}_n, \mathit{options})$
2. Sélectionner le meilleur modèle : $h^* \leftarrow \mathit{Selection}(\mathcal{D}_{app}, \mathcal{P})$
3. Évaluer le modèle retenu : $\mathit{Perf} \leftarrow \mathit{EvaluerPerformance}(\mathcal{D}_{test}, h^*)$

Fonction $h^* \leftarrow \mathit{Selection}(\mathcal{D}, \mathcal{P})$:

1. Redécouper les données $(\mathcal{D}_{app}, \mathcal{D}_{val}) \leftarrow \mathit{SplitData}(\mathcal{D}, \mathit{options})$
2. Pour $p_i \in \mathcal{P}$:
   2.1 Apprendre le modèle : $h_i \leftarrow \mathit{Apprentissage}(\mathcal{D}_{app}, p_i)$
   2.2 Évaluer le modèle : $\mathit{Perf}_i \leftarrow \mathit{EvaluerPerformance}(\mathcal{D}_{val}, h_i)$
3. Sélectionner le meilleur hyper-paramètre : $p^* \leftarrow \mathbf{arg\,min}_i \mathit{Perf}_i$
4. $h^* \leftarrow \mathit{Apprentissage}(\mathcal{D}, p^*)$

Comparaison de modèles

1. Choisir une mesure de performance pertinente S
2. Pour chaque base $\mathcal{D}_j$:
   - 2.1 Réaliser k découpages $\mathcal{D}_{j,app_k}$ / $\mathcal{D}_{j,test_k}$
   - 2.2 Pour chaque méthode $\mathcal{A}_i$:
     - 2.2.1 $h_{i,j,k} \leftarrow Selection(\mathcal{D}_{j,app_k}, \mathcal{P})$
     - 2.2.2 Evaluer sa performance $S_{i,j,k}$ sur $\mathcal{D}_{j,test_k}$
   - 2.3 Calculer la performance moyenne $S_{i,j}$
3. Comparaison globale ou sur chaque base avec un test de significativité

Principe :

- On vérifie une hypothèse statistique sur les performances des modèles
- L'hypothèse usuelle est que les performances des modèles sont équivalentes
- Le test statistique permet de rejeter ou non cette hypothèse
- Le test porte sur les distributions des performances des modèles
- Si les performances sont significativement différentes, on peut conclure que les modèles sont différents et que l'un est meilleur que l'autre

- Comparer les performances de **deux** modèles sur N datasets
- Soit $S_{i,j}$ la performance de $\mathcal{A}_i$ sur $\mathcal{D}_j$
- Hypothèse testée : les distributions $S_{1,j}, S_{2,j}, \forall j$ sont équivalentes
- Procédure :
  - $S_j = |S_{1,j} - S_{2,j}|$
  - Calcul des rangs R_j de S_j (1 pour le plus petit S_j, 2 pour le suivant, etc.)
  - *Signed rank sum* :
$$W = \sum_j \text{sign}(S_j) R_j$$
  - Calcul d'une p-value pour W (table de Wilcoxon)
  - Si $p < \alpha$, on rejette l'hypothèse d'équivalence

- Comparer les performances de $k > 2$ modèles sur N datasets
- Soit $r_{i,j}$ le rang de $\mathcal{A}_i$ sur $\mathcal{D}_j$
- Hypothèse testée : les $r_{i,j}, \forall i, j$ sont équivalents, i.e. tous les algorithmes se valent
- Procédure :
  - Calcul des rangs moyens $\bar{r}_i = \frac{1}{n} \sum_j r_{i,j}$
  - Calcul de la statistique de Friedman :

$$\chi_F^2 = \frac{12N}{k(k+1)} \left(\sum_i \bar{r}_i^2 - \frac{k(k+1)^2}{4} \right)$$

  - Calcul d'une p-value pour χ_F^2 (table de Friedman)
  - Si $p < \alpha$, on rejette l'hypothèse d'équivalence

- Si hypothèse rejetée par Friedman, comparaison deux à deux avec un test post-hoc
- Nemenyi calcule un seuil de différence significative entre les rangs moyens

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

- Si la différence entre les rangs moyens de deux classifieurs est supérieure à CD, alors les performances sont significativement différentes