S1-MAJEUR - Apprentissage Automatique

Chapitre 4 - Réaliser un projet de machine learning de A à Z

Petit rappel

Un problème de machine laerning

Contexte :

- Soit un ensemble de données $D = \{(x_i, y_i) \in X \times Y, i = 1, \ldots, n\}$.
- Les couples (x, y) sont générés à partir d'une distribution $P(X, Y)$, généralement inconnue.

Objectif :

- Apprendre une fonction de prédiction $h : \hat{y} = h(x)$, pour tout $x \in X$, où $\hat{y}$ est la prédiction de Y pour x.
- Pour chaque $(x_i, y_i) \in D$, souhaitons que $h(x_i)$ prédise correctement la valeur y_i.
- Plus crucial encore, visons à ce que h puisse prédire correctement la valeur y_j pour toute instance future $x_j \notin D$, démontrant ainsi sa capacité à généraliser et sa performance en généralisation.

les modèles du chapitre 3

Régression Linéaire:

$$Y = \omega_0 + \omega_1 X^1 + \omega_2 X^2 + \ldots + \omega_p X^p$$

Régression Logistique(pour des problèmes de classification binaire):

$$P(Y = 1) = \frac{1}{1 + e^{-(\omega_0 + \omega_1 X^1 + \omega_2 X^2 + \ldots + \omega_p X^p)}}$$

Ridge Regression(introduit une pénalité L2 pour contrôler la complexité du modèle):

$$\text{Minimiser } J(\omega) = \sum_{i=1}^{n} (y_i - \omega_0 - \sum_{j=1}^{p} \omega_j X^{ij})^2 + \lambda \sum_{j=1}^{p} \omega_j^2$$

Je rappelle qu'ici que notre fameux biais c'est b = W_0

les modèles du chapitre 3

Lasso Regression(introduit une pénalité L1 pour encourager la parcimonie dans les coefficients):

$$\text{Minimiser } J(\omega) = \sum_{i=1}^{n}(y_i - \omega_0 - \sum_{j=1}^{p}\omega_j X^{ij})^2 + \lambda \sum_{j=1}^{p}|\omega_j|$$

ElasticNet(combine les pénalités L1 et L2 pour profiter des avantages des deux):

$$\text{Minimiser } J(\omega) = \sum_{i=1}^{n}(y_i - \omega_0 - \sum_{j=1}^{p}\omega_j X^{ij})^2 + \lambda_1 \sum_{j=1}^{p}|\omega_j| + \lambda_2 \sum_{j=1}^{p}\omega_j^2$$

les métriques du chapitre 3

Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

où n est le nombre d'observations, y_i sont les valeurs réelles, et $\hat{y}_i$ sont les valeurs prédites.

Le coefficient de corrélation de Pearson r entre deux variables X et Y est donné par :

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

Où :

- n est le nombre d'observations,
- X_i et Y_i sont les valeurs des observations,
- $\bar{X}$ et $\bar{Y}$ sont les moyennes de X et Y, respectivement.

1 Le preprocessing avant tout

Techniques de Prétraitement des Données

1. Normalisation (définir une plage) des données.
2. Standardisation (réduire et centrer) des données.
3. Gestion des valeurs manquantes (imputation).
4. Encodage des variables catégorielles.
5. Réduction de la dimension (PCA, LDA).
6. Traitement des données déséquilibrées.
7. Suppression des valeurs aberrantes.
8. Transformation log, exponentielle, etc.
9. Extraction de caractéristiques (Feature Engineering) voir les deux slides qui suivent.
10. Discrétisation des variables continues.
11. Création de nouvelles variables.

Corrélation entre Caractéristiques et Élimination

Problème de Corrélation entre Caractéristiques :

- **Redondance d'Information :** Des caractéristiques fortement corrélées apportent des informations similaires.
- **Instabilité des Coefficients :** Les modèles peuvent devenir sensibles aux petites variations des données.
- **Interprétation Difficile :** Difficulté à attribuer l'importance individuelle des caractéristiques.
- **Corrélation n'implique pas Causalité :** Une forte corrélation entre deux caractéristiques ne signifie pas nécessairement une relation de cause à effet.

Corrélation entre Caractéristiques et Élimination

Élimination des Caractéristiques Corrélées :

- **Analyse de Corrélation :** Identifiez les paires de caractéristiques fortement corrélées.
- **Seuil de Corrélation :** Établissez un seuil au-delà duquel les caractéristiques sont considérées corrélées.
- **Retrait ou Combinaison :** Retirez l'une des caractéristiques de chaque paire ou combinez-les en une seule.
- **Utilisation de Méthodes de Sélection :** La méthode RFE éliminent automatiquement les caractéristiques moins informatives.

2 La séléction des caractéristiques

Sélection des Caractéristiques

Objectif : Identifier les caractéristiques les plus informatives pour améliorer les performances du modèle.

Approches Courantes :

- **Filtre :** Évaluation des caractéristiques indépendamment de l'algorithme d'apprentissage.
- **Enveloppe :** Utilisation d'un algorithme pour évaluer la performance des sous-ensembles de caractéristiques.
- **Incorporée :** Sélection pendant le processus d'apprentissage.

Considérations :

- **Redondance :** Éviter les caractéristiques similaires.
- **Irrelevance :** Éliminer les caractéristiques non informatives.

Méthodes de Sélection des Caractéristiques - État de l'Art

Méthodes Couramment Utilisées :

- **RFE (Recursive Feature Elimination) :** Élimination itérative des caractéristiques les moins importantes.
- **LASSO (Least Absolute Shrinkage and Selection Operator) :** Intègre une pénalité pour favoriser la sparsité.
- **SVM-RFE :** Utilise des SVM pour évaluer l'importance des caractéristiques.
- **Forward Selection / Backward Elimination :** Construction ou élimination séquentielle des caractéristiques.
- **Random Forest Importance :** Évalue l'importance des caractéristiques via un ensemble de forêts aléatoires.

Critères de Sélection :

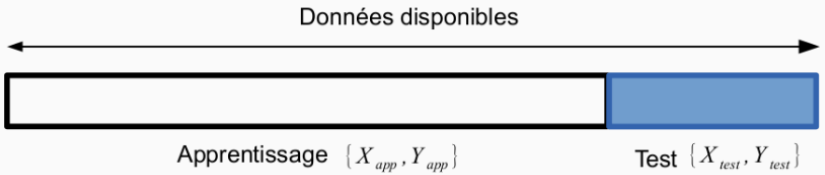
- **Performance du Modèle :** Mesure de la contribution des caractéristiques à la performance globale.
- **Stabilité :** La sélection reste cohérente avec des variations dans l'ensemble de données.

Division des données en X en X_app ou X_train et X_test

Estimation des Capacités de Généralisation - Ensembles Apprentissage/Test

Découper aléatoirement D_n en deux sous-ensembles disjoints D_{app} et D_{test}:

- $D_{\text{app}} = \{(x_i, y_i), i = 1, \ldots, n_{\text{app}}\}$: données servant à l'apprentissage de h.
- $D_{\text{test}} = \{(x_i, y_i), i = 1, \ldots, n_{\text{test}}\}$: données servant à 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.
- D_{test} n'est utilisé qu'une seule fois!

Fiabiliser l'Estimation - Méthodes de Division des Données

Pour fiabiliser l'estimation, on reproduit l'opération pour plusieurs D_{test} différents:

- En reproduisant l'estimation sur plusieurs découpages apprentissage/test, on fiabilise l'estimation (performances moyennes).
- Cela permet aussi de calculer des intervalles de confiance, des écarts-types, etc.

Plusieurs méthodes:

- **Séparations Aléatoires Répétées:** Plusieurs découpages aléatoires $D_{\text{app}}/D_{\text{test}}$ (pour la classification : "stratifiées", c'est-à-dire en conservant les proportions de classes).
- **Validation Croisée (K-fold Cross-Validation):** Détails plus loin.
- **Leave-One-Out:** Apprentissage sur $n-1$ données et test sur la donnée restante, répété n fois.
- **Bootstrap:** Tirage aléatoire avec remise dans D et mesure sur les données restantes (out-of-bag), répété plusieurs fois.

Qualité d’un modèle

rappel de la différence entre un risque

- **L'erreur réelle :** mesure la performance réelle du modèle sur de nouvelles données.

- **L'erreur empirique :** évalue la performance sur l'ensemble d'entraînement.

- **L'erreur résiduelle :** quantifie la différence entre les prédictions du modèle et les valeurs réelles dans l'ensemble d'entraînement.

Fonction de Coût et Classification

Fonction de Coût :

- Mesure de la capacité à généraliser basée sur une fonction de perte.
- Objectif : Pénaliser les erreurs de prédiction de $h(x)$ par rapport à y.
- Pour la classification binaire avec $Y = \{-1, 1\}$:
  - Coût 0-1 (zero-one loss) :

$$\mathcal{L}(h(x), y) = \begin{cases} 0 & \text{si } yh(x) > 0 \\ 1 & \text{si } yh(x) \leq 0 \end{cases}$$

- Mesure le nombre d'erreurs de classification.

Fonction Risque, Apprentissage, et Décision Optimal

Fonction Risque en Généralisation :

- Mesure la performance en généralisation en fonction de la perte
- Risque réel : $R(h) = \mathbb{E}_{X,Y}[\mathcal{L}(h(X), Y)]$

Risque Empirique :

- Utilisation des observations $D = \{(x_i, y_i)\}$.
- Risque empirique : $R_n(h) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h(x_i), y_i)$.
- Objectif : Minimiser le risque empirique $h_n = \arg\min_{h \in H} R_n(h)$.

Risque Empirique, Sur-apprentissage, et Espace d'Hypothèses

Risque Empirique et Sur-apprentissage :

- Sélectionner h basé sur $R_n(h_n)$ n'est généralement pas pertinent.
- Il est possible d'obtenir $R_n(h_n) = 0$ avec $R(h_n)$ élevé.
- L'enjeu est de minimiser la différence entre le risque empirique et le risque réel (generalization gap).

Espace d'Hypothèses H:

- Représentation graphique de l'espace d'hypothèses H avec coût $\mathcal{L}$.
- Algorithme A cherche h pour minimiser $R(h)$ en se basant sur $R_n(h_n)$

Contrôle de la complexité

Contrôler la Complexité : Régularisation

Minimisation du Risque Empirique Régularisé :

$$\min_h \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h(x_i), y_i) + \lambda \Omega(h)$$

- $\Omega(h)$ est une fonction de régularisation et $\lambda > 0$ un hyper-paramètre de régularisation.
- $\lambda \gtrsim 1 \rightarrow$ favorise une h de faible complexité.
- Exemple : nous y reviendrons dans les SVM .

Contrôler la Complexité : Régularisation avec contrainte

Minimisation du Risque Empirique Régularisé :

$$\min_h \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h(x_i), y_i) + \lambda \Omega(h)$$

- $\Omega(h)$ est une fonction de régularisation et $\lambda > 0$ un hyper-paramètre de régularisation.
- $\lambda \gtrsim 1 \rightarrow$ favorise une h de faible complexité.
- Exemple : nous y reviendrons dans les SVM .
- Les contraintes peuvent s'écrire plus simplement :

$$yi \cdot h(x_i) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

Contrôler la Complexité : Régularisation sans contrainte

Minimisation du Risque Empirique Régularisé :

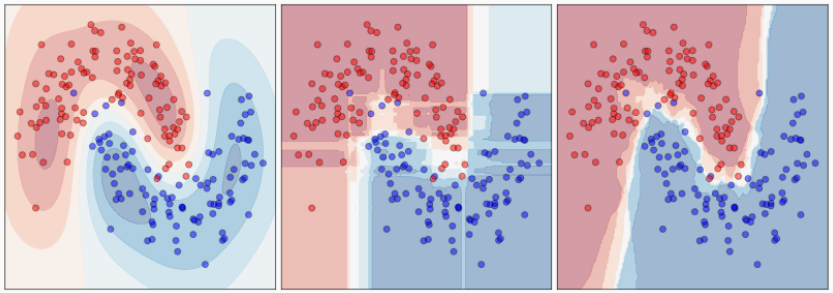
$$\min_h \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(h(x_i), y_i) + \lambda \Omega(h)$$

- $\Omega(h)$ est une fonction de régularisation et $\lambda > 0$ un hyper-paramètre de régularisation.
- $\lambda \gtrsim 1 \rightarrow$ favorise une h de faible complexité.
- Exemple : nous y reviendrons dans les SVM .
- On peut réécrire le problème d'optimisation sans contraintes :

$$\min_h \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - yi \cdot h(x_i)) + \lambda \|w\|^2$$

avec $\mathcal{L}(h(x_i), y_i) = \max(0, 1 - yi \cdot h(x_i))$, $\Omega(h) = \|w\|^2$, et $\lambda = C$.

Méthodes d'Apprentissage et Hyper-paramètres



Questions Essentielles :

- SVM : Quel noyau choisir ? En cas de RBF, quelles valeurs pour γ et C ?
- Forêts Aléatoires : Combien d'arbres et quelle profondeur pour les arbres ?
- k-Plus Proches Voisins (kNN) : Avec combien de voisins ?

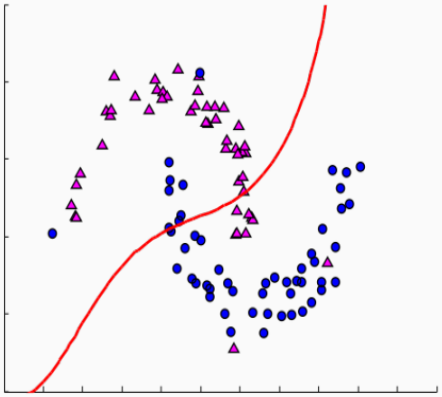
Défis :

- Comment choisir la meilleure fonction de prédiction h parmi ces méthodes et paramètres ?
- Comment garantir que h possède la capacité à généraliser efficacement ?

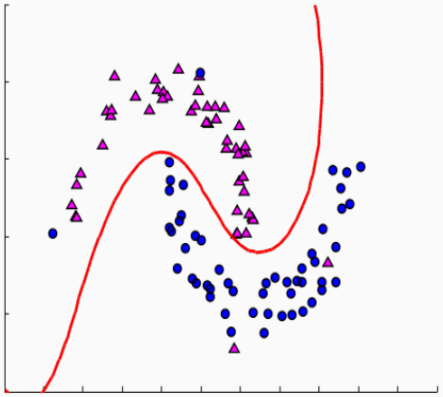
Contrôler la Complexité

Influence de l'hyper-paramètre C

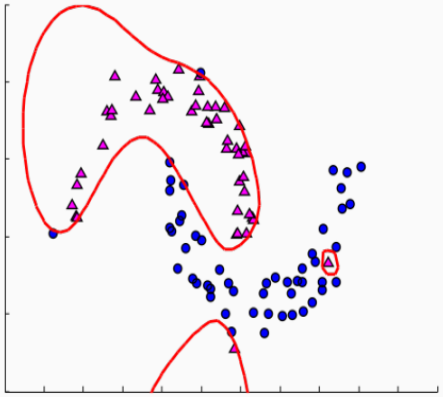
Ici avec un modèle de SVM polynomial :



C trop petit (sous-apprentissage)



C correct



C trop grand (sur-apprentissage)

Evaluaton de modèle

Sélection et Évaluation de Modèle

Objectifs :

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

Remarques :

- Ce qui va être dit s'applique à d'autres types de problèmes d'apprentissage.

Matrice de Confusion

Matrice de Confusion pour la Classification Binaire:

	Vérité Terrain (y)	**Positifs**	**Négatifs**
Valeurs Prédites ($\hat{y}$)	**Positifs**	(TP)	(FP)
	Négatifs	(FN)	(TN)

- TP (True Positive) : nombre de positifs classés positifs (bonnes prédictions).
- FP (False Positive) : nombre de négatifs classés positifs (erreurs).
- FN (False Negative) : nombre de positifs classés négatifs (erreurs).
- TN (True Negative) : nombre de négatifs classés négatifs (bonnes prédictions).

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

Mesures de Performances

Critères Classiques:

- Taux d'erreur = $\frac{FP+FN}{TP+TN+FP+FN}$ (on veut qu'il soit bas)
- Taux de bonne classification = $\frac{TP+TN}{TP+TN+FP+FN}$ (on veut qu'il sopit élevé)

Critères Utilisés en Recherche d'Information:

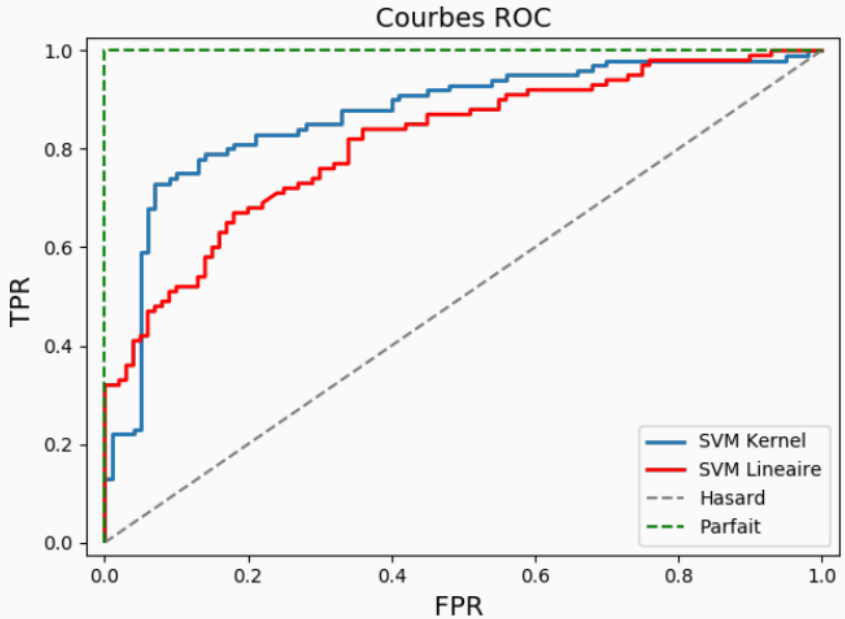
- Précision = $\frac{TP}{TP+FP}$ (on veut qu'elle soit élevé ou bonne)
- Rappel = Taux de vrais positifs = $\frac{TP}{TP+FN}$
- F-mesure = $\frac{2 \times \text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$ (on veut qu'il soit élevé)

Plusieurs autres mesures existent, chacune adaptée à des contextes spécifiques.

Mesures de Performances - Courbe ROC

Courbe ROC (Receiver Operating Characteristic):

- **TPR** : Taux de vrais positifs (fraction des positifs effectivement détectés) et **FPR** : Taux de faux positifs (fraction des négatifs incorrectement détectés).
- Courbe ROC : **TPR = f(FPR)** (pour tout point de fonctionnement/seuil de décision) . FRP= $\frac{FP}{TN+FP}$
- Permet de comparer visuellement plusieurs modèles, pour plusieurs compromis **TPR/FPR**.



Mesures de Performances - Aire sous la Courbe ROC

Aire sous la Courbe ROC (AUC):

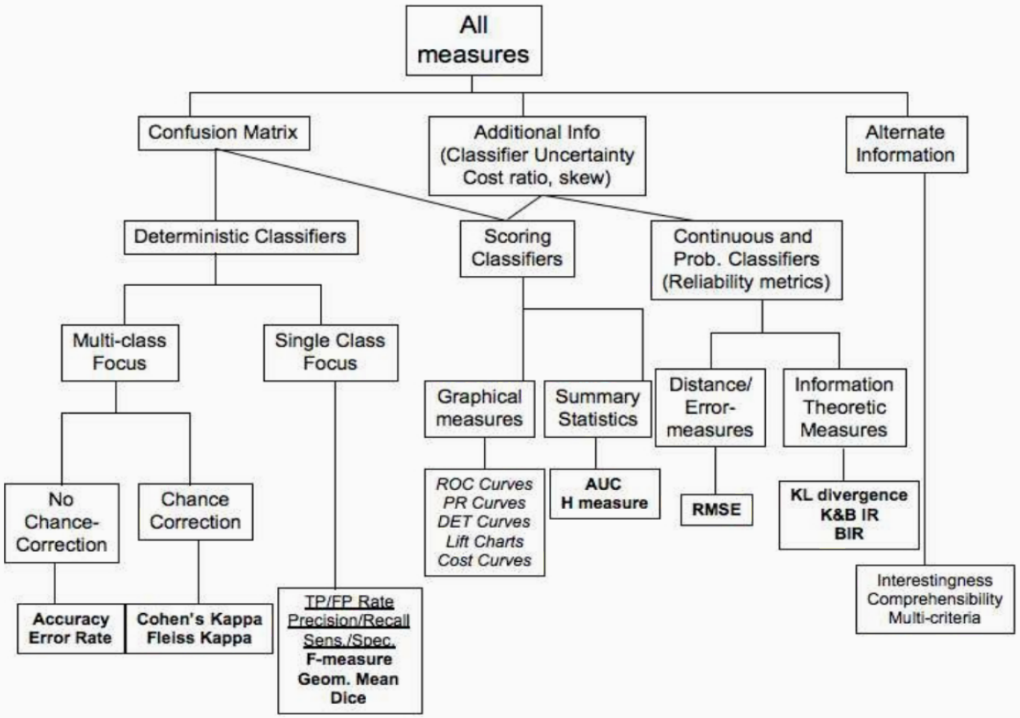
- Soit $D = \{(x_i, y_i = 1) \cup (x_j, y_j = -1), i = 1, \ldots, n_+, j = 1, \ldots, n_-\}$.
- L'AUC (Area Under the ROC Curve) est définie par :

$$AUC = \frac{1}{n_+ \times n_-} \sum_{i=1}^{n_+} \sum_{j=1}^{n_-} 1[h(x_i) > h(x_j)]$$

- AUC est comprise entre 0 et 1 s.
- Privilégie la fonction de décision telle que $h(x_i) > h(x_j)$, $\forall (y_i = 1, y_j = -1)$.

Mesures de Performances - Choix de la Mesure

- Il existe de nombreuses autres mesures de performances.
- Chacune d'elles peut désigner un modèle différent comme étant "le meilleur".
- Le choix de la bonne mesure de performances dépend de la tâche d'apprentissage, des modèles étudiés, de la problématique spécifique, etc.



Selection de modèle : validation croisée

Sélection de Modèle

L'objectif est de choisir le modèle le plus adapté parmi une sélection, en tenant compte des données disponibles.

- Quelle valeur d'hyper-paramètres offre les meilleures performances de généralisation ?
- Quel algorithme d'apprentissage convient le mieux à un problème donné ?
- Quelles caractéristiques sélectionner (sélection de caractéristiques) ?
- Quelle architecture adopter (noyaux pour les SVM, nombre de neurones et de couches pour les réseaux de neurones, nombre de classifieurs pour les ensembles, etc.) ?

Sélection de Modèle - Ensembles de Validation

Comment choisir le meilleur modèle sans toucher aux D_{test}:

1. Diviser aléatoirement $D_n = D_{\text{app}} \cup D_{\text{val}} \cup D_{\text{test}}$.
2. Entraîner chaque modèle possible sur D_{app}.
3. Évaluer ses performances sur D_{val}.
4. Sélectionner le modèle offrant les meilleures performances sur D_{val}.
5. Tester le modèle retenu sur D_{test}.

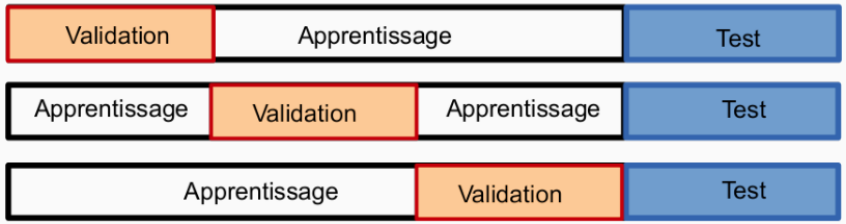
Remarque:

- D_{test} n'est utilisé qu'une seule fois!
- Le modèle retenu est celui qui excelle sur D_{val}.

Sélection de Modèle - Validation Croisée (K-fold Cross-Validation)

En cas de faible volume de données dans D_n (ou pour une estimation plus fiable):

1. Diviser aléatoirement $D_n = D_{\text{app}} \cup D_{\text{test}}$.
2. Diviser aléatoirement $D_{\text{app}} = D_1 \cup \ldots \cup D_K$ en K sous-ensembles disjoints (folds).
3. Pour $k = 1, \ldots, K$:
   - 3.1 Mettre de côté D_k.
   - 3.2 Entraîner h sur les $K - 1$ folds restants.
   - 3.3 Évaluer ses performances R_k sur D_k.
4. Calculer la moyenne des K mesures de performances R_k.



En pratique : protocole expérimentale et rigourux

En pratique - Méthodologie pour le Choix d'Hyper-paramètres

Objectif : Sélectionner la fonction de décision offrant les meilleures performances sur des données futures, avec une famille de modèles et différents hyper-paramètres $H = \{p_1, p_2, \ldots\}$.

Procédure :

1. Découper les données $(D_{\text{app}}, D_{\text{test}}) \leftarrow$ `SplitData`(D, options)
2. Sélectionner le meilleur modèle : $h^* \leftarrow$ `Selection`(D_{app}, H)
3. Évaluer le modèle retenu : Perf $\leftarrow$ `EvaluerPerformance`(D_{test}, h^*)

En pratique - Méthodologie pour le Choix d'Hyper-paramètres (Suite)

Fonction $h^* \leftarrow$ **Selection**(D, H) :

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

En pratique - Méthodologie pour Comparer Plusieurs Algorithmes d'Apprentissage

Méthodologie :

1. Choisir une mesure de performance pertinente R.
2. Pour chaque base D_j:
   - 2.1 Réaliser k découpages $D_{j,\text{app}k}/D_{j,\text{test}k}$.
   - 2.2 Pour chaque méthode A_i:
     - 2.2.1 Déterminer les meilleurs hyper-paramètres avec la fonction `Selection`.
     - 2.2.2 Apprendre le modèle $h_{i,j,k}$ avec $D_{j,\text{app}k}$.
     - 2.2.3 Évaluer sa performance $R_{i,j,k}$ sur $D_{j,\text{test}k}$.
   - 2.3 Calculer la performance moyenne $R_{i,j}$.
3. Comparer les performances sur chaque base et/ou globalement avec un test statistique de significativité.

Explicabilité dans les Modèles d'Apprentissage

Explicabilité : Capacité à comprendre et expliquer les décisions prises par un modèle.

- **Interprétabilité des Caractéristiques :** Comprendre l'importance des caractéristiques pour les décisions.
- **Visualisation :** Utilisation de graphiques pour représenter les relations.
- **Méthodes Lien-Locaux :** Analyse des prédictions autour d'une instance spécifique.
- **Méthodes Globales :** Compréhension du modèle dans son ensemble.

Une bonne explicabilité renforce la confiance et facilite la détection des biais. Exemple d'une méthode : Shape .