

Machine Learning, méthodes et solutions

Évaluation et Apprentissage de modèles

En classification :

Précision

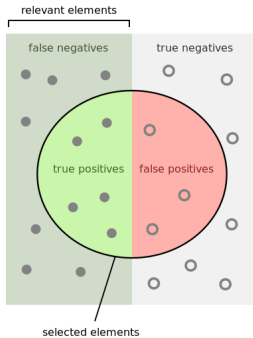
$$\frac{\text{vrais positifs}}{\text{vrais positifs} + \text{faux positifs}}$$

Rappel

$$\frac{\text{vrais positifs}}{\text{vrais positifs} + \text{faux négatifs}}$$

F-mesure moyenne harmonique entre précision et rappel (aussi appelée F1 score)

Évaluation — outils — précision, rappel



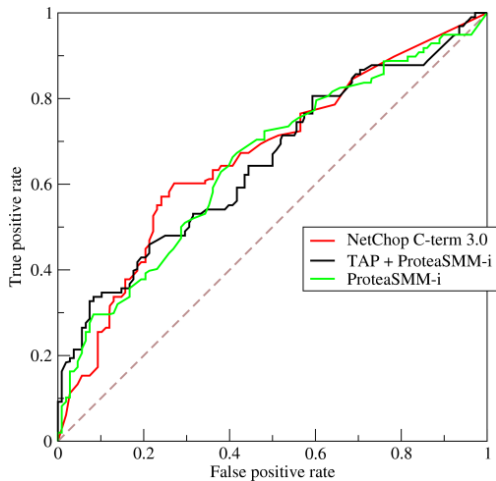
How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

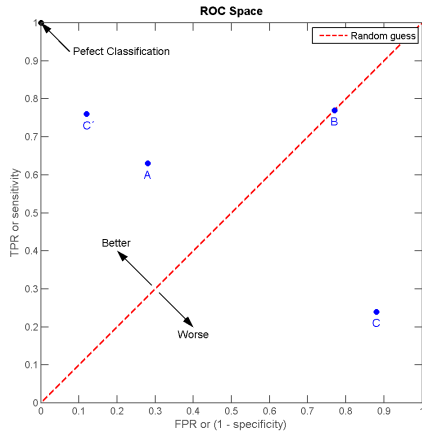
How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

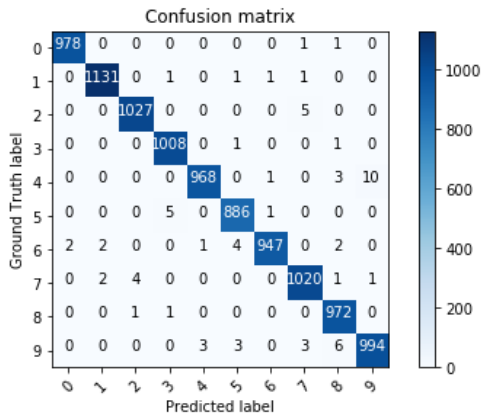
Outils — courbe ROC



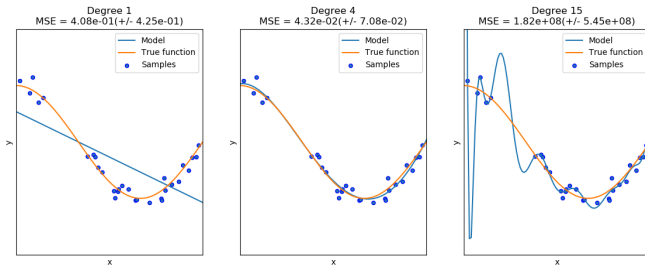
Outils — courbe ROC



Évaluation — outils — matrice de confusion

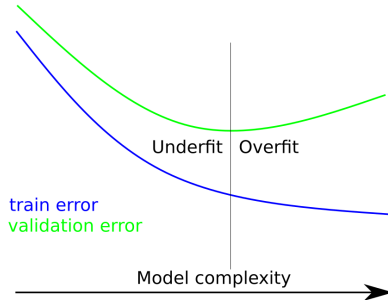


Entraînement supervisé d'un modèle — overfit



Problème : trop minimiser la perte n'est pas bon !

Apprentissage de modèles



→ Minimiser la perte sur un ensemble de validation

Séparation des données :

- ensemble d'entraînement
- ensemble de validation pour mesurer la généralisation
- ensemble de test (pour éviter le biais statistique)

→ Split 60/20/20 habituel.

Apprentissage de modèles

Idéalement : Cross Validation Pour « perdre » moins de données et mieux tester la généralisation, cross-validation :



Ici, 4-fold cross-validation.

Apprentissage de modèles

1 - initialisation aléatoire du modèle

2 - Tant que(critère arrêt == 0)

- Selection aléatoire d'un **batch** de données
- **Forward** : Passe avant du **batch** dans le modèle
- Calcul de l'erreur par rapport aux sorties attendues
- **Backward** : Rétropropagation du gradient de l'erreur en fonction des paramètres dans le modèle (mise à jour du modèle)
- Calcul critère arrêt

3 - Calcul de l'erreur sur un échantillon de données **qui n'ont JAMAIS été vues par le modèle pendant l'apprentissage !**

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon
 - recommandations : $y_i = \text{liste d'items } x_{k \neq i}$

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon
 - recommandations : $y_i = \text{liste d'items } x_{k \neq i}$
 - réduction de dimensionnalité : $y_i = x_i$ projeté dans moins de features

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon
 - recommandations : $y_i = \text{liste d'items } x_{k \neq i}$
 - réduction de dimensionnalité : $y_i = x_i$ projeté dans moins de features
- on définit quand même une perte (loss)

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon
 - recommandations : $y_i = \text{liste d'items } x_{k \neq i}$
 - réduction de dimensionnalité : $y_i = x_i$ projeté dans moins de features
- on définit quand même une perte (loss)

Densité intra- et inter-clusters en clustering

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon
 - recommandations : $y_i = \text{liste d'items } x_{k \neq i}$
 - réduction de dimensionnalité : $y_i = x_i$ projeté dans moins de features
- on définit quand même une perte (loss)

Densité intra- et inter-clusters en clustering

Ou utiliser des données supervisées...

Étant donné des exemples x_i , trouver un modèle h :

- but moins défini qu'en supervisé :
 - clustering : $h(x_i) = \hat{y}_i = \text{cluster de } x_i$
 - détection d'anomalies : $y_i = 1$ si anomalie, 0 sinon
 - recommandations : $y_i = \text{liste d'items } x_{k \neq i}$
 - réduction de dimensionnalité : $y_i = x_i$ projeté dans moins de features
- on définit quand même une perte (loss)

Densité intra- et inter-clusters en clustering

Ou utiliser des données supervisées... Pourcentage d'info perdue pour la réduction de dimensionnalité

- extrêmement importante pour compléter les analyses après les retours business
- ensemble de bonnes pratiques

- garder une trace exacte du preprocessing

- garder une trace exacte du preprocessing
- de préférence utiliser des notebooks

- garder une trace exacte du preprocessing
- de préférence utiliser des notebooks
- faire attention au random (utiliser des seeds)

- garder une trace exacte du preprocessing
- de préférence utiliser des notebooks
- faire attention au random (utiliser des seeds)
- définir les datasets utilisés, dates comprises

- garder une trace exacte du preprocessing
- de préférence utiliser des notebooks
- faire attention au random (utiliser des seeds)
- définir les datasets utilisés, dates comprises
- garder une trace de l'environnement

Méta-paramètres et Régularisation

Les méta-paramètres forment l'ensemble des prétraitements, la forme et les contraintes appliquées au modèle **AVANT** son apprentissage.

- Forme : Nombre de couches?, de quelle taille? ...
- L'algorithme d'optimisation (SGD, adaboost, adam,...)
- Méthodes de régularisation (norme des paramètres dans la loss, bruitage, dropout, ...)

Régularisation
 \approx
empêcher le surapprentissage

