Concepts fondamentaux de l'apprentissage automatique

Jian Tang

HEC Montréal

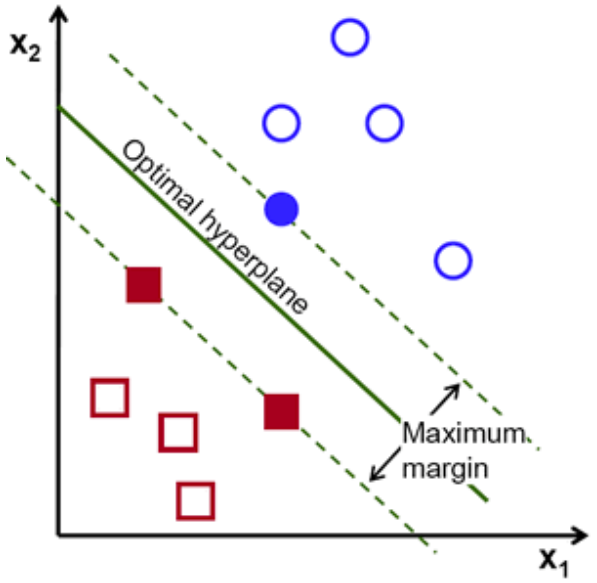
Institut IA Mila-Québec

Courriel: jian.tang@hec.ca

Apprentissage automatique

- “L'apprentissage automatique est un **champ d'étude de l'intelligence artificielle** qui se fonde sur des approches mathématiques et statistiques pour donner aux **ordinateurs** la capacité d'« apprendre » à partir de **données**, c'est-à-dire d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune.”

-Wikipedia



Machines à vecteurs de support (SVM)

Extraction Artisanale de caractéristiques

Classification Simple
e.g., SVM, LR

Algorithme d'apprentissage

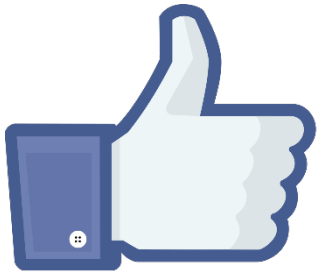
- Un algorithme d'apprentissage automatique est un algorithme capable d'apprendre à partir de données (ou d'expérience)

- Apprentissage: "A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P** if its performance at tasks in **T**, as measured by **P**, improves with experience **E**." - Mitchell (1997)

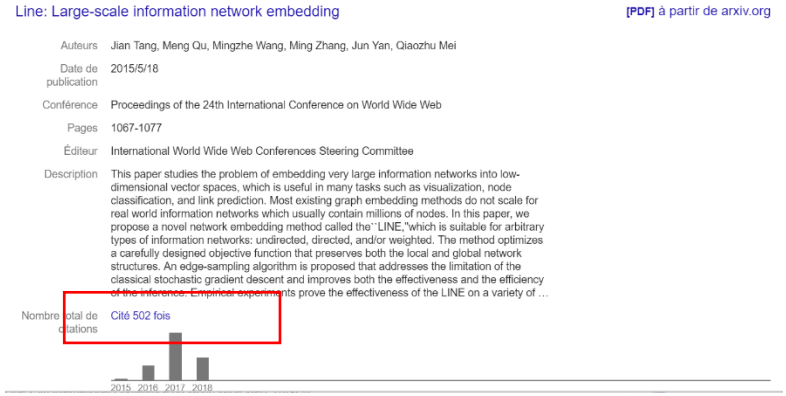
- Expérience **E**: un ensemble d'exemples.
  Chaque exemple *x* est représentée comme un vecteur d'**attributs** de grande dimension $x \in R^n$
  - Un exemple pourrait être une image représentée par les pixels à l'intérieur de celle-ci

Tâche: Régression

- Assigner un vecteur d'attributs à une valeur continue $f: x \in R^d \rightarrow R$
- Le but est de prédire correctement la valeur cible



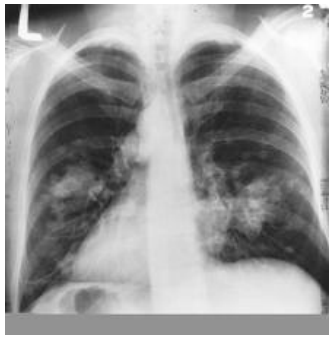
X: (attributs de l'utilisateur, attributs du message)
Y: nombre de j'aime (like)



X: (attributs de l'auteur, attributs de l'article)
Y: nombre de citations

Tâche: Classification

- Assigner un vecteur de valeurs réelles x à K classes distinctes $C = \{C_k\}_{k=1,\ldots,K}, i.e., f_\theta : x \in R^d \rightarrow C$



X: un ensemble d'intensité de pixels
Y: cancer présent/cancer absent

Most Helpful Customer Reviews

56 of 63 people found the following review helpful

★★★★☆ **Can A Reference Book Be Too Thorough?**

By B.L. on January 9, 2011

Format: Paperback

Programming Python is a book designed to take people who know Python and guide them on how to actually make it do things in the real world. It's important to note that the material in here (In the December 2010 4th edition) is for 3.X versions of Python and only deals with 2.X to the extent that the versions overlap, so you'll be better off with an earlier edition of the book (or another book designed to deal thoroughly with both versions) if you're working on a project that needs to work uusing earlier versions of Python.

X: critique d'utilisateur
Y: sentiment (positif/neutre/négatif)

Tâche : Estimation de densité

- Assigner un vecteur d'attributs x à une valeur $p_{model}: x \in R^d \rightarrow R$, où $p_{model}(x)$ est une fonction de densité
  - Génération de données

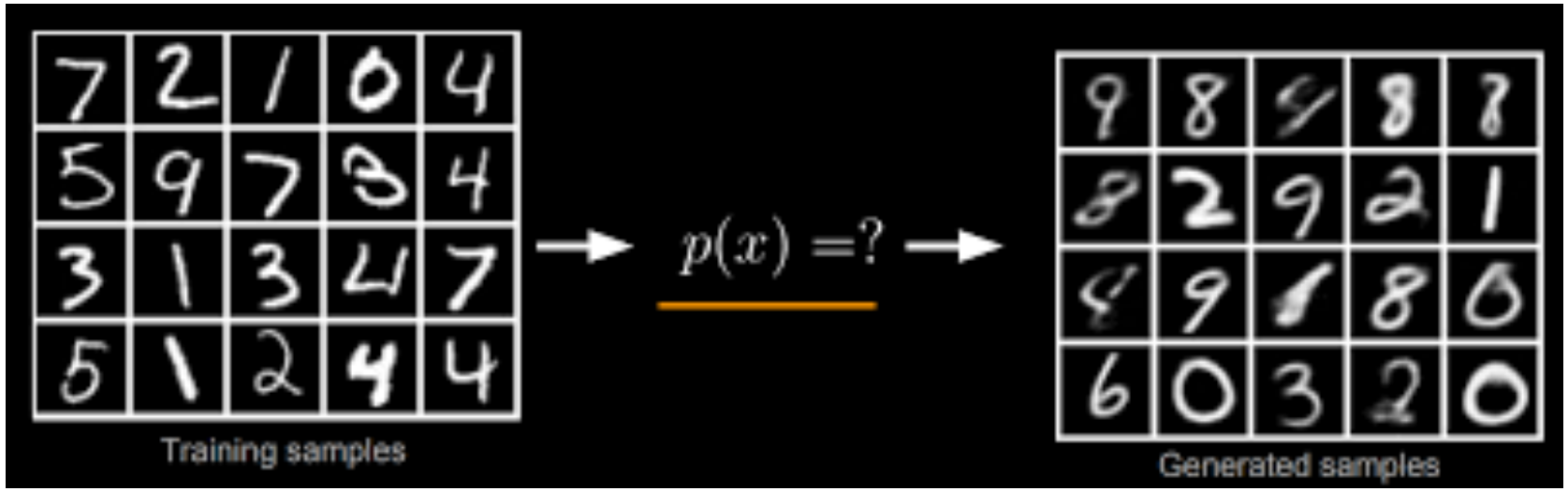


Image from Internet

Mesure de performance: P

- Une mesure quantitative P doit être conçue pour évaluer l’efficacité d'un algorithme d'apprentissage automatique
  - Spécifique à la tâche
- Ex. classification: précision (accuracy)
  - Le pourcentage d'exemples classifiés correctement
- La performance est généralement évaluée sur un jeu de données jamais observé (jeu de données test).

Expérience: E

- Algorithme d'apprentissage automatique:
  - Supervisé
  - Non supervisé
- Supervisé: chaque exemple est associé à une classe/valeur cible
  - Ex. classification ou régression
- Non supervisé: aucune classe/valeur cible n'est fournie
  - Ex. estimation de densité, partitionnement de données « clustering », réduction de dimension
- Pas couvert durant la session: apprentissage par renforcement
  - L'expérience n'est pas fixe mais elle est générée dynamiquement en interagissant avec l'environnement

Exemple: Régression Logistique (K = 2)

- Pour une classification binaire, la probabilité à posteriori de la classe $\mathcal{C}_1$ peut être écrite comme une fonction sigmoïdale

$$\mathrm{p}(\mathcal{C}_1|\boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{x}^T\boldsymbol{w}-b)} = \sigma(\boldsymbol{x}^T\boldsymbol{w}+b)$$

- où $\boldsymbol{w}$ sont les poids des attributs et $\boldsymbol{b}$ le terme de biais. La probabilité de l'autre class se définit comme suit:

$$p(\mathcal{C}_2|\mathbf{x}) = 1 - p(\mathcal{C}_1|\mathbf{x}),$$

Maximum de vraisemblance pour la Régression Logistique

- Nous avons observé un jeu de données d'entrainement

$$\{\mathbf{x}_n, t_n\},\ n = 1, .., N;\ t_n \in \{0, 1\}.$$

- En maximisant la probabilité d'obtenir la bonne classe, on obtient la fonction de vraisemblance suivante:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^{N} \left[y_n^{t_n} (1 - y_n)^{1-t_n} \right], \qquad y_n = \sigma(\boldsymbol{x_n^T w})$$

La fonction d'entropie croisée

- En prenant le valeur négative de la log vraisemblance, on peut définir la fonction d'erreur d'entropie croisée (que l'on veut minimiser):

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{X},\mathbf{w}) = -\sum_{n=1}^{N}\left[t_n \ln y_n + (1-t_n)\ln(1-y_n)\right] = \sum_{n=1}^{N} E_n.$$

- Ici, $E_n = -t_n \log y_n - (1-t_n)\log(1-y_n)$ est l'entropie croisée entre les deux distributions binaires $\mathrm{P}_{\mathrm{data}} = (t_n, 1-t_n)$ et $\mathrm{P}_{\mathrm{model}} = (y_n, 1-y_n)$

Multi-classes (K > 2) avec fonction softmax

- Définir une fonction linéaire pour chaque classe:

Class 1: $\mathbf{x}^{\mathbf{T}}\mathbf{w_1}$

Class 2: $\mathbf{x}^{\mathbf{T}}\mathbf{w_2}$

....

Class K: $\mathbf{x}^{\mathbf{T}}\mathbf{w_K}$

- Normaliser les scores avec une fonction softmax

$$P(\mathcal{C}_k|\,\mathbf{x}) = \frac{\exp(\mathbf{x}^{\mathbf{T}}\mathbf{w_k})}{\sum_{i=1}^{K}\exp(\mathbf{x}^{\mathbf{T}}\mathbf{w_i})}$$

- Afin de définir la probabilité d'appartenance à la classe $\mathcal{C}_k$

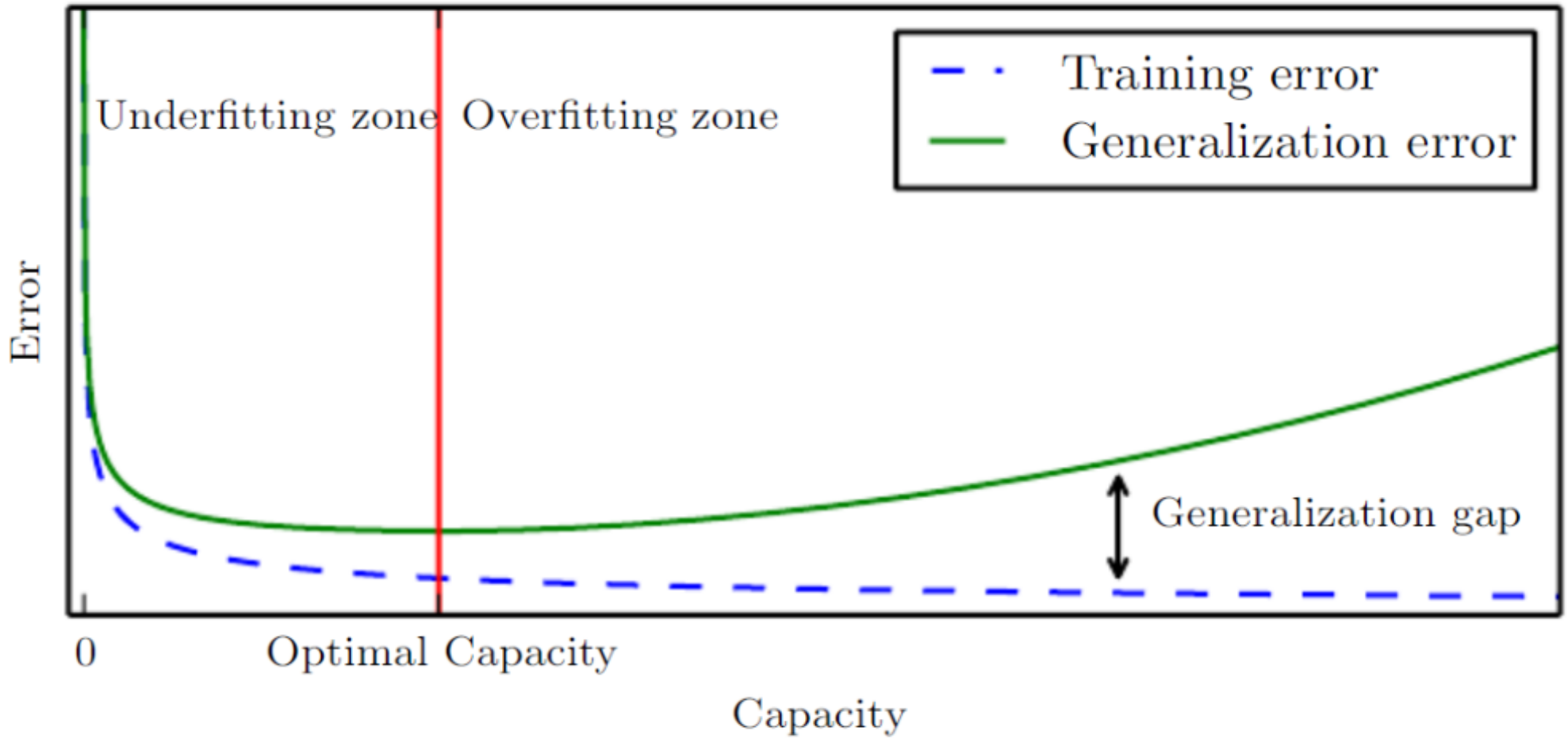
Capacité d'un modèle
Sous-Entrainement et Surentrainement

- Le but d'un modèle d'apprentissage automatique est de maximiser son pouvoir de **généralisation**
  - Bien prédire pour des donnés jamais observées
- Données d'entrainement => erreur d'entrainement
- Données test => erreur test (erreur de généralisation)
- Pour une régression linéaire:
  - Entrainer le modèle en minimisant l'erreur d'entrainement
  - Évaluer la performance du modèle selon l'erreur test

Capacité d'un modèle
Sous-Entrainement et Surentrainement

- Capacité d'un modèle: la capacité à s'ajuster à une variété de fonctions
  - Les modèles avec plus de paramètres ont d'habitude une plus grande capacité.
- Sous-Entrainement: Le modèle n'est pas capable d'obtenir une erreur suffisamment basse sur le jeu de données d'entrainement
- Surentrainement : Le modèle performe bien sur les données d'entrainement mais pas sur les données test

Erreur d'un modèle en fonction de sa capacité



Régularisation

- Technique permettant d'éviter le surentrainement
  - Exprimant certaines préférences pour différentes fonctions
- Régression logistique régularisée

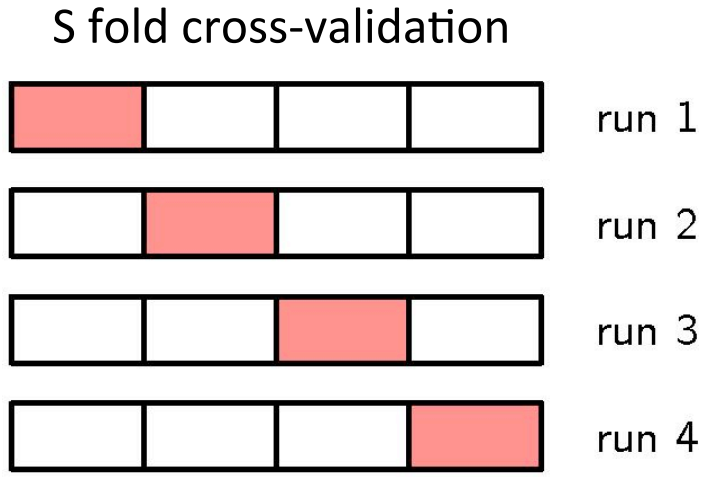
$$O = -\log p(\mathbf{T}|\mathbf{X}, \mathbf{w}) + \lambda \left|\left|\boldsymbol{w}\right|\right|_2^2$$

- Ceci est aussi connu comme la régularisation L2 ou weight decay

Validation croisée

- Diviser le jeu de données en 3
  - Entrainement: utilisé pour apprendre les paramètres du modèle
  - Validation: utilisé pour sélectionner un modèle et des hyperparamètres (Ex. régularisation)
  - Test: utilisé pour évaluer la performance du modèle

- Validation croisée à *S* blocs
  - Utilise le plus de données possibles

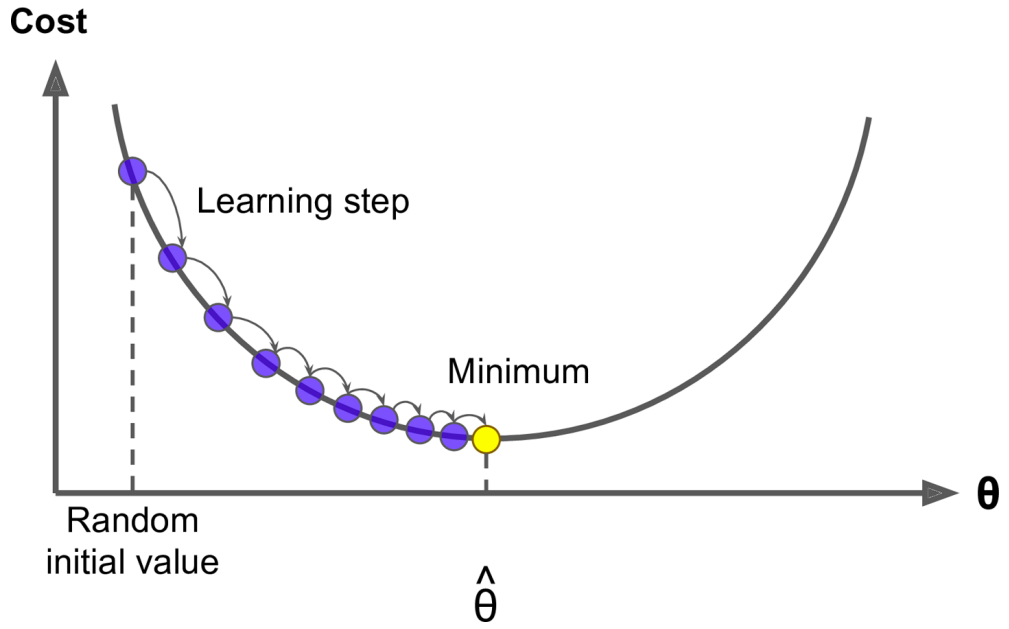


Algorithme du gradient

- L‘algorithme du gradient est un algorithme itératif d’optimisation qui permet de trouver le minimum d‘une fonction (Ex. la log vraisemblance négative)
- Pour une fonction F(x) à un point **a**, F(x) *décroit plus rapidement* si l'on va dans la direction du gradient négatif de **a**

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \gamma \nabla F(\mathbf{a}_n)$$

Quand le gradient devient zero, nous avons atteint un minimum local



Algorithme du gradient stochastique

- On veut minimiser la fonction d'erreur d'entropie croisée (cross-entropy) par rapport au paramètre w :

$$\nabla_w E(w) = \frac{1}{n}\sum_{i=1}^{n} \nabla_w E_i$$

- *n* peut devenir très grand, ce qui est coûteux en temps de calcul
- L'algorithme du gradient stochastique fait une approximation du gradient à l'aide d'échantillon aléatoire

Un échantillon : $\nabla_w E(w) \approx \nabla_w E_i$

Un lot (“Batch”) d'échantillon: $\nabla_w E(w) \approx \frac{1}{B}\sum_{i=1}^{B} \nabla_w E_i$

Lecture

- Livre: Deep Learning
  - Chap. 2, 3, 5

À faire pour le prochain cours

- Enregistrer votre équipe d'étudiants pour la présentation et le projet de groupe.
  - Votre équipe devrait être la même pour les deux activités