Apprentissage Supervisé

Boi Faltings

Laboratoire d'Intelligence Artificielle
boi.faltings@epfl.ch
http://moodle.epfl.ch/

Rappel: inférences logiques

Considérez:

a) `oiseau(Tweety)`
b) `vole(Tweety)`
c) $(\forall$ `x)` `oiseau(x)` $\Rightarrow$ `vole(x)`

3 Types d'inférences:

- **déduction**: a), c) $\rightarrow$ b)
- **abduction**: b), c) $\rightarrow$ a)
- **induction**: a), b) $\rightarrow$ c)

Induction = Apprentissage à partir d'exemples

Expériences $\Rightarrow$ nouvelles connaissances

- Exemple: Quelques expériences de piments:

`grand,allongé,rouge`	`piquant`
`grand,rond,vert`	$\neg$`piquant`
`petit,allongé,jaune`	$\neg$`piquant`
`petit,allongé,rouge`	`piquant`
`petit,rond,rouge`	$\neg$`piquant`

- $\Rightarrow$ *induction* d'un *modèle*, p.ex. une règle:

`allongé,rouge` $\Rightarrow$ `piquant`

La validité de l'induction

- Hypothèse d'un monde clos:

  Tous les exemples possibles ont été considérés.

- $\Rightarrow$ le plus d'exemples considérés, le moins probable que l'hypothèse soit fausse. ("Big Data")
- Théorie de l'apprentissage (p.ex. PAC) quantifie le nombre d'exemples nécessaires pour garantir la qualité.
- Données = troisième type de ressource (comme temps de calcul et mémoire).

Types d'apprentissage

- Supervisé: trouver un modèle à partir d'exemples classifiés.
  Exemple: apprendre des règles pour reconnaître les piments piquants, les spam, les mauvais payeurs, etc.
- Non-supervisé: trouver un modèle en structurant des exemples non-classifiés.
  Exemple: grouper les clients d'un site web pour optimiser la structure d'accès.

Représentation du problème

Les exemples doivent être representés par des attributs
Problème de la sélection des attributs: l'utilité des attributs peut parfois se révéler uniquement dans leur combinaison.
Exemple:

- $x, y, z \in [-1..1]$, `piquant` si $x * y * z > 0$.
- x, y et z seuls n'ont aucune corrélation avec `piquant`.
- seule la combinaison compte!
- $\Rightarrow$ recherche combinatoire!

Grand problème de l'apprentissage.

Buts de l'apprentissage

- Classification:
  appartenance à une classe: piquant, spam, etc.
- Régression:
  valeur d'un attribut, p. ex. un prix, une couleur.
- Renforcement:
  amélioration incrémentale en renforcant les succès et supprimant les échecs.
- Autres structures de données, par exemple des recommandations de produits.

Dans ce cours: Classification et Régression

Classification supervisé

Etant donnés:

un ensemble $\mathcal{P}$ d'instances d'un concept

un ensemble $\mathcal{N}$ de non-instances du concept

trouver

un modèle du concept qui couvre toutes les instances de $\mathcal{P}$ et aucune instance de $\mathcal{N}$.

Exemples de concepts: piments piquants, mauvais payeurs, etc.

Types de modèles

Modèles simples (paramétrique):

fonction paramétrisée des attributs

Exemples:

- conjonction d'attributs: toute instance ayant la combinaison d'attributs est membre du concept.
- frontière de décision: toute instance pour laquelle $f(x) > 0$ est membre du concept.
- approximation par régression: classification $= f(x)$

Types de modèles (2)

Modèles structurés (non-paramétrique):

décomposition de l'ensemble d'exemples et apprentissage d'une classification paramétrique pour chacun des sous-ensembles.

Examples:

- arbres de décision: décomposition jusqu'à ce que les classes soient très homogènes.
- boosting: combinaison pondérée de classifications.

Techniques

taille	forme	couleur	
grand (1)	allongé (1)	rouge (1)	piquant
grand (1)	rond (0)	vert (2)	¬piquant
petit (0)	allongé (1)	rouge (1)	piquant
petit (0)	rond (0)	rouge (1)	¬piquant
petit (0)	allongé (1)	jaune (0)	¬piquant

- Logique: allongé $\wedge$ rouge $\Rightarrow$ piquant
- Frontière: couleur + 2 * forme $\geq$ 3
- Instance: chercher un exemple de la même forme et couleur.

Modèle logique

1. `grand,allongé,rouge,piquant`
2. `grand,rond,vert,`$\neg$`piquant`
3. `petit,allongé,rouge,piquant`
4. `petit,rond,rouge,`$\neg$`piquant`

Attributs partagés par tous les exemples positifs:

$$\mathcal{D}_0 = \{\texttt{allongé},\ \texttt{rouge}\}$$

Tout modèle conjonctif doit être un sous-ensemble de $\mathcal{D}_0$.

Spécialisation et généralisation

Deux types de recherche:

- spécialisation: commencer avec une description ayant peu d'attributs et *spécialiser* en rajoutant des attributs pour n'inclure que les exemples positives.
  Exemple: $\{\texttt{rouge}\} \Rightarrow \{\texttt{rouge, allongé}\}$.
- généralisation: commencer avec une description avec beaucoup d'attributs et *généraliser* en laissant tomber des attributs en veillant à ne pas inclure des exemples negatifs.
  Exemple: $\mathcal{D}_0 = \{\texttt{allongé, rouge}\} \Rightarrow \{\texttt{allongé}\}$

Concepts disjonctifs

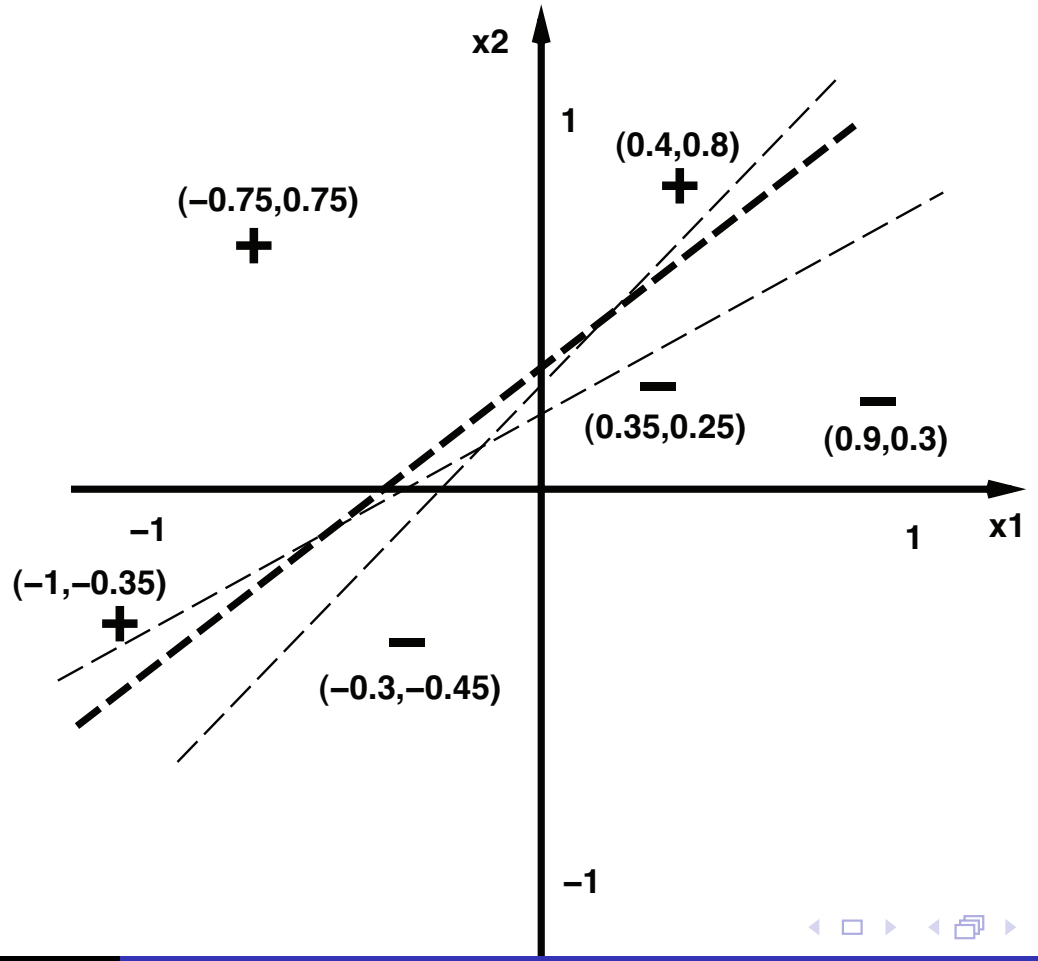
`grand,allongé,rouge,piquant`
`petit,allongé,rouge,piquant`
`petit,rond,vert,piquant`
`grand,rond,vert,¬piquant`
`petit,allongé,jaune,¬piquant`
`petit,rond,rouge,¬piquant`

$\mathcal{D}_0$ est vide $\Rightarrow$ ne peut pas être représenté par une seule conjonction, même avec généralisation.
$\Rightarrow$ il faut un modèle structuré.

Attributs numériques

Pas possible d'énumérer les modèles

$\Rightarrow$ modèles simples = frontières de décision



Frontières de décision linéaires

- Exemples représentés par des vecteurs d'attributs continus $X = (x_1, ..., x_n)$
- On rajoute $x_0 = 1$ pour pouvoir s'éloigner de l'origine.
- Frontière = fonction linéaire:

$$\delta(W, X) = w_0 \cdot x_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + ...w_n \cdot x_n$$

- Modèle paramétrisé par vecteur W des poids
- Décision = fonction de seuil:
  - $C(X) = 1$ (+) si $\delta(W, X) \geq 0$, et
  - $C(X) = 0$ (-) autrement.

Perceptron

- Perceptron = modèle simplifié d'un neuron.
- Paramètres: vecteur de poids $W = (w_0, .., w_n)$.
- Entrée: vecteur $X = x_1, .., x_n$ (+ constante x_0).
- applique la frontière linéaire $\delta(W, X)$.
- Sortie = 1 si $\delta \geq 0$, 0 autrement.

Apprentissage de frontières de décision

Règle du *perceptron*:

Function PERCEPTRON($\mathcal{P}, \mathcal{N}$)
W $\leftarrow$ (1,0,0,...,0)
repeat
  $X_i \leftarrow$ choix alléatoire $\in \mathcal{P} \cup \mathcal{N}$
  if $C(W, X_i) \neq class(X_i)$ **then**
    if $class(X_i) = +$ **then**
      W $\leftarrow$ W $+ \sigma \cdot X_i$
    else
      W $\leftarrow$ W - $\sigma \cdot X_i$
until W ne change plus
return W

σ = coefficient d'apprentissage (par exemple =1)

Analogie aux neurones $\Rightarrow$ réseaux de neurones artificiels.

Descente de gradient stochastique

L'algorithme du perceptron est une *descente de gradient*:

- classification erronée: besoin de corriger le vecteur W.
- gradient $\frac{\delta(W \cdot X)}{\delta W} \propto X$.
- longueur de pas $= \sigma$.
- stochastique: sélection aléatoire de l'exemple.

$\Rightarrow$ en général, bonne convergence.

Classe d'algorithmes beaucoup utilisée pour l'optimisation de modèles induits!

Exemple (Perceptron)

$\sigma = 1$

Exemple	W	Val.	Class.	W_{modif}
(1,0.4,0.8) +	(1,0,0)	1	+	(1,0,0)
(1,0.35,0.25) -	(1,0,0)	1	+	(0,-0.35,-0.25)
(1,0.9,0.3) -	(0,-0.35,-0.25)	-0.39	-	(0,-0.35,-0.25)
(1,-0.75,0.75) +	(0,-0.35,-0.25)	0.075	+	(0,-0.35,-0.25)
(1,-1,-0.35) +	(0,-0.35,-0.25)	0.4375	+	(0,-0.35,-0.25)
(1,-0.3,-0.45) -	(0,-0.35,-0.25)	0.2175	+	(-1,-0.05,0.2)
(1,-1,-0.35) +	(-1,-0.05,0.2)	-1.02	-	(0,-1.05,-0.15)
(1,0.4,0.8) +	(0,-1.05,-0.15)	-0.54	-	(1,-0.65,0.65)
....				

Un exemple doit parfois passer plusieurs fois.

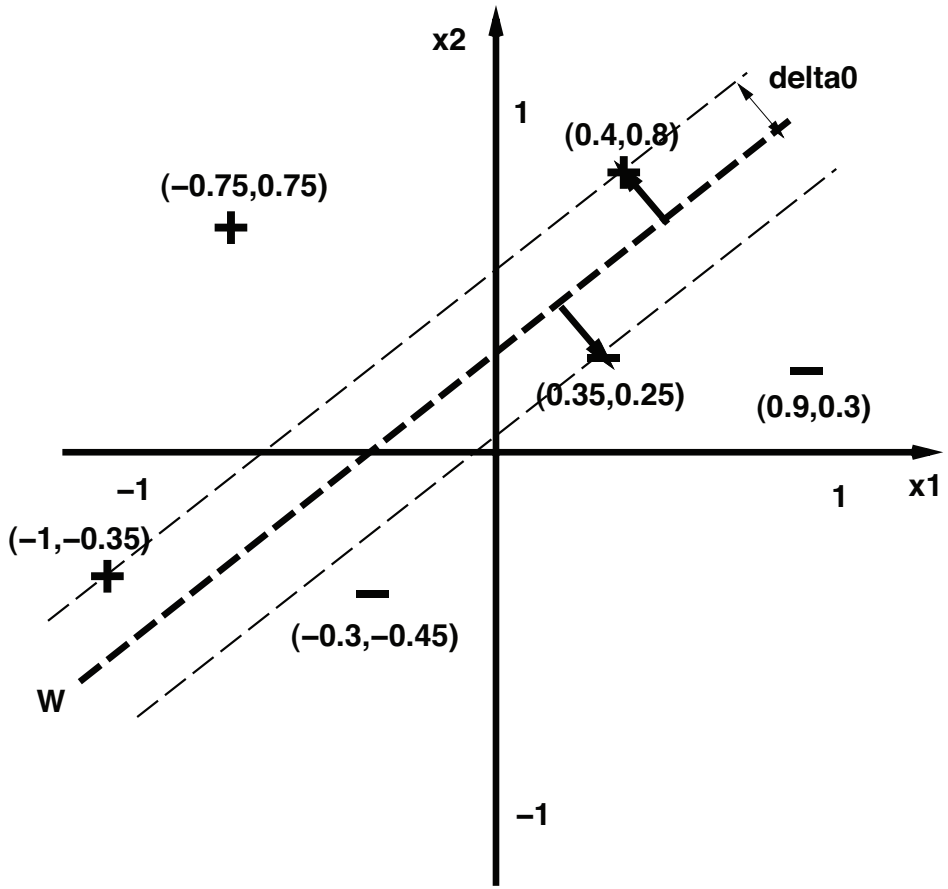
La qualité du résultat n'est pas garantie!

Support Vector Machines (SVM)

maximiser la séparation des exemples

$\Rightarrow$ choisir W tel que:

- toutes les instances positives: distance $\delta > \delta_0$
- toutes les instances négatives: distance $\delta < -\delta_0$
- δ_0 est maximal



Trouver la séparation optimale

- Normaliser w pour que $X \cdot W = 1$ pour un point à distance δ_0.
  $\Rightarrow$ problème d'optimisation quadratique:
  - objectif: maximiser δ_0 = minimiser $|W|^2$
  - contraintes:
    pour tout exemple positif, $X \cdot W > 1$
    pour tout exemple négatif, $X \cdot W < -1$
- n exemples: résolvable en $O(n^3)$ temps et $O(n^2)$ mémoire.
- Dans la solution, certaines instances se trouveront à une distance δ_0 de la frontière de décision $\Rightarrow$ support vectors.

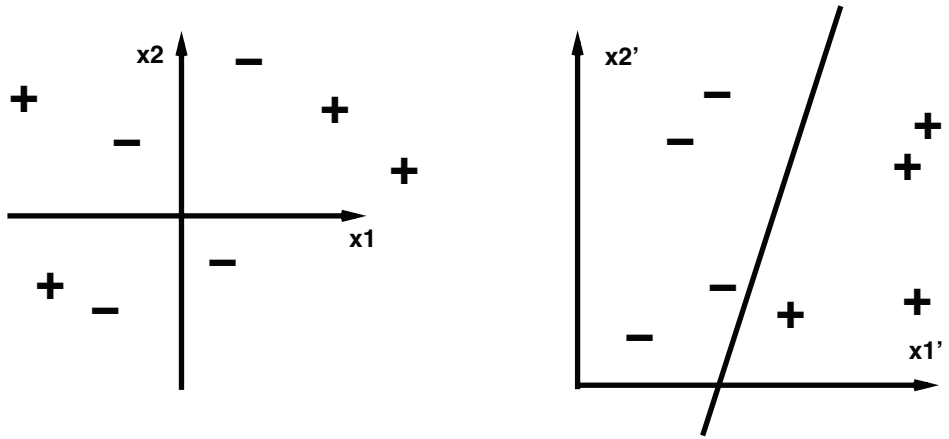
Instances non-séparables

Lorsque les exemples ne sont pas séparables, on peut

- admettre que certains exemples ne sont pas correctement classifiés. On peut déduire la somme des erreurs de δ_0 lors de la maximisation de la surface et ainsi optimiser une combinaison des critères.
- introduire une transformation non-linéare qui les rend séparables (kernel function).

Transformation non-linéaire

S'il n'y a pas de frontière linéaire, on peut transformer chaque coordonnée par une transformation $\phi(x)$ et ainsi transformer les instances dans un autre espace.
Exemple: $x'_1 = x_1^2, x'_2 = x_2^2$



rend les exemples séparables.

Expression dans une SVM

L'optimisation n'utilise que les produits pointés de vecteurs:

$$min\phi(W) \cdot \phi(W)$$

avec contraintes

$$\phi(X_i) \cdot \phi(W) > 1/ < -1$$

Remplacer $\phi(X) \cdot \phi(Y)$ par une fonction Kernel $K(X, Y)$.
Exemple:

$$\begin{aligned} K(X,Y) &= (X \cdot Y)^2 \Rightarrow \phi = (x_1^2, \sqrt{(2)}x_1x_2, x_2^2) \\ K(X,Y) &= x_1^2y_1^2 + 2x_1y_1x_2y_2 + x_2^2y_2^2 = \phi(X) \cdot \phi(Y) \end{aligned}$$

$\Rightarrow minK(W, W)$, contraintes $K(X_i, W) > 1/ < -1$
L'optimisation ne construit jamais ϕ, juste $K(X, Y)$.

Autres fonctions Kernel

- "Radial basis function":

$$K(X, Y) = e^{-(X-Y)^2/2\sigma^2}$$

$\Rightarrow \underline{\phi}$ serait d'une dimensionalité infinie (et heureusement pas construite explicitement!)

- En général, les fonctions Kernel sont des fonctions de similarité.
- Pas toutes les fonctions Kernel sont admissibles.

Régression

- Pour la prévision d'une valeur inconnue y, il faut un modèle $y = f(X)$, X = vecteur de traîts.
- Le modèle est optimisé sur les exemples connus, de façon à ce qu'on obtienne une bonne approximation.
- On espère ainsi que la prévision sera bonne sur les nouvelles données également.
- Notation: $X(i)$ = i-ème exemple, $\underline{X} = \{X(1), X(2), ...\}$ = ensemble des exemples connus.

Régression linéaire

- Approximation linéaire d'une fonction y = f(X)

$$y = w_0 + w_1 x_1 + ... + w_k x_k + \mathcal{L}$$

- $\mathcal{L}$ = l'erreur d'approximation
- Intuition: choisir W pour minimiser $\mathcal{L}$ sur les exemples.
- Formellement: choisir le W le plus probable etant données l'ensemble des observations $\underline{X}$: maximiser

$$p(W|\underline{X}) = \frac{p(\underline{X}|W)p(W)}{p(\underline{X})}$$

- Si toutes les W ont la même probabilité:
  = Maximum likelihood: choisir W qui maximise probabilité des observations $p(\underline{X}|W)$.

Maximiser $p(\underline{X}|W)$

- Pour chaque exemple i avec erreur de prédiction $\mathcal{L}(i)$:

$$\mathcal{L}(i) = y(i) - (w_0 + w_1 x_1(i) + ... + w_k x_k(i))$$

  la probabilité $p(i)$ d'observer $y(i)$ est la probabilité de $\mathcal{L}(i)$.
- Erreurs sont distribuées selon une distribution Gaussienne:

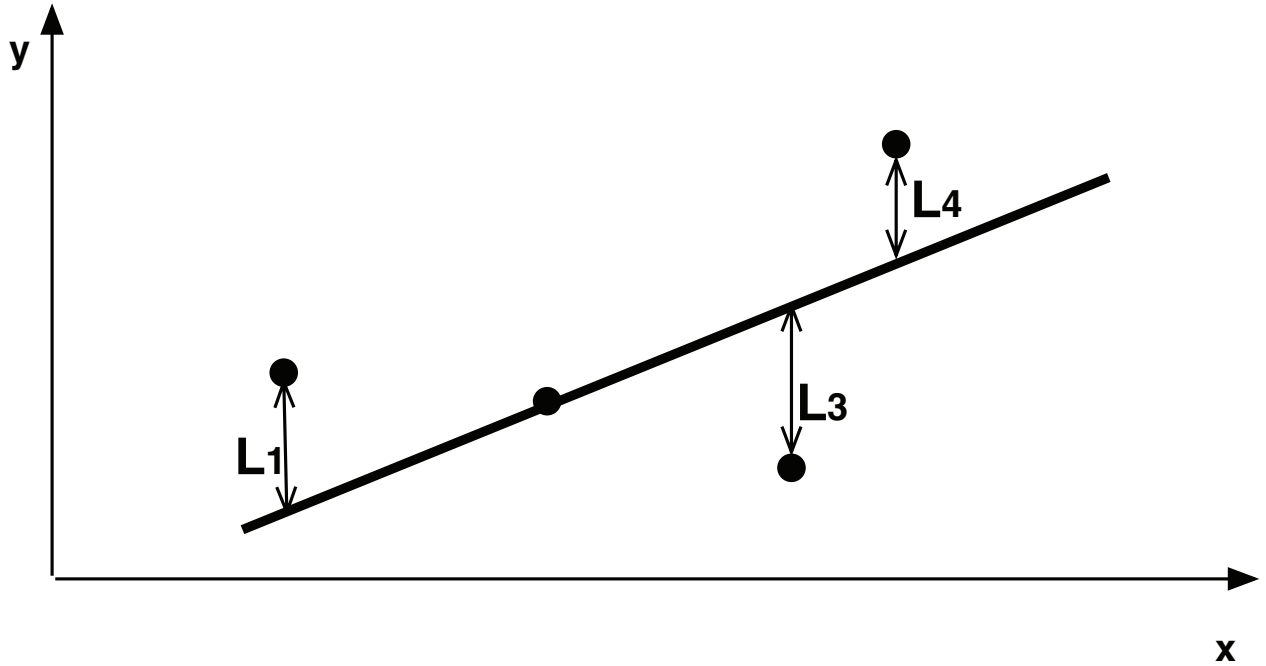
$$p(i) = p(\mathcal{L}(i)) = \alpha e^{-w\mathcal{L}(i)^2}$$

- Pour *maximiser* la probabilité de tous les exemples $\prod_{X(i)\in\underline{X}} p(i)$, on doit *minimiser*

$$\sum_{X(i)\in\underline{X}} -\log p(i) = \sum_i \mathcal{L}(i)^2$$

Moindres carrées

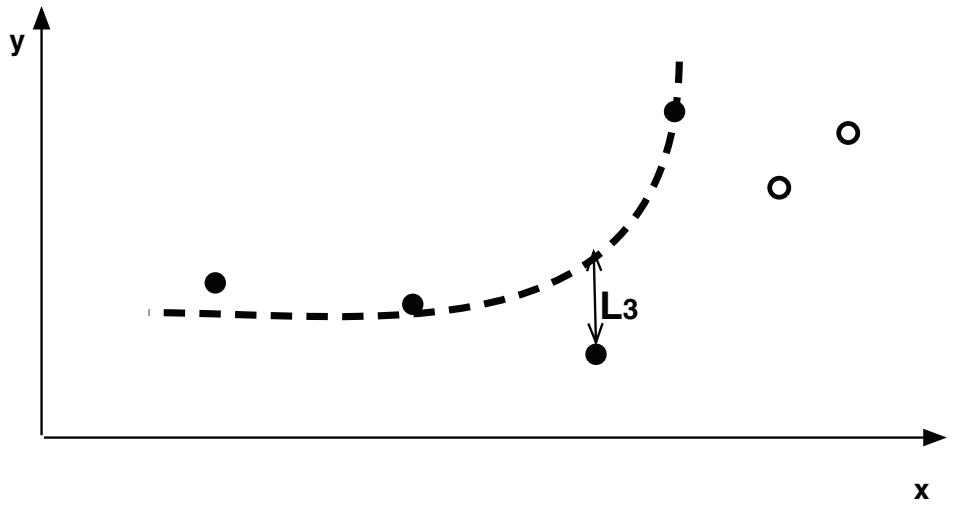
Modèle optimal = approximation des moindres carrés:



$\Rightarrow$ droite qui minimise $\sum_{i=1}^{4} L_i^2$
s'étend à des polynômes:
régression linéaire dans un espace 1, x, x^2, ..., x_n.

Surapprentissage (Overfitting)

Un polynôme non-linéaire permet souvent de mieux approximer les points:



Cependant, il y a un compromis à faire:

- Biais du modèle: capacité du modèle de représenter la réalité.
- Variance de l'apprentissage: erreur du modèle à cause d'une courbe déterminée par les erreurs d'observation.

Régularisation

- Supposons que certains W sont plus probables que d'autres: modèles simples, petits coefficients.

$\Rightarrow$ choisir W pour maximiser non simplement $p(\underline{X}|W)$, mais:

$$p(W|\underline{X}) = p(\underline{X}|W)\frac{p(W)}{p(\underline{X})}$$

$p(\underline{X})$ pas influencé par $W \Rightarrow$ *minimiser*:

$$-\ln p(\underline{X}|W) - \ln p(W)$$

$\ln p(W)$ est un *régularisateur*:

- si tous les W sont équiprobables, il n'a pas d'influence.
- autrement, il impose une distribution des coefficients.
- si les w_i doivent être distribuées selon Gaussienne avec moyenne 0, alors le regularisateur sera $\sum_k -w_k^2$.

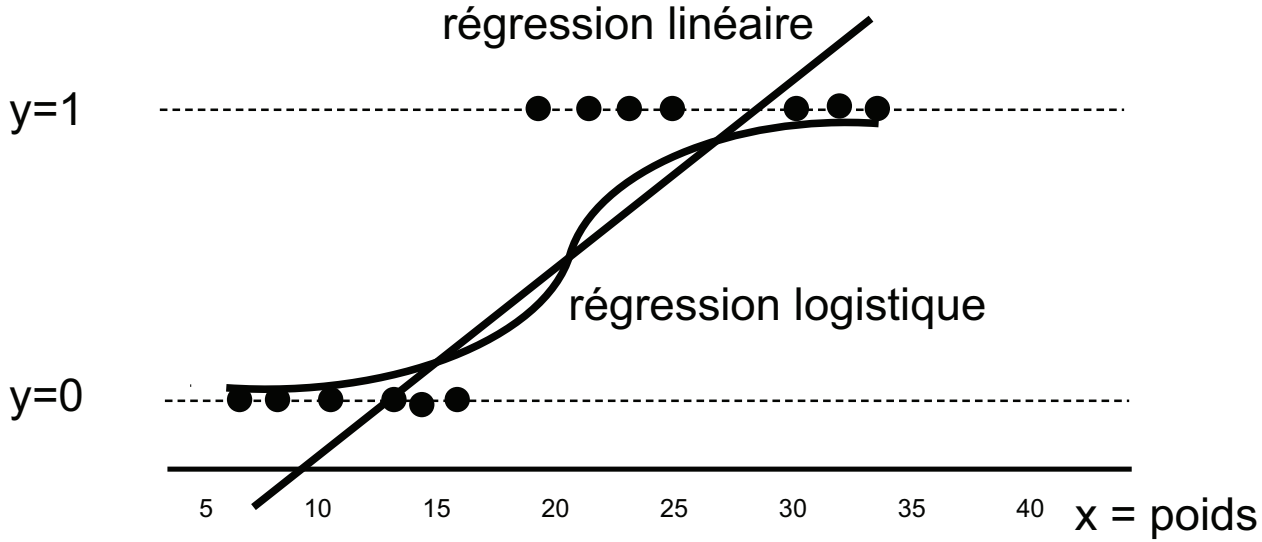
Fonction du régularisateur

- L'optimisation de $\ln p(\underline{X}|W)$ a tendance à produire des coefficients extrêmes.
- Le régularisateur compense cette tendance: les coefficients extrêmes ne sont pas probables.
- Autre régularisateur courant: $\sum_k w_k$ (correspond à une distribution exponentielle).

Classification par Régression

Deux classes c_1 et c_2: définir $y = p(c_1)$

Exemple: distinguer chats et chiens par leur poids:



Faiblesses:

- y peut devenir < 0 ou > 1 et donc ne pas être une probabilité.
- Le problème définit un seul et unique poids pour chaque classe et laisse pas de marge aux variations.

Transformation logistique

A la place de la fonction linéaire, appliquer la transformation:

$$p(y = c_1|X, W) = \frac{e^{W \cdot X}}{1 + e^{W \cdot X}} = \frac{e^{w_0 + w_1 x_1 + ... + w_k x_k}}{1 + e^{w_0 + w_1 x_1 + ... + w_k x_k}}$$

- toujours entre 0 et 1.
- courbe sigmoïde qui admet une marge de valeurs de x pour une même classe.
- appelée souvent *softmax*

Régression logistique

$$p(y = c_1|X, W) = \frac{e^{W \cdot X}}{1 + e^{W \cdot X}} = \frac{1}{1 + e^{-W \cdot X}}$$

$$p(y = c_2|X, W) = 1 - \frac{e^{W \cdot X}}{1 + e^{W \cdot X}} = \frac{1}{1 + e^{W \cdot X}}$$

et donc (si $c_1 = 1$ et $c_2 = -1$):

$$p(y = y_i|X, W) = \frac{1}{1 + e^{-y(i)W \cdot X(i)}}$$

$\Rightarrow$ choisir W pour maximiser:

$$p(\underline{Y}|\underline{X}, W) = \sum_i - \ln(1 + e^{-y(i)W \cdot X(i)})$$

Application: filtre spam

- Les messages électroniques non-solicités (spam) sont un problème majeur.
- SpamAssasin, un logiciel gratuit, est le noyau de nombreux outils anti-spam.
- Utilise différents critères prédéterminés et une méthode de frontière de décision pour le combiner.
- Utilise l'algorithme du perceptron car l'algorithme du SVM est trop lourd pour tourner comme partie du client courriel.
- voir `http://spamassassin.apache.org/`.

Résumé

Apprentissage supervisé à partir d'exemples:

- Modèles de classification logiques.
- Frontières de décision:
  - règle du perceptron
  - support vector machine
- Régression linéaire et logistique.
- Classification par régression logistique.