Méthodes numériques pour le traitement de la langue naturelle

Benoit Favre

Aix-Marseille Université

Master Info

Classifieurs / Régression

- Prédire un label à partir de données représentés sous forme de traits (*features*)
- ou prédire des valeurs

Introduction

Les entrées

- Features : caractéristiques $\boldsymbol{x} \in \mathcal{X}$ d'un individu
- Classe : une catégorie $y \in \mathcal{Y}$
- Exemple : un couple $(y^\star, \boldsymbol{x})$

Classifieur

- Trouver une fonction $f(y, \boldsymbol{x})$ telle que $y^\star = \text{argmax}_y\, f(y, \boldsymbol{x})$
- Cas non séparable : trouver $f()$ qui minimise l'erreur

$$\text{minimiser} \sum_i erreur(\hat{y}_i, y_i^\star) \tag{1}$$

$$\hat{y}_i = \underset{y}{\text{argmax}}\, f(y, \boldsymbol{x}_i) \tag{2}$$

Features

Catégories

- Couleur des yeux : bleu, vert, marron, rouge
- Distance : proche, loin, très loin
- étiquette linguistique

Valeurs binaires (présence, seuil...)

- Contient ou pas le mot “sport” ?
- Traits morphologiques : se termine par “ant” ?
- Poids $>$ 70kg

Valeurs numériques

- Taille : 170cm, 180cm, 230cm
- Température extérieure : -5C, 25C, 37.5C
- Distance entre “l’OM” et “gagner” : 3 mots
- représentation vectorielle des mots

Problèmes de classification

- Prédire la météo de demain à Marseille
  Classes : soleil, pluie, vent, gris
  Entrées : Pluviométrie, vent, température, cartes météo des 10 dernières années
  Features : Mesures sur les 5 derniers jours, mesures en moyenne à la même date
- Reconnaisance de chiffres écrits
  Classes : 0-9
  Entrées : images redimensionnées à 100×100 pixels, en 2 couleurs
  Features : le pixel (i, j) est noir ou blanc
- Gestion du risque de crédit
  Classes : acceptation, rejet
  Entrées : données sur le client
  Features : revenus mensuels, montant des mensualités, historique du client, quantité empruntée, catégorie socio-professionnelle

Jeux de données

Entraînement, Développement, Test

- Entraînement : apprentissage des paramètres du modèle
- Développement : sélection de modèle
- Test : calcul des performances

Notion de risque

- Taux d'erreur sur l'entraînement $\rightarrow$ sur le test ?
- Attention au **sur-apprentissage**

K-fold

- Diviser les données en K ensembles
- Faire une rotation sur le test, le reste pour l'entraînement
- Performance = moyenne sur les K ensembles

Arbres de décision

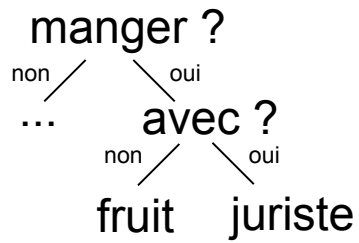
Jeu du devin `http://fr.akinator.com/`

- Poser une série de questions sur les features afin de déterminer la classe
  - Est-ce que c'est un humain ? oui
  - Est-ce que ses yeux sont bleu ? non ...

Apprentissage pour n classes

- Énumérer toutes les questions possibles sur $\boldsymbol{x}$
- Répéter : choisir la question qui minimise un critère
  - Entropie : $H(y) = -\sum P(y) \log_2 P(y)$

je mange avec mon **avocat**
je mange un **avocat**



Classifieur Bayesien Naïf

Problème de classification

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y|\boldsymbol{x}) = \underset{y}{\operatorname{argmax}}\, \frac{P(y)P(\boldsymbol{x}|y)}{P(\boldsymbol{x})} = \underset{y}{\operatorname{argmax}}\, P(y)P(\boldsymbol{x}|y)$$

Mais comme $\boldsymbol{x} = x_1 \dots x_n$:

$$\begin{aligned} P(\boldsymbol{x}|y) &= P(x_1 \dots x_n|y) \\ &= P(x_1|y)P(x_2 \dots x_n|y, x_1) \\ &= P(x_1|y)P(x_2|y, x_1)P(x_3 \dots x_n|y, x_1, x_2) \\ &= P(x_1|y) \dots P(x_n|y, x_1, \dots x_{n-1}) \end{aligned}$$

Hypothèse d'indépendance (le côté naïf) :

$$P(\boldsymbol{x}|y) = \prod_i P(x_i|y)$$

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y) \prod_i P(x_i|y)$$

Bayes Naïf (exemple)

Avocat : **f**ruit ou **j**uriste ? $P(f) = 0.3, \quad P(j) = 0.7$

Feature	$P(x_i = 1\|f)$	$P(x_i = 1\|j)$
manger=m	0.1	0.02
pousser=p	0.08	0.0001
plaider=l	0.0001	0.1
défendre=d	0.001	0.07

On observe : *je mange un avocat* ($x = \{m = 1, p = 0, l = 0, d = 0\}$)

$$\begin{aligned} P(f|x) &= P(f)P(m = 1|f)P(p = 0|f)P(l = 0|f)P(d = 0|f) \\ &= P(f)P(m = 1|f)\,(1 - P(p = 1|f))\,(1 - P(l = 1|f))\,(1 - P(d = 1|f)) \\ &= 0.3 \times 0.1 \times (1 - 0.0001) \times (1 - 0.1) \times (1 - 0.07) \simeq 0.0251 \\ P(j|x) &= 0.7 \times 0.02 \times (1 - 0.08) \times (1 - 0.0001) \times (1 - 0.001) \simeq 0.0128 \end{aligned}$$

On choisi le fruit.

Bayes Naïf (Lissage et valeurs continues)

Que se passe-t-il si $P(x_i|y) = 0$?

- Estimation de probabilité : tout n'a pas été observé
- Lissage : ajouter 1 à tous les comptes

$$P(a) = \frac{cpt(a) + 1}{\sum_i (cpt(i) + 1)}$$

Comment faire pour prendre en compte des valeurs continues ?

- Quantifier l'espace : $P(x_i > valeur)$ (histogrammes)
- Forme de distribution *a priori*
  - Gaussienne
  - Mixture de gaussiennes
  - ...

Perceptron

Idée : regarder les exemples un par un, favoriser la bonne classe et pénaliser une mauvaise prédiction
Début

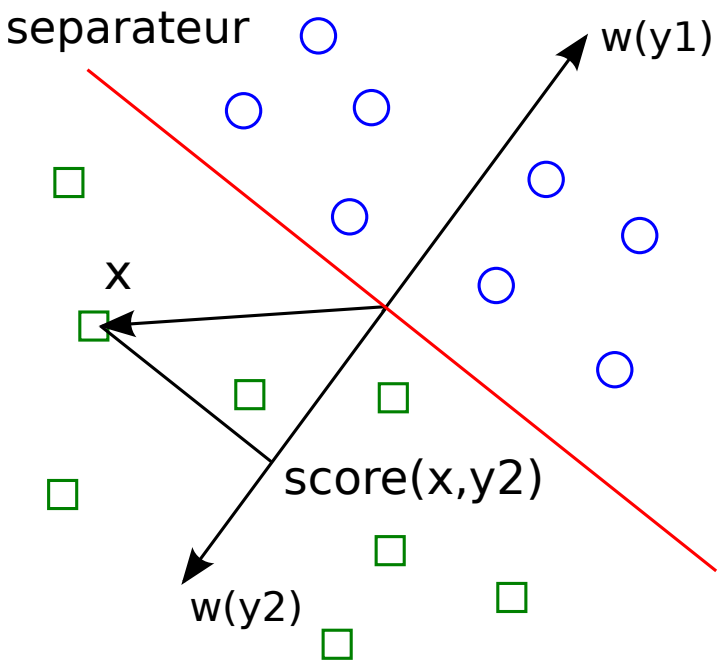
- Un vecteur de poids par classe $\boldsymbol{w}_y = \{0\}^n \quad \forall y$.
- $score(y, \boldsymbol{x}) = \boldsymbol{w}_y^T \boldsymbol{x} = \sum_i \boldsymbol{w}_y[i] \times \boldsymbol{x}[i]$.

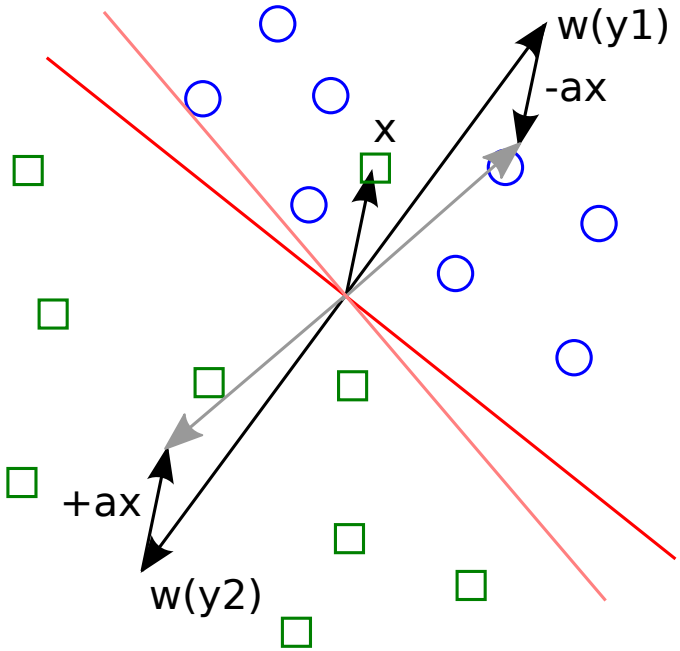
Algorithme : pour chaque exemple $(y^\star, \boldsymbol{x})$

1. voir $\boldsymbol{x}$, prédire $\hat{y} = \text{argmax}_y \, score(y, \boldsymbol{x})$
2. si $\hat{y} \neq y^\star$, mettre à jour :
   - $\boldsymbol{w}_{y^\star} = \boldsymbol{w}_{y^\star} + \alpha \boldsymbol{x}$
   - $\boldsymbol{w}_{\hat{y}} = \boldsymbol{w}_{\hat{y}} - \alpha \boldsymbol{x}$

- Boucler plusieurs fois pour converger
- Taux d'apprentissage $\alpha = \frac{1}{\sqrt{\text{nombre de m-à-j}}}$
- Extension vers réseaux de neurones multicouches (MLP, CNN, . . .)

Perceptron (Fonctionnement)





Perceptron (exemple)

x : couleurs {rouge, vert, bleu}, y : {sombre, clair}

- $\boldsymbol{w}_{sombre} = \{0, 0, 0\}$
- $\boldsymbol{w}_{clair} = \{0, 0, 0\}$

Exemples :

- 1 :(sombre, {35, 65, 128})
- 2 :(clair, {255, 0, 0})
- 3 :(sombre, {53, 0, 90})

Itérations :

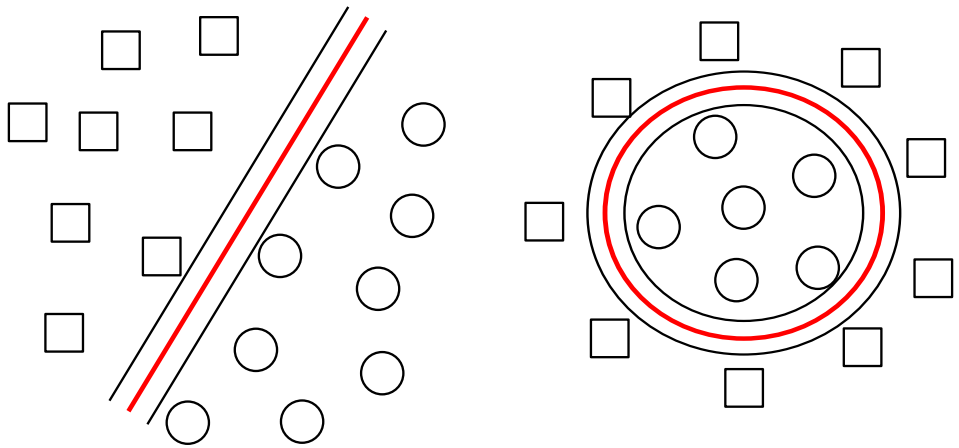
1. example 1 : $score_{sombre} = 0$, $score_{clair} = 0$
   - (m-à-j) $\rightarrow \alpha = 1$, $\boldsymbol{w}_{sombre} = \{35, 65, 128\}$, $\boldsymbol{w}_{clair} = \{-35, -65, -128\}$.
2. exemple 2 : $score_{sombre} = 8925$, $score_{clair} = -8925$
   - (m-à-j) $\rightarrow \alpha \simeq 0.7$, $\boldsymbol{w}_{sombre} = \{-143.5, 65, 128\}$, $\boldsymbol{w}_{clair} = \{143.5, 65, 128\}$.
3. exemple 3 : $score_{sombre} = 7605.5$, $score_{clair} = -7505.5$
   - (pas de m-à-j) ...

Support Vector Machines

Minimiser $||\boldsymbol{w}||$ tel que :

$$score(y_i^\star, \boldsymbol{x}_i) - score(y, \boldsymbol{x}_i) \geq error(y_i^\star, y) \quad \forall y \in \mathcal{Y}, \forall i$$

- Projection dans un espace de plus grande dimension par fonction noyau $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_j)$



Boosting

On peut faire un bon classifieur à partir de plusieurs (pas trop) mauvais classifieurs.

- *Par exemple : utiliser les features comme des arbres de décisions à un seul niveau.*

Adaboost :

1. Les exemples d'apprentissage ont un poids équiprobable
2. Sélectionner le classifieur qui permet de minimiser l'erreur pondérée
3. Appliquer le classifieur puis augmenter le poids des exemples mal classifiés
4. Boucler en (2) jusqu'à convergence

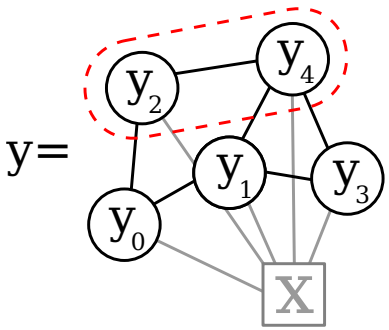
Prédictions structurées

Prédiction d'un ensemble de label

Prédictions structurées

Une prédiction est un ensemble d'étiquettes $\boldsymbol{y} = (y_1, \ldots, y_n)$

- Ajout de dépendances locales dans les features $f(y_2, y_4, \boldsymbol{x}) = 1$.



Qu'est-ce que ça change ?

- Prédiction : $\text{argmax}_{\boldsymbol{y}} f(\boldsymbol{y}, \boldsymbol{x})$
- Taux d'erreur : $error(\boldsymbol{y}^\star, \boldsymbol{y})$

Hidden Markov Models

Modèle probabiliste de séquences.

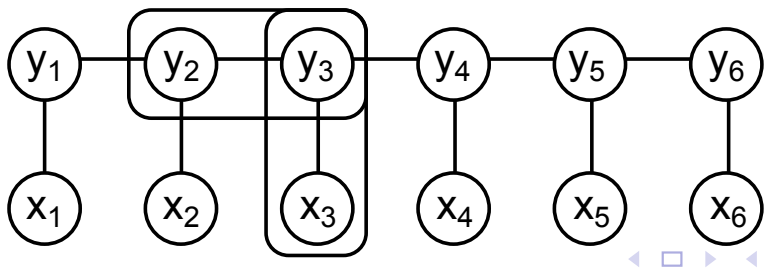
$$\begin{aligned} P(\boldsymbol{y}|\boldsymbol{x}) &= P(y_1 \dots y_n|\boldsymbol{x}) \\ &= P(y_n|y_1 \dots y_{n-1}, \boldsymbol{x}) P(y_1 \dots y_{n-1}|\boldsymbol{x}) \\ &= \prod_i P(y_i|y_1 \dots y_{i-1}, \boldsymbol{x}) \end{aligned}$$

- Hypothèse d'horizon limité :

$$P(y|x) = \prod_i P(y_i|y_{i-1}, \boldsymbol{x})$$

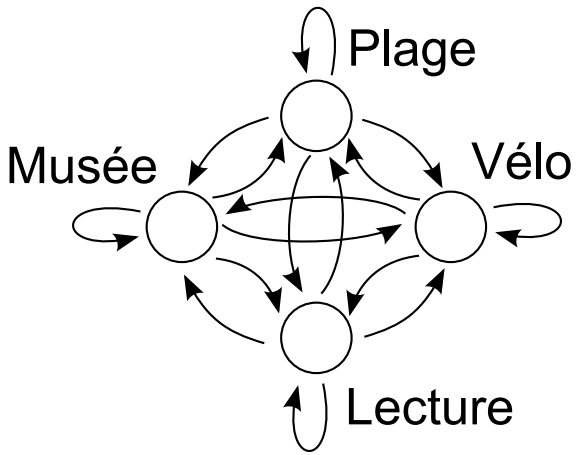
- Hypothèse d'indépendance des observations :

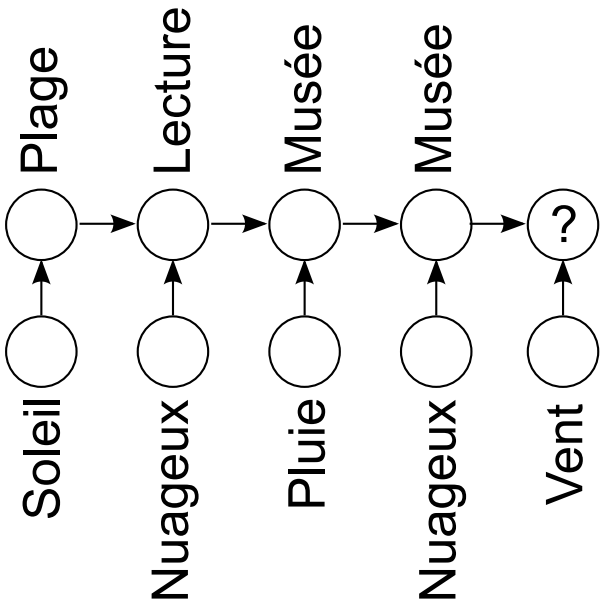
$$P(y|x) = \prod_i P(y_i|y_{i-1}) P(y_i|x_i)$$



HMM : Exemple

- x_i : Temps (Pluie, Nuageux, Soleil, Vent)
- y_i : Activité (Plage, Vélo, Musée, Lecture)





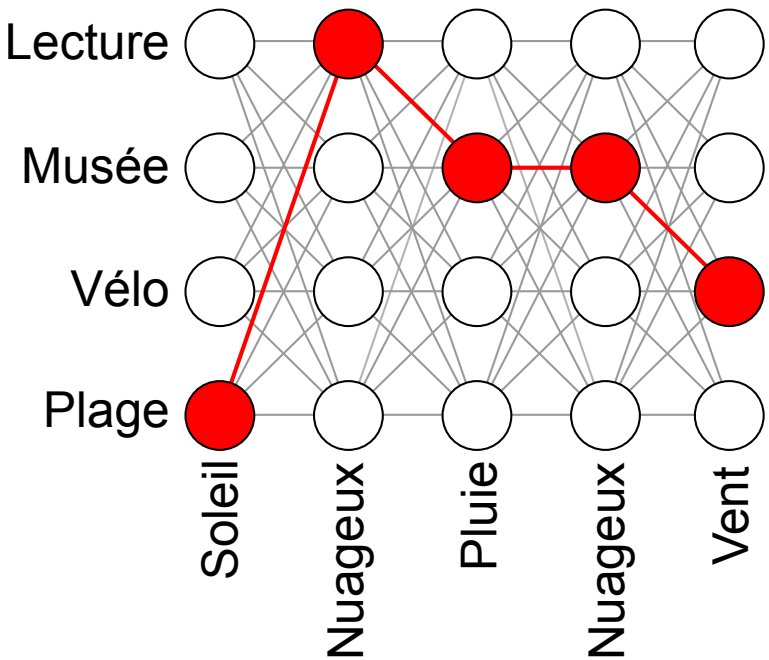
	$y_{i-1} = p$	$y_{i-1} = v$	$y_{i-1} = m$	$y_{i-1} = l$	$x_i = p$	$x_i = n$	$x_i = s$	$x_i = v$
$y_i = p$	0.2	0.3	0.4	0.1	0.1	0.3	0.2	0.4
$y_i = v$	0.1	0.2	0.3	0.4	0.3	0.2	0.4	0.1
$y_i = m$	0.4	0.1	0.2	0.3	0.2	0.4	0.1	0.3
$y_i = l$	0.3	0.4	0.1	0.2	0.4	0.1	0.3	0.2

Algorithme de Viterbi

Prédiction d'une séquence de y_i avec des dépendances entre y_{i-1} et y_i.

$$\hat{\boldsymbol{y}} = \underset{y_0,\ldots,y_n}{\operatorname{argmax}} \sum_i \log P(y_i|y_{i-1}) + logP(y_i|x_i)$$

Équivalent au plus court chemin dans un graphe



Conditional Random Fields

Modèle probabiliste paramétré par $\boldsymbol{w}$

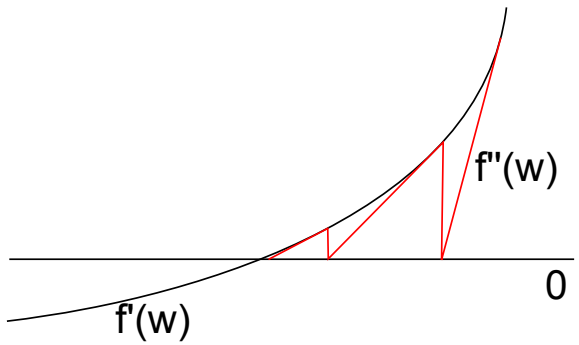
$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} exp(\sum_i \sum_j w_j f_j(y_i, y_{i-1}, \boldsymbol{x}))$$

$$Z(\boldsymbol{x}) = \sum_{\boldsymbol{y}} exp(\sum_i \sum_j w_j f_j(y_i, y_{i-1}, \boldsymbol{x}))$$

Maximiser la vraisemblance des données d'apprentissage :

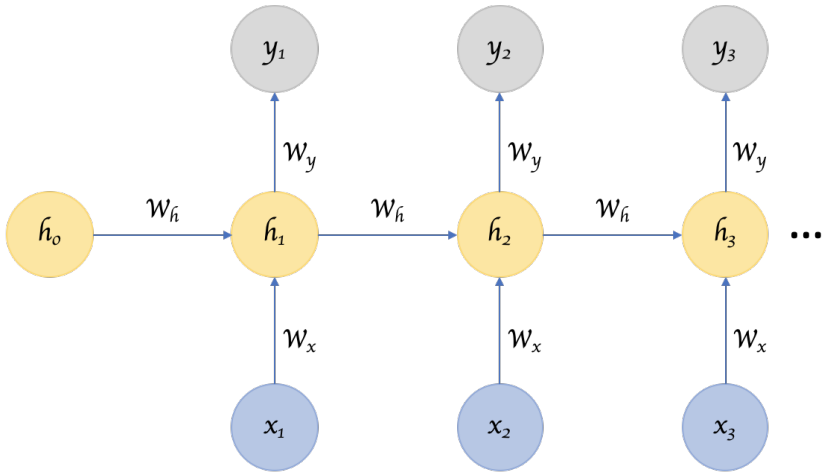
$$\hat{\boldsymbol{w}} = argmax_{\boldsymbol{w}} \prod_k P(\boldsymbol{y}_{(k)}|\boldsymbol{x}_{(k)}, \boldsymbol{w})$$

Optimisation numérique par Méthodes quasi-Newtoniennes (LBFGS)



Réseaux de neurones récurrents

- Recurrent Neural Network (RNN)
- Long Short Term Memory (LSTM)
- etc.



Application : étiquetage morpho-syntaxique

Parties de discours

- Objectif : Associer des catégories aux mots
- Principales catégories
  - nc : nom commun (arbre, ciel, neige)
  - np : nom propre (Luminy, Jean, iPhone)
  - v : verbe (mangeais, parlerons, aller)
  - adj : adjectif (petit, grande, jaune)
  - adv : adverbe (bien, passionnément)
  - det : déterminant (le, la, ces, mon,)
  - prep : préposition (de, à, dans, pour)
  - cs : conjonctions de subordination (que, comme, quand)
  - cc : conjonctions de coordination (où, mais, car, donc)
  - pro : Pronoms (lui, qui)
  - cl : Pronoms clitiques faibles (je, tu, il, elle)
  - interjections, mots étrangers, ponctuation...

Ambiguïté

- Chaque mot peut avoir plusieurs catégories
- Exemple : *la souris le fait*
  - la : nc, det, pro
  - souris : v, nc
  - le : det, pro
  - fait : nc, v, adj
- Comment trouver la catégorie en contexte ?
- Dictionnaire de catégories possibles
  - Lefff (`http://www.labri.fr/perso/clement/lefff/`)
  - Lexique 3 (`http://www.lexique.org/`)

Méthodes

- Décisions locales
  - règles
  - classifieur
- Décisions globales
  - Modèles de séquences

Méthodes à base de règles

- Connaissances linguistiques
  - formalisation de connaissances linguistiques pour enlever l'ambigüité
  - donne de bons résultats pour l'étiquetage en POS (Brill Tagger)
- Problèmes
  - difficultés pour écrire les règles
    - ex : *le* + *président* = *(président,nom)*
    - mais : *ils le président*
  - problèmes des mots hors-vocabulaire
  - difficulté pour produire des scores de confiance (pipeline)

Classifieurs

- Chaque POS est prédit en fonction du contexte d'apparition du mot
  - exemple : *ils le* **président** *avec les*

$(w_{i-2}$=*ils*$)(w_{i-1}$=*le*$)(w_i$=**président**$)(w_{i+1}$=*avec*$)(w_{i+2}$=*les*$)$=¿ *Verbe*

- Avantage : simplicité
- Inconvénients : pas d'optimisation globale de la séquence
  - décisions indépendantes

Modèle de séquence

- Modèle de Markov caché

$$P(det\ nc\ pro\ v|la\ souris\ le\ fait) \sim P(det|la)P(nc|souris)P(pro|le)P(v|fait) \times P(det|\langle deb\rangle)P(nc|det)P(pro|nc)P(v|pro)P(\langle fin\rangle|v)$$

- Conditional Random Fields
  - Information de séquence : y_i, y_{i-1}
  - Ambiguïté des mots : y_i, x_i
  - Morphologie : commence par une majuscule, contient des chiffres, préfixes, suffixes
- Réseaux de neurones récurrents
  - apprentissage de représentation
  - dépendances de longueur variable (LSTM, attention model, . . .)

Conclusions

Utiliser une méthode numérique pour une tâche de TAL ?

1. définir la tâche
   - entrée ?
   - sortie ?
   - s'assurer qu'il s'agit bien d'une tâche de TAL
2. Choisir une représentation des données
   - représentation symbolique explicite
   - représentation continue apprise sur corpus
3. Choisir un type d'inférence
   - prédiction locale
   - prédiction structurée (séquence, arbre, graphe)
   - génération/traduction (*end-to-end*)
4. Choisir une méthode d'inférence
5. Evaluer