

PROJET FINAL

# **Normalisation de concepts médicaux de comptes-rendus médicaux**

***Réalisé par***

Mohamed DJERRAB  
Sophia HAKAM  
Romaric KANYAMIBWA  
Boussad MERHANE  
Kayim SAID BACAR

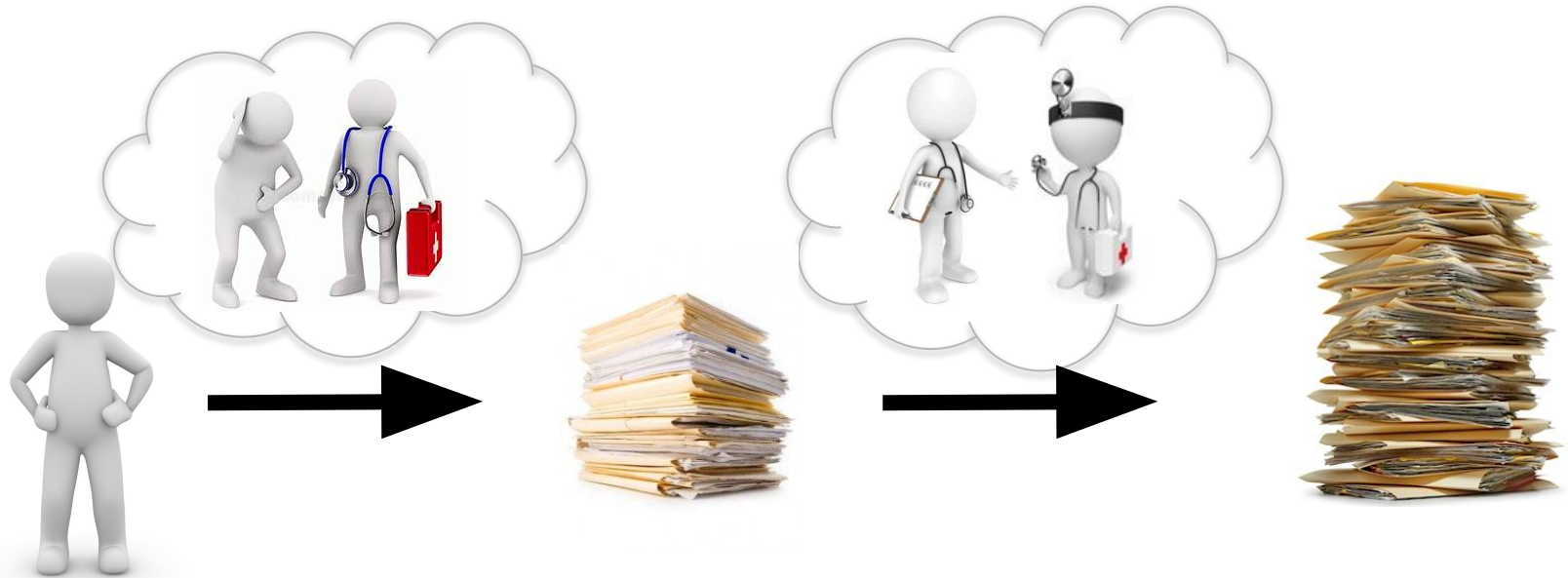
**Encadrant**

Xavier TANNIER

**MAIN 5**

*Polytech Sorbonne*  
Année 2018-2019

# Contexte



# Problématique

Simplifier la lecture de compte-rendus médicaux

Synthétiser en mots-clés

Normaliser en concepts :

→ Créer un lien entre les mentions de **concepts issues des textes** et ceux d'une **base de connaissances**

# Base de données UMLS

## Répartition en catégories

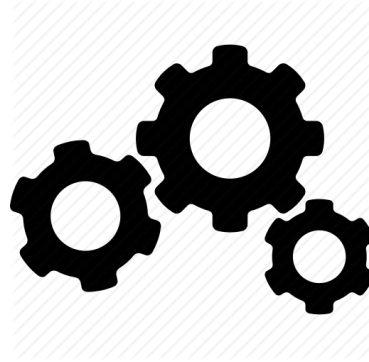
Signes et Symptômes  
Partie anatomique  
Éléments chimiques  
Organismes vivants  
Maladie  
Troubles  
Concept et idées

Partitionnement de chaque base

**Code CUI** (Concept Unique Identifiers)

**Nom** (expression ou à un groupe de mots)

**Catégorie** (concepts médicaux)

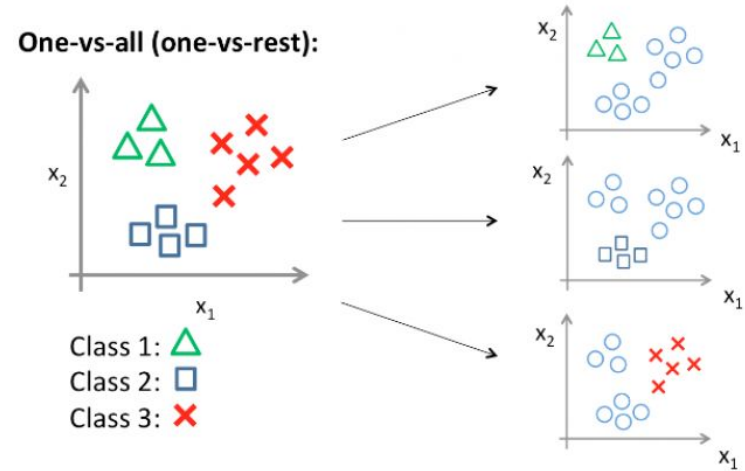


# Pre-processing

Recherche de la catégorie d'une expression donnée

# Méthode 1 : Régression Logistique

- La régression logistique est un algorithme d'apprentissage statistique très connue.
- Découper le problème de classification multi-classes en une multitude de problèmes de classification binaires.



# Résultats

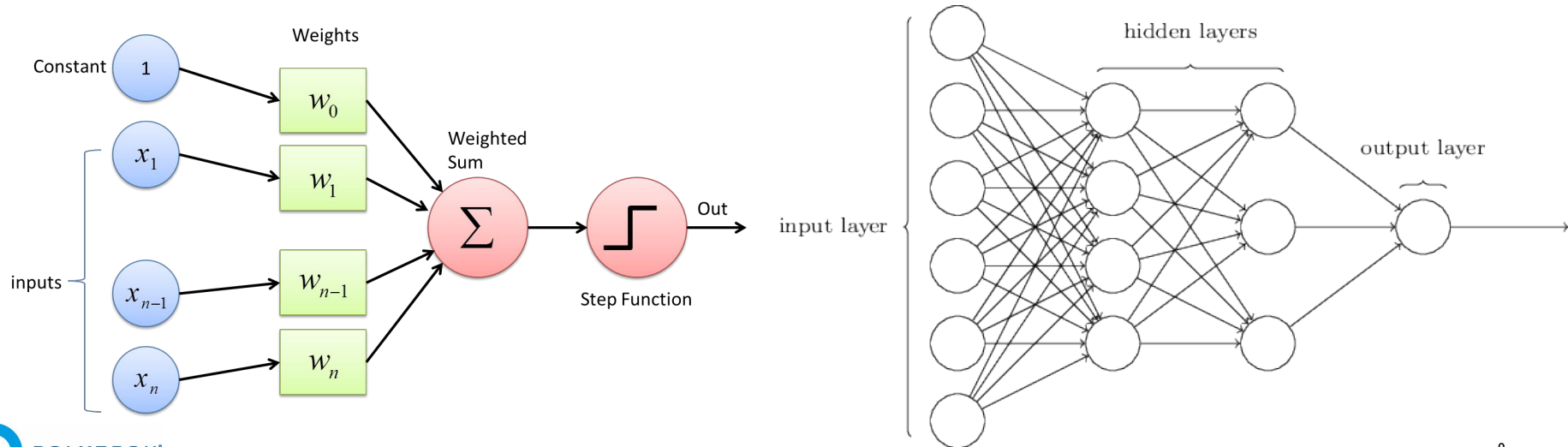
## Régression Logistique

	precision	recall	f1-score	support
0	0.97	0.95	0.96	5346
1	0.98	0.99	0.98	5647
2	0.95	0.94	0.95	5514
3	0.95	0.91	0.93	5480
4	0.85	0.93	0.89	5721
5	0.93	0.89	0.91	5594
micro avg	0.94	0.94	0.94	33302
macro avg	0.94	0.94	0.94	33302
weighted avg	0.94	0.94	0.94	33302

# Méthode 2 : Multi Layer Perceptron (MLP)

Le MLP est un algorithme de Deep Learning utilisé en classification supervisée.

Chaque neurone est un perceptron :





# Résultats

## Multi Layer Perceptron

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 1000)	15001000
activation_1 (Activation)	(None, 1000)	0
dropout_1 (Dropout)	(None, 1000)	0
dense_2 (Dense)	(None, 200)	200200
activation_2 (Activation)	(None, 200)	0
dropout_2 (Dropout)	(None, 200)	0
dense_3 (Dense)	(None, 20)	4020
activation_3 (Activation)	(None, 20)	0
dropout_3 (Dropout)	(None, 20)	0
dense_4 (Dense)	(None, 9)	189
activation_4 (Activation)	(None, 9)	0
Total params: 15,205,409		
Trainable params: 15,205,409		
Non-trainable params: 0		
Train on 26164 samples, validate on 2908 samples		

- 4 couches de neurones  
avec ReLu comme fonction d'activation

Précision MLP < Précision Régression logistique

Test accuracy: 0.8380795191465

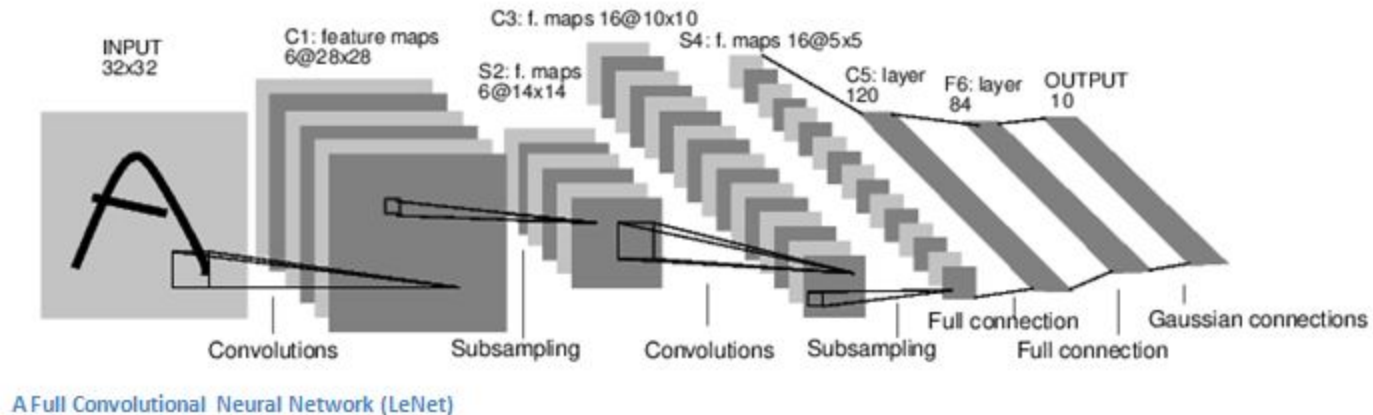


# Partie 1

Recherche des codes associés à une expression

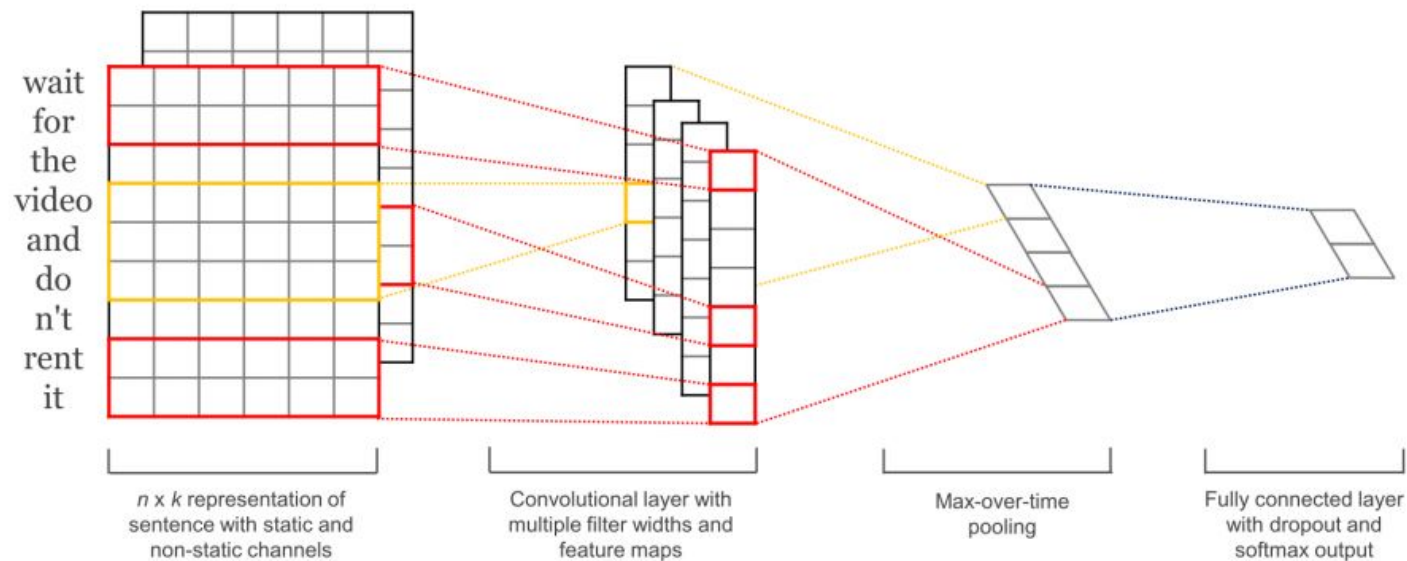
# Une première approche

## Convolutional Neural Network (CNN)



# CNNs & NLP

”Comment les CNNs s’appliquent-ils au NLP ?”

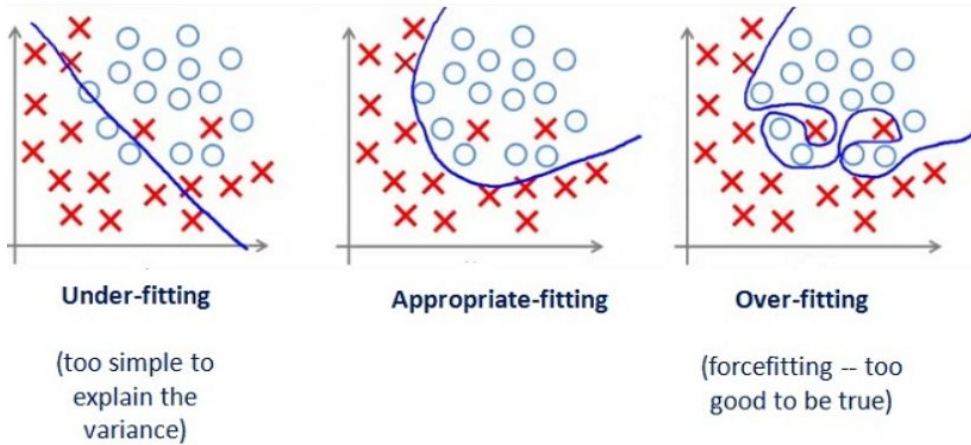


# Résultats

## CNN

Catégorie de fichiers	Train Size	Test Size	Nombre de code unique	Précision en %
Disorders	19473	499	18372	4.0081
Disorders	18474	1498	18372	3.5381
Disorders	15981	3991	18372	2.3051
Disorders	16978	4992	20080	2.8446
Disorders	18976	5989	22582	3.6233

# Problème ?



Classification de 20 000 données  
sur 18 300 classes (Codes)

- 1 code → 1 expressions



Classification CNN classique

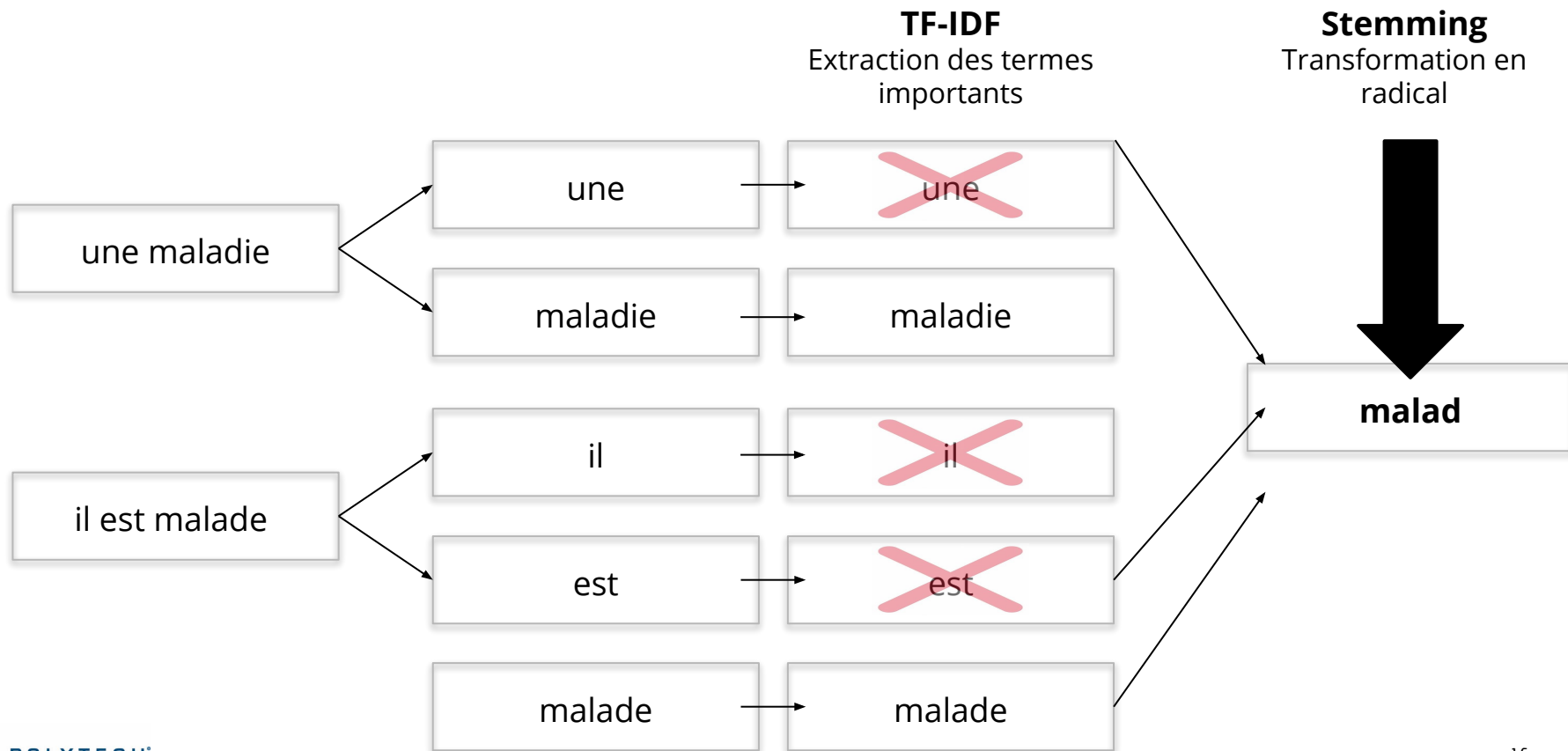


# Solutions ?

- Autres méthodes de recherche des codes associés à une expression
  - créer un dictionnaire où à un radical correspond plusieurs codes
  - rechercher du code le plus proche à partir des similarités fourni par Gensim

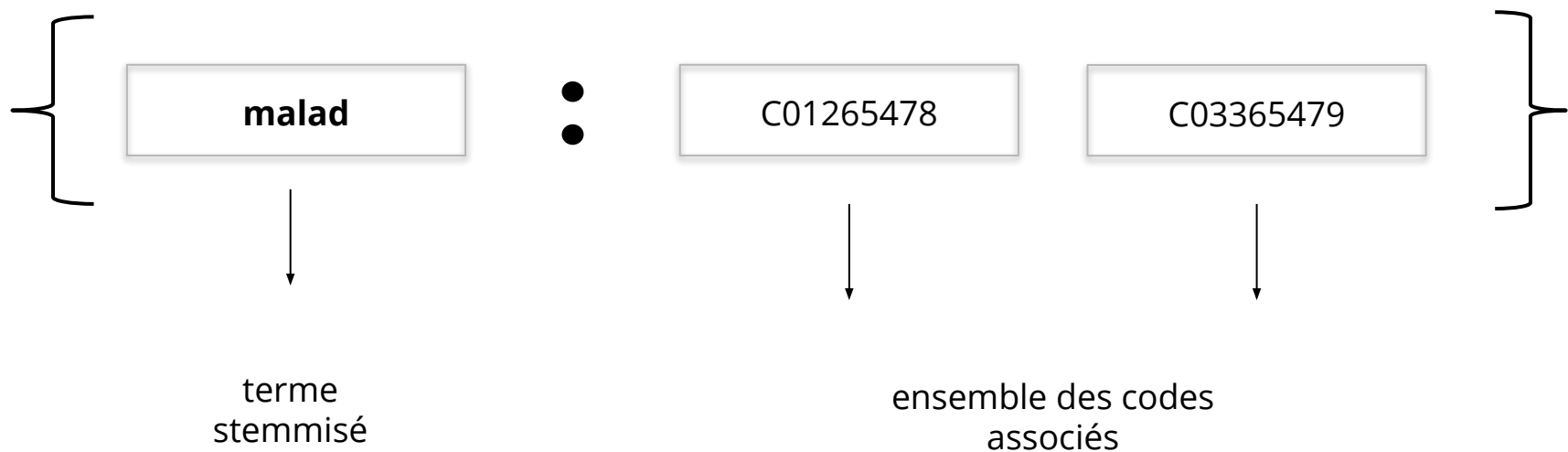


# Construction d'un dictionnaire





# Construction d'un dictionnaire



# TF-IDF

## Term Frequency-Inverse Document Frequency

Définir l'importance d'un mot-clé ou d'une phrase dans un document.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

$tf_{i,j}$  nombre d'occurrence de  $i$  dans  $j$

$$idf(w) = \log\left(\frac{N}{df_t}\right)$$

$df$  nombre de documents contenant  $i$

$N$  nombre total de documents

Le produit de ces deux métriques nous donne la formule du TF-IDF qui indique la pertinence d'un mot-clé pour un document sous la forme d'un score notée  $w$

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

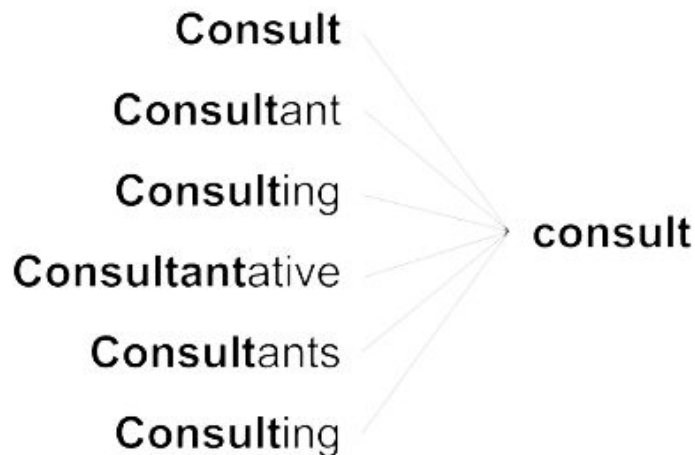
# Stemming

“Création d’un radical”

L’algorithme le plus courant

→ l’**algorithme de Porter** (Porter, 1980).

Il est composé de 5 phases de réduction de mots, appliquées de manière séquentielle.



# Stemming

Ces règles utilisent des **méthodes de "mesure" de mots** indiquant le nombre de syllabes d'un mot

But : vérifier si le mot est assez long pour qu'il soit considéré comme :  
- un **suffixe** plutôt qu'une partie d'un **radical** d'un mot.

Exemple de règles :

SSSES → SS

IES → I

SS → SS

S → S

...

Par exemple,

"replacement" → "replac"

"cement" → "c"

# Application

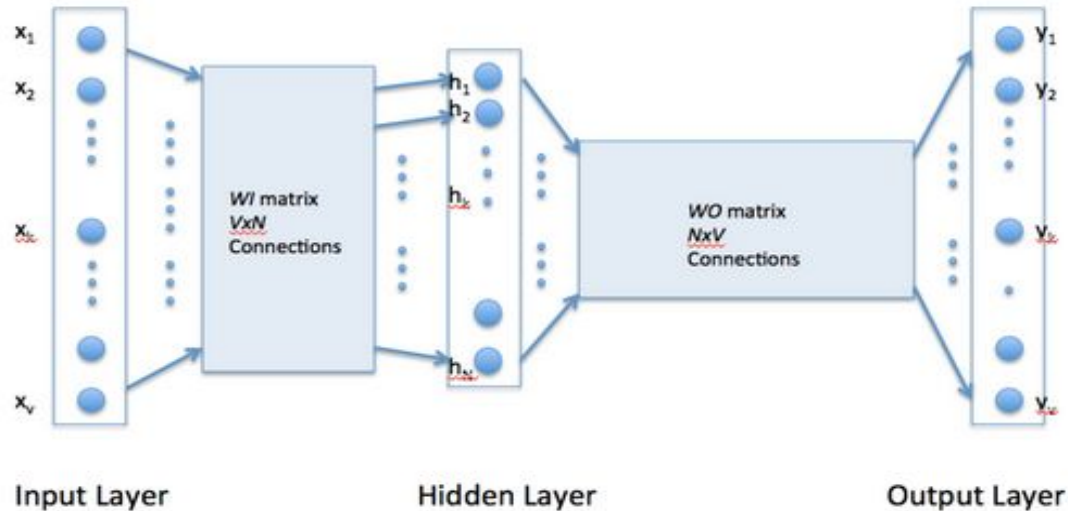
- Chaque catégorie possède son dictionnaire.
- On recherche dans le dictionnaire de cette catégorie les codes associées aux radicaux extraits.
  - On compte leur occurrence.
  - On choisit le **maximum** à l'aide de la méthode *Counter* de la bibliothèque python *collections*.

Dans certains cas, il peut arriver qu'aucun code ne se distingue des autres.

- Les **scores** du TF-IDF sont utiles pour mettre en exergue le code correspondant à l'expression la plus proche de l'expression test.

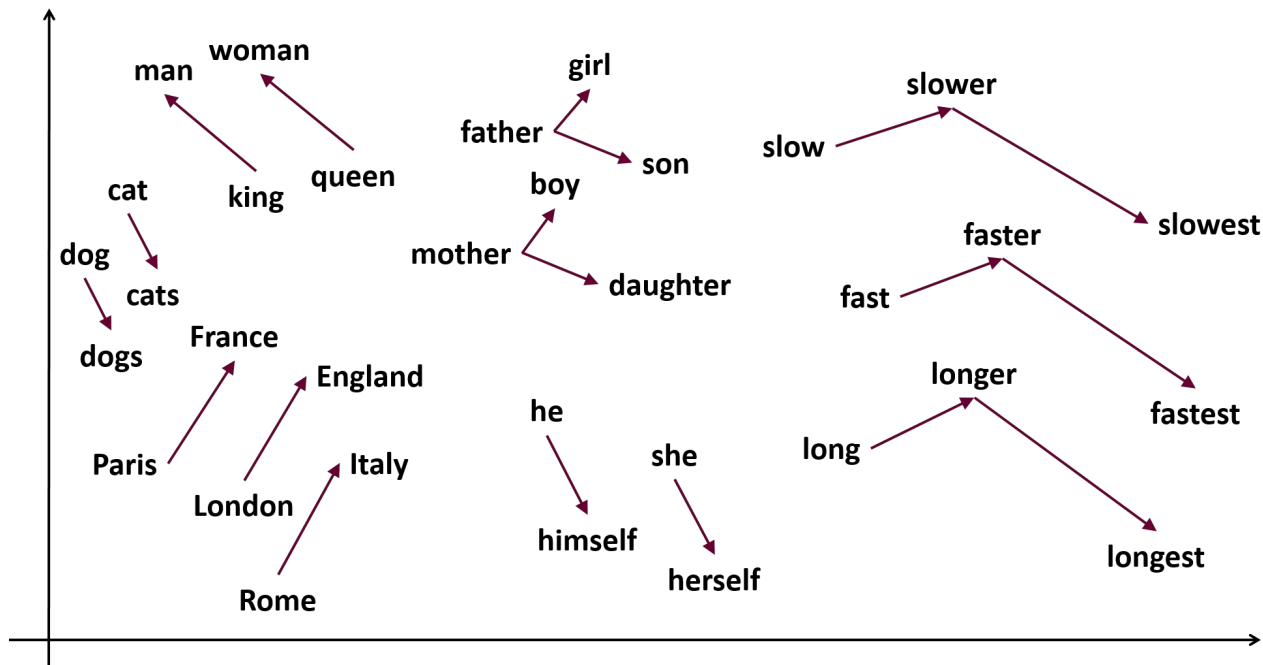
# Word2Vec

- CBOW (Continuous bag of Words)
- Skip-Gram model



# Word2Vec

King - Man + Woman = Queen



# Trouver l'expression la plus proche

Mise en place du modèle

## Pre-processing

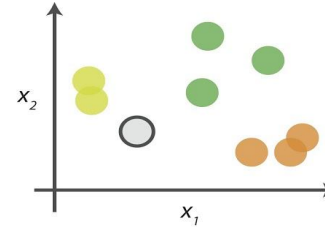
- Création d'un dictionnaire reliant les codes et les expressions
- Parcours du dictionnaire et création d'un vecteur moyen pour chaque expression



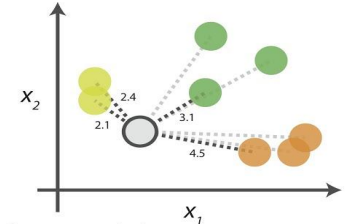
# Trouver l'expression la plus proche

Mise en place du modèle

Utilisation d'un modèle des **k plus proche voisins** à partir de nos vecteurs moyennés











On souhaite classer l'élément gris dans une des 3 classes possibles






On va calculer chaque distance pour évaluer les points plus proches

Distance des points

		2.1	→ 1 <sup>er</sup>
		2.4	→ 2 <sup>ème</sup>
		3.1	→ 3 <sup>ème</sup>
		4.5	→ 4 <sup>ème</sup>

On récupère le classement des k points les plus proches ici on a choisi  $k=4$

Classe  gagne  
Notre Point  est donc affecté à la classe .

2  
1  
1

On détermine la classe qui revient le plus de fois et on l'attribue à notre point

# Evaluation

Trouver l'expression la plus proche

- Expression Test → Vectorisation via TF-IDF  
→ Application du modèle logistique → Récupération de l'origine du fichier
  
- Expression test → Nouvelle vectorisation (à l'aide du Word2Vec)  
→ Application du KNN → Récupération de l'expression la plus proche et le code

# Résultats

Catégorie de fichiers	Précision en %	Nombre de code unique
Troubles	79.01	10250
Eléments chimiques	70.31	11412
Concept et idées	51.58	11936
Signes et Symptômes	50.20	12300
Parties anatomiques	35.12	15606

# Conclusion

- Utilisation des méthodes de classification
- Beaucoup de Labels vs peu de données
- Ouverture sur le NLP
- Zero Shot Learning

# MERCI POUR VOTRE ATTENTION



Vos questions sont  
les bienvenues