

Cours 1 : Représentations vectorielles

François HU - 05/07/2021

Data Scientist au DataLab de la Société Générale Assurances

Doctorant à l'ENSAE-CREST

Les cours se trouvent ici : <https://curiousml.github.io/>

Sommaire

1. Quelques motivations

2. Words Embeddings

- Représentations vectorielles des mots :
one-hot-encoding ?
- word embedding
- word2vec
- doc2vec

3. Topic modeling

- Représentation vectorielle des documents / topics
- Latent Semantic Analysis (**LSA**)
- Probabilistic LSA (**PLSA**)
- Latent Dirichlet Allocation (**LDA**)

Introduction

Représentations vectorielles

Deep Learning pour NLP

Active Learning

Quelques motivations

Contextes **non-supervisé**

Représenter **numériquement des mots** de sorte que la relation suivante ait un sens :

$$\text{vect}(\text{king}) - \text{vect}(\text{man}) + \text{vect}(\text{woman}) = \text{vect}(\text{queen})$$

Représenter **numériquement des thèmes (topics) cachés** et les **documents** :

- créer des **systèmes de recommandation**
(utilisés par les e-commerçants, les moteurs de recherche, ...)
- **catégorisation** de textes
- processus d'**exploration des données**
- en bio-informatique : **extraire des connaissances cachées** des données biologiques (molécules d'ADN)

2. Word Embeddings

Représentation vectorielle des mots : *one-hot-encoding* ?

objectif

texte à apprendre : mon client est content de son assurance → 

texte à prédire : l'assuré est satisfait de notre contrat → ?

Représentation vectorielle des mots : *one-hot-encoding* ?



1. définir le vocabulaire

```
V = {'a' : 1, ...  
     'assurance' : 59, ...  
     'assuré' : 62, ...  
     'client' : 698, ...  
     'content' : 772, ...  
     'contrat' : 899, ...  
     'satisfait' : 8 201, ...  
     'zoo' : 9 999, ...  
     <UNK> : 10 000 }
```

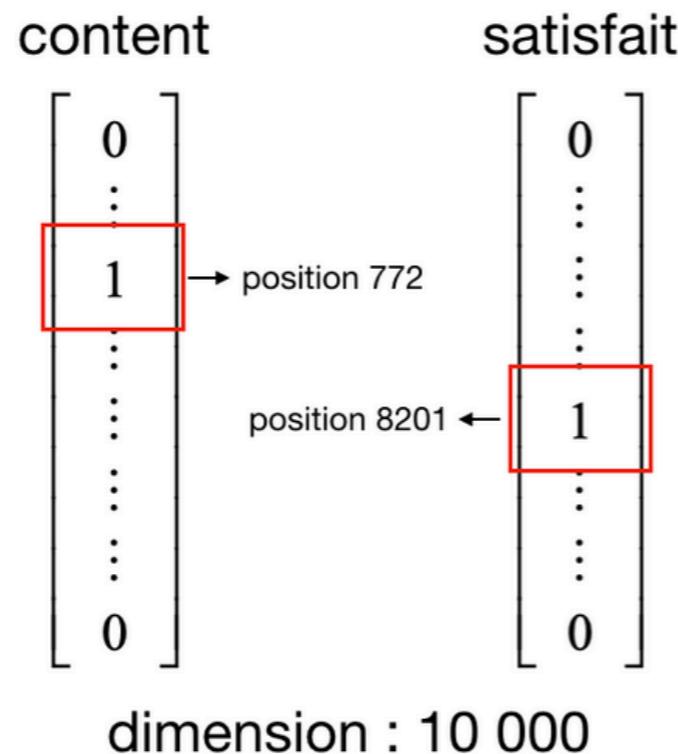
Représentation vectorielle des mots : *one-hot-encoding* ?



1. définir le vocabulaire

2. one-hot encoding

```
V = {'a' : 1, ...
      'assurance' : 59, ...
      'assuré' : 62, ...
      'client' : 698, ...
      'content' : 772, ...
      'contrat' : 899, ...
      'satisfait' : 8 201, ...
      'zoo' : 9 999, ...
      <UNK> : 10 000 }
```



Approche naïve de vectorisation :

approche one-hot

approche TF-IDF

Représentation vectorielle des mots : *one-hot-encoding* ?

Remarques

objectif

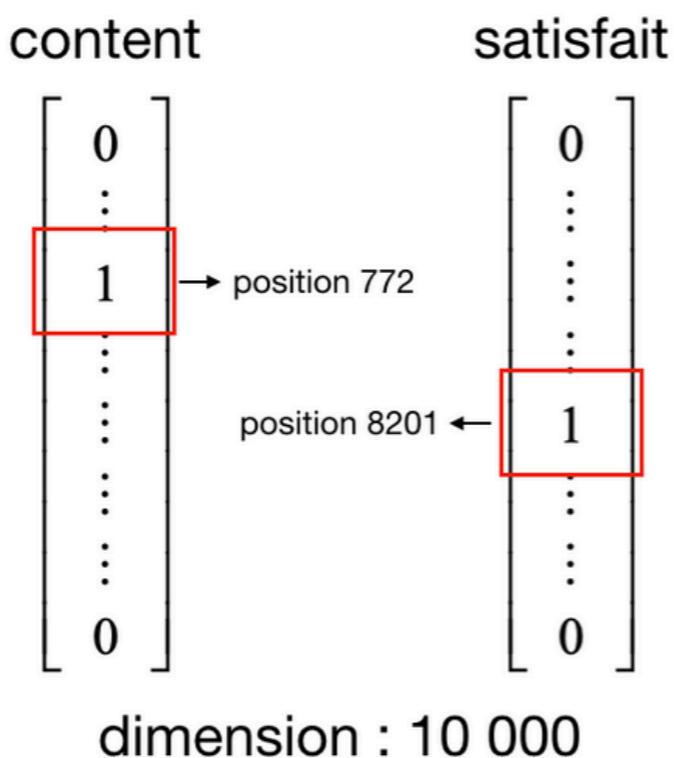
texte à apprendre : mon client est content de son assurance → 

texte à prédire : l'assuré est satisfait de notre contrat → ?

1. définir le vocabulaire

2. one-hot encoding

```
V = {'a' : 1, ...
      'assurance' : 59, ...
      'assuré' : 62, ...
      'client' : 698, ...
      'content' : 772, ...
      'contrat' : 899, ...
      'satisfait' : 8 201, ...
      'zoo' : 9 999, ...
      <UNK> : 10 000 }
```



Approche naïve de vectorisation :

approche one-hot

approche TF-IDF

Problèmes :

vecteurs creux de grandes dimensions

sauvegardent des mots sans prendre en compte le contexte :

vect (« content »)

7

vect (« satisfait »)

Représentation vectorielle des mots : *one-hot-encoding* ?

Remarques

objectif

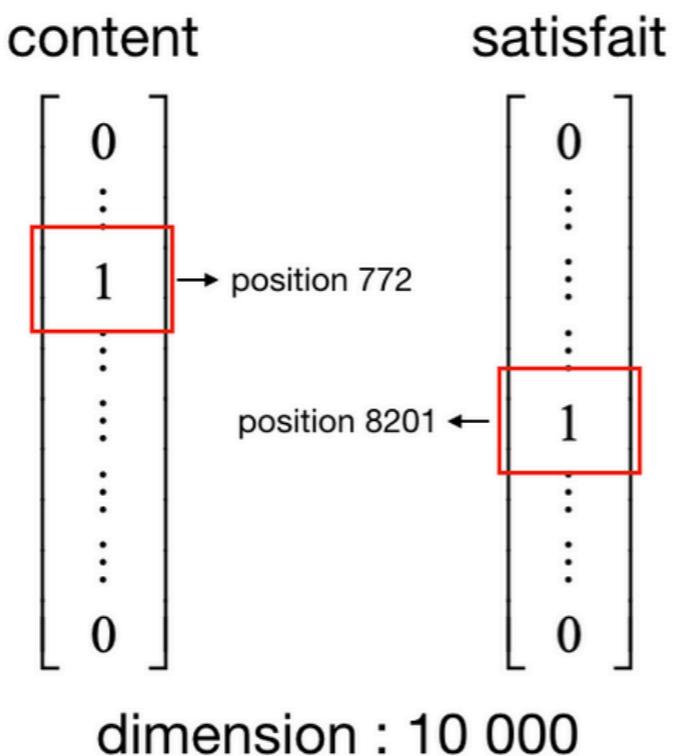
texte à apprendre : mon client est content de son assurance → 

texte à prédire : l'assuré est satisfait de notre contrat → ?

1. définir le vocabulaire

2. one-hot encoding

```
V = {'a' : 1, ...
      'assurance' : 59, ...
      'assuré' : 62, ...
      'client' : 698, ...
      'content' : 772, ...
      'contrat' : 899, ...
      'satisfait' : 8 201, ...
      'zoo' : 9 999, ...
      <UNK> : 10 000 }
```



Approche naïve de vectorisation :

approche one-hot

approche TF-IDF

Problèmes :

vecteurs creux de grandes dimensions

sauvegardent des mots sans prendre en compte le contexte :

vect (« content »)

7

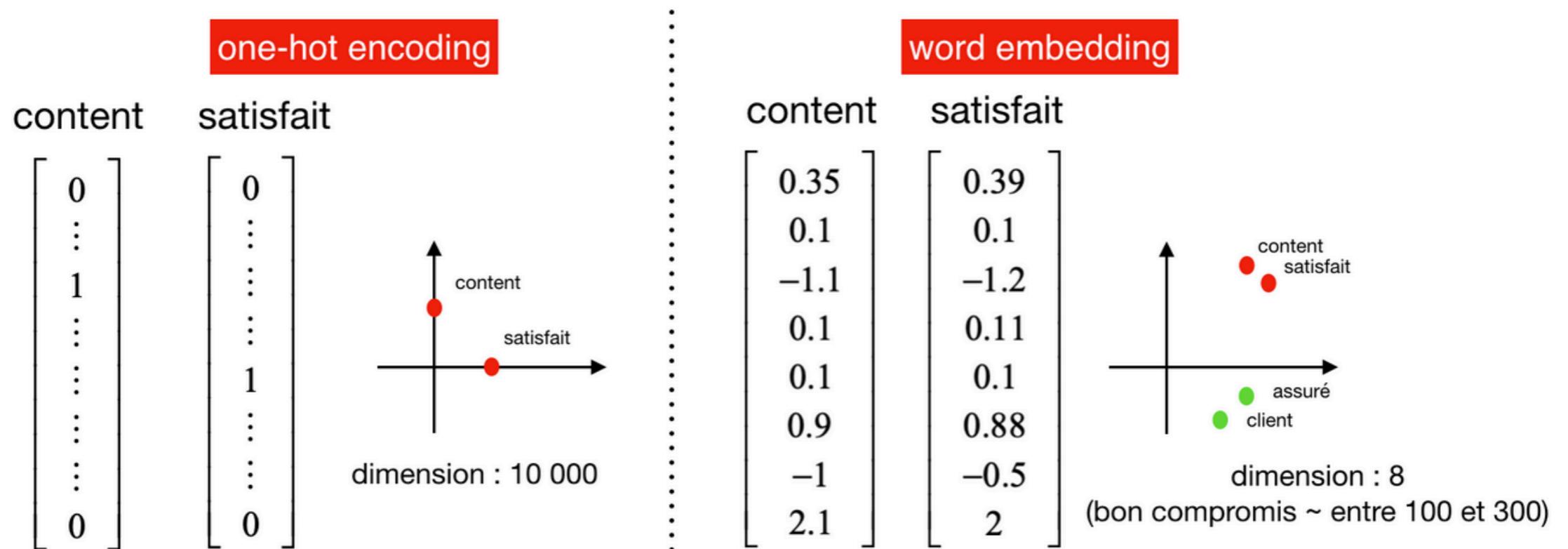
vect (« satisfait »)

Solution :

approche word embedding

Word Embedding

Word embedding (plongement de mot en français) : vectorisation des mots de sorte que les mots apparaissant dans des contextes similaires ont des significations apparentées



Vecteurs denses de plus petites dimensions

$$\text{mot1} \approx \text{mot2} \implies \text{word_embedding}(\text{mot1}) \approx \text{word_embedding}(\text{mot2})$$

Word embedding populaire : **word2vec**

Word2vec

Principe : utiliser les réseaux de neurones pour construire ces vecteurs

Deux architectures de word2vec : **Continuous Bag-Of-Words** (CBOW) et **Skip-Gram**

Exemple de texte :

notre client est content de son assurance automobile

notre client est content de son assurance automobile
client content assurance automobile

Word2vec

Principe : utiliser les réseaux de neurones pour construire ces vecteurs

Deux architectures de word2vec : **Continuous Bag-Of-Words** (CBOW) et **Skip-Gram**

Exemple de texte :

notre client est content de son assurance automobile

notre client est content de son assurance automobile

client content assurance automobile

Générer des observations pour word2vec pour une fenêtre de taille 2

cible	contexte
observation 1 :	client content assurance automobile
observation 2 :	client content assurance automobile
observation 3 :	client content assurance automobile
observation 4 :	client content assurance automobile

Word2vec

Principe : utiliser les réseaux de neurones pour construire ces vecteurs

Deux architectures de word2vec : **Continuous Bag-Of-Words** (CBOW) et **Skip-Gram**

Exemple de texte :

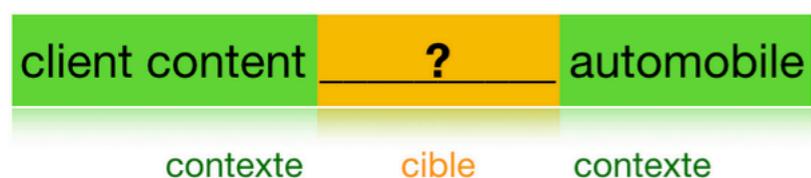
notre client est content de son assurance automobile
notre client est content de son assurance automobile
client content assurance automobile

Générer des observations pour word2vec pour une fenêtre de taille 2

	cible	contexte
observation 1 :	client	content assurance automobile
observation 2 :	client	content assurance automobile
observation 3 :	client	content assurance automobile
observation 4 :	client	content assurance automobile

Modèle CBOW :

cherche à prédire un mot à partir du contexte



réseaux de neurones

prédir assurance

Modèle skip-gram :

cherche à prédire les mots du contexte à partir d'un mot

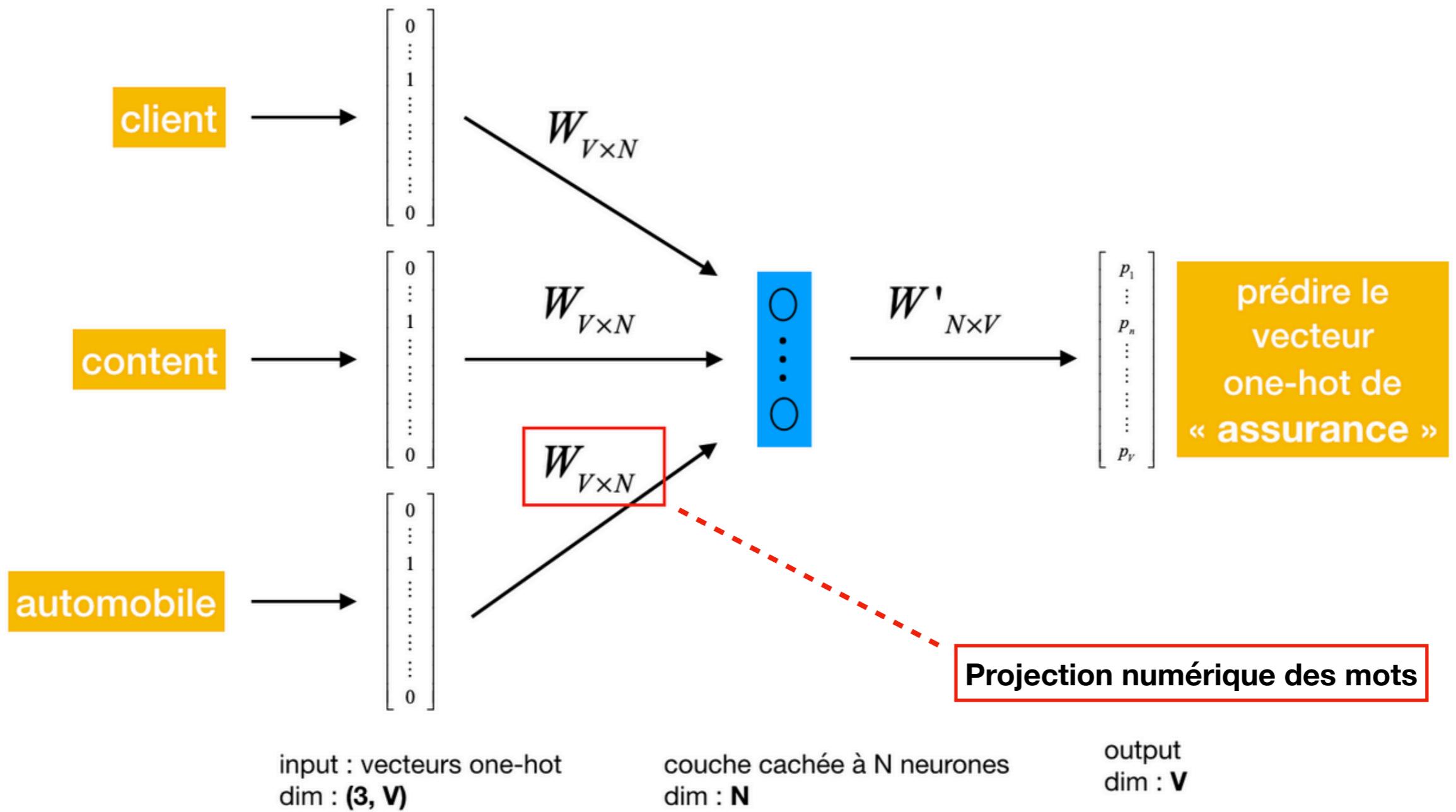


réseaux de neurones

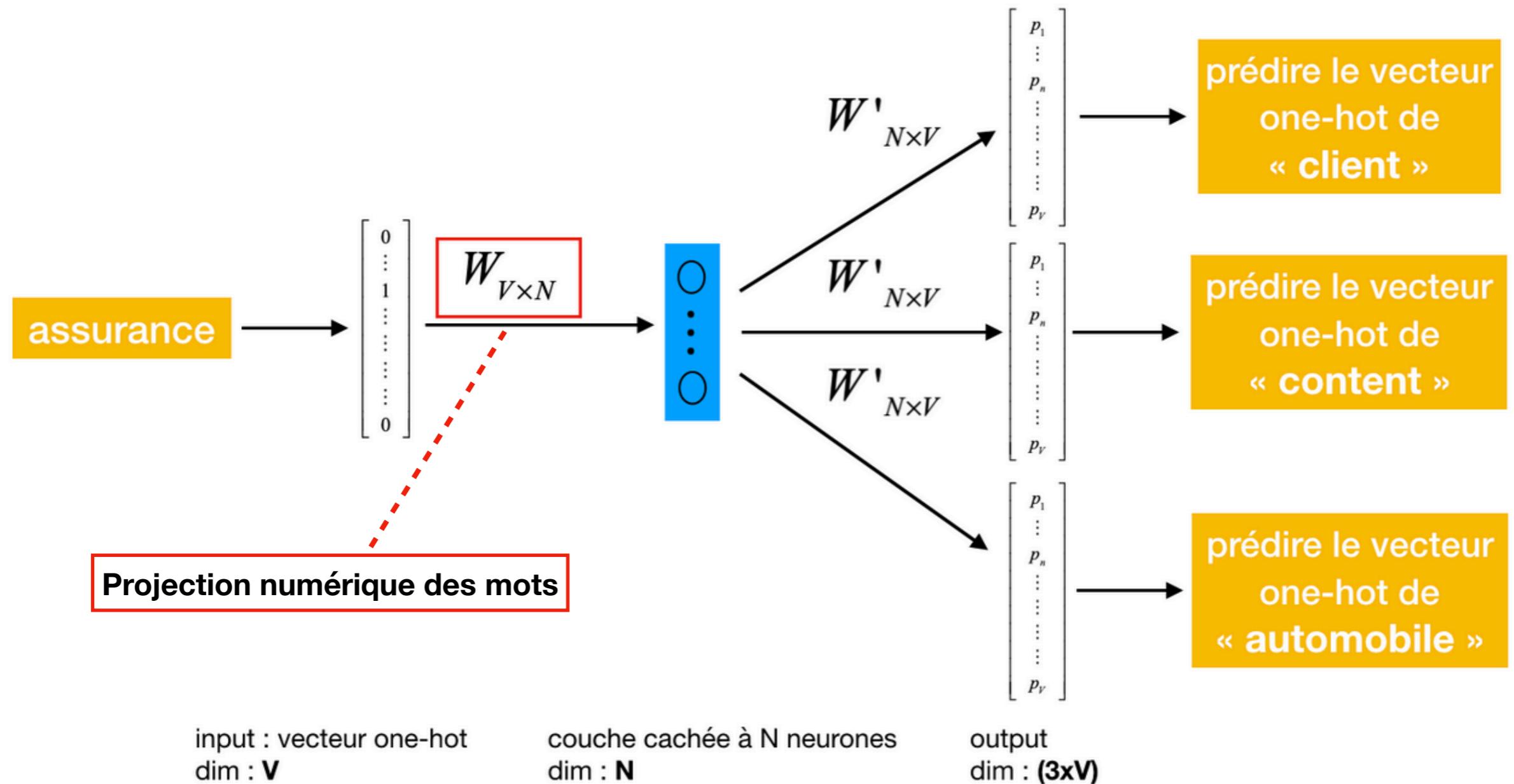
prédir : client content

automobile

Word2vec : modèle CBOW

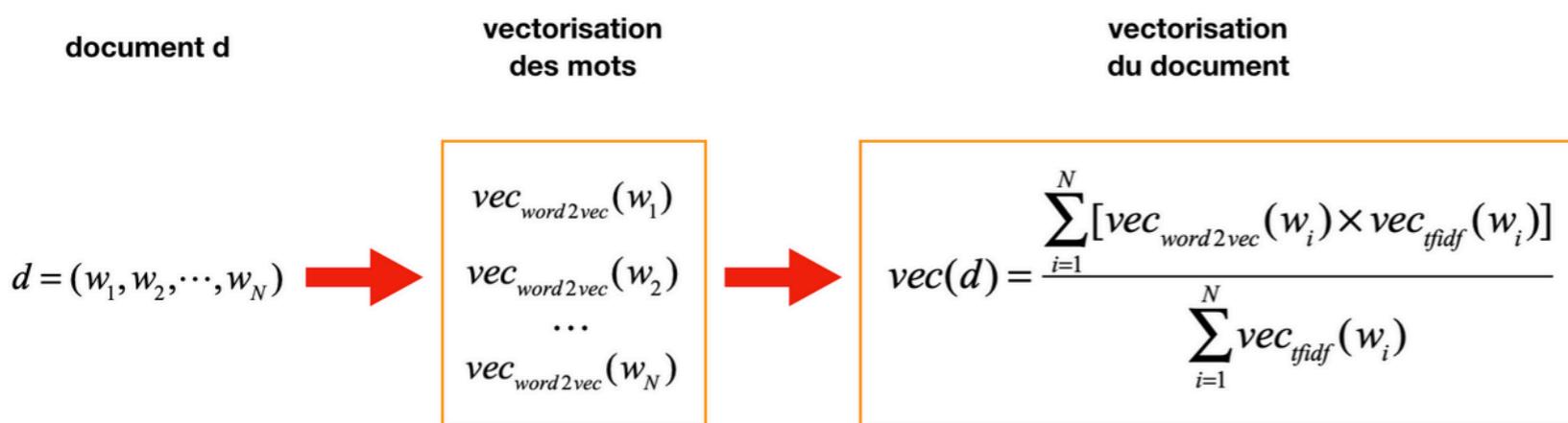
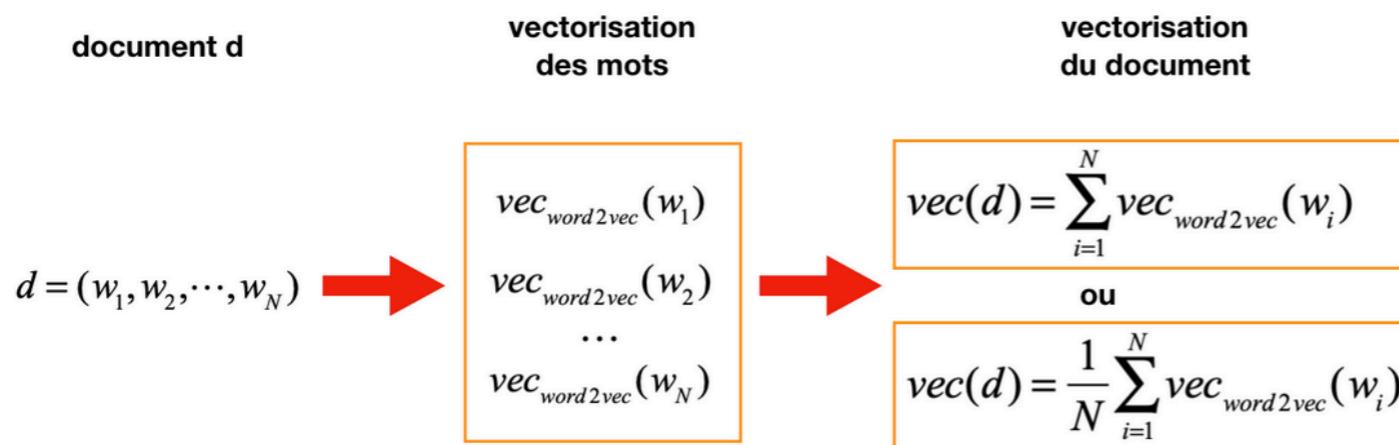


Word2vec : modèle Skip-Gram



classification de textes avec word2vec

agrégation puis classification classique :



doc2vec puis classification classique : voir [section suivante](#) pour le modèle doc2vec

classification avec les réseaux de neurones : voir [cours 2](#) pour les modèles séquentiels

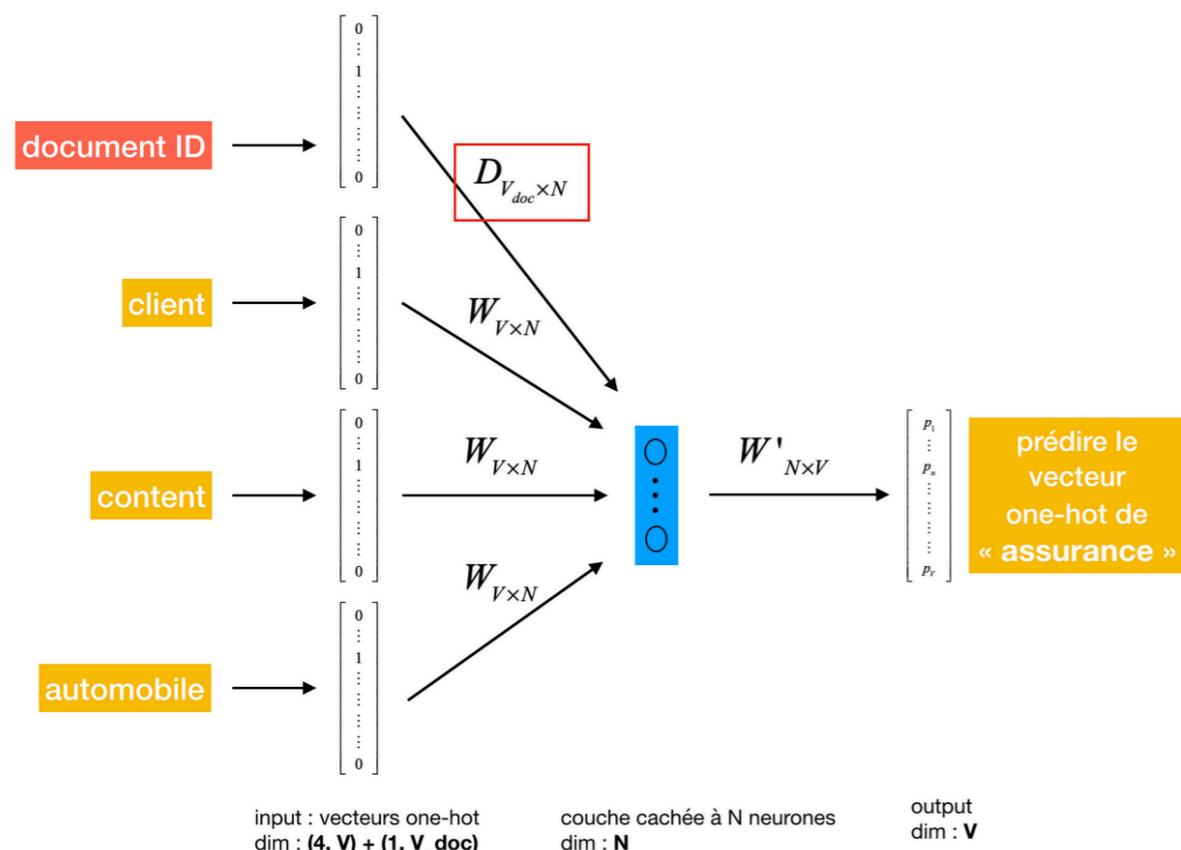
classification de textes avec doc2vec

doc2vec

vectorisation des **documents** (resp. des mots) de sorte que les **documents** (resp. les mots) apparaissant dans des contextes similaires ont des significations apparentées

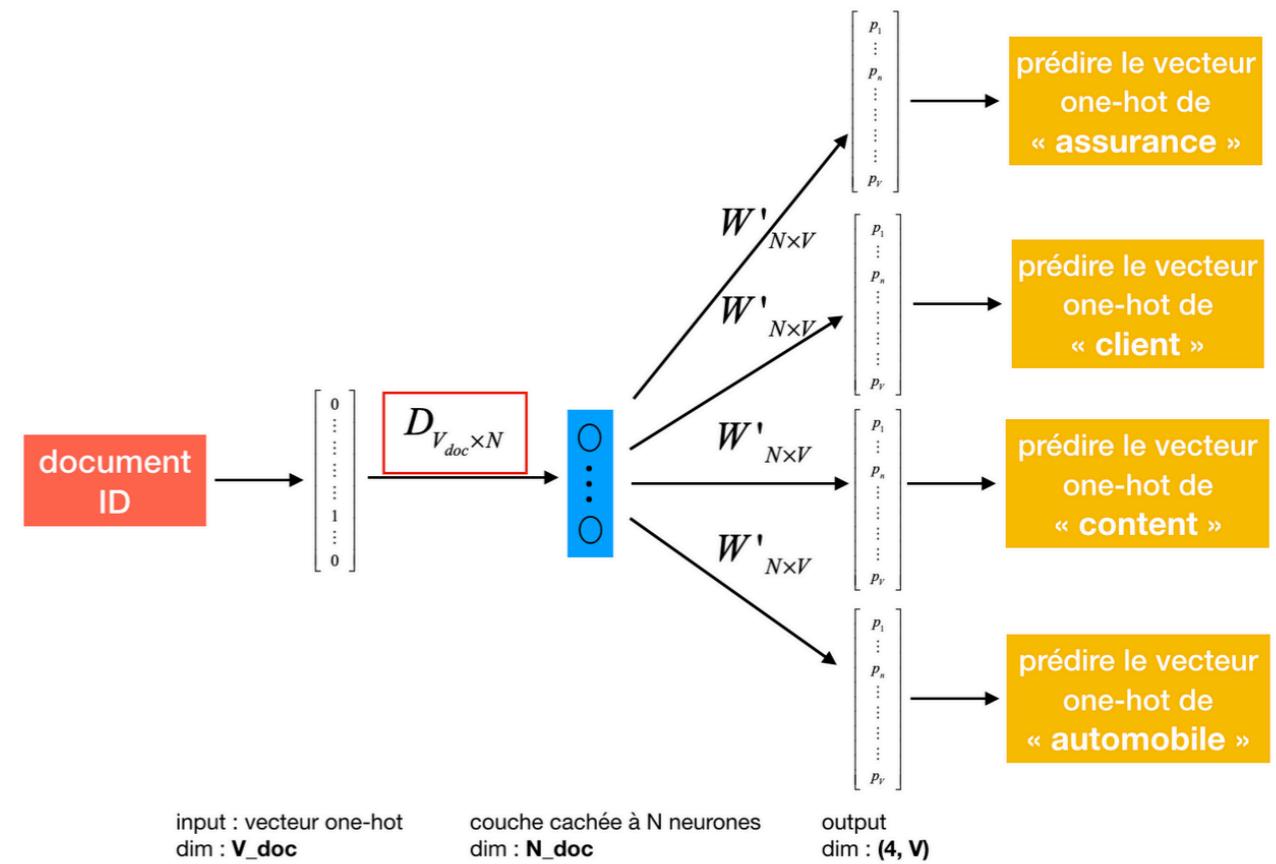
Distributed Memory (DM) :

$$p(w_i \mid w_{i-h}, \dots, w_{i+h}, d)$$

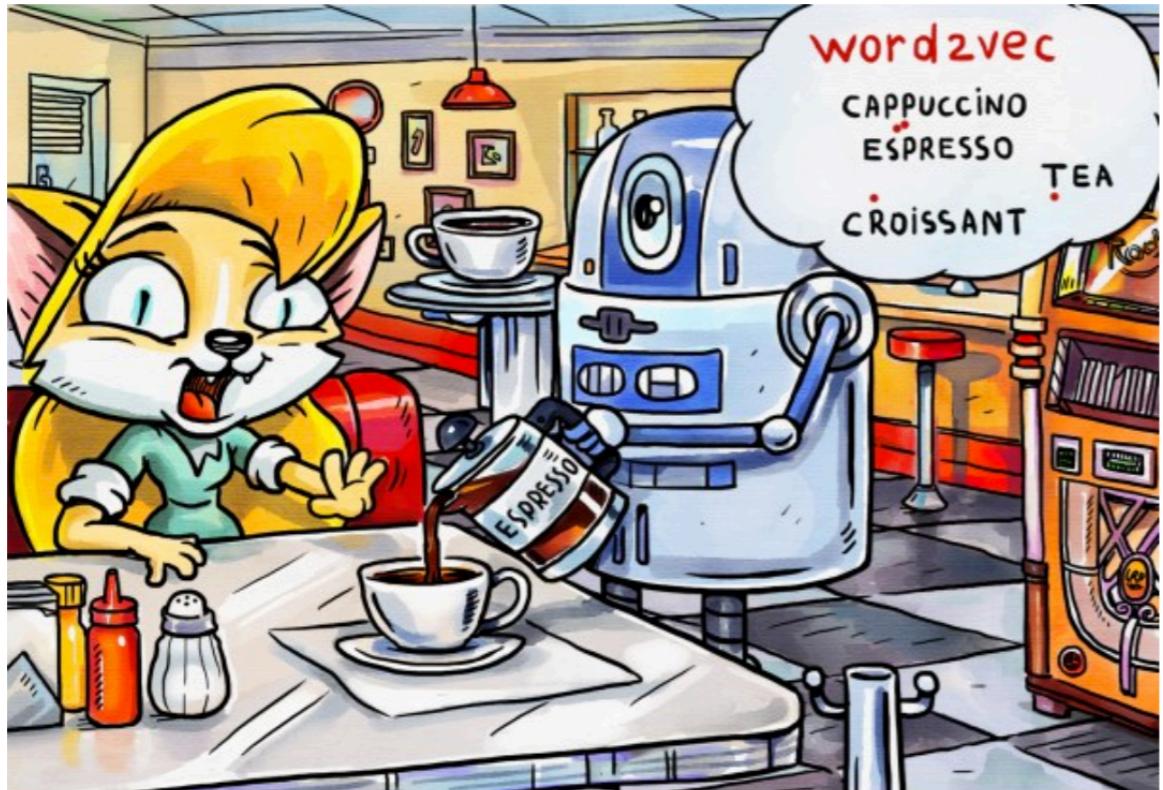


Distributed Bag-Of-Words (DBOW) :

$$p(w_{i-h}, \dots, w_{i+h} \mid d)$$



Résumé



- Espresso? But I ordered a cappuccino!
- Don't worry, the cosine distance between them is so small
that they are almost the same thing.

word embedding :

- word2vec :
CBOW, skip-gram

doc embedding :

- doc2vec :
DM, DBOW

classification de textes avec ces embeddings

auteur : Dmitry Malkov, data scientist et dessinateur de BD chez Data Monsters

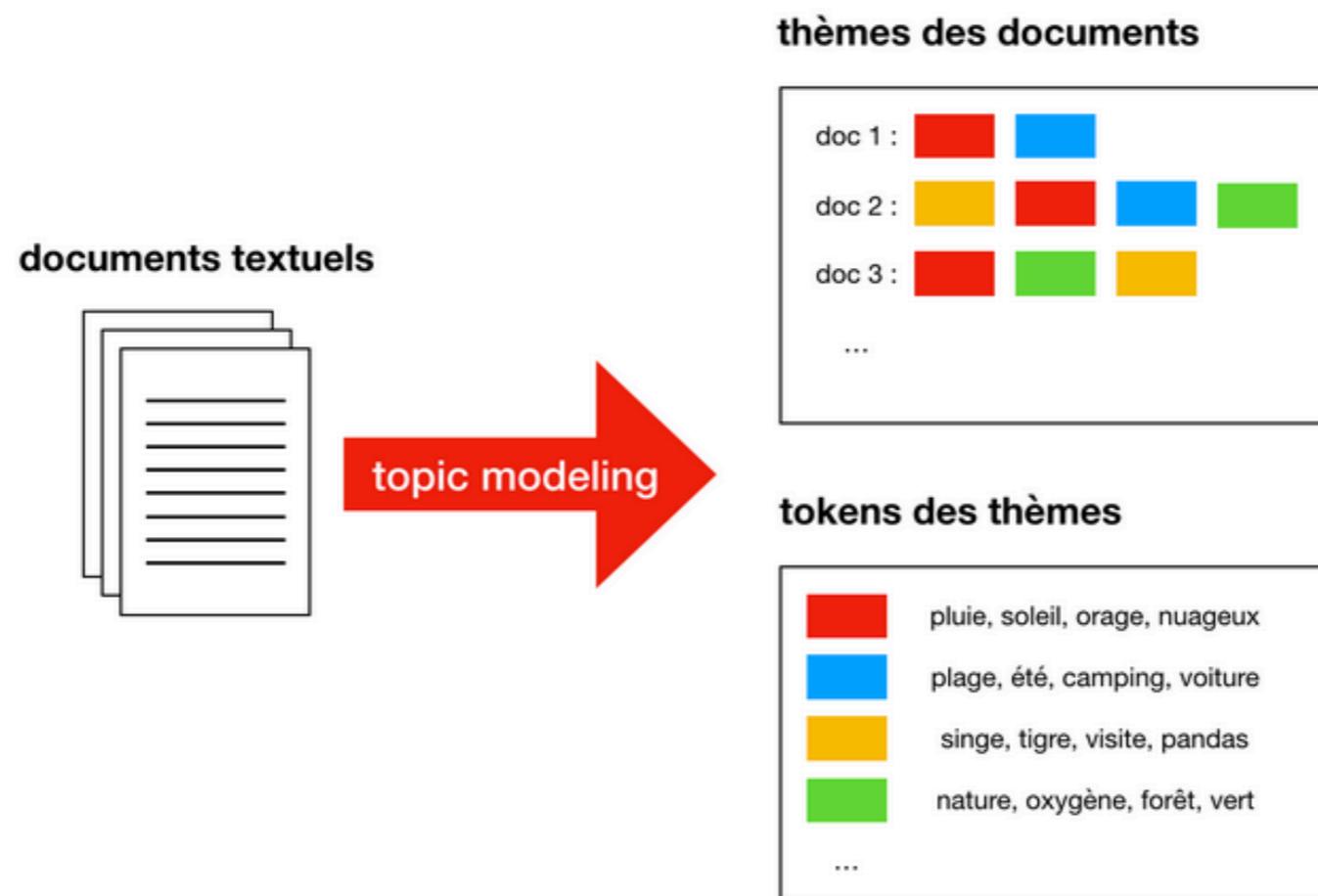
3. Topic modeling

Représentation vectorielle des documents / topics

Topic modeling (modélisation thématique en français) : un modèle statistique permettant de découvrir les "sujets" cachés qui apparaissent dans une collection de documents.

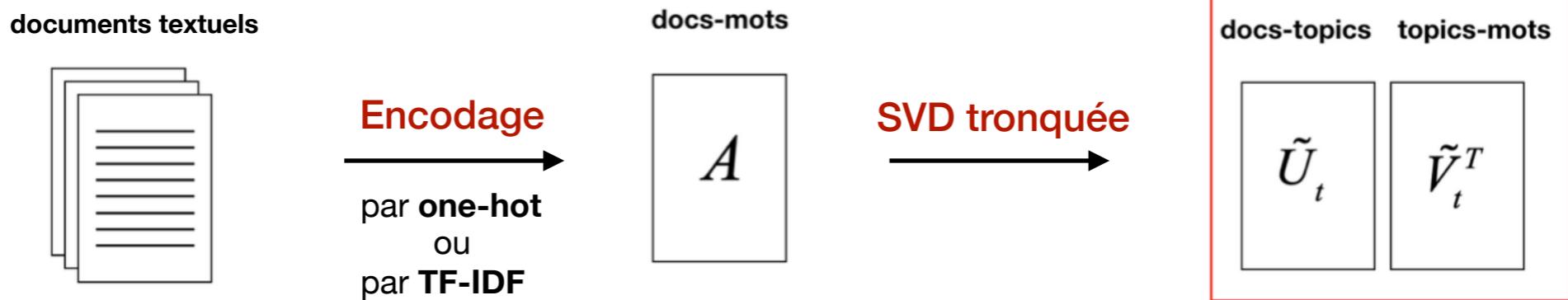
Idée :

- chaque **document** est vu comme un **mélange de topics**
- chaque **topic** est vu comme une **collection de mots**

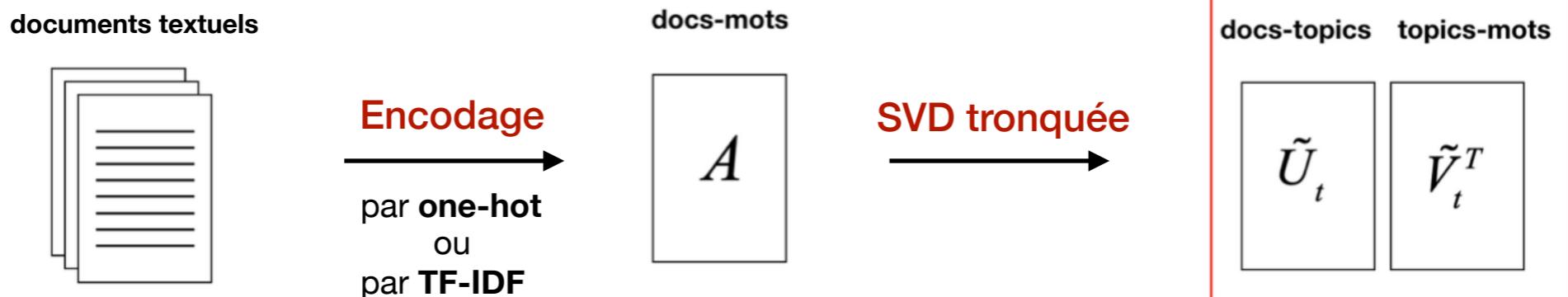


Topic model le plus populaire : LSA, **LDA**

Latent Semantic Analysis (1/2)



Latent Semantic Analysis (1/2)



Décomposition en Valeurs Singulières (SVD) :

$$A = U \times S \times V^T$$

factorisation matricielle

matrice unitaire $n \times d$

matrice unitaire $n \times n$

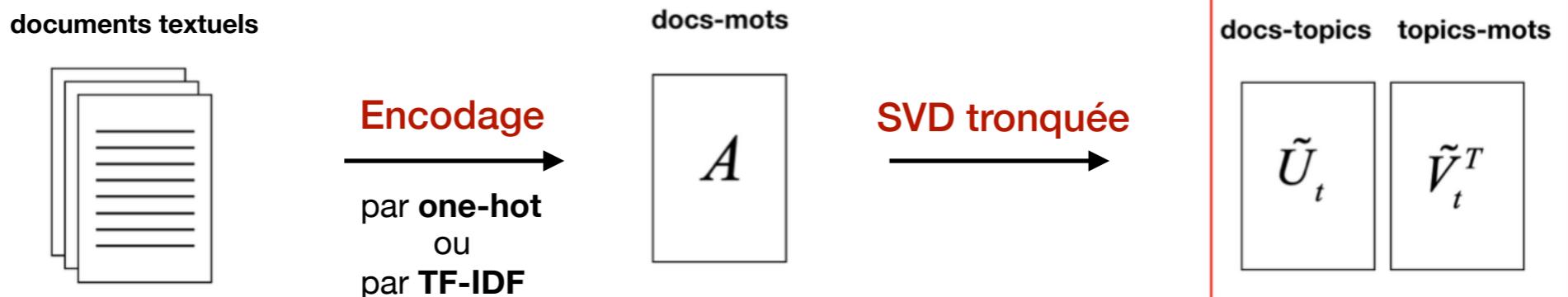
matrice diagonale
(valeurs singulières de A)
ps : on les range par ordre décroissant

matrice unitaire $n \times d$

$d \times d$

V^T

Latent Semantic Analysis (1/2)



Décomposition en Valeurs Singulières (SVD) :

factorisation matricielle

$$A = U \times S \times V^T$$

matrice unitaire matrice diagonale
(valeurs singulières de A)
ps : on les range par ordre décroissant

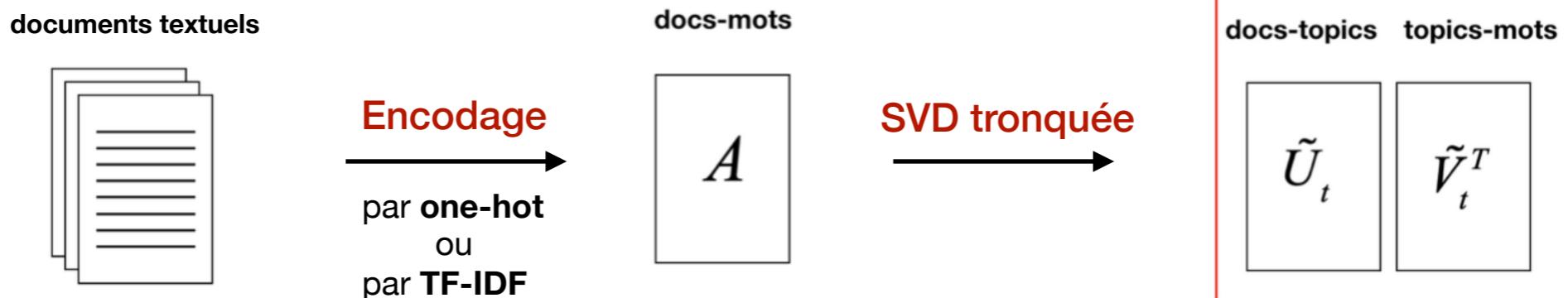
Dimensions: $A: n \times d$, $U: n \times n$, $S: n \times d$, $V^T: d \times d$

SVD tronquée :

$$A \approx U_t \times S_t \times V_t^T$$

Dimensions: $A: n \times d$, $U: n \times n$, $S: n \times d$, $V^T: d \times d$

Latent Semantic Analysis (1/2)



Décomposition en Valeurs Singulières (SVD) :

factorisation matricielle

$$A = U \times S \times V^T$$

matrice unitaire $n \times n$ matrice diagonale $n \times d$ matrice unitaire $d \times d$

matrice unitaire $n \times d$ matrice diagonale (valeurs singulières de A)
ps : on les range par ordre décroissant

SVD tronquée :

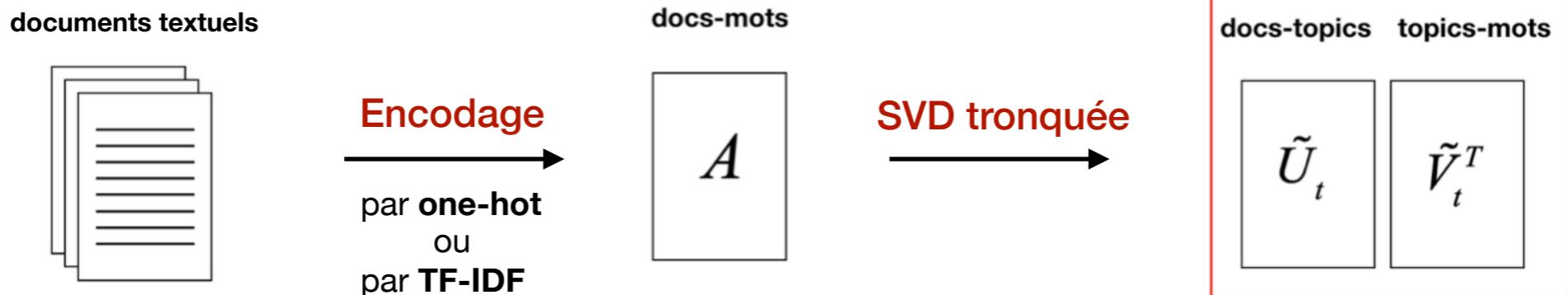
$$A \approx U_t \times S_t \times V_t^T$$

matrice unitaire $n \times t$ matrice diagonale $t \times t$ matrice unitaire $t \times d$

matrice unitaire $n \times d$ matrice diagonale $n \times d$ matrice unitaire $d \times d$

Comment choisir \tilde{U}_t et \tilde{V}_t^T ?

Latent Semantic Analysis (1/2)



Décomposition en Valeurs Singulières (SVD) :

factorisation matricielle

$$A = U \times S \times V^T$$

$n \times d$ $n \times n$ $n \times d$ $d \times d$
 matrice unitaire matrice diagonale
 (valeurs singulières de A)
 ps : on les range par ordre décroissant

SVD tronquée :

$$A \approx U_t \times S_t \times V_t^T$$

$n \times d$ $n \times t$ $t \times t$ $t \times d$
 $\tilde{U}_t = U_t \times S_t$ $\tilde{V}_t^T = V_t^T$

Comment choisir \tilde{U}_t et \tilde{V}_t^T ?

$$A \approx \tilde{U}_t \times \tilde{S}_t \times \tilde{V}_t^T$$

$n \times d$ $n \times t$ $t \times t$ $t \times d$
 $\tilde{U}_t = U_t \times \sqrt{S_t}$ $\tilde{V}_t^T = V_t^T \times \sqrt{S_t}$

Latent Semantic Analysis (2/2)

Remarques :

- approche **algèbre linéaire**
- évaluer la similarité entre les **documents** (similarité cosinus)
- évaluer la similarité entre les **mots** (similarité cosinus)

Désavantages :

- vecteurs difficilement **interprétable**
- besoin d'un **grand ensemble de docs et de vocabulaires** pour obtenir une bonne précision

Latent Semantic Analysis (2/2)

Remarques :

- approche **algèbre linéaire**
- évaluer la similarité entre les **documents** (similarité cosinus)
- évaluer la similarité entre les **mots** (similarité cosinus)

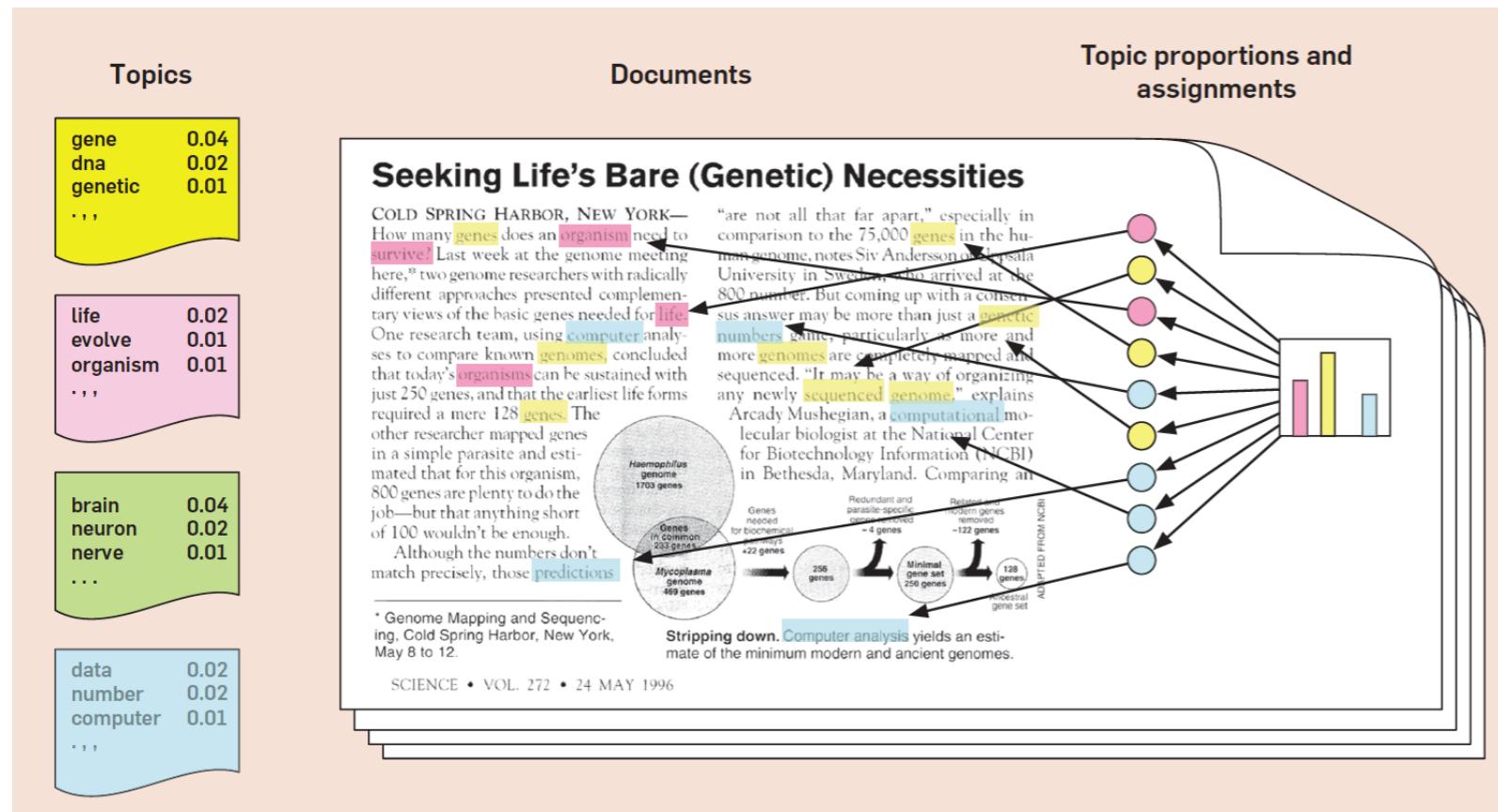
Désavantages :

- vecteurs difficilement **interprétable**
- besoin d'un **grand ensemble de docs et de vocabulaires** pour obtenir une bonne précision



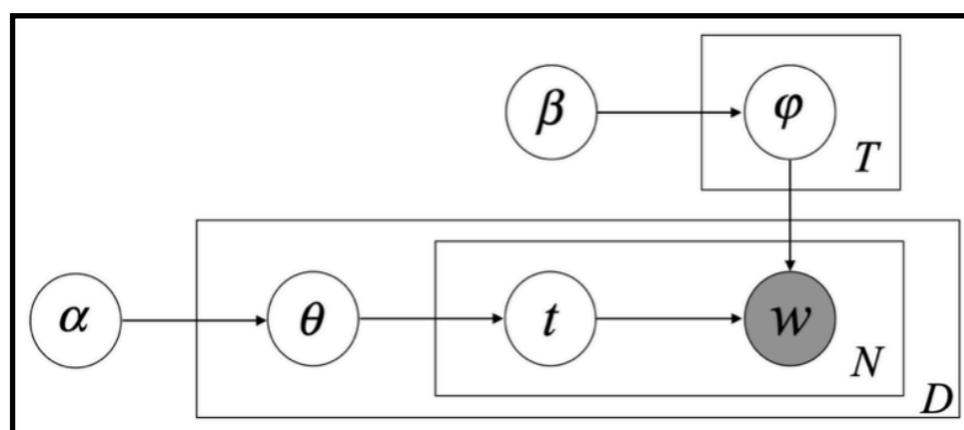
Solution : une **version probabiliste** avec LDA

Latent Dirichlet Allocation : définition



comment est généré un document d d'après le modèle LDA ?

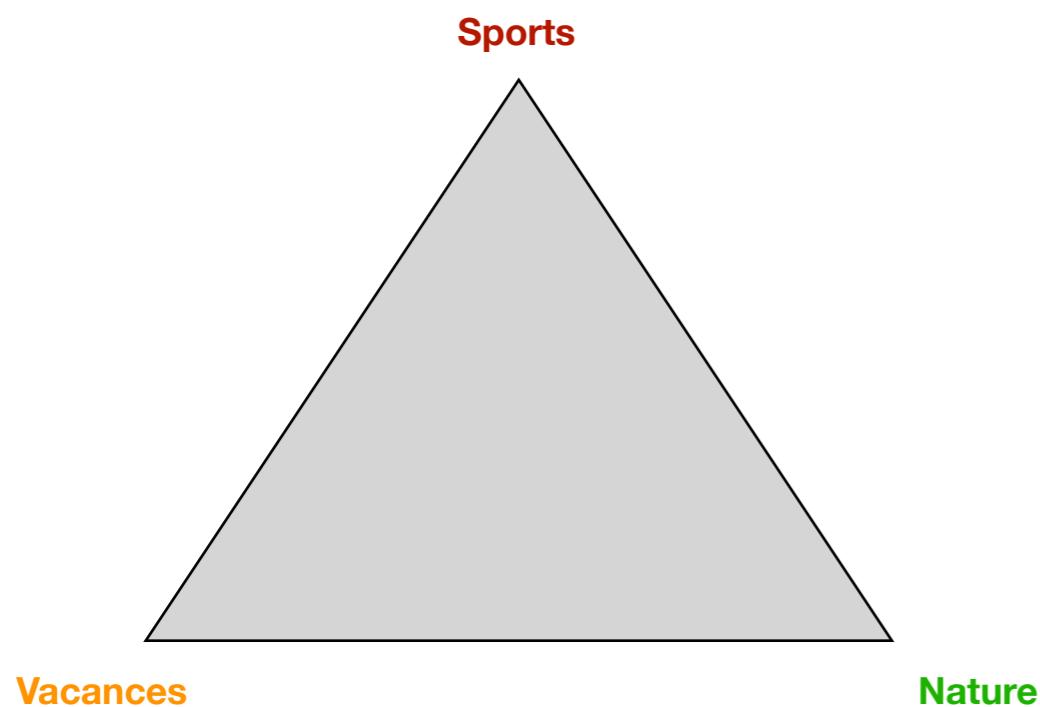
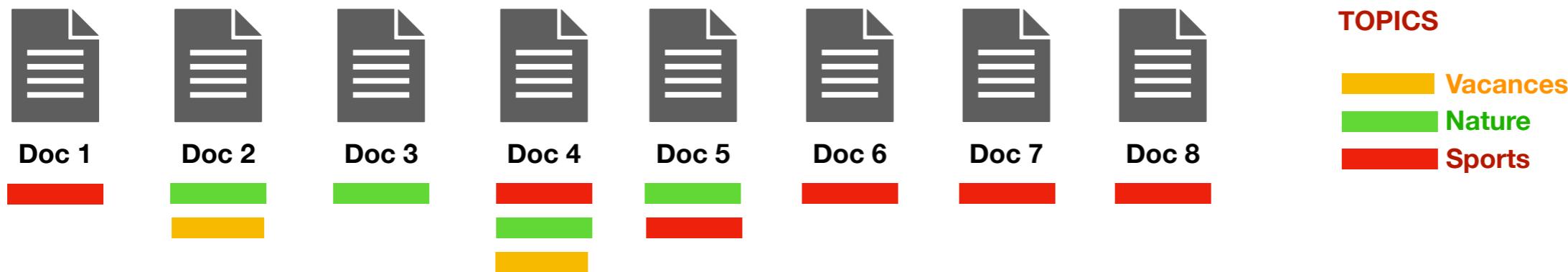
un document $d = (w_1, \dots, w_W)$ est généré de la manière suivante : pour tout $w \in d$



- $\theta_d \sim Dir(\alpha)$, $\phi_t \sim Dir(\beta)$
- $t \sim Multinomial(\theta_{\cdot d})$, $w \sim Multinomial(\phi_{\cdot t})$

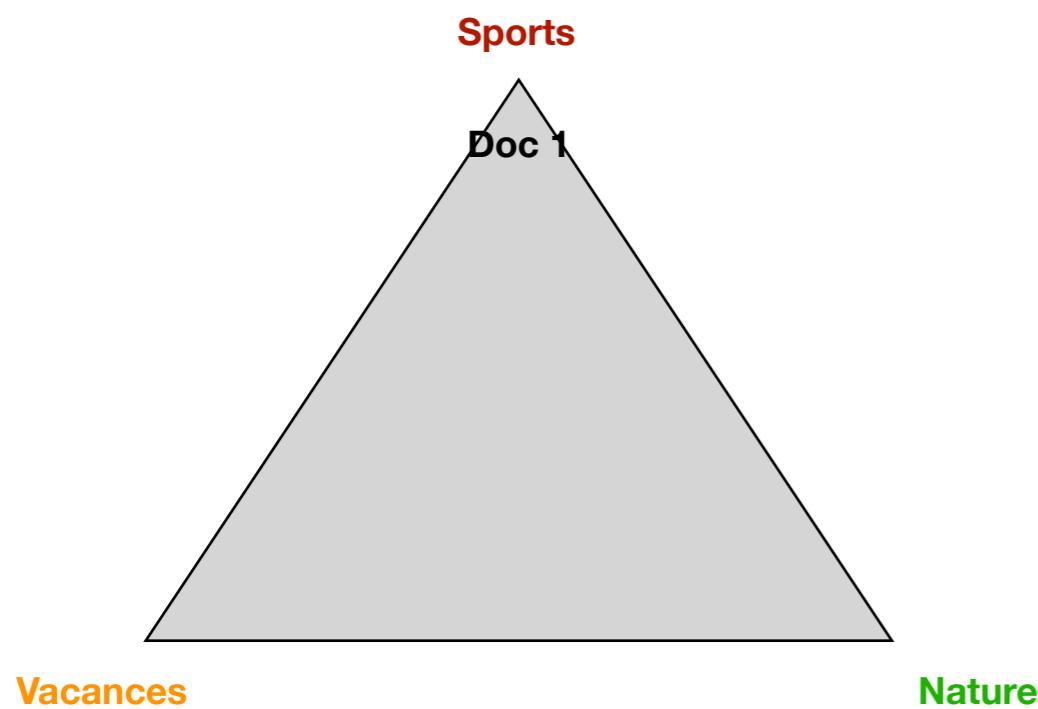
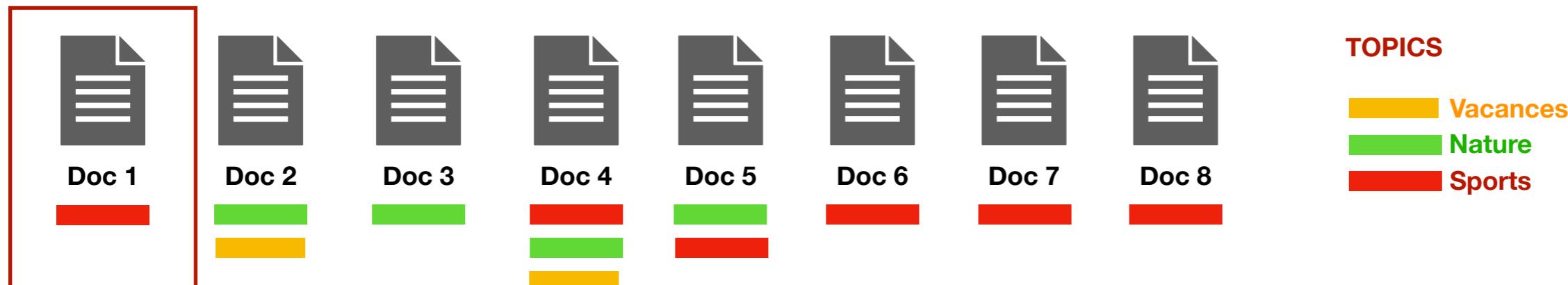
Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :



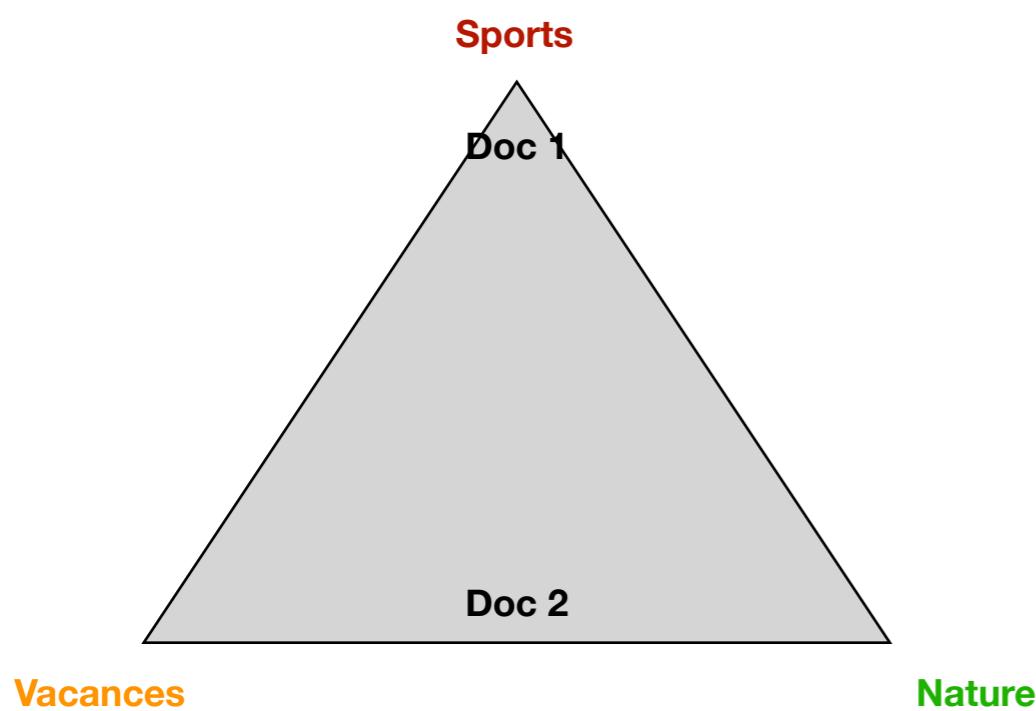
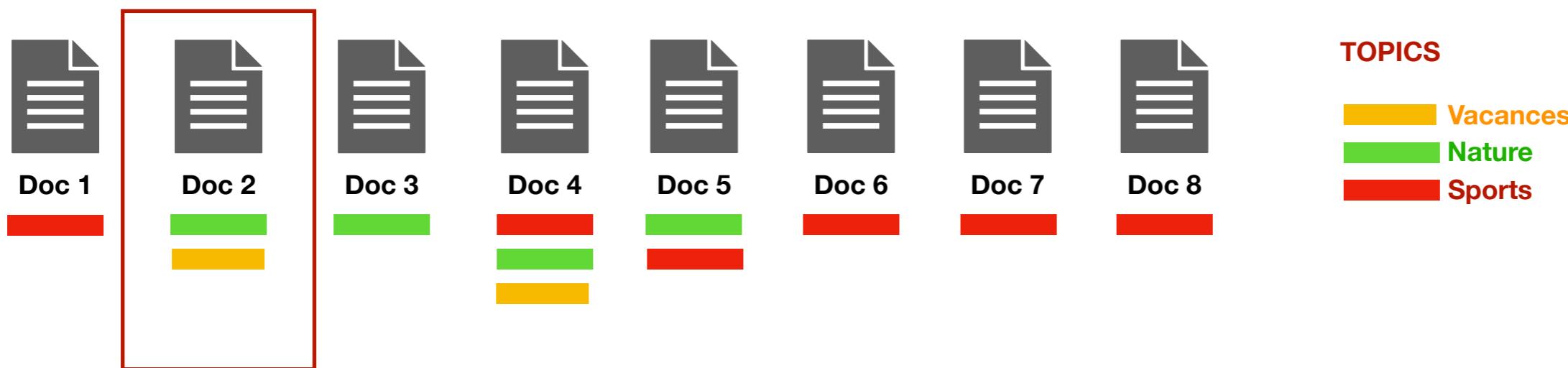
Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :



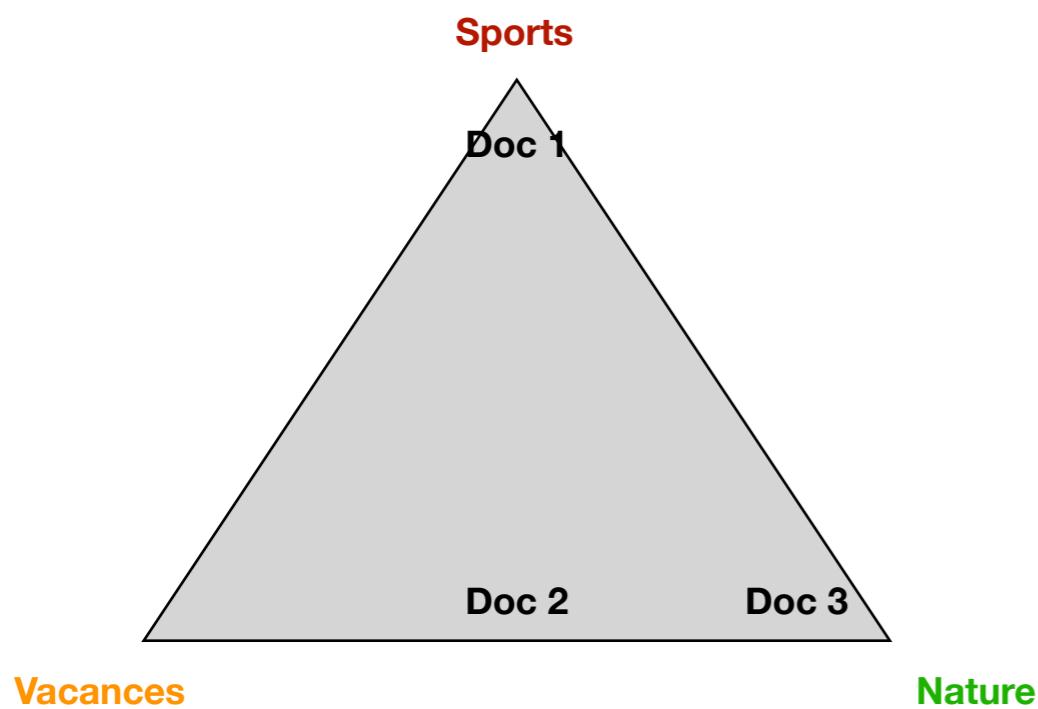
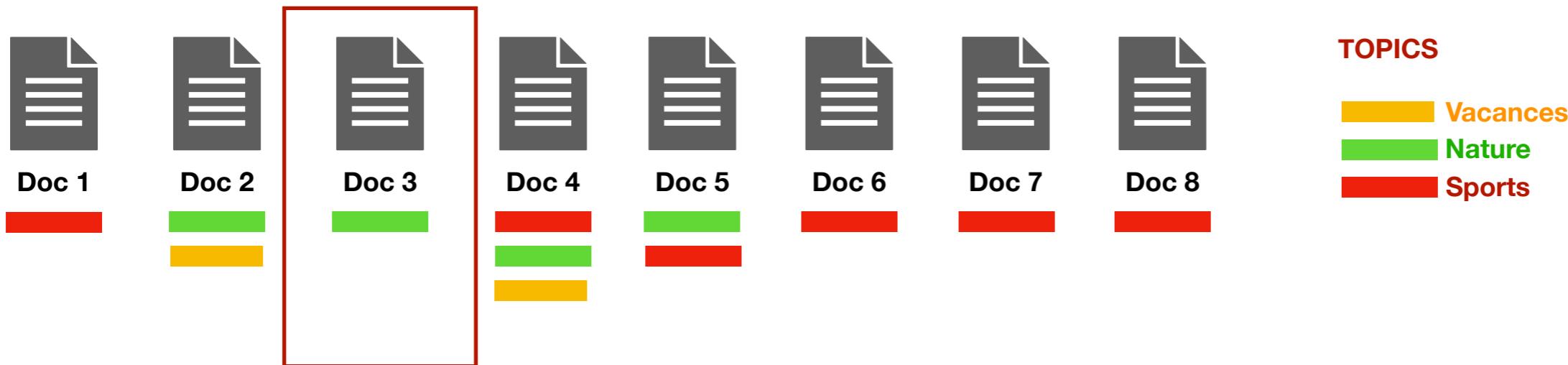
Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :



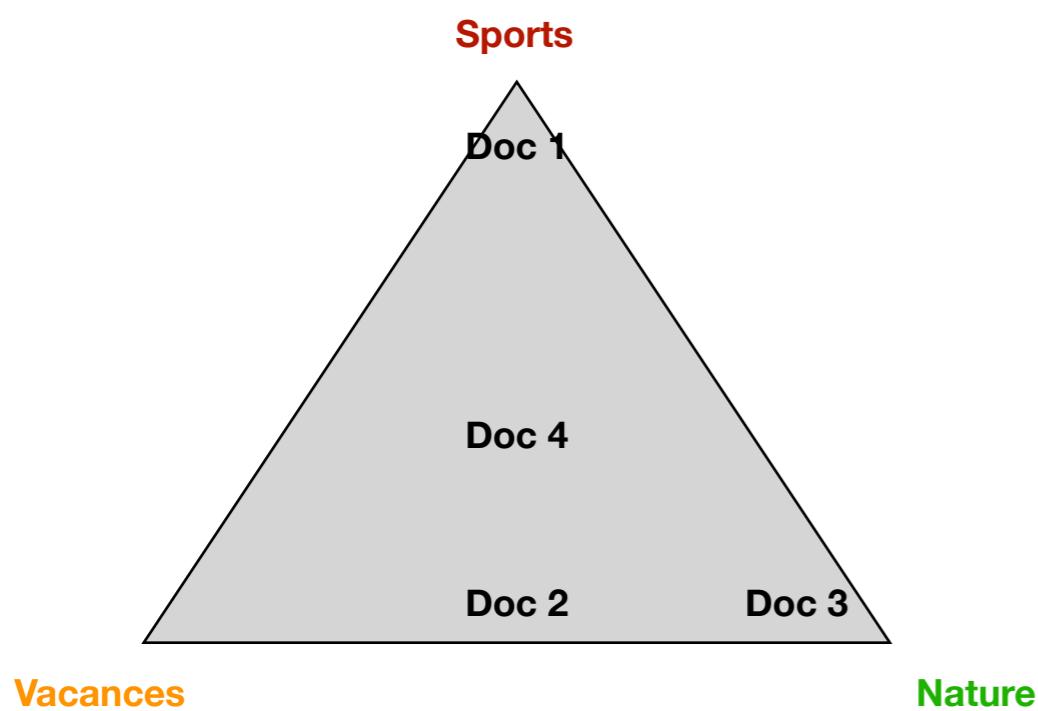
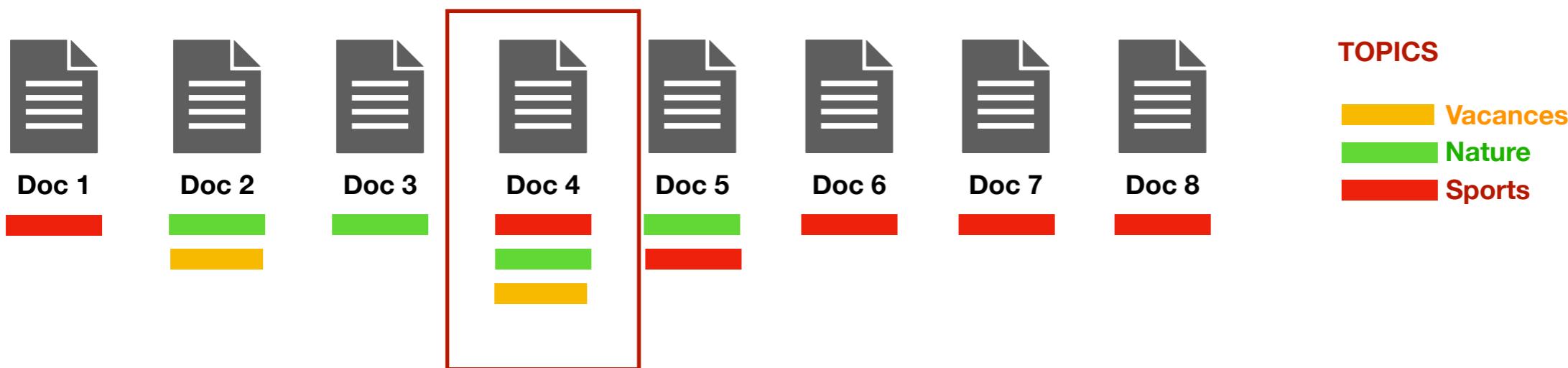
Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :



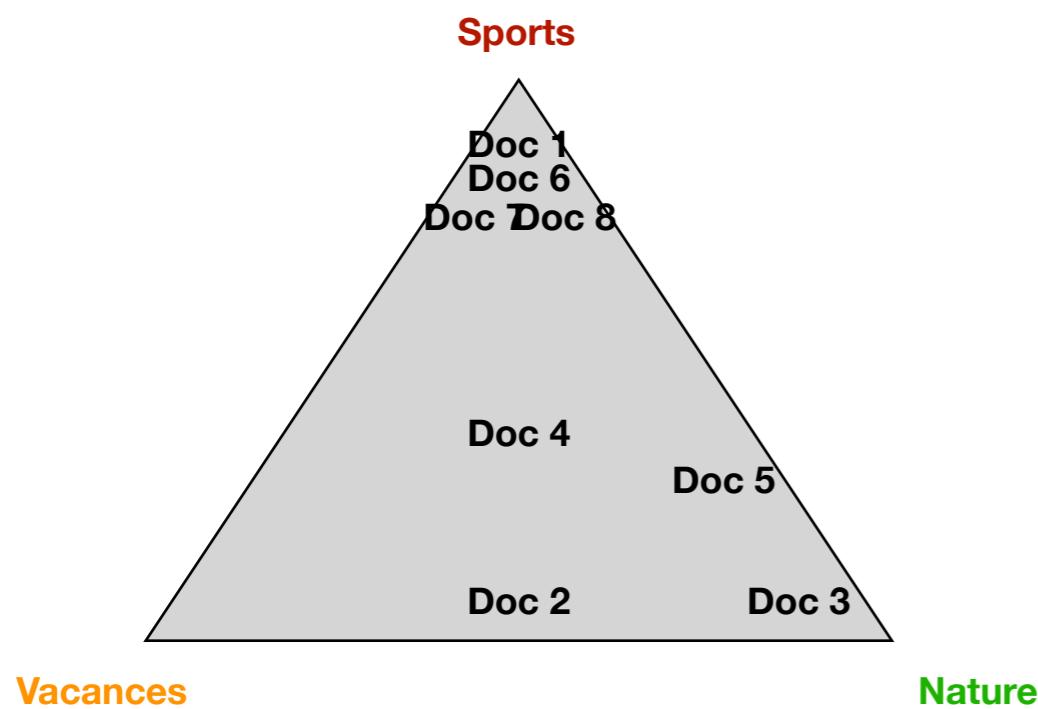
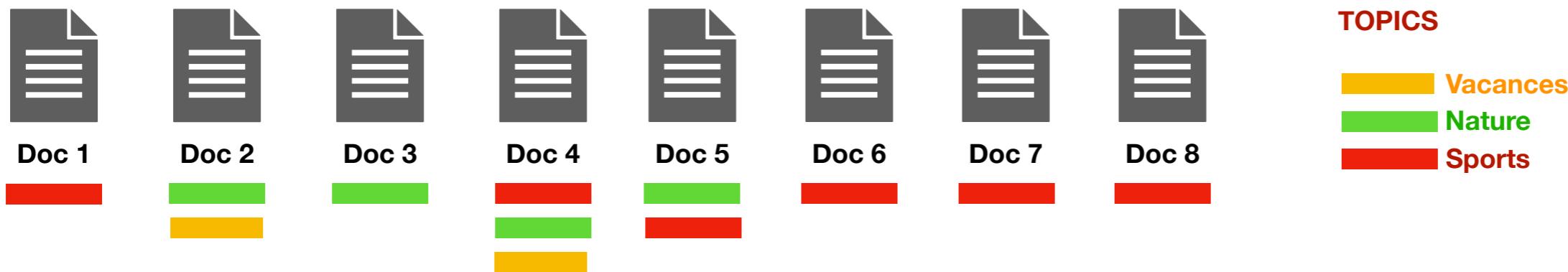
Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :



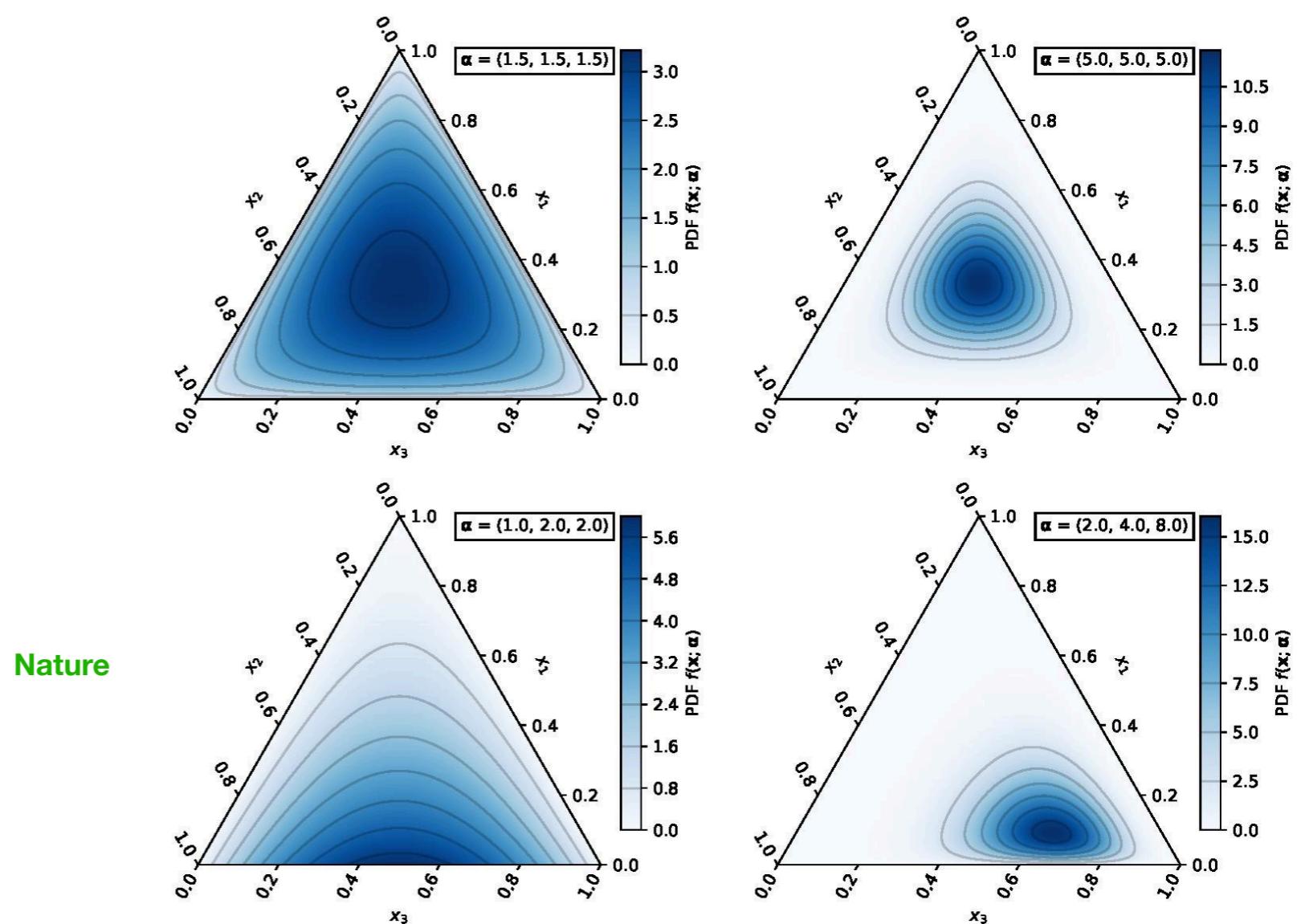
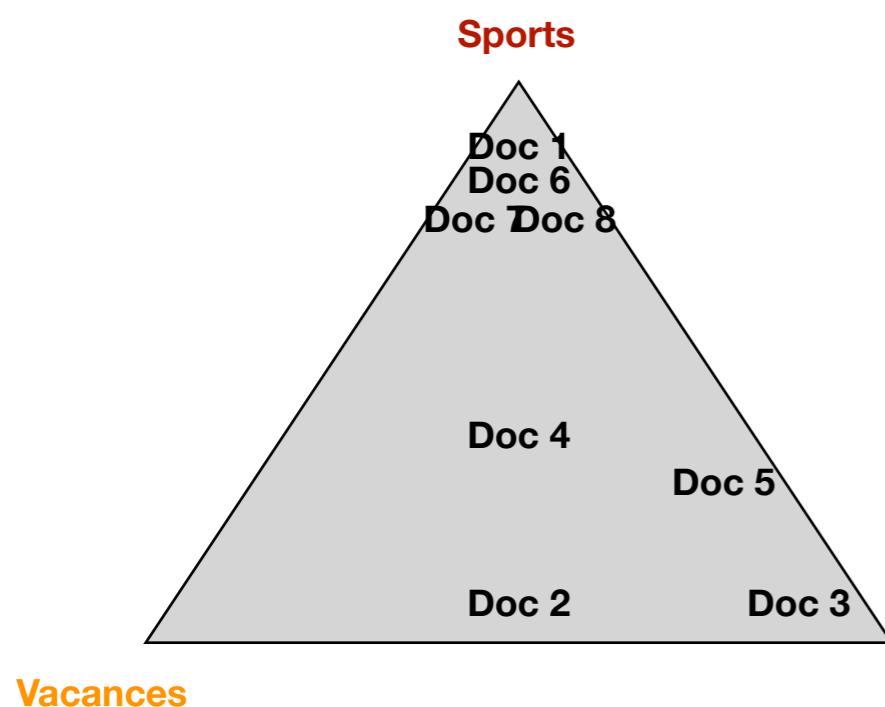
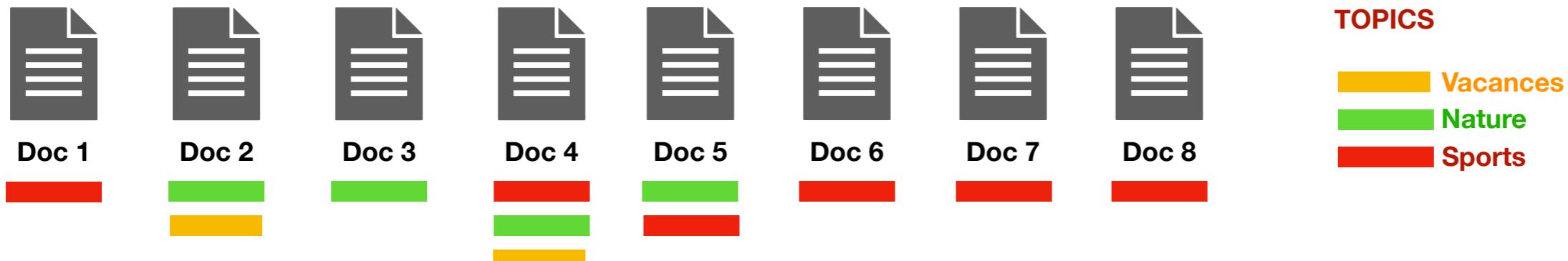
Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :

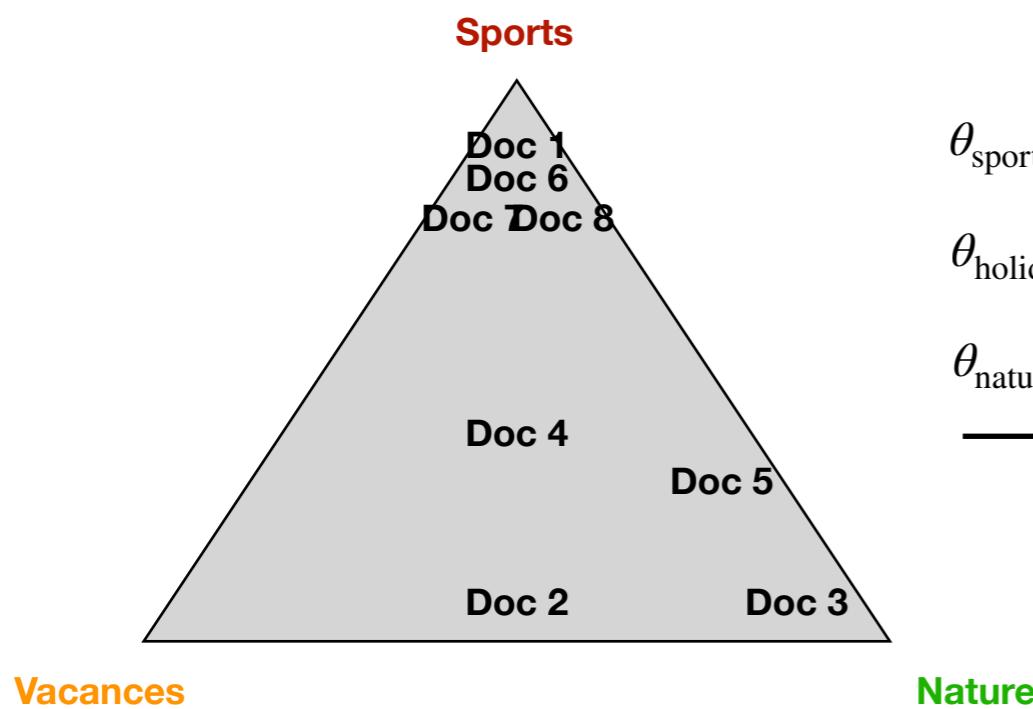


Latent Dirichlet Allocation : zoom sur Dirichlet

Dirichlet distribution (dimension 3) :



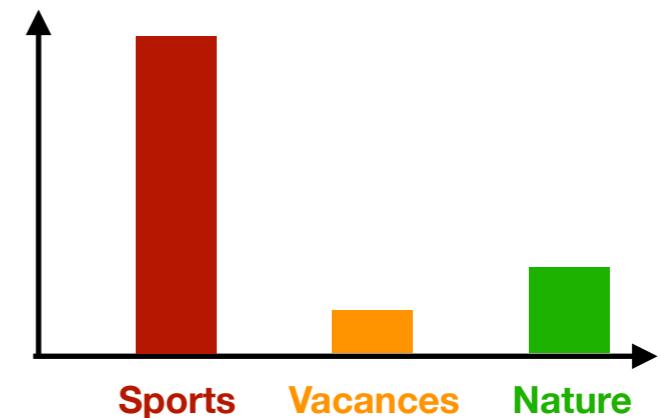
Latent Dirichlet Allocation : zoom sur Dirichlet



$$\theta_{\text{sports}} = P(\text{sports} | \alpha) = 0.7$$

$$\theta_{\text{holiday}} = P(\text{vacances} | \alpha) = 0.1$$

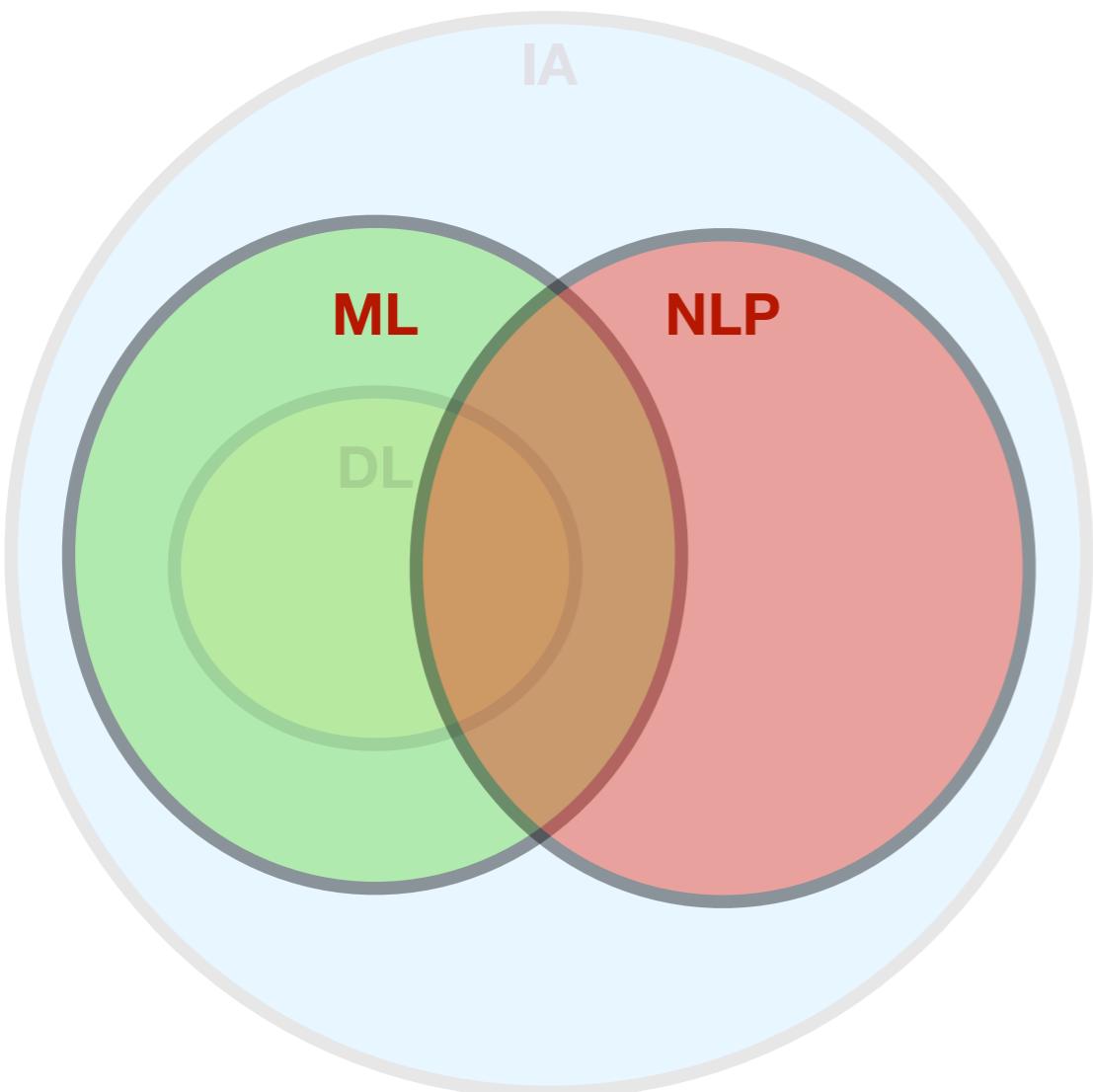
$$\theta_{\text{nature}} = P(\text{nature} | \alpha) = 0.2$$



Dirichlet distribution
« distribution de distribution »

Multinomial distribution

Résumé



Topic modeling

Topic models :

- LSA
- LDA

TP sur le topic modeling :
<https://curiousml.github.io/>