OpenClassrooms

Soutenance: Projet 7

Topic Modeling - Evaluation des Performances

Edward Levavasseur

Updated: 2021/06/24

Problématique

La classification de textes se fait de manière supervisée (*topic classification*) et de manière non-supervisée (*topic modeling*).

Les méthodes non-supervisées sont celles qui sont le plus susceptible de générer des résultats absurdes, et en même temps celles qui sont le plus difficile à évaluer.

Problématique

La classification de textes se fait de manière supervisée (*topic classification*) et de manière non-supervisée (*topic modeling*).

Les méthodes non-supervisées sont celles qui sont le plus susceptible de générer des résultats absurdes, et en même temps celles qui sont le plus difficile à évaluer.

Comment évaluer les performances des différents modèles non-supervisés de manière systématique?

Problématique

La classification de textes se fait de manière supervisée (*topic classification*) et de manière non-supervisée (*topic modeling*).

Les méthodes non-supervisées sont celles qui sont le plus susceptible de générer des résultats absurdes, et en même temps celles qui sont le plus difficile à évaluer.

Comment évaluer les performances des différents modèles non-supervisés de manière systématique?

- Score de Cohérence

Plan

1. Présentation des données
2. Méthodes
3. Evaluation
4. Nouveaux Algorithmes
5. Conclusion

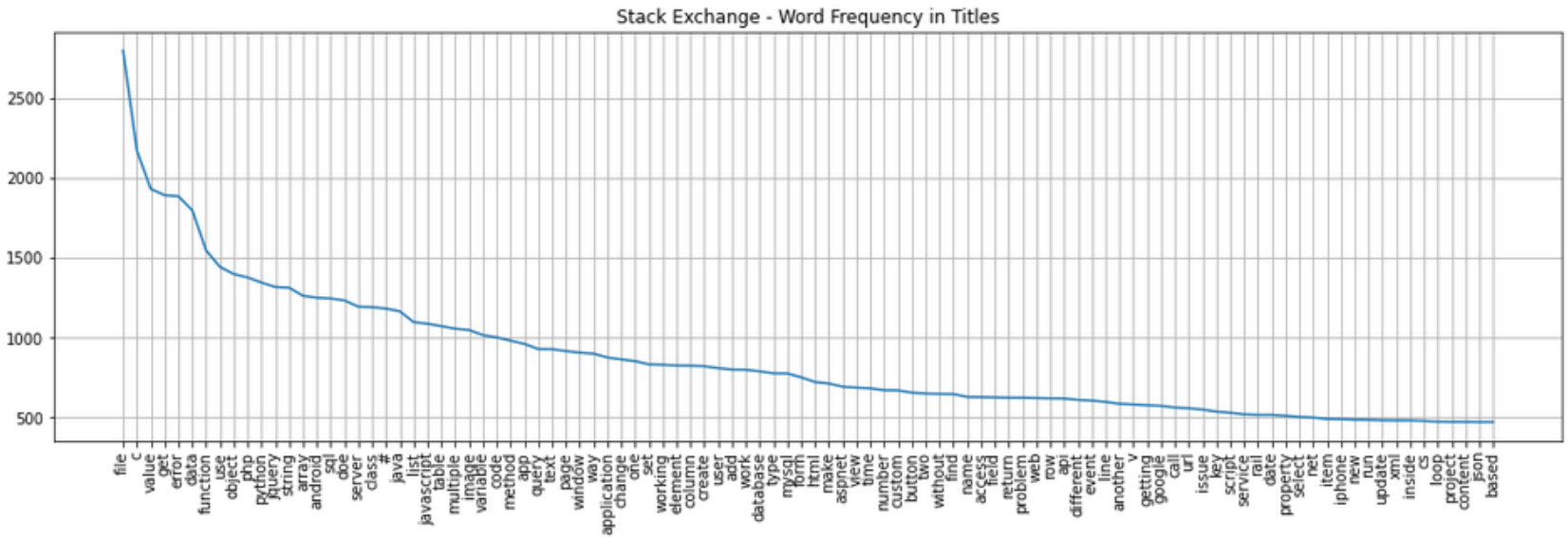
Présentation des données

Questions Stack Overflow

- 50.000 questions de Stack Overflow
  - Titre
  - Question
- Tonkeize
- Lemmatize
- Suppression des Stop Words

Bag of Words

- Je garde les 500 mots les plus utilisés dans les titres



- Bag Of Words avec seulement les features

Méthodes

Word Embedding: *CountVectorizer* (1/2)

- Représente chaque texte dans un espace vectoriel

	access	accessing	action	activity	add	...	write	writing	wrong	xcode	xml
10	0	0	0	0	0	...	1	0	0	0	0
11	0	0	0	0	0	...	0	0	0	0	0
12	3	2	0	0	0	...	0	0	0	0	2
13	0	0	0	0	0	...	0	0	0	0	0
14	0	0	0	0	1	...	0	0	1	0	0
15	0	0	0	0	0	...	0	0	1	0	0
16	0	0	0	0	0	...	0	0	0	0	0
17	0	0	0	0	0	...	0	0	0	0	0
18	0	0	0	0	0	...	1	0	0	0	0
19	0	0	0	0	0	...	0	0	0	0	0

- Compte le nombre de fois que chaque feature est utilisé dans chaque texte

Word Embedding: *TF-IDF* (2/2)

- Représente chaque texte dans un espace vectoriel

	access	accessing	action	activity	...	writing	wrong	xcode	xml
10	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000
11	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000
12	0.204243	0.200478	0.0	0.0	...	0.0	0.000000	0.0	0.16123
13	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000
14	0.000000	0.000000	0.0	0.0	...	0.0	0.176641	0.0	0.00000
15	0.000000	0.000000	0.0	0.0	...	0.0	0.105907	0.0	0.00000
16	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000
17	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000
18	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000
19	0.000000	0.000000	0.0	0.0	...	0.0	0.000000	0.0	0.00000

- Compte:
  - $P(terme \mid document) \times log\left(\frac{1}{P(documents\ avec\ terme \mid tous\ documents)}\right)$

Production de Topics : *LDA* (1/3)

- 2 types de méthodes (Kherwa and Bansal (2020)):
  - Probabilistes (LDA)
  - Non-probabilistes (NNMF, LSA)

Production de Topics : *LDA* (1/3)

- 2 types de méthodes (Kherwa and Bansal (2020)):
  - Probabilistes (LDA)
  - Non-probabilistes (NNMF, LSA)

- Hypothèses du modèle:
  - Probabilité pour un document d'appartenir à un topic suit une loi de dirichlet
  - Probabilité pour un mot d'appartenir à un document suit une aussi loi de dirichlet

Production de Topics : *LDA* (1/3)

- 2 types de méthodes (Kherwa and Bansal (2020)):
  - Probabilistes (LDA)
  - Non-probabilistes (NNMF, LSA)
- Hypothèses du modèle:
  - Probabilité pour un document d'appartenir à un topic suit une loi de dirichlet
  - Probabilité pour un mot d'appartenir à un document suit une aussi loi de dirichlet
- Optimization des paramêtres des lois de dirichlet
- Représente chaque texte dans un simplex de probabilité d'appartenir à chaque topic.

- Decomposition de la matrice du word embedding:

$$\underset{n\times m}{X} = \underset{n\times k}{W} \times \underset{k\times m}{H}$$

- W est la matrice avec 50.000 textes dans un espace à k Topics

- Decomposition de la matrice du word embedding:

$$\underset{n\times m}{X} = \underset{n\times k}{W} \times \underset{k\times m}{H}$$

- W est la matrice avec 50.000 textes dans un espace à k Topics
- W et H sont trouvés en résolvant:

$$\underset{W,H}{min} \; ||X - WH||^2 \quad s.c \; W \geq 0, \; H \geq 0$$

- Solution trouvée par descente de gradient

- Decomposition de la matrice du word embedding:

$$\underset{n\times m}{X} = \underset{n\times k}{U} \times \underset{k\times k}{\Sigma} \times \underset{k\times m}{V}$$

- U est la matrice avec 50.000 textes dans un espace à k Topics

- Decomposition de la matrice du word embedding:

$$\underset{n\times m}{X} = \underset{n\times k}{U} \times \underset{k\times k}{\Sigma} \times \underset{k\times m}{V}$$

- U est la matrice avec 50.000 textes dans un espace à k Topics
- U, Σ et V sont trouvés en :
  - Calculant les valeurs propres de X
  - Déterminant les vecteurs propres associés
- Solution Analystique existe

Evaluation

Score de Cohérence

- Après identification de k topics
- Evaluation de chaque topic: score de cohérence

Score de Cohérence

- Après identification de k topics
- Evaluation de chaque topic: score de cohérence
  - Similarité entre chaque paire de mots importants du topic:

$$UMass(w_i, w_j) = log\left(\frac{D(w_i, w_j)+\epsilon}{D(w_i)}\right)$$

Score de Cohérence

- Après identification de k topics
- Evaluation de chaque topic: score de cohérence
  - Similarité entre chaque paire de mots importants du topic:

$$UMass(w_i, w_j) = log\left(\frac{D(w_i, w_j)+\epsilon}{D(w_i)}\right)$$

  - Somme des UMASS de chaque paire de mot:

$$Coherence(T_i) = \sum_{i=1}^{10} \sum_{j=1}^{10} UMass(w_i, w_j)$$

Score de Cohérence

- Après identification de k topics
- Evaluation de chaque topic: score de cohérence
  - Similarité entre chaque paire de mots importants du topic:

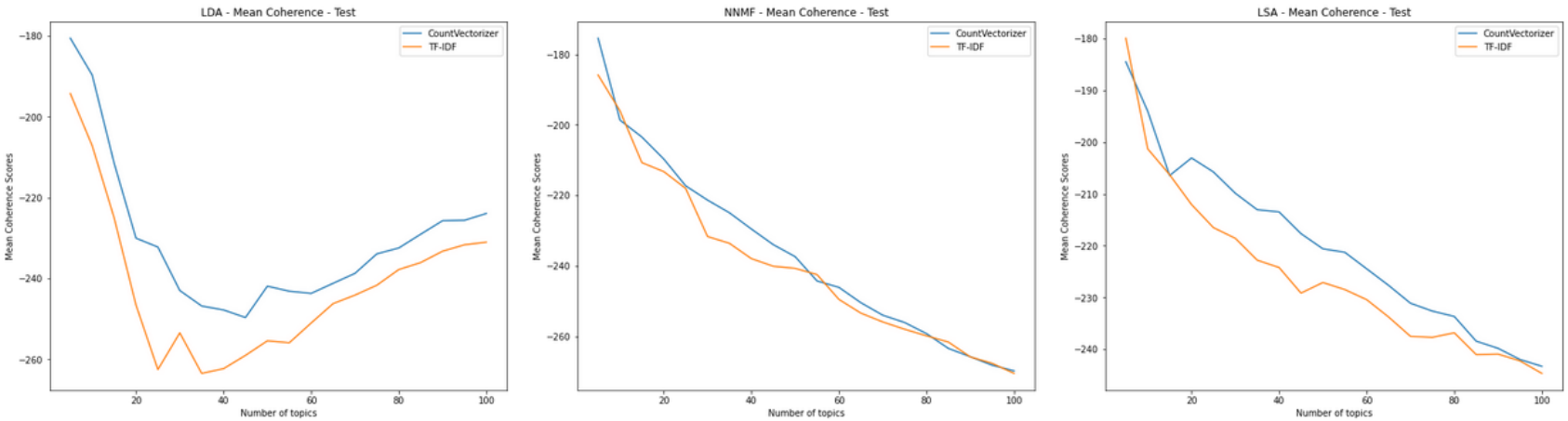
$$UMass(w_i, w_j) = log\left(\frac{D(w_i,w_j)+\epsilon}{D(w_i)}\right)$$

  - Somme des UMASS de chaque paire de mot:

$$Coherence(T_i) = \sum_{i=1}^{10}\sum_{j=1}^{10} UMass(w_i, w_j)$$

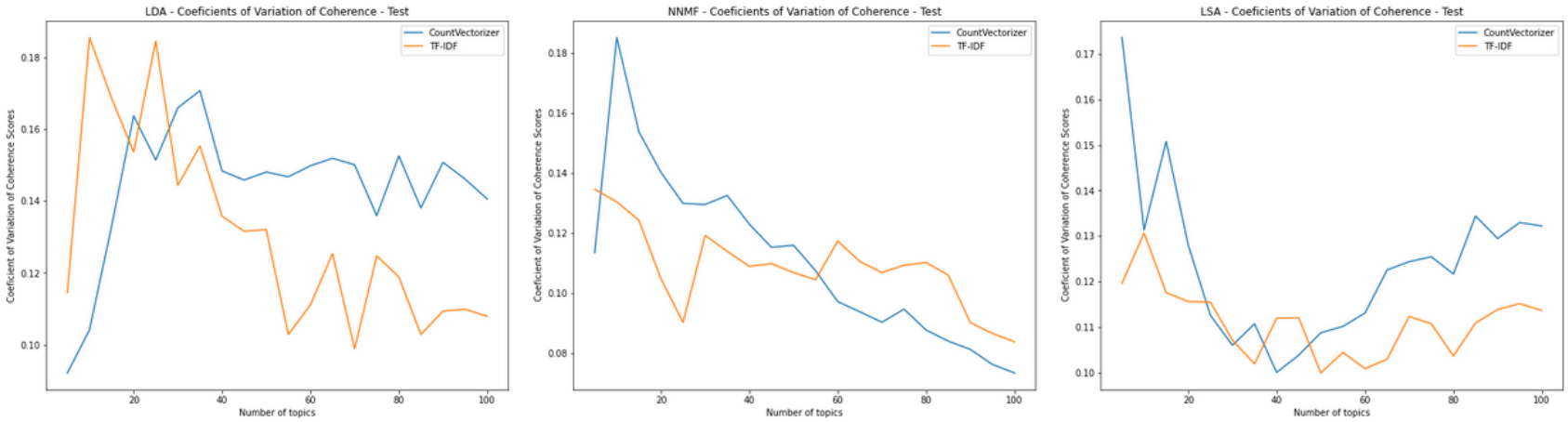
- Cohérence moyenne des k topics
- Coéficient de variation des k topics

TF-IDF vs CountVectorizer (1/2)



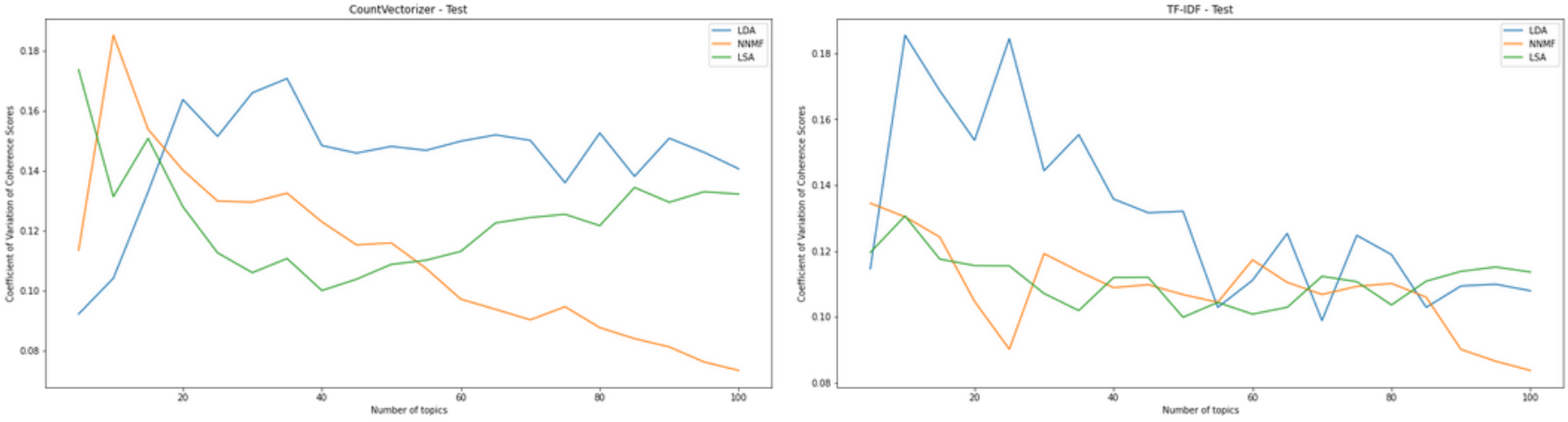
- *CountVectorizer* génère une meilleur cohérence moyenne que *TF-IDF*
  - Pour tout nombre de topics
  - Pour tout modèle de *topic modeling* (LDA, NNMF, LSA)

TF-IDF vs CountVectorizer (2/2)



- *CountVectorizer* et *TF-IDF* génèrent autant d'inégalité dans les cohérences des topics

LDA vs NNMF vs LSA (2/2)



- *LDA* génère globalement une plus grande inégalité dans les cohérences des topics que *NNMF* et *LSA*

Nouveaux Algorithmes

Top2Vec

- Algorithme qui fait tout le travail:
  - Word2Vec:
    - Calcule la probabilité qu'un mot soit suivi d'un autre
    - Calcule la similarité entre des bi-grammes ou n-grammes
  - Doc2Vec:
    - Représente les textes (n-grammes) et les features (mono-gramme) dans un même espace mutli-dimensionnel
  - UMAP: Réduction des dimensions
  - HDBSCAN: Identification de clusters $\Longrightarrow$ Topics
- Résultats:
  - Nombre de Topics: 374
  - Cohérence Moyenne: -364.858
  - Coéficient de variation de la Cohérence: 0.1948

BERT

- Similaire à *Top2Vec*:
  - Word2Vec:
    - Calcule la probabilité qu'un mot soit suivi d'un autre
    - Calcule la similarité entre des bi-grammes ou n-grammes
  - Représentation des textes dans un espace multi-dimensionnel
  - UMAP: Réduction des dimensions
  - HDBSCAN: Identification de clusters $\Longrightarrow$ Topics
  - TF-IDF: Appliqué à l'intérieur des topics
    - Identifie les mots de chaque topic

- Résultats:
  - Nombre de Topics: 594
  - Cohérence Moyenne: -249.961
  - Coéficient de variation de la Cohérence: 0.284

Conclusion

Conclusion (1/2)

- Objectif :
  - Evaluer différentes méthodes de Topic Modeling
- Application du score de cohérence :
  - CountVectorizer vs TF-IDF
  - LDA vs NNMF vs LSA

Conclusion (1/2)

- Objectif :
  - Evaluer différentes méthodes de Topic Modeling
- Application du score de cohérence :
  - CountVectorizer vs TF-IDF
  - LDA vs NNMF vs LSA
- LSA performe mieux que NNMF:
  - 2 à 3 fois plus rapide
  - Cohérence moyenne plus élevée pour tout nombre de topics
- LSA performe mieux que LDA:
  - 2 à 3 fois plus rapide
  - Cohérence moyenne plus élevée si Topics < 80

Conclusion (2/2)

- Nouveaux Algorithmes :
  - Top2Vec
  - Bert

Conclusion (2/2)

- Nouveaux Algorithmes :
  - Top2Vec
  - Bert
- Avantage :
  - fait tout le travail
- Inconvénient :
  - ne permet pas de choisir le nombre de Topics

Conclusion (2/2)

- Nouveaux Algorithmes :
  - Top2Vec
  - Bert
- Avantage :
  - fait tout le travail
- Inconvénient :
  - ne permet pas de choisir le nombre de Topics
- Top2Vec :
  - Assez mauvaise cohérence dans l'ensemble
- BERT:
  - Certains topics avec très haute cohérence, d'autres un peu moins