# Recommandation personnalisée et Machine Learning

Adrien Todeschini
Chief Data Scientist

scorelab

1er Févr. 2018
1ère Journée Entreprises - Aquitaine
Optimisation, Recherche Opérationnelle, Intelligence Artificielle

scorelab

Présentation générale

# ML4U

Recommandation personnalisée pour les conférences de Machine Learning

# Scorelab

# Scorelab

- Startup bordelaise créée en avril 2016.
- Spécialisée en Data Science et Machine Learning.
- Des produits innovants.
- Des prestations sur mesure.

# L'équipe

scorelab

**Guillaume Forcade**

CEO & Co-founder

**Jean-Baptiste Pautrizel**

PhD Physics & Co-founder

**Adrien Todeschini**

PhD Stat. ML & Data Scientist

**Louis Amon**

Back-end Engineer & CTO

**Kevin Baudin**

MSc. Stat. & Data Scientist

**Wassek Al Chahid**

Full Stack Dev.

**Yang Zheng**

Stagiaire Data Scientist

**Ngoc-Phi Tran**

Stagiaire Data Scientist

# Nos partenaires

scorelab

Incubateur Régional d'Aquitaine
Création d'entreprises innovantes

Région Nouvelle-Aquitaine

bpifrance

La French Tech

AQUITI Gestion

JEI jeune entreprise innovante

Crédit d'impôt recherche – Agrément CIR

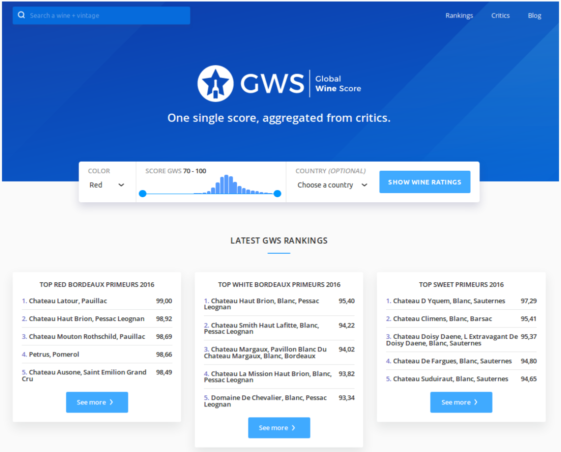
AMIES – Les maths vecteur d'innovation

Institut de Mathématiques de Bordeaux

Institut Européen de Chimie et Biologie – IECB

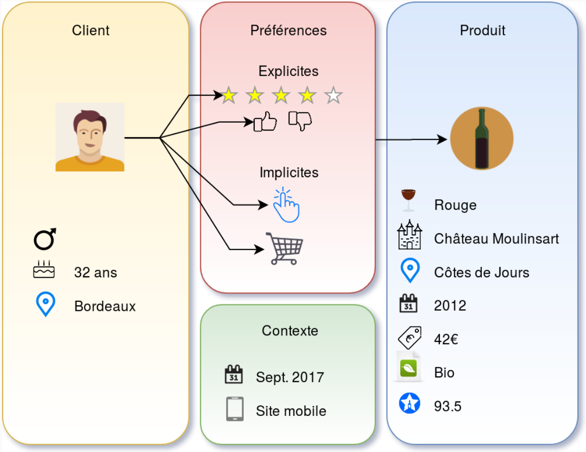
# globalwinescore.com

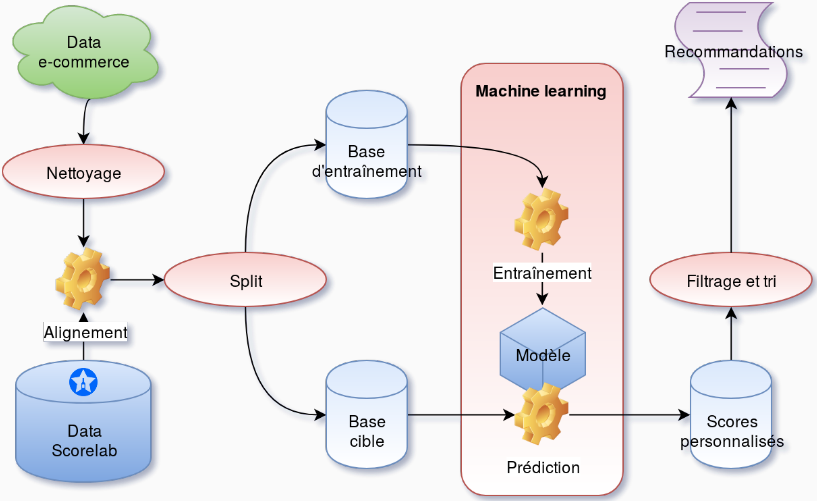
scorelab



scorelab

## Recommandation personnalisée pour les sites d'e-commerce de vin.




Data
e-commerce
Nettoyage
Alignement
Data
Scorelab
Split
Base
d'entraînement
Base
cible
Machine learning
Entraînement
Modèle
Prédiction
Scores
personnalisés
Filtrage et tri
Recommandations

# Prestations



Notre expertise en Data Science & Machine Learning au service de vos solutions innovantes.

## Projets en cours

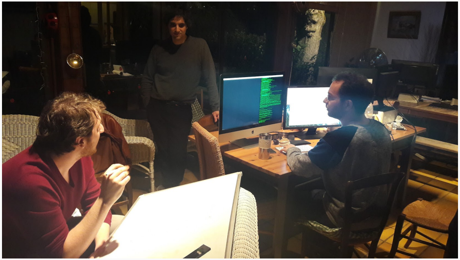
- Estimateur de prix immobillier.
- Credit scoring dans le courtage immobilier.
- Moteur de recommandation pour la parapharmacie.
- Cacao trading system.

# Side projects

scorelab

## ML4U

Recommandation personnalisée pour les conférences de Machine Learning



## `noodle.scorelab.io`

Doodle for notation

# ML4U

# Machine Learning hype

scorelab

- Un domaine de recherche très tendance.
- International Conference on Machine Learning (ICML, juil. 2017) :
    - 433 papiers acceptés / 1701 soumissions (25.5%).
    - ~ 3000 participants.
- Neural Information Processing Systems (NIPS, déc. 2017) :
    - 679 papiers acceptés / 3240 soumissions (20.9%).
    - ~ 6000 participants.

http://www.aiindex.org/2017-report.pdf

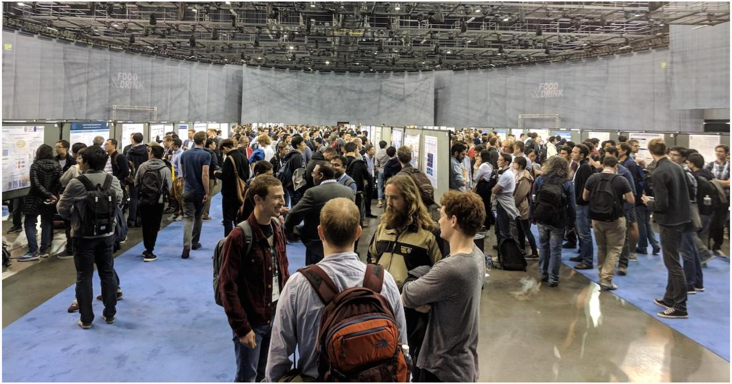
# NIPS conference



NIPS 2017, Long Beach, USA

# NIPS conference

scorelab

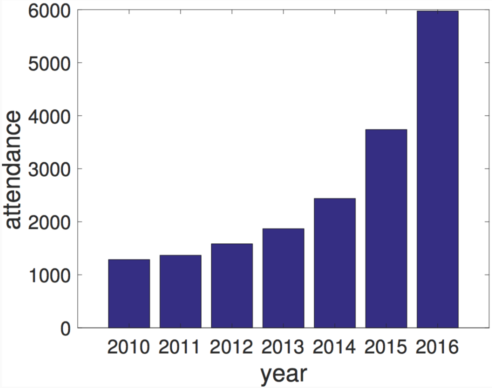


NIPS 2017, poster sessions

Jordan Novet, CNBC

# NIPS conference

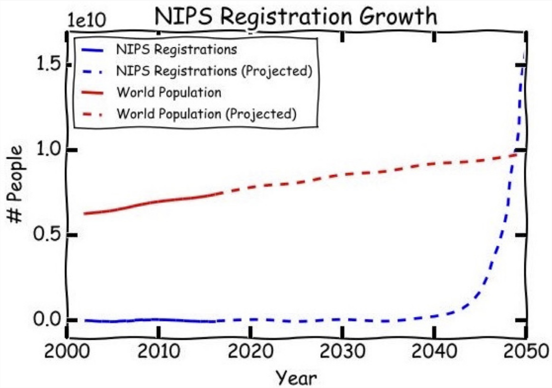
scorelab




NIPS registrations

http://answeron.com/machine-learning-bubble

# NIPS conference



# NIPS conference : articles acceptés

scorelab

## Industriels

1. Google : 60 (8.8%)

4. Microsoft : 40 (5.9%)

7. Deepmind : 31 (4.6%)

13. IBM : 16 (2.4%)

25. Facebook : 11 (1.6%)

32. Tencent AI Lab : 9 (1.3%)

34. OpenAI : 8 (1.2%)

34. Adobe : 8 (1.2%)

## Académiques

2. Carnegie Mellon University : 48 (7.1%)

3. Massachusetts Institute of Technology : 43 (6.3%)

5. Stanford University : 39 (5.7%)

6. University of California, Berkeley : 35 (5.2%)

8. University of Oxford : 22 (3.2%)

9. University of Illinois at Urbana-Champaign : 20 (2.9%)

10. Georgia Institute of Technology : 18 (2.7%)

11. Princeton : 17 (2.5%)

11. ETH Zurich : 17 (2.5%)

14. Inria : 15 (2.2%)

14. Harvard University : 15 (2.2%)

https://unsupervisedmethods.com/nips-accepted-papers-stats-26f124843aa0

# Le projet ML4U

scorelab

- Application de recommandation personnalisée pour les conférences de Machine Learning.
- Open source, collaboratif et expérimental.
- Equipe de chercheurs et ingénieurs internationale (Oxford, Warwick, Montréal, Bordeaux).
- Version Bêta pour NIPS 2017 : `nips17.ml`.

# Principaux collaborateurs

scorelab

**Yee Whye Teh**

Prof. @ Univ. Oxford

**Laurent Charlin**

Assist. Prof. @ HEC Montreal

**Sebastien Doncker**

CEO @ Snark Factory

**Valerio Perrone**

PhD student @ Univ. Warwick

**Huoai Phuoc Truong**

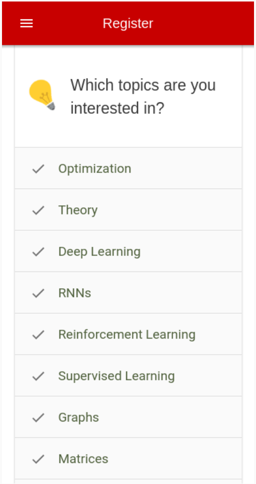
Soft. Engineer @ Google

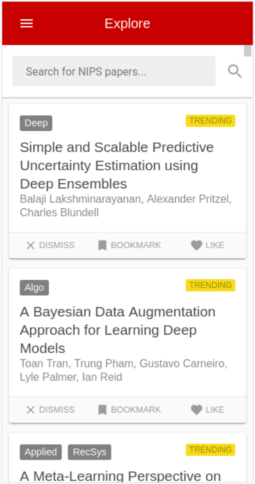
**Adrien Todeschini**

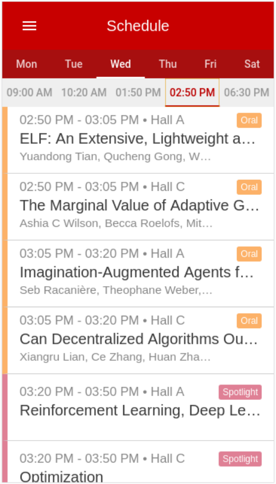
Data Scientist @ Scorelab

# nips17.ml

scorelab



scorelab

Basé sur le modèle *Collaborative Topic Regression* [Wang and Blei, 2011] qui permet de compenser le *cold-start* en combinant :

- *Topic modeling* : *Latent Dirichlet Allocation* [Blei et al., 2003, LDA].
- Filtrage collaboratif : *Probabilistic Matrix Factorization* [Mnih and Salakhutdinov, 2008, PMF].

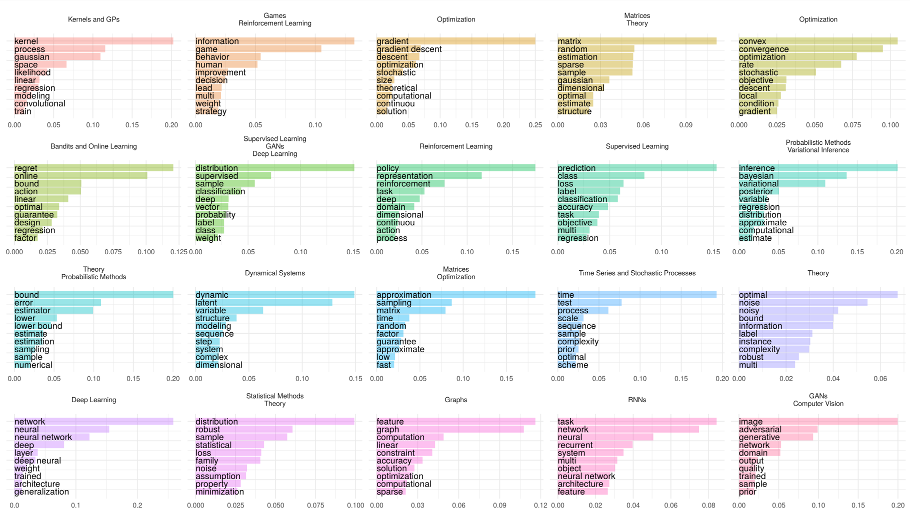
A inspiré le moteur de recommandation du New York Times.

`http ://open.blogs.nytimes.com/2015/08/11/building-the-next-new-york-times-recommendation-engine`

- Documents : titres + résumés d'articles.
- Vocabulaire : uni- et bi-grammes, *stopwords* retirés (mots trop ou peu fréquents).
- *Bag-of-words* : chaque document est un ensemble non ordonné de termes appartenant au vocabulaire.
- On suppose l'existence de $\boldsymbol{K}$ topics latents $(\boldsymbol{\beta_1, \ldots, \beta_K})$.
- Chaque topic $\boldsymbol{k}$ est une distribution sur le vocabulaire.
- Chaque document $\boldsymbol{j}$ est un mélange des topics.

# *Latent Dirichlet Allocation*

scorelab




20 topics labellisés, NIPS 2017

Pour chaque document $j$ de taille $m_j$ ,

- Proportions des topics :

$$\theta_j \sim \textbf{Dirichlet}\left(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\right)$$

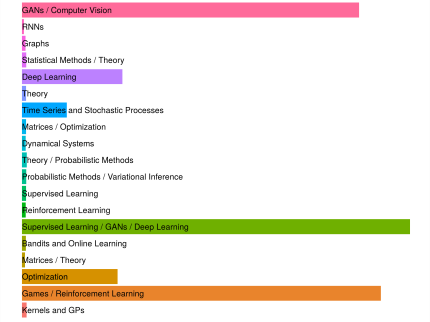
- Pour chaque terme $w_{j\ell}$ avec $\ell = 1, \ldots, m_j$,
  - Choix du topic :

$$z_{j\ell} \sim \textbf{Discrete}\left(\theta_j\right).$$

  - Choix du terme au sein du topic :

$$w_{j\ell} \sim \textbf{Discrete}\left(\beta_{z_{j\ell}}\right).$$

Improved Training of Wasserstein GANs : Generative Adversarial Networks (GANs) are powerful generative models, but suffer from training instability. The recently proposed Wasserstein GAN (WGAN) makes significant progress toward stable training of GANs, but can still generate low-quality samples or fail to converge in some settings. We find that these training failures are often due to the use of weight clipping in WGAN to enforce a Lipschitz constraint on the critic, which can lead to pathological behavior. We propose an alternative method for enforcing the Lipschitz constraint : instead of clipping weights, penalize the norm of the gradient of the critic with respect to its input. Our proposed method converges faster and generates higher-quality samples than WGAN with weight clipping. Finally, our method enables very stable GAN training : for the first time, we can train a wide variety of GAN architectures with almost no hyperparameter tuning, including 101-layer ResNets and language models over discrete data. We further demonstrate state of the art inception scores on CIFAR-10, and provide samples on higher resolution datasets.



# Latent Dirichlet Allocation

scorelab

Improved Training of Wasserstein GANs Generative Adversarial Networks (GANs) are powerful generative models, but suffer from training instability. The recently proposed Wasserstein GAN (WGAN) makes significant progress toward stable training of GANs, but can still generate low quality samples or fail to converge in some settings. We find that these training failures are often due to the use of weight clipping in WGAN to enforce a Lipschitz constraint on the critic, which can lead to pathological behavior. We propose an alternative method for enforcing the Lipschitz constraint: instead of clipping weights, penalize the norm of the gradient of the critic with respect to its input. Our proposed method converges faster and generates higher quality samples than WGAN with weight clipping. Finally, our method enables very stable GAN training: for the first time, we can train a wide variety of GAN architectures with almost no hyperparameter tuning, including 101 layer ResNets and language models over discrete data. We further demonstrate state of the art inception scores on CIFAR 10, and provide samples on higher resolution datasets.

topic

- Games / Reinforcement Learning
- Supervised Learning / GANs / Deep Learning
- Deep Learning
- NA
- Optimization
- Time Series and Stochastic Processes
- GANs / Computer Vision

Pour chaque paire user-document $(i, j)$, on suppose les scores

$$s_{ij} \sim \mathcal{N}\left(u_i{}^T v_j,\ c_{ij}^{-1}\right)$$
$$u_i \sim \mathcal{N}\left(\mu_i + \epsilon_i,\ \lambda_u^{-1} I_K\right)$$
$$v_j \sim \mathcal{N}\left(\theta_j + \delta_j,\ \lambda_v^{-1} I_K\right)$$

où $u_i$ et $v_j$ sont des vecteurs de features latentes (topics) de taille $K$.

### *Implicit feedback*

$s_{ij} \in \{0, 1\}$ (*not liked* / *liked*).

On pose $c_{ij} = \begin{cases} a, & \text{si } s_{ij} = 1 \\ b, & \text{si } s_{ij} = 0 \end{cases}$ avec $a > b > 0$.

La confiance accordée à l'observation d'un *like* (signal fort) est supérieure à celle accordée à l'observation d'un *not liked* (signal faible).

La matrice de scores $\boldsymbol{S} = (s_{ij})$ admet une factorisation

$$\boldsymbol{S} \simeq \boldsymbol{U} \times \boldsymbol{V^T}.$$

$$
\begin{array}{ccc}
 & & \text{Documents} \\
 & \text{Features} \rightarrow & \begin{pmatrix} \square & \square & \square & \square & \square & \cdots \\ \square & \square & \square & \square & \square & \cdots \\ \square & \square & \square & \square & \square & \cdots \end{pmatrix} \\
 & \downarrow & \\
\text{Users} & \begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix} & \begin{pmatrix} \square & \square & \square & \square & \square & \cdots \\ \square & \square & \square & \square & \square & \cdots \\ \square & \square & \square & \square & \square & \cdots \\ \square & \square & \square & \square & \square & \cdots \\ .. & .. & .. & .. & .. & \cdots \end{pmatrix} \\
 & \leftarrow \boldsymbol{K} \rightarrow &
\end{array}
$$

# Features latentes avec biais



$$\underbrace{\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix}}_{\substack{\boldsymbol{U} \\ \text{features users}}} = \underbrace{\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix}}_{\substack{\boldsymbol{\mu} \\ \text{cold-start}}} + \underbrace{\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix}}_{\substack{\boldsymbol{\epsilon} \\ \text{collaborative}}}$$

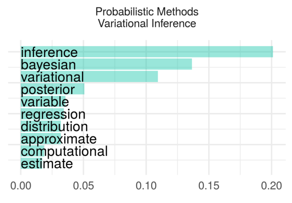
$$\underbrace{\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix}}_{\substack{\boldsymbol{V} \\ \text{features documents}}} = \underbrace{\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix}}_{\substack{\boldsymbol{\theta} \\ \text{cold-start}}} + \underbrace{\begin{pmatrix} \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \\ .. & .. & .. \end{pmatrix}}_{\substack{\boldsymbol{\delta} \\ \text{collaborative}}}$$

- Biais pour le cold-start :
  - $\boldsymbol{\mu}$ : préférences renseignées par chaque user.
  - $\boldsymbol{\theta}$ : *topic-proportions* des documents issues du *topic model* (LDA).
- Features collaboratives à inférer : $\boldsymbol{\epsilon}$ et $\boldsymbol{\delta}$.

# Procédure simplifiée

scorelab

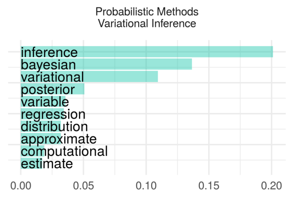
1. Inférer les topics : $\beta$ et $\theta$ (algo. EM variationnel).
2. Labelliser les topics.
3. Récupérer les topics d'intérêt des users : $\mu$.
4. Calculer les recommandations *cold-start*.
5. Toutes les 60 sec.,
   5.1 Actualiser la matrice de *likes* : $S$.
   5.2 Inférer les features collaboratives : $\epsilon$ et $\delta$ (algo. ALS).
   5.3 Calculer les recommandations collaboratives.



# Procédure simplifiée

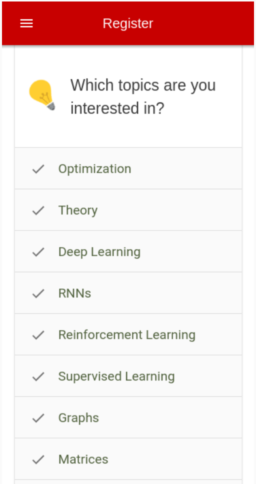
scorelab

1. Inférer les topics : $\beta$ et $\theta$ (algo. EM variationnel).
2. **Labelliser les topics.**
3. Récupérer les topics d'intérêt des users : $\mu$.
4. Calculer les recommandations *cold-start*.
5. Toutes les 60 sec.,
   - 5.1 Actualiser la matrice de *likes* : $S$.
   - 5.2 Inférer les features collaboratives : $\epsilon$ et $\delta$ (algo. ALS).
   - 5.3 Calculer les recommandations collaboratives.



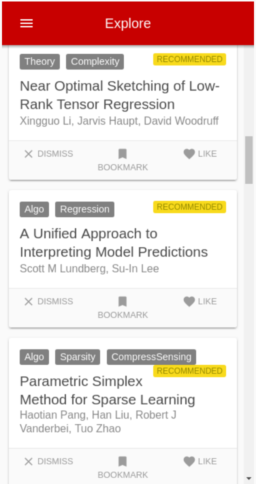
# Procédure simplifiée

1. Inférer les topics : $\beta$ et $\theta$ (algo. EM variationnel).
2. Labelliser les topics.
3. **Récupérer les topics d'intérêt des users : $\mu$.**
4. Calculer les recommandations *cold-start*.
5. Toutes les 60 sec.,
   5.1 Actualiser la matrice de *likes* : $S$.
   5.2 Inférer les features collaboratives : $\epsilon$ et $\delta$ (algo. ALS).
   5.3 Calculer les recommandations collaboratives.



# Procédure simplifiée

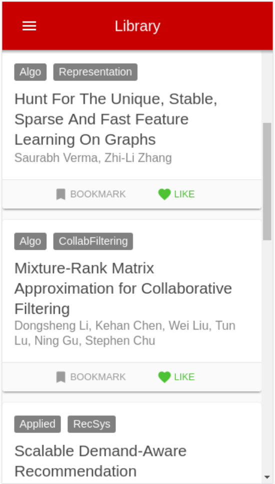
1. Inférer les topics : $\beta$ et $\theta$ (algo. EM variationnel).
2. Labelliser les topics.
3. Récupérer les topics d'intérêt des users : $\mu$.
4. **Calculer les recommandations *cold-start*.**
5. Toutes les 60 sec.,
    5.1 Actualiser la matrice de *likes* : $S$.
    5.2 Inférer les features collaboratives : $\epsilon$ et $\delta$ (algo. ALS).
    5.3 Calculer les recommandations collaboratives.



# Procédure simplifiée

scorelab

1. Inférer les topics : $\beta$ et $\theta$ (algo. EM variationnel).
2. Labelliser les topics.
3. Récupérer les topics d'intérêt des users : $\mu$.
4. Calculer les recommandations *cold-start*.
5. Toutes les 60 sec.,
    5.1 Actualiser la matrice de *likes* : $S$.
    5.2 Inférer les features collaboratives : $\epsilon$ et $\delta$ (algo. ALS).
    5.3 Calculer les recommandations collaboratives.



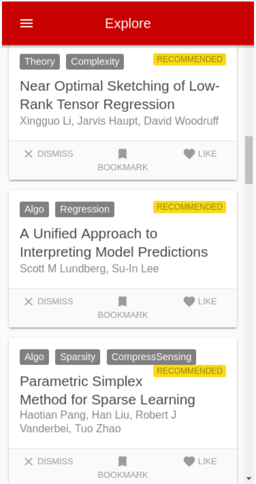
# Procédure simplifiée



1. Inférer les topics : $\boldsymbol{\beta}$ et $\boldsymbol{\theta}$ (algo. EM variationnel).
2. Labelliser les topics.
3. Récupérer les topics d'intérêt des users : $\boldsymbol{\mu}$.
4. Calculer les recommandations *cold-start*.
5. Toutes les 60 sec.,
   5.1 Actualiser la matrice de *likes* : $\boldsymbol{S}$.
   5.2 Inférer les features collaboratives : $\boldsymbol{\epsilon}$ et $\boldsymbol{\delta}$ (algo. ALS).
   5.3 Calculer les recommandations collaboratives.

# Procédure simplifiée

1. Inférer les topics : $\beta$ et $\theta$ (algo. EM variationnel).
2. Labelliser les topics.
3. Récupérer les topics d'intérêt des users : $\mu$.
4. Calculer les recommandations *cold-start*.
5. **Toutes les 60 sec.,**
   - 5.1 Actualiser la matrice de *likes* : $S$.
   - 5.2 Inférer les features collaboratives : $\epsilon$ et $\delta$ (algo. ALS).
   - 5.3 **Calculer les recommandations collaboratives.**



# Références

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003).
**Latent Dirichlet Allocation.**
*Journal of Machine Learning Research*, 3(Jan) :993–1022.

Mnih, A. and Salakhutdinov, R. R. (2008).
**Probabilistic Matrix Factorization.**
In *Advances in Neural Information Processing Systems*, pages 1257–1264.

Wang, C. and Blei, D. M. (2011).
**Collaborative topic modeling for recommending scientific articles.**
In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456. ACM.

Merci de votre attention !

scorelab

https ://scorelab.io

scorelab_io

adrien@scorelab.io

adrtod