

UNIVERSITÉ CLAUDE BERNARD LYON I

MASTER DATA SCIENCE

PROJET DATA MINING

---

# Détection d'événements sur Twitter

---

*Auteurs :*

Gregory HOWARD 11207726

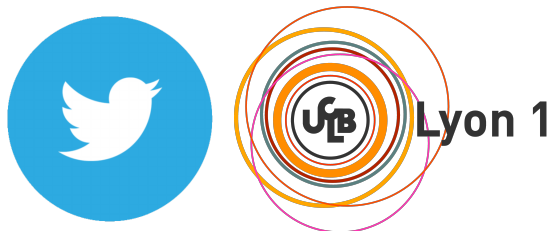
Marine RUIZ 11208141

Jules SAUVINET 11412086

*Professeur :*

Marc PLANTEVIT

21 janvier 2017



## Résumé

La détection d'évènements est un des sujets de recherche les plus importants dans l'analyse des réseaux sociaux. Les flux de données provenant des plates-formes de ces réseaux contiennent, la plupart du temps, beaucoup d'informations. On peut traiter et analyser ces informations dans le but de faire de la détection d'évènement(s).

Néanmoins, une grande partie des données à traiter est bruitée, notamment par des comptes Twitter entretenus par des robots. Ces comptes postent des tweets en masse avec une fréquence de publication constante, ce qui nous empêchent de donner une bonne description d'un événement ou alors on peut créer de faux événements.

Voilà pourquoi il est important de comprendre comment atténuer l'influence du bruit pour faire de la détection événements. Notre objectif a été de détecter des événements à partir de tweets en exploitant des informations spatio-temporelles et textuelles en essayant d'écarter les tweets de robots.

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Description de nos données</b>	<b>2</b>
<b>3</b>	<b>Le pré-traitement des données</b>	<b>3</b>
<b>4</b>	<b>L'approche globale</b>	<b>3</b>
<b>5</b>	<b>La détection d'évènements multi-échelle</b>	<b>4</b>
5.1	Adaptation de l'algorithme de détection multi-échelle et intégration du filtrage des robots . . . . .	4
<b>6</b>	<b>Description des éléments utiles à la détection d'événements</b>	<b>5</b>
6.1	Le tweet . . . . .	5
6.2	La matrice de similarité . . . . .	5
6.3	Le clustering . . . . .	5
6.4	L'événement . . . . .	5
<b>7</b>	<b>Les options sur le clustering</b>	<b>7</b>
7.1	L'élasticité . . . . .	7
7.2	La géolocalisation . . . . .	7
7.3	La loi de Poisson géographique . . . . .	7
<b>8</b>	<b>La détection d'évènement avec DBScan</b>	<b>7</b>
<b>9</b>	<b>Résultats</b>	<b>7</b>
<b>10</b>	<b>Conclusion</b>	<b>7</b>

# 1 Introduction

Notre objectif a été de détecter des événements à partir de tweets en exploitant des informations spatio-temporelles et textuelles. Notre approche utilise la technique de détection d'événements multi-échelles dans les réseaux sociaux introduite dans l'article de recherche [1] et développé par AHMED ANES BENDIMERAD et AIMENE BELFODIL. Nous avons adapté cette version à nos données à laquelle nous avons ajouté des mécanismes permettant de filtrer les tweets envoyés par des robots en amont du clustering permettant la détection d'événements. Nous avons également développé une approche de détection spatio-temporelle d'événements avec l'algorithme de clustering DBScan puis nous avons comparé les résultats des deux types de clustering.

Nous appellerons événement (au sens événement que nous souhaitons détecter) un phénomène physique survenant en un point et pendant une durée bien déterminé. Un événement est intéressant s'il suscite un nombre suffisant de tweets. Ces tweets peuvent se situer sur le lieu de l'événement ou bien ailleurs, notamment si celui-ci est retransmis par les médias. Les tweets sur l'événement peuvent se produire avant, pendant et après l'événement. Ainsi, la détection d'événement doit réussir à intégrer ces dispersions pour localiser dans les temps et l'espace les événements et c'est ce qui la rend complexe.

## 2 Description de nos données

Nous avons effectué notre détection d'événements à partir d'un fichier de 1,7 Go comportant 10 982 005 tweets récupérés sur Twitter. Ces tweets ont été postés dans l'agglomération newyorkaise ou par des utilisateurs newyorkais du 21 juillet 2015 au 16 novembre 2015. Chaque tweet contient les informations suivantes :

- un identifiant
- la date et l'heure à laquelle le tweet a été posté
- la position de la personne quand le tweet a été posté (latitude et longitude)
- l'identifiant du lieu géographique où a été posté le tweet
- le lieu du tweet au moment où il a été posté
- l'ensemble des références et hashtags du tweet
- l'identifiant de la personne qui a posté le tweet
- le pseudo de la personne qui a posté le tweet
- le nom complet de la personne qui a posté le tweet
- l'identifiant du lieu de création du compte de la personne qui a posté le tweet
- l'information de la certification ou non du compte de l'utilisateur ayant posté le tweet
- le nombre de personnes qui suivent la personne qui a posté le tweet
- le nombre d'amis de la personne qui a posté le tweet
- le nombre de tweet déjà posté par la personne qui a posté le tweet

Pour la détection d'événements, nous nous sommes servi de l'identifiant du tweet, du nom de la personne qui a posté le tweet, des hashtags, de la date et de la position d'envoi du tweet.

### 3 Le pré-traitement des données

Nous avons utilisé Knime pour effectuer un premier filtrage des robots. Pour cela, nous classons dans l'ordre décroissant les utilisateurs en fonction du nombre de tweets qu'ils ont postés. Puis, nous supprimons les 0.5% premiers utilisateurs à partir de ce classement. Ce pré-traitement nous permet d'homogénéiser nos données mais aussi d'augmenter de façon évidente la pertinence des événements détectés.

Nous avons basé ce choix de filtrage sur les deux critères suivants. En effet, d'une part, il y a une probabilité importante qu'un utilisateur qui poste en très grande quantité des tweets soit un robot. D'autre part, nous nous sommes dit qu'un utilisateur réel ayant une activité sur Twitter bien supérieure à celle des autres utilisateurs, avait en moyenne moins de contenu descriptif d'évènement dans ses tweets et plus de "tweets bruits", c'est à dire des tweets nous donnant aucune information sur le déroulement d'un quelconque évènement (e.g " #lol", ou encore " #oklm avec mon fré"). Nous avons ensuite confirmé cette intuition empiriquement en observant notre jeu de données.

Cette technique étant très approximative, nous ne l'utilisons que pour filtrer les utilisateurs qui publient, vraiment de façon évidente, plus que tous les autres utilisateurs.

La dimension géographique étant cruciale dans notre méthode de clustering, nous avons également filtré les tweets ne possédant pas de position géographique dans le module Knime. Nous avons donc donné en entrée à Knime notre fichier de 1,7 Go contenant 10 982 005 tweets et nous avons obtenu, à la fin du traitement, un fichier de 250 Mo contenant 1 600 000 tweets.

Nous avons joint le workflow Knime robotFilter.knwf permettant de faire ce filtrage à l'archive de rendu du projet.

### 4 L'approche globale

Pour notre premier algorithme, nous faisons des sous-ensembles de tweets en découpant nos données par date. On a ainsi un ensemble de tweets par jour. On applique, sur chaque ensemble journalier de tweets, l'algorithme de détection d'évènement multi-échelle.

Pour notre second algorithme, nous appliquons le premier algorithme mais nous supprimons avant tout traitement, les comptes robots que nous sommes susceptibles d'avoir dans nos données à l'aide du module Knime décrit précédemment puis par autre filtre que nous décrirons plus tard. En effet, comme expliqué dans la partie précédente, il est fort probable qu'ils faussent notre clustering.

De plus, chacun de nos événements est décrit par 20 hashtags pertinents. Ces hashtags pertinents sont définis comme étant les plus récurrents dans les tweets contenus dans l'évènement. Cependant, certains hashtags sont présents dans la description de beaucoup d'autres événements, ce qui les rend moins pertinents. Nous voulons les détecter et ne pas les prendre en compte lors du clustering afin d'avoir une vraie description de l'évènement.

## 5 La détection d'évènements multi-échelle

La technique de détection d'évènements n'est autre que celle évoquée dans l'article [1].

L'algorithme calcule la matrice de similarité et construit les clusters.

MOI

MOI

MOI

### 5.1 Adaptation de l'algorithme de détection multi-échelle et intégration du filtrage des robots

On enlève ceux qui sont régi par une loi Géométrique (ici ou dans pretraitement des données)

## 6 Description des éléments utiles à la détection d'événements

### 6.1 Le tweet

Un tweet est décrit par :

- son identifiant
- le nom du Tweetos
- les hashtags
- la date et l'heure à laquelle le Tweetos a tweeté
- sa position au format latitude/longitude au moment du tweet
- La date

Notre jeu de données de tweets s'étale sur 103 jours. Puisque nous voulons faire de la détection d'événements journaliers et localisés, nous avons découpé notre jeu de données initial en 103 sous-ensembles journaliers. Chaque sous-ensemble de tweets contient, approximativement, plus de 15 000 tweets.

Nous appliquons donc notre algorithme de détection d'événements sur chacun de nos sous-ensembles de tweets journaliers.

### 6.2 La matrice de similarité

La matrice de similarité est une matrice triangulaire supérieure de taille (nombre de tweets) \* (nombre de tweets).

Rappeler ce qui a été dit dans la partie 5

- Décrire la construction
- reciter l'article en parlant rapidement de la compression de Haar

### 6.3 Le clustering

Le clustering se fait à partir de la matrice de similarité qui elle-même a été obtenue en traitant un sous-ensemble de tweets journaliers.

Rappeler ce qui a été dit dans la partie 5

- Redonner l'algorithme
- Citer le jeu et expliquer rapidement ce qu'on maximise / minimise dans l'algorithme

### 6.4 L'événement

On décrit un événement par :

- une heure de début et une heure de fin qui permettent de calculer une durée
- une position centrale (longitude, latitude) calculée à partir de la moyenne de toutes les positions (moyenne des longitudes, moyenne des latitudes)
- le nombre de personnes différentes ayant posté des tweets
- la liste des 20 hashtags les plus importants permettant de le décrire

Nous récupérons tous les clusters obtenus lors du clustering et ne gardons que les clusters pertinents. Un cluster est pertinent si le nombre de personnes ayant tweeté est assez important, s'il y a assez de tweets et si la proportion de tweets par personne n'est pas exagérée. Ce cluster devient alors un événement.

## 7 Les options sur le clustering

### 7.1 L'élasticité

On suppose qu'un hashtag est fréquent pour un ensemble de tweets donné s'il apparaît plus de 20 fois.

Ce nombre est statique, ce qui ne permettrait pas de détecter de petits événements dans une journée où il y a eu peu de tweets. A l'inverse, on détecte beaucoup d'événements sur une journée où il y a eu beaucoup de tweets. Cela peut être intéressant, mais il est parfois plus judicieux de s'adapter au nombre de tweets postés dans une même journée. Ainsi, on définit un hashtag fréquent s'il apparaît dans plus de 20% des tweets.

Pour coder cette option nous avons défini un booléen, que l'on met à Vrai pour activer l'élasticité et à Faux pour ne pas l'activer.

Les valeurs 20 et 20% peuvent être modifiées grâce aux variables globales définies au début du module d'exécution Python.

### 7.2 La géolocalisation

### 7.3 La loi de Poisson géographique

## 8 La détection d'événement avec DBScan

## 9 Résultats

## 10 Conclusion

Les combinaisons intéressantes  
Quand, comment

## Acknowledgments

Nos remerciements vont à Marc Plantevit pour l'enseignement de son cours "Data Mining" à l'Université Claude Bernard Lyon I et son accompagnement durant ce projet.

## Références

- [1] X. Dong, M. D., Calabrese, and P. Frossard. Multiscale event detection in social media. *Data Min Knowl Disc*, June 2015. 2, 4