Guide Complet du Projet

Event Detection in Finance using Hierarchical Clustering Algorithms on News and Tweets

Reproduction et Extension de Carta et al. (2021)

Master 2 MOSEF — Université Paris 1 Panthéon-Sorbonne

Maeva & Roland

Février 2026

SOMMAIRE

1. Répartition des rôles
2. Planning détaillé (8–19 février)
3. Trame de modélisation (code)
4. Extensions proposées
5. Structure du rapport
6. Paramètres clés à respecter

PARTIE 1 : COMPRÉHENSION DÉTAILLÉE DU PAPIER

1. Vue d'ensemble et Objectif du Papier

Le papier de Carta et al. (2021) propose une **méthode de détection d'événements financiers en temps réel** basée sur le clustering hiérarchique, combinant deux sources de données textuelles :

- **Articles de presse** (newswires) → source autoritaire, peu bruitée → base qualitative
- **Tweets** (Stocktwits) → source bruitée mais reflétant l'opinion publique → mesure quantitative (résonance)

Objectif double :

- Identifier des groupes (clusters) d'articles sémantiquement liés, correspondant à des événements réels
- Détecter les **« hot events »** en mesurant leur écho sur les réseaux sociaux → génération d'alertes

Innovation clé : L'intégration de sources hétérogènes (presse + social media) pour une représentation riche des événements, avec un système d'alerte basé sur la résonance sociale.

2. Architecture du Pipeline (Figure 1)

Le pipeline se décompose en **7 étapes séquentielles**, exécutées quotidiennement :

```
Stock Prices + News Articles → LEXICON GENERATION
      → FEATURE ENGINEERING (News Modeling)
                → NEWS CLUSTERING
  → OUTLIER REMOVAL → RELEVANT WORDS EXTRACTION
Tweets → TWEETS MODELING → TWEETS ASSIGNMENT
                → ALERT GENERATION
```

3. Étape 1 : LEXICON GENERATION

3.1 Principe

L'objectif est de créer un **lexique dynamique, temporel et spécifique au domaine financier**. Ce lexique capture les mots qui ont un impact mesurable sur le marché. Il est **régénéré chaque jour** pour capturer l'effet de nouveaux mots/événements.

3.2 Procédure détaillée

Pour un jour d donné :

- **1. Collecte des articles** : tous les articles mentionnant « Standard & Poor's », « S&P 500 » ou « SPX » publiés dans la fenêtre [d–28, d–1] (4 semaines).
- **2. Prétraitement textuel** : tokenization (NLTK), conversion en minuscules, suppression des stopwords (Stanford CoreNLP), stemming (NLTK).
- **3. Filtrage de fréquence** : supprimer les mots dans >90% des documents (trop fréquents) et dans <10 documents (trop rares).

- **4. Matrice document-terme** : lignes = articles, colonnes = termes, valeurs = présence/absence (dummy 0/1).
- **5. Calcul du score f(j)** pour chaque mot j (voir formule ci-dessous).
- **6. Sélection** : mots avec f(j) ≥ P80 (positifs) ou f(j) ≤ P20 (négatifs).

3.3 Formule Mathématique — Marginal Screening

Pour N articles collectés dans la période [d–l, d–1] :

$$f(j) = (1/N) \times \Sigma_{k=1..N} X_k^{(j)} \cdot \delta_d^{(k)}$$

Où :

- $X_k^{(j)}$: variable dummy (1 si le terme j apparaît dans l'article k, 0 sinon)
- $\delta_d^{(k)}$: rendement (return) du S&P 500 le jour d pour l'article k = $(close_d - close_{d-1}) / close_{d-1}$

Interprétation : f(j) est la pente d'une **régression marginale** (marginal screening, Genovese et al. 2012). Prouvé plus efficace que le Lasso en haute dimension. Propriété Sure Screening : Prob(S_hat = S) → 1 quand N → ∞.

3.4 Sélection des mots du lexique

- **Mots positifs** : $f(j) \geq t^+$ (seuil au 80e percentile) → associés à des hausses de prix
- **Mots négatifs** : $f(j) \leq t^-$ (seuil au 20e percentile) → associés à des baisses de prix
- Les mots entre les deux seuils sont éliminés (peu informatifs)

3.5 Paramètres utilisés par les auteurs

Paramètre	Valeur
Fenêtre temporelle (l)	28 jours (4 semaines)
Seuil haute fréquence	>90% des documents
Seuil basse fréquence	<10 documents
Percentile positif (t+)	80e percentile
Percentile négatif (t–)	20e percentile

4. Étape 2 : FEATURE ENGINEERING

Transformer chaque article en un **vecteur dense** (document embedding) capturant sa sémantique, en ne retenant que les mots pertinents du lexique.

Procédure :

- **1.** Prétraitement : tokenization, lowercase, stopwords removal
- **2.** Filtrage par le lexique du jour courant
- **3.** Word embeddings via **Word2Vec pré-entraîné** (Google News, 300 dimensions, 3M mots)
- **4.** Document embedding = **moyenne des word embeddings** des mots filtrés

$$doc\_embedding(a) = (1/|W_a|) \times \Sigma_{w \in W_a} word2vec(w)$$

Note : le lexique sert de filtre sémantique — seuls les mots ayant un impact marché sont conservés.

5. Étape 3 : NEWS CLUSTERING

5.1 Agglomerative Clustering

Algorithme hiérarchique **bottom-up** : chaque document commence comme son propre cluster, puis les clusters sont fusionnés successivement selon un critère de liaison.

Paramètre	Valeur	Justification
Critère de linkage	Average Linkage	Minimise la distance moyenne entre paires
Métrique	Distance cosinus	Standard pour documents textuels
Look-back window	7 jours	Articles de la semaine précédente
Range k	2 à 10	Testé puis optimisé par silhouette

5.2 Silhouette Maximization

Le nombre de clusters k est **choisi automatiquement** chaque jour via maximisation du score de silhouette :

```
silhouette(i) = (b(i) − a(i)) / max(a(i), b(i))
```

- **a(i)** = distance moyenne entre i et tous les points de son cluster (cohésion)
- **b(i)** = distance moyenne entre i et les points du cluster le plus proche (séparation)
- Valeur $\in$ [–1, 1] : +1 = parfait, 0 = frontière, –1 = mal clusteré
- Score global = moyenne sur tous les échantillons

5.3 Calcul des centroïdes

$$\text{centroid}_c = \textbf{median}(\{\text{doc\_embedding}(a) : a \in \text{cluster}_c\})$$

Choix de la médiane (et non la moyenne) : plus robuste aux outliers et au bruit.

5.4 Pourquoi Agglomerative Clustering ?

Comparé à K-Means, K-Medians, K-Medoids, l'agglomerative clustering offre : meilleur Silhouette, meilleur Dunn Index, plus de clusters extraits (meilleure séparabilité), overlapping similaire des mots pertinents.

6. Étape 4 : OUTLIER REMOVAL

Certains articles ne reportent pas d'événements actuels (anniversaires, analyses générales...) → bruit dans les clusters. Méthode à **deux critères** :

- **Coefficient de silhouette par échantillon** : identifie les documents à la frontière entre clusters
- **Distance cosinus au centroïde** : identifie les documents faiblement corrélés à leur cluster

Procédure :

- 1. Trier les échantillons par chaque métrique
- 2. Seuil de coupure au **15e percentile**
- 3. Outlier = sous le seuil dans **au moins une** des deux métriques → supprimé
- 4. Recalculer les centroïdes (médiane) des clusters affectés

Amélioration observée : ~50% d'amélioration du Silhouette et du Dunn Index.

7. Étape 5 : RELEVANT WORDS EXTRACTION

- 1. Appliquer un modèle **TF-IDF** sur l'ensemble des articles (features = mots du lexique)
- 2. Obtenir un vecteur TF-IDF par document

- 3. Regrouper par cluster, calculer la **moyenne** des vecteurs TF-IDF
- 4. Trier par score décroissant → sélectionner les **top-10 mots** par cluster

$$TF\text{-}IDF(t, d) = TF(t, d) \times IDF(t) \quad \text{avec} \quad IDF(t) = \log(N / df(t))$$

8. Étape 6 : TWEET ASSIGNMENT

Enrichir chaque cluster de news avec des tweets sémantiquement corrélés pour évaluer la résonance.

- 1. Collecter les tweets du jour le plus récent (cashtag $SPX)
- 2. Dédoublonner (anti-spam)
- 3. Modéliser chaque tweet : suppression ponctuation, tokenization, filtrage par lexique, embedding = moyenne word2vec
- 4. Assignation : **similarité cosinus** tweet vs chaque centroïde → assigner au plus proche
- 5. Condition : distance < **tweet distance threshold = 0.5**, sinon écarté

$$cos\_sim(a, b) = (a \cdot b) / (||a|| \times ||b||)$$

9. Étape 7 : ALERT GENERATION

Détecter les **hot events** en mesurant la proportion de tweets assignés :

$$assigned\_ratio(d) = nb\ tweets\ assignés / nb\ total\ tweets\ du\ jour\ d$$

Si assigned_ratio(d) > **alert threshold (3%)**, une alerte est générée. Les auteurs testent des seuils de 1% à 5%.

10. Évaluation Expérimentale

10.1 Données

Source	Contenu	Période	Volume
Dow Jones DNA	Articles financiers anglais	Juin 2016 – Mars 2020	8 403 articles
Stocktwits ($SPX)	Tweets financiers	Juin 2016 – Mars 2020	283 473 tweets
S&P 500	Prix OHLCV quotidien	Juin 2016 – Mars 2020	~960 jours

10.2 Ground Truth pour les alertes

$$\delta_d = |close_{d+7} - close_d| / close_d$$

- Un jour d est un **event day** si $\delta_d > 0.02$ (2% variation hebdomadaire)
- Event days consécutifs agrégés en événements (tolérance : 3 jours d'interruption)
- ~15% des jours sont marqués comme event days

10.3 Métriques

- **Recall** = événements détectés / total événements
- **Precision** = alertes correctes (hits) / total alertes
- **F-score** = 2 × (precision × recall) / (precision + recall)

10.4 Résultats clés

- Avec seuil 1% : recall ≈ 95%, précision > 50%
- Agglomerative Clustering surpasse K-Means, K-Medians, K-Medoids
- Clusters capturent correctement : Brexit, Elections US 2016, Trade War, Covid-19
- Fausses alertes souvent liées aux rapports trimestriels

PARTIE 2 : ORGANISATION DU PROJET

1. Répartition des Rôles

Roland — Source 1 (News GDELT) + Source 3 (S&P 500)

- Extraction et nettoyage des articles GDELT (scraping des URLs)
- Prétraitement des données de prix S&P 500
- Lexicon Generation (possède news + prix)
- Feature Engineering des news
- News Clustering + Outlier Removal + Relevant Words Extraction

Maeva — Source 2 (Tweets Entreprises Énergie)

- Nettoyage et extraction des tweets en CSV
- EDA complète des tweets + Preprocessing NLP
- Tweet Modeling (embedding)
- Tweet Assignment aux clusters de Roland
- Alert Generation + Évaluation

2. Planning Détaillé (8–19 février)

Phase 1 : DATA / EDA / PREPROCESSING (8–14 février)

Roland :

Jour	Tâche	Livrable
8–9 fév	Finaliser scraping GDELT (newspaper3k)	news_2019_2023.csv complet
9–10 fév	Nettoyage News (doublons, articles vides, dates)	news_clean.csv
10–11 fév	EDA News (distribution, sources, longueur)	Notebook + section rapport
11–12 fév	Prétraitement prix S&P 500 (rendements δ_d)	sp500_returns.csv
12–13 fév	Preprocessing NLP News (tokenize, stem, stopwords)	news_preprocessed.csv
13–14 fév	Lexicon Generation (formule f(j), filtrage, percentiles)	lexicons/ + rapport

Maeva :

Jour	Tâche	Livrable
8–9 fév	Extraction tweets en CSV (parser le format brut)	tweets_clean.csv
9–10 fév	EDA Tweets (entreprise, temporel, engagement, langues)	Notebook + section rapport
10–11 fév	Filtrage (spam, doublons, non-anglais, non-pertinents)	tweets_filtered.csv
11–12 fév	Analyse lien S&P 500 (corrélation volume/variations)	Analyse + rapport
12–13 fév	Preprocessing NLP (tokenize, URLs, mentions, lowercase)	tweets_preprocessed.csv
13–14 fév	Rédaction rapport EDA	rapport_eda_maeva.md

Phase 2 : MODÉLISATION (14–18 février)

Samedi 14 février soir — Mise en commun #1 : présenter résultats EDA, vérifier cohérence, valider lexiques.

Roland (14–18 fév) :

Jour	Tâche
14–15 fév	Feature Engineering : Word2Vec Google News 300d, filtrage lexique, doc embeddings
15–16 fév	News Clustering : Agglomerative (sklearn), cosine, average linkage, silhouette max
16–17 fév	Outlier Removal : silhouette + distance centroïde, seuil 15e percentile
17 fév	Relevant Words : TF-IDF sklearn, top-10 mots par cluster
17–18 fév	Visualisations : t-SNE, boxplots, comparaison K-Means/K-Medoids/K-Medians

Maeva (14–18 fév) :

Jour	Tâche
14–15 fév	Tweet Modeling : même Word2Vec, filtrage par lexique Roland, tweet embeddings
15–16 fév	Tweet Assignment : cosine similarity vs centroïdes, threshold 0.5
16–17 fév	Alert Generation : assigned_ratio, seuil 3%, identification alertes
17 fév	Évaluation : ground truth $\delta_d > 0.02$, precision/recall/F-score (seuils 1–5%)
17–18 fév	Extension comparative (Word2Vec vs alternatives, analyse sentiment)

Phase 3 : FINALISATION (18–20 février)

Date	Tâche
18 fév	Rédaction finale individuelle + documentation code
Jeudi 19 fév	MISE EN COMMUN #2 : fusionner rapports, adapter slides, répétition
Vendredi 20 fév	SOUTENANCE

3. Trame de Modélisation (Code Python)

3.1 Lexicon Generation — Roland

```
def generate_lexicon(news_df, prices_df, day, lookback=28,
                     freq_high=0.9, freq_low_abs=10,
                     percentile_pos=80, percentile_neg=20):
    # 1. Fenetre temporelle [d-28, d-1]
    start = day - pd.Timedelta(days=lookback)
    articles = news_df[(news_df['date']>=start) & (news_df['date']<day)]

    # 2. Matrice document-terme (dummy 0/1)
    vectorizer = CountVectorizer(binary=True, max_df=freq_high, min_df=freq_low_abs)
    X = vectorizer.fit_transform(articles['text_preprocessed'])

    # 3. Rendements du jour suivant
    delta = articles['date'].apply(lambda d: get_return(prices_df, d)).values

    # 4. f(j) = (1/N) * X.T @ delta
    f_scores = (X.T @ delta) / X.shape[0]

    # 5. Selection par percentiles
    t_pos = np.percentile(f_scores, percentile_pos)
    t_neg = np.percentile(f_scores, percentile_neg)
    terms = vectorizer.get_feature_names_out()
    lexicon = terms[(f_scores >= t_pos) | (f_scores <= t_neg)]
    return lexicon
```

3.2 Feature Engineering + Clustering — Roland

```
# Document Embedding
w2v_model = api.load('word2vec-google-news-300')
def doc_embedding(tokens, lexicon, model, dim=300):
    filtered = [w for w in tokens if w in lexicon and w in model]
    if not filtered: return np.zeros(dim)
    return np.mean([model[w] for w in filtered], axis=0)

# Agglomerative Clustering + Silhouette Maximization
def cluster_news(embeddings, k_range=range(2, 11)):
    best_k, best_score, best_labels = 2, -1, None
    for k in k_range:
        model = AgglomerativeClustering(n_clusters=k, metric='cosine',
                                        linkage='average')
        labels = model.fit_predict(embeddings)
        score = silhouette_score(embeddings, labels, metric='cosine')
        if score > best_score:
            best_k, best_score, best_labels = k, score, labels
    # Centroides = mediane par cluster
    centroids = {c: np.median(embeddings[best_labels==c], axis=0)
                 for c in range(best_k)}
    return best_labels, centroids, best_k
```

3.3 Tweet Assignment + Alert Generation — Maeva

```
# Tweet Assignment
def assign_tweets(tweet_embs, centroids, threshold=0.5):
    centroid_matrix = np.array(list(centroids.values()))
    assignments, assigned_count = {}, 0
    for i, emb in enumerate(tweet_embs):
        if emb is None: continue
        sims = cosine_similarity([emb], centroid_matrix)[0]
        best_idx = np.argmax(sims)
        if (1 - sims[best_idx]) < threshold:
            c = list(centroids.keys())[best_idx]
            assignments.setdefault(c, []).append(i)
            assigned_count += 1
    return assignments, assigned_count
```

```
# Alert Generation
def alert(assigned_count, total_tweets, threshold=0.03):
    ratio = assigned_count / total_tweets if total_tweets > 0 else 0
    return ratio > threshold, ratio

# Evaluation (Ground Truth)
def ground_truth(prices_df, var_threshold=0.02, gap=3):
    prices_df['weekly_var'] = abs(
        (prices_df['close'].shift(-7) - prices_df['close']) / prices_df['close']
    )
    prices_df['is_event'] = prices_df['weekly_var'] > var_threshold
    # Agreger event days consecutifs (tolerance gap jours)
    ...
```

4. Extensions Proposées

Extension 1 : Nouveau secteur

Application au secteur énergie (BHP, BP, TotalEnergies, FMC, Stora Enso) au lieu du S&P 500 global. Les auteurs mentionnent que l'approche peut être étendue à différents domaines.

Extension 2 : Source alternative GDELT

GDELT (gratuit) vs Dow Jones DNA (commercial). Comparaison de la qualité des clusters obtenus.

Extension 3 : Période 2019–2023

Nouveaux événements majeurs : Covid complète, crise énergétique 2021-22, guerre Ukraine/Russie, inflation 2022-23.

Extension 4 : Modèles comparatifs

BERT/FinBERT au lieu de Word2Vec pour les embeddings. DBSCAN/HDBSCAN au lieu de l'agglomerative. Analyse de sentiment en complément.

Extension 5 : Tweets multi-entreprises

Propagation d'événements entre entreprises du même secteur (événement BP → impact tweets TotalEnergies ?).

5. Structure du Rapport

Section	Contenu	Rédacteur	Deadline
1. Introduction	Contexte, objectif, contributions	Commun	18 fév
2. Revue de littérature	Résumé du papier + positionnement	Commun	17 fév
3. Méthodologie	Pipeline complet (7 étapes + formules)	Commun	15 fév
4.1 Données News + Prix	GDELT + S&P 500 + EDA	Roland	14 fév
4.2 Données Tweets	420K tweets + EDA + preprocessing	Maeva	14 fév
5.1–5.3 Lexique + Clustering	Lexiques, silhouette, t-SNE, comparaison algos	Roland	18 fév
5.4 Tweets + Alertes	Assignment, alertes, precision/recall	Maeva	18 fév
5.5–6–7 Extensions + Discussion	Résultats extension, forces/limites, conclusion	Commun	19 fév

6. Récapitulatif des Paramètres Clés

Paramètre	Valeur	Étape
Fenêtre lexique	28 jours	Lexicon Generation
Seuil haute fréquence	>90% docs	Lexicon Generation
Seuil basse fréquence	<10 docs	Lexicon Generation
Percentile positif	80e	Lexicon Generation
Percentile négatif	20e	Lexicon Generation
Word2Vec dimension	300	Feature Engineering
Look-back window clustering	7 jours	News Clustering
Linkage	Average	News Clustering
Distance	Cosine	News Clustering
Range k	2 à 10	News Clustering
Centroïde	Médiane	News Clustering
Outlier percentile	15e	Outlier Removal
Tweet distance threshold	0.5	Tweet Assignment
Alert threshold	3% (tester 1–5%)	Alert Generation
Ground truth variation	2% hebdomadaire	Évaluation
Gap tolerance	3 jours	Évaluation