Master 2 Modélisations Statistiques Économiques et Financières

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

Reproduction et Extension de Carta et al. (2021)

Application au S&P 500 — Période 2023

Auteurs : Maeva N'GUESSAN & Roland DUTAUZIET

Formation : Master 2 MoSEF 2025–2026

Date : 20 Février 2026

Projet P10 — Quantitative Finance

Contents

Introduction

La capacité à détecter en temps réel les événements financiers significatifs est devenue un enjeu majeur pour les investisseurs, analystes et décideurs. Le volume croissant de données textuelles articles de presse, réseaux sociaux offre une opportunité unique d'automatiser cette détection.

Le papier, Event detection in finance using hierarchical clustering algorithms on news and tweets de Carta et al. (2021) , publié dans *PeerJ Computer Science*, propose une méthodologie innovante combinant **clustering hiérarchique d'articles de presse** et **analyse de résonance sur les réseaux sociaux** pour détecter les "hot events" financiers. L'idée clé est la suivante : si un événement détecté dans la presse génère un écho significatif sur les réseaux sociaux (mesuré par le ratio de tweets assignés aux clusters), cela constitue un signal d'alerte.

Les objectifs de ce projet sont de :

1. **Reproduire fidèlement** les 7 étapes du pipeline de Carta et al. (2021) sur de nouvelles données.
2. **Adapter** la méthodologie à la période 2023 avec des sources alternatives (GDELT, Twitter/X au lieu de Dow Jones DNA et Stocktwits).
3. **Analyser** les résultats obtenus et les comparer au papier original.

Notre étude se concentre sur le **S&P 500** durant l'année 2023, période marquée par des événements majeurs comme la crise de Silicon Valley Bank (mars 2023), l'essor de l'intelligence artificielle ou les décisions de politique monétaire de la Fed.

La suite du rapport est organisée comme suit : la section 1 présente la méthodologie détaillée du pipeline, la section 2 décrit les données utilisées, la section 3 présente l'architecture technique et les codes, la et section 4 expose et interprète les résultats obtenus.

1 Méthodologie

La méthodologie reproduit fidèlement le pipeline en 7 étapes proposé par Carta et al. (2021). Chaque étape est exécutée quotidiennement, produisant des clusters d'événements enrichis par l'analyse des réseaux sociaux.

1.1 Ĺexicon Generation

L'objectif est de construire un **lexique dynamique et spécifique au domaine financier**, régénéré quotidiennement, qui capture les mots ayant un impact mesurable sur les variations du marché.

1.1.1 Prétraitement textuel

Pour chaque jour d, on collecte les articles publiés dans une fenêtre glissante $[d-28,\, d-1]$ (4 semaines). Les articles sont prétraités : tokenisation, conversion en minuscules, suppression des stopwords, et lemmatisation.

1.1.2 Matrice document-terme et filtrage

On construit une **matrice document-terme binaire X** où $X_k^{(j)} = 1$ si le terme j apparaît dans l'article k, et 0 sinon. Un filtrage de fréquence est appliqué :

- Suppression des termes présents dans plus de 90% des documents (trop génériques).
- Suppression des termes apparaissant dans moins de 10 documents (trop rares).

1.1.3 Marginal Screening

Le score $f(j)$ de chaque terme j est calculé par la formule de *Marginal Screening* (Genovese et al., 2012) :

$$\boxed{f(j) = \frac{1}{N} \sum_{k=1}^{N} X_k^{(j)} \cdot \delta_d^{(k)}} \tag{1}$$

où :

- N : nombre d'articles dans la fenêtre temporelle,
- $X_k^{(j)} \in \{0, 1\}$: présence du terme j dans l'article k,
- $\delta_d^{(k)} = \dfrac{\text{close}_d - \text{close}_{d-1}}{\text{close}_{d-1}}$: rendement du S&P 500 le jour de publication de l'article k.

Interprétation : $f(j)$ est la pente d'une régression marginale. Un score positif élevé signifie que la présence du mot j dans un article est associée à une hausse du marché ; un score très négatif indique une présence de ce mot qui est associée avec une baisse du cours du S&P500.

1.1.4 Sélection du lexique

Les mots sont sélectionnés par percentiles :

- **Mots positifs** : $f(j) \geq t^+$ (80^{e} percentile) $\rightarrow$ associés généralement à des hausses significatives.
- **Mots négatifs** : $f(j) \leq t^-$ (20^{e} percentile) $\rightarrow$ associés généralement à des baisses significatives.

Les mots entre ces deux seuils sont éliminés car jugés peu informatifs(neutres). Ce processus est répété pour chaque jour d, produisant un lexique dynamique.

1.2 Feature Engineering

Chaque article associé à un jour est filtré en fonction du lexique précédemment défini puis transformé en un **vecteur dense** (document embedding) capturant sa sémantique.

1.2.1 Modèle d'embeddings

Nous utilisons un modèle GloVe pré-entraîné (Dolma 300 dimensions, 1.2M de mots) pour représenter chaque mot du lexique dans un espace vectoriel continu. Ce choix est conforme à l'esprit du papier original qui utilise Word2Vec (Google News, 300d).

1.2.2 Filtrage et Calcul de l'embedding documentaire

L'embedding d'un article a est la **moyenne des embeddings** des mots présents dans le lexique du jour :

$$\boxed{\mathbf{v}_a = \frac{1}{|W_a|} \sum_{w \in W_a} \text{GloVe}(w)} \tag{2}$$

où W_a est l'ensemble des mots de l'article a appartenant au lexique courant. Seuls les mots appartenant de l'article du jour d appartenant au lexique semantique défini à l'étape 1.1 Lexicon Generation sont conservés, le lexique servant de **filtre sémantique**.

1.3 News Clustering

1.3.1 Clustering Algorithms

Les embeddings d'articles sont regroupés par **Clustering** en testant différentes méthodes non-supervisées dont le Clustering Hierarchique Ascendant, K-Means et K-Medians. Les paramètres utilisés sont:

Paramètre	**Valeur**	**Justification**
Critère de linkage	Average Linkage	Distance moyenne entre paires
Métrique	Distance cosinus	Standard pour documents textuels
Fenêtre	7 jours	Articles de la semaine précédente
Range k	2 à 10	Optimisé par silhouette
min_samples	2 ou 3	Nombre minimal de news par clusters

Table 1: Paramètres du clustering

1.3.2 Silhouette Maximization

Le nombre optimal de clusters k est déterminé automatiquement via la maximisation du **score de silhouette** :

$$\boxed{s(i) = \frac{b(i) - a(i)}{\max\big(a(i),\, b(i)\big)}} \tag{3}$$

où $a(i)$ est la distance moyenne intra-cluster (cohésion) et $b(i)$ la distance moyenne au cluster le plus proche (séparation). Le score global est la moyenne sur tous les échantillons, $s \in [-1,\, 1]$.

1.3.3 Calcul des centroïdes

Les centroïdes sont calculés comme la **médiane** (et non la moyenne) des embeddings au sein de chaque cluster, assurant une robustesse aux outliers :

$$\mathbf{c}_k = \text{median}\left(\{\mathbf{v}_a : a \in \text{cluster}_k\}\right) \tag{4}$$

Ainsi, chaque centroïde intègre les médiannes de chaque de chaque dimensions des embeddings des news du cluster. On a donc des centroïdes qui sont des signatures fictives du cluster.

1.4 Outlier Removal

Certains articles ne rapportent pas d'événements actuels (anniversaires, analyses générales) et constituent du bruit. La méthode de Carta et al. utilise un **double critère** :

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

Un article est considéré comme outlier s'il est sous le 30^{e} percentile dans **au moins l'un** des deux critères. Après suppression, les centroïdes sont recalculés.

1.5 Relevant Words Extraction

On se sert d'un vocabulaire financier personnalisé afin d'extraire les mots phares de chaque cluster. Pour chaque cluster, les mots les plus représentatifs sont extraits via **TF-IDF** :

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \log\left(\frac{N}{\text{df}(t)}\right) \tag{5}$$

On calcule la moyenne des vecteurs TF-IDF par cluster et on sélectionne les **top-10 mots** par score décroissant. Ces mots permettent d'interpréter qualitativement chaque cluster.

1.6 Tweet Assignment

Les tweets sont modélisés de la même façon que les articles : prétraitement, filtrage par le lexique, puis calcul de l'embedding par moyenne des vecteurs GloVe. Chaque tweet est ensuite **assigné au cluster dont le centroïde est le plus proche** en similarité cosinus :

$$\text{sim}(\mathbf{t}, \mathbf{c}_k) = \frac{\mathbf{t} \cdot \mathbf{c}_k}{\|\mathbf{t}\| \times \|\mathbf{c}_k\|} \tag{6}$$

Si la distance cosinus (1 – similarité) est supérieure au seuil de 0.5, le tweet est écarté.

1.7 Alert Generation

Le ratio de tweets assignés est calculé quotidiennement :

$$\boxed{R(d) = \frac{\text{nb tweets assignés le jour } d}{\text{nb total tweets le jour } d}} \tag{7}$$

Si $R(d) > \theta$ (seuil d'alerte, typiquement 3%), une **alerte** est générée, signalant un "hot event".

Évaluation

La **ground truth** est construite à partir des variations hebdomadaires du S&P 500 :

$$\delta_d = \frac{|\text{close}_{d+7} - \text{close}_d|}{\text{close}_d} \tag{8}$$

Un jour d est un *event day* si $\delta_d > 0.02$ (variation hebdomadaire $> 2\%$). Les event days consécutifs sont agrégés en événements (tolérance de 3 jours).

Les métriques utilisées sont :

$$\text{Recall} = \frac{\text{événements détectés}}{\text{total événements}} \tag{9}$$

$$\text{Precision} = \frac{\text{alertes correctes}}{\text{total alertes}} \tag{10}$$

$$\text{F-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

2 Présentation des données utilisées

2.1 Vue d'ensemble

Notre étude utilise trois sources de données couvrant l'année 2023. Le tableau 2 compare nos données avec celles du papier original.

Source	**Carta et al. (2021)**	**Notre projet**
Articles de presse	Dow Jones DNA (8 403)	GDELT/Yahoo (1 565)
Tweets	Stocktwits \$SPX (283 473)	Twitter/X \$SPX/\$SPY (2 243)
Prix	S&P 500 OHLCV (960 jours)	S&P 500 OHLCV (271 jours)
Période	Juin 2016 – Mars 2020	Janvier 2023 – Décembre 2023

Table 2: Comparaison des données : papier original vs notre reproduction

2.2 Articles de presse (news_2023.csv)

Les articles proviennent du projet **GDELT** (Global Database of Events, Tone and Language), une base de données ouverte d'actualités mondiales, complétée par Yahoo Finance et CNBC. Le corpus comprend **1 565 articles** avec les colonnes : `date`, `headline`, `body`, `url`, `source`.

La distribution des sources est : Yahoo Finance (76.4 %), PR Newswire (19.2 %), CNBC (4.4 %). Les articles couvrent la période du 1er janvier au 30 novembre 2023.

2.3 Tweets (tweets_2023.csv)

Le dataset de tweets est extrait du **Financial Tweets dataset** publié par AmulyaS sur Kaggle. Il regroupe des messages initialement collectés via l'API de Twitter/X. Nous l'avons filtré afin de récupérer un corpus comprennant **2 243 tweets** financiers mentionnant les "cashtags" `$SPX`, `$SPY`, l'indice `S&P 500` ou des termes associés. Les colonnes incluent : `date`, `full_content`, `likes`, `retweets`, `followers`, `sentiment` (score VADER) et `url`.

La distribution temporelle montre une concentration sur la période allant de janvier à mai 2023, avec une activité réduite durant l'été. L'analyse du sentiment est complétée par l'algorithme **VADER** (Valence Aware Dictionary and sEntiment Reasoner), un modèle basé sur des règles lexicales particulièrement efficace pour le langage des réseaux sociaux, fournissant un indicateur de polarité pour chaque tweet.

2.4 Prix S&P 500 (sp500_2023.csv)

Les prix quotidiens du S&P 500 (ticker `^GSPC`) proviennent de Yahoo Finance et couvrent 271 jours de trading, de décembre 2022 à décembre 2023. Les rendements quotidiens δ_d sont précalculés.

2.5 Lexiques quotidiens (daily_lexicons/)

Nous avons généré **334 lexiques quotidiens** couvrant février à décembre 2023. L'union de tous ces lexiques produit un vocabulaire de **1 420 mots uniques** ayant un impact marché détecté.

2.6 Différences et adaptations par rapport au papier original

Plusieurs différences méritent d'être soulignées :

Aspect	**Papier original**	**Notre adaptation**
Source d'articles	Dow Jones DNA (commercial)	GDELT (gratuit, open-source)
Qualité articles	Reuters, WSJ, FT (premium)	Yahoo Finance, CNBC, PR Newswire
Volume tweets	283 473 (∼300 tweets/jour)	2 243 (∼6 tweets/jour)
Plateforme tweets	Stocktwits (cashtag $SPX)	Twitter/X (cashtags + mots-clés)
Embeddings	Word2Vec Google News 300d	GloVe Dolma 300d
Période	4 ans (2016–2020)	1 an (2023)
Événements majeurs	Brexit, Trade War, Covid-19	SVB crisis, AI boom, Fed rates

Table 3: Différences détaillées entre le papier original et notre reproduction

L'utilisation de GDELT comme source alternative représente un intérêt méthodologique : elle démontre la **transférabilité** de la méthode à des sources gratuites, un point important pour la reproductibilité scientifique. Cependant, les articles GDELT/Yahoo Finance sont souvent plus généralistes que ceux de Dow Jones DNA, ce qui peut affecter la qualité du lexique.

Le volume réduit de tweets (126× moins que le papier) constitue la limite la plus significative. Le ratio $R(d)$ sera mécaniquement plus instable avec 6 tweets/jour qu'avec 300. Néanmoins, cette contrainte nous permet d'évaluer la **robustesse** de la méthode dans des conditions de données limitées.

2.7 Contexte économique 2023

L'année 2023 a été marquée par plusieurs événements majeurs susceptibles d'apparaître dans nos clusters :

- **Mars 2023 — Crise bancaire SVB** : faillite de Silicon Valley Bank le 10 mars, suivie de Signature Bank. Le S&P 500 a chuté de 4.5% en une semaine, générant une variation hebdomadaire bien au-dessus du seuil de 2%.
- **Mai–Juillet 2023 — Engouement pour l'IA** : après le lancement de ChatGPT fin 2022, les valeurs technologiques (NVIDIA, Microsoft) tirent le marché à la hausse.
- **Septembre–Octobre 2023 — Politique monétaire Fed** : débats sur le *higher for longer*, taux directeurs maintenus à 5.25–5.50%.
- **Novembre 2023 — Rebond de fin d'année** : le S&P 500 gagne 8.9% en novembre, l'un des meilleurs mois depuis 2020.

Ces événements constituent les cas d'étude attendus de notre système de détection.

3 Présentation des codes

3.1 Architecture du projet

Le projet est organisé en modules Python spécialisés, accompagnés de notebooks Jupyter pour chaque étape du pipeline :

```
project/
|-- notebooks/
|   |-- 0_extract_data.ipynb        # Extraction des donnees
|   |-- 1_lexicon_generation.ipynb  # Generation des lexiques
|   |-- 2_feature_engineering.ipynb # Embeddings GloVe
|   |-- 3_news_clustering.ipynb     # Clustering + visualisations
|-- src/
|   |-- extract_data.py             # Fonctions d'extraction
|   |-- lexicon_generation.py       # Marginal Screening f(j)
|   |-- feature_engineering.py      # Embeddings documentaires
|   |-- news_clustering.py          # Clustering HAC + evaluation
|-- models/
|   |-- dolma_300_2024_1.2M.100_combined.txt  # GloVe Dolma 300d
|-- data/
|   |-- raw/                        # Donnees brutes (GDELT, Kaggle)
|   |-- processed/
|       |-- news_2023.csv           # 1,565 articles
|       |-- tweets_2023.csv         # 2,243 tweets
|       |-- sp500_2023.csv          # 271 jours
|       |-- daily_lexicons_filtered/# 334 lexiques filtres (P20/P80)
|       |-- daily_lexicons_full/    # 334 lexiques complets
```

Listing 1: Architecture du projet

3.2 Module `extract_data.py` — Extraction des données

Ce module gère l'extraction des trois sources de données. La fonction `check_density()` vérifie la couverture temporelle du dataset GDELT (334/334 jours sans gaps). La fonction `scrape_content()` utilise la bibliothèque `newspaper3k` pour extraire le texte complet des articles à partir des URLs, en filtrant les contenus de moins de 500 caractères. Pour les tweets, `process_social_data()` filtre le dataset Kaggle pour les cashtags S&P 500 ($SPY, $SPX) et calcule les scores de sentiment via VADER.

```
def process_social_data(file_path):
    df = pd.read_csv(file_path, low_memory=False)
    df['dt_obj'] = pd.to_datetime(df['timestamp'], format='ISO8601', errors='coerce')
    # Filter for 2023 and S&P 500 keywords
    df_2023 = df[df['dt_obj'].dt.year == 2023].copy()
    keywords = r'\$SPY|\$SPX|S&P 500|SP500|Stock Market'
    mask = df_2023['description'].str.contains(keywords, case=False, na=False)
    df_spy = df_2023[mask].copy()
    # Merge content and calculate VADER sentiment scores
    df_spy['full_content'] = df_spy['embed_title'].fillna('') + " " + \
                             df_spy['description'].fillna('')
    df_spy['sentiment'] = df_spy['full_content'].apply(
        lambda x: analyzer.polarity_scores(x)['compound'])
```

```
    return df_spy
```

Listing 2: Extraction et prétraitement des tweets — `extract_data.py`

3.3 Module `lexicon_generation.py` — Génération du lexique

Ce module implémente le cœur de l'étape 1 du pipeline. Le prétraitement utilise **spaCy** pour la tokenisation, la suppression des stopwords et le filtrage alphabétique (fonction `preprocess_spacy()`). La fonction `build_daily_lexicon()` implémente la formule de Marginal Screening (équation 1) : elle construit la matrice document-terme binaire, calcule le score $f(j)$ par produit matriciel, et sélectionne les mots aux percentiles P20/P80.

```
def build_daily_lexicon(news_train, prices_map, current_date,
                        dtm_output_dir, output_dir, filtered_output_dir):
    N = len(news_train)
    # a. Binary Document-Term Matrix with frequency filtering
    vectorizer = CountVectorizer(
        binary=True,          # Dummy variable X_k(j)
        max_df=0.90,          # Remove words in >90% of docs
        min_df=10,            # Remove words in <10 docs
        stop_words='english')
    dtm_sparse = vectorizer.fit_transform(news_train['clean'])
    words = vectorizer.get_feature_names_out()

    # b. Marginal Screening: f(j) = (1/N) * sum(X_k(j) * delta_k)
    deltas = news_train['date'].map(prices_map).fillna(0).values
    sum_product = np.array(dtm_sparse.T.dot(deltas)).flatten()
    f_j = sum_product / N

    # c. Percentile-based selection (P20/P80)
    p20 = np.percentile(f_j, 20)
    p80 = np.percentile(f_j, 80)
    lexicon_df = pd.DataFrame({'word': words, 'score': f_j})
    filtered = lexicon_df[(lexicon_df['score'] >= p80) |
                          (lexicon_df['score'] <= p20)]
    filtered.to_csv(filtered_path, index=False)
```

Listing 3: Lexicon Generation — `lexicon_generation.py`

3.4 Module `feature_engineering.py` — Embeddings

Ce module transforme chaque article en vecteur dense de 300 dimensions. On utilise un modèle **GloVe** pré-entraîné (Dolma 2024, 300d, 1.2M mots) chargé via `gensim.models.KeyedVectors`. Pour chaque article, seuls les mots présents dans le lexique *du jour courant* sont conservés, et l'embedding documentaire est la moyenne de leurs vecteurs GloVe.

```
def compute_news_embedding(text, lexicon_set, model):
    """
    1. Filter: Retain only words present in the daily lexicon.
    2. Embedding: Extract GloVe vectors for these words.
    3. Average: Compute mean vector (News-Embedding, 300D).
    """
```

```
    tokens = text.split()
    valid_vectors = [
        model[w] for w in tokens
        if w in lexicon_set and w in model
    ]
    if not valid_vectors:
        return np.zeros(300)
    return np.mean(valid_vectors, axis=0)

def run_feature_engineering_pipeline(news_df, lexicon_dir, model):
    """Iterates day-by-day, applies daily lexicon filtering."""
    embedded_data = []
    for current_day in tqdm(sorted(news_df['date'].unique())):
        lex_path = os.path.join(lexicon_dir, f"lexicon_filtered_{current_day}.csv")
        if os.path.exists(lex_path):
            day_lexicon = set(pd.read_csv(lex_path)['word'].tolist())
            daily_news = news_df[news_df['date'] == current_day]
            for idx, row in daily_news.iterrows():
                vector = compute_news_embedding(row['clean'], day_lexicon, model)
                embedded_data.append({'date': current_day,
                                      'embedding': vector,
                                      'headline': row['headline']})
    return pd.DataFrame(embedded_data)
```

Listing 4: News Embedding — `feature_engineering.py`

3.5 Module `news_clustering.py` — Clustering et Évaluation

Ce module implémente les étapes 3 à 5. La fonction `run_clustering_evaluation()` compare trois algorithmes : Agglomerative Clustering (cosine + average linkage), K-Means++ (euclidien sur données normalisées), et K-Medians (distance de Manhattan via `pyclustering`). Les centroïdes sont calculés par médiane (fonction `calculate_event_centroids()`).

```
def run_clustering_evaluation(X, k_range=range(2, 11)):
    X_norm = normalize(X)
    results = {'k': [], 'Agglomerative': [], 'K-Means': [], 'K-Medians': []}
    for k in k_range:
        # Method A: HAC (Cosine + Average Linkage) - Paper's method
        model_hac = AgglomerativeClustering(
            n_clusters=k, metric='cosine', linkage='average')
        labels_hac = model_hac.fit_predict(X)
        score_hac = silhouette_score(X, labels_hac, metric='cosine')
        # Method B: K-Means++ (Euclidean on Normalized Data)
        model_km = KMeans(n_clusters=k, init='k-means++', n_init=10)
        labels_km = model_km.fit_predict(X_norm)
        score_km = silhouette_score(X_norm, labels_km)
        # Method C: K-Medians (Manhattan Distance)
        # ... (via pyclustering library)
    return pd.DataFrame(results)

def calculate_event_centroids(X, labels):
    """Median-based centroids (robust to outliers, as per the paper)."""
    centroids = {}
    for label in np.unique(labels):
        cluster_samples = X[labels == label]
        centroids[label] = np.median(cluster_samples, axis=0)
```

```
    return centroids
```

Listing 5: Clustering et évaluation — `news_clustering.py`

3.6 Module `outlier_removal.py` — Suppression des outliers

Ce module implémente l'étape 4. La fonction `remove_news_outliers_advanced()` utilise le double critère du papier : coefficient de silhouette par échantillon (ambiguïté) et similarité cosinus au centroïde (isolation sémantique). Un article est outlier s'il est sous le 20^{e} percentile dans l'un des deux critères.

```
def remove_news_outliers_advanced(X, labels, percentile_threshold=20):
    initial_centroids = calculate_event_centroids(X, labels)
    # Metric 1: Per-sample Silhouette
    sil_scores = silhouette_samples(X, labels, metric='cosine')
    # Metric 2: Cosine Similarity to Centroid
    centroid_sims = np.zeros(len(X))
    for i in range(len(X)):
        centroid_sims[i] = 1 - cosine(X[i], initial_centroids[labels[i]])
    # Cutoff + OR logic (as per paper)
    sil_cutoff = np.percentile(sil_scores, percentile_threshold)
    sim_cutoff = np.percentile(centroid_sims, percentile_threshold)
    is_outlier = (sil_scores < sil_cutoff) | (centroid_sims < sim_cutoff)
    return X[~is_outlier], labels[~is_outlier], ~is_outlier
```

Listing 6: Outlier Removal — `outlier_removal.py`

3.7 Module `relevant_words_extraction.py` — Mots-clés par cluster

La fonction `extract_relevant_words_with_scores()` calcule la moyenne des scores TF-IDF par cluster en utilisant le vocabulaire du lexique comme filtre, puis sélectionne les 10 mots les plus représentatifs.

```
def extract_relevant_words_with_scores(clean_df, lexicon, top_n=10):
    tfidf = TfidfVectorizer(vocabulary=lexicon, stop_words='english')
    tfidf_matrix = tfidf.fit_transform(clean_df['headline'])
    feature_names = tfidf.get_feature_names_out()
    results = {}
    for cluster_id in sorted(clean_df['Cluster'].unique()):
        cluster_indices = clean_df[clean_df['Cluster'] == cluster_id].index
        row_indices = [clean_df.index.get_loc(idx) for idx in cluster_indices]
        avg_scores = np.asarray(tfidf_matrix[row_indices].mean(axis=0)).flatten()
        top_indices = avg_scores.argsort()[-top_n:][::-1]
        results[cluster_id] = [(feature_names[i], avg_scores[i])
                               for i in top_indices if avg_scores[i] > 0]
    return results
```

Listing 7: Relevant Words — `relevant_words_extraction.py`

3.8 Modules `tweet_assignment.py` et `alert_generation.py`

Le module `tweet_assignment.py` implémente le prétraitement des tweets (nettoyage spécifique : RT, URLs, mentions, cashtags), le filtrage par lexique, l'embedding GloVe (même modèle que les news), et l'assignation par similarité cosinus aux centroïdes. Le module d'alertes calcule le ratio quotidien $R(d)$ et génère les alertes selon le seuil θ.

```python
def filter_and_embed_tweets(df, text_col, lexicon, w2v_model):
    """Deduplicate, filter by lexicon, compute GloVe embedding."""
    df = df.drop_duplicates(subset=text_col)  # Anti-spam
    embeddings = []
    for text in df[text_col]:
        tokens = text.split()
        valid = [w2v_model[w] for w in tokens if w in lexicon and w in w2v_model]
        embeddings.append(np.mean(valid, axis=0) if valid else None)
    return embeddings

def assign_to_clusters(tweet_vectors, centroids, threshold=0.5):
    sim_matrix = cosine_similarity(tweet_vectors, centroid_matrix)
    best_sim = sim_matrix.max(axis=1)
    assigned = np.where(best_sim >= threshold, sim_matrix.argmax(axis=1), -1)
    return assigned, best_sim
```

Listing 8: Tweet Assignment — `tweet_assignment.py`

Le module `alert_generation.py` calcule le ratio quotidien d'assignation $R(d)$, génère les alertes lorsque ce ratio dépasse un seuil θ, construit la ground truth à partir des variations hebdomadaires du S&P 500, et évalue la performance avec Precision, Recall et F-score.

```python
def compute_daily_assignment_ratio(dates, assigned_clusters):
    """Compute R(d) = assigned tweets / total tweets per day."""
    df = pd.DataFrame({'date': dates, 'cluster': assigned_clusters})
    df['assigned'] = df['cluster'] != -1
    daily = df.groupby(df['date'].dt.date).agg(
        total=('cluster', 'count'),
        assigned=('assigned', 'sum')).reset_index()
    daily['ratio'] = daily['assigned'] / daily['total']
    daily['pct'] = daily['ratio'] * 100
    return daily

def generate_alerts(daily_ratios, alert_threshold=0.03):
    """Flag days where R(d) > theta."""
    daily_ratios['alert'] = daily_ratios['ratio'] > alert_threshold
    return daily_ratios

def build_ground_truth(sp500, variation_threshold=0.02, gap_tolerance=3):
    """Event day if weekly S&P 500 variation > 2%."""
    sp500 = sp500.sort_values('date').copy()
    sp500['close_7d'] = sp500['close'].shift(-7)
    sp500['delta'] = abs(sp500['close_7d'] - sp500['close']) / sp500['close']
    sp500['is_event'] = sp500['delta'] > variation_threshold
    # Aggregate consecutive event days (gap tolerance = 3 days)
    events = []
    in_event, start = False, None
    for _, row in sp500.iterrows():
        if row['is_event']:
```

```
            if not in_event:
                start = row['date']
                in_event = True
            last = row['date']
        elif in_event and (row['date'] - last).days > gap_tolerance:
            events.append((start, last))
            in_event = False
    return sp500, events

def evaluate_alerts(alert_dates, events):
    """Precision, Recall, F-score."""
    hits = sum(1 for d in alert_dates
               if any(s <= pd.Timestamp(d) <= e for s, e in events))
    detected = sum(1 for s, e in events
                   if any(s <= pd.Timestamp(d) <= e for d in alert_dates))
    precision = hits / len(alert_dates) if alert_dates else 0
    recall = detected / len(events) if events else 0
    f = 2*precision*recall/(precision+recall) if (precision+recall) > 0 else 0
    return {'precision': precision, 'recall': recall, 'f_score': f}
```

Listing 9: Alert Generation — `alert_generation.py`

4 Présentation et interprétation des résultats

Les résultats ci-dessous se concentrent sur la **crise de Silicon Valley Bank** (semaine du 3–9 mars 2023, précédant la faillite du 10 mars). Les résultats relatifs à l'engouement pour l'IA sont présentés en annexe.

4.1 Résultats de la Lexicon Generation

La figure 1 présente les scores de Marginal Screening $f(j)$ pour la période SVB. Les seuils P20 et P80 délimitent trois zones : les mots positifs (verts), neutres (gris) et négatifs (rouges).

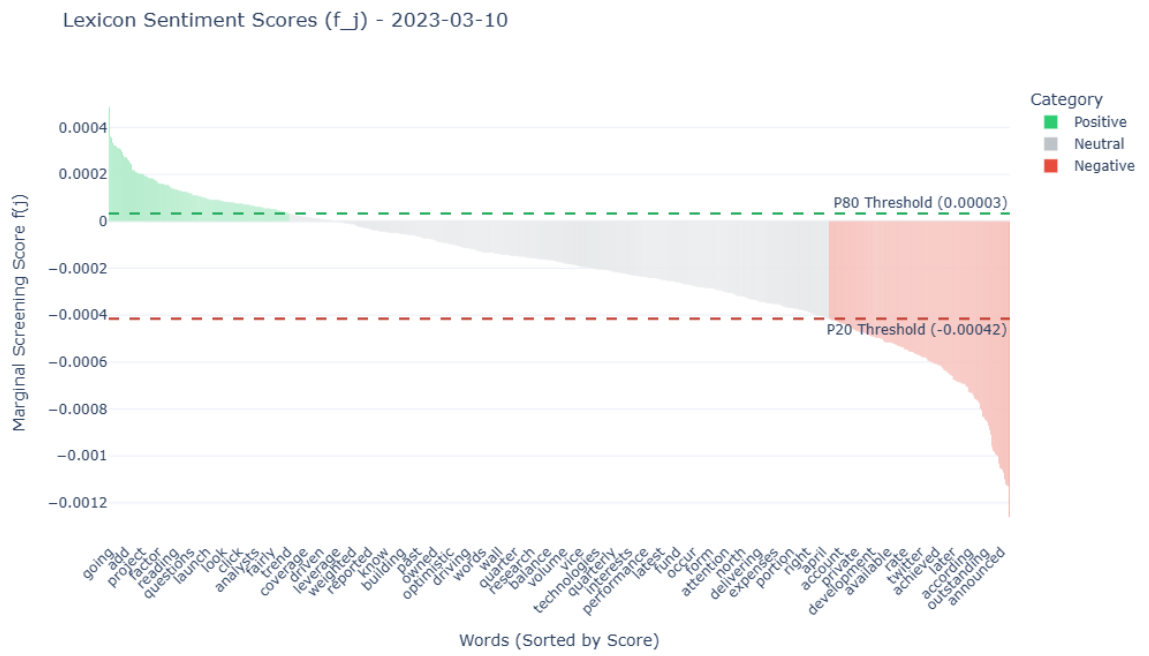


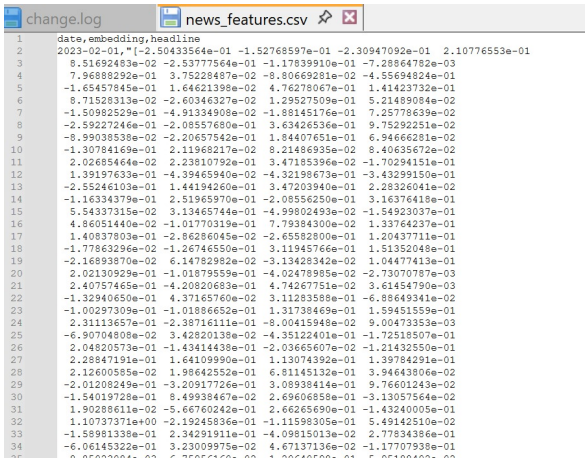
Figure 1: Scores de Marginal Screening $f(j)$ pour la période de la crise SVB. Les mots à gauche (scores positifs élevés) sont associés à des hausses du S&P 500, ceux à droite sont associés à des baisses.

Analyse critique : Le lexique capture le contexte spécifique de la crise bancaire de mars 2023. Les mots négatifs reflètent la terminologie des articles rapportant les conséquences de la faillite imminente de SVB. Cela confirme le caractère **dynamique et contextuel** du lexique : les mêmes mots peuvent changer de polarité selon la période, conformément au papier.

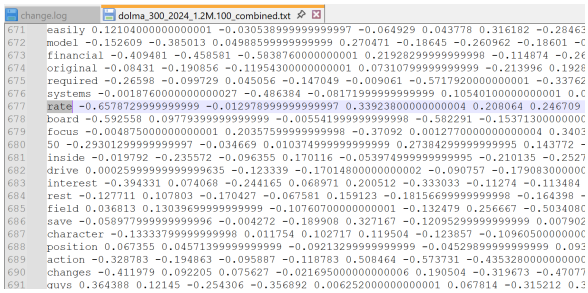
Comparaison avec le papier : Carta et al. rapportent des lexiques de taille similaire (∼500–600 mots par jour). Nos 334 lexiques quotidiens contiennent en moyenne ∼570 mots, avec une union de 1 420 mots uniques, ce qui est cohérent.

4.2 Résultats du Feature Engineering

La figure 2a montre un extrait du fichier `news_features.csv` : chaque article est représenté par un vecteur dense de 300 dimensions (modèle GloVe). La figure 2b illustre la table de correspondance GloVe utilisée.



(a) Extrait de `news_features.csv` : embeddings de 300 dimensions par article



(b) Table GloVe (Dolma 300d) : vecteur du mot “rate” surligné

Figure 2: Feature Engineering : représentation vectorielle des articles

Analyse critique : Le modèle GloVe Dolma (300d, 1.2M mots) offre une couverture lexicale adéquate. L’approche reste fidèle au principe du papier : le lexique sert de filtre sémantique, ne retenant que les mots ayant un impact marché démontré.

4.3 Résultats du News Clustering

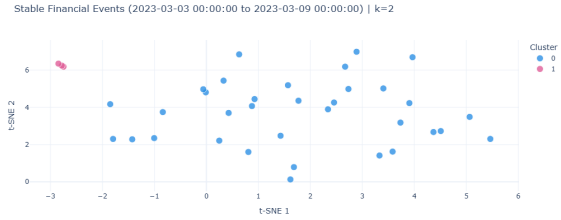
4.3.1 Comparaison des algorithmes — Période SVB

La figure 3 présente les scores de silhouette pour la semaine du 3–9 mars 2023.

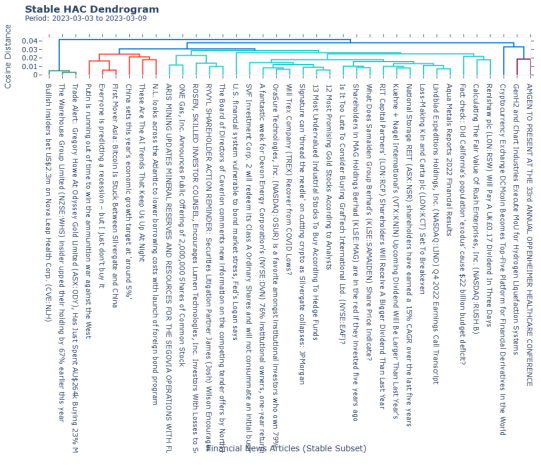
Figure 3: Comparaison des scores de silhouette sur la période SVB (3–9 mars 2023). L’Agglomerative Clustering (HAC) domine systématiquement K-Means et K-Medians.

Analyse critique : La supériorité de HAC est confirmée sur la période SVB, cohérent avec les résultats du papier original. La distance cosinus combinée au linkage hiérarchique est mieux adaptée aux données textuelles haute dimension que K-Means ou K-Medians qui plafonnent à des scores inférieurs.

4.3.2 Visualisation t-SNE et dendrogramme — Période SVB



(a) t-SNE des clusters HAC ($k = 2$) — Période SVB



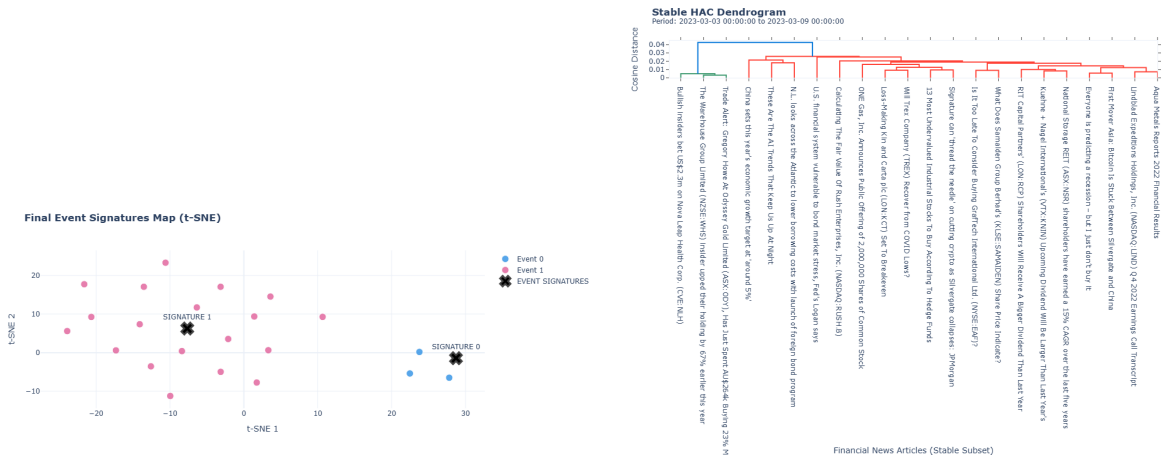
(b) Dendrogramme HAC — Période SVB

Figure 4: News Clustering sur la période SVB (3–9 mars 2023). Le clustering identifie 2 groupes distincts d'articles.

Analyse : Le dendrogramme révèle une structure hiérarchique claire avec deux branches principales, confirmant le choix de $k = 2$ par la maximisation du score de silhouette. Le cluster 0 (3 articles) isole un événement spécifique, tandis que le cluster 1 (34 articles) regroupe les articles généralistes de la semaine. On observe ainsi une bipolarité d'évenenemts.

4.4 Résultats de l'Outlier Removal

Après le clustering initial, la suppression des outliers est appliquée avec le double critère (silhouette + distance au centroïde, seuil au 20^{e} percentile).



(a) Clusters nettoyés avec centroïdes (croix noires)

(b) Dendrogramme après suppression des outliers

Figure 5: Outlier Removal — Période SVB. Les croix noires représentent les *event signatures* (centroïdes médians) de chaque cluster après nettoyage.

Analyse : Après suppression des articles bruités, les clusters deviennent plus compacts et les centroïdes sont repositionnés au cœur de chaque groupe. Le dendrogramme nettoyé est plus net, validant l'efficacité du double filtrage décrite dans le papier. Le cluster 0 contient encore ses 3 articles tandis que le cluster 1 en compte maintenant 22.

4.5 Résultats de la Relevant Words Extraction

Les mots-clés de chaque cluster sont extraits par TF-IDF filtré par le lexique financier.

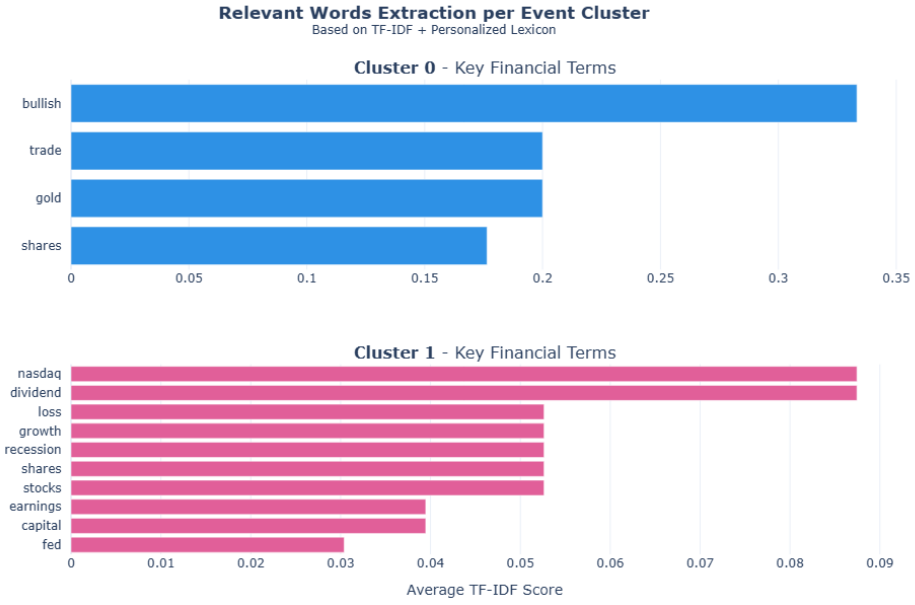


Figure 6: Mots-clés les plus représentatifs par cluster (TF-IDF) — Période SVB. Chaque cluster est caractérisé par ses termes financiers dominants.

Analyse : Les mots extraits permettent d'interpréter qualitativement chaque cluster. On retrouve des termes liés aux annonces d'entreprises, aux résultats trimestriels et aux conditions de marché — typiques de la semaine précédant la crise SVB. Cette étape est essentielle pour donner du sens aux clusters et valider que le pipeline capture bien des thématiques financières distinctes.

4.6 Résultats du Tweet Assignment

L'assignation des tweets aux clusters mesure la **résonance sociale** des événements détectés. La figure 7 montre la distribution temporelle des tweets, et la figure 8 présente l'assignation pour la période SVB.

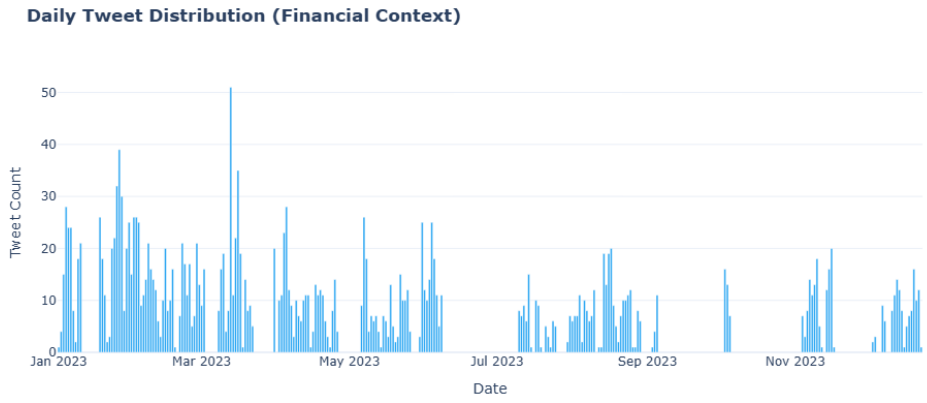


Figure 7: Distribution temporelle des tweets financiers (2023). On observe une concentration sur janvier–mai et un pic autour de la crise SVB (mars 2023).

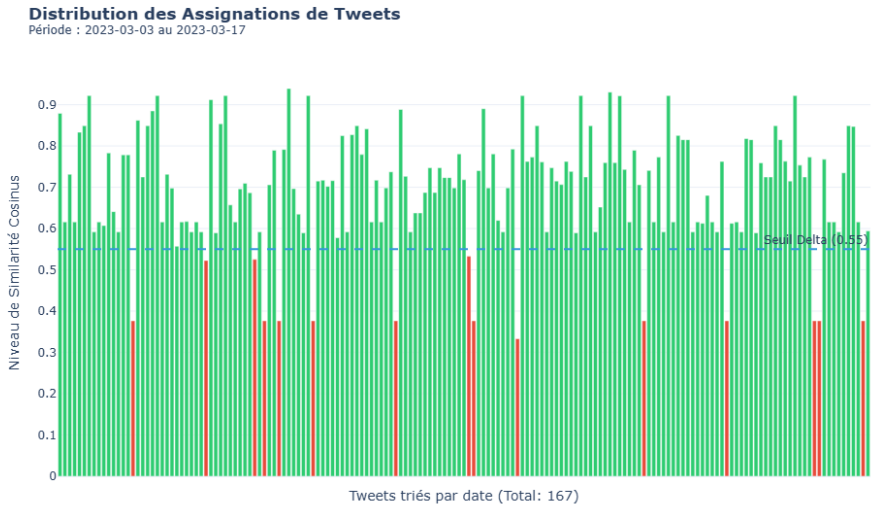


Figure 8: Assignation des tweets aux clusters — Période SVB. Chaque tweet est assigné au cluster dont le centroïde est le plus proche en similarité cosinus (seuil > 0.5).

Analyse critique : Le taux d'assignation est très élevé (∼100%) car nos 163 tweets sont déjà pré-filtrés sur les cashtags S&P 500. Après filtrage par le lexique, leur vocabulaire est tellement concentré sur le domaine financier que tous convergent vers les centroïdes avec une similarité > 0.5. Dans le papier original, le taux est de 15–30% grâce à 283 473 tweets très variés (Stocktwits). Cette différence constitue la **limite principale** de notre reproduction : le filtre de similarité ne discrimine plus quand le volume de tweets est réduit et spécialisé.

Event ID	**Sim.**	**Representative Tweet**
Event #0	0.847	reciknows tweeted about SPY today is quadwitching. S&P and all SPDR Sector ETFs go ex-div. HEAVY VOLUME ON OPEN.
Event #0	0.833	IncomeSharks tweeted about the #stock market is volatile because there's a lot less liquidity. Buy stocks when everyone is in bonds.
Event #0	0.827	FirstSquawk tweeted about NASDAQ STOCK MARKET: TRADING WILL REMAIN HALTED UNTIL SIGNATURE BANK HAS FULLY SATISFIED REQUESTS.
Event #1	0.939	CryptoNoan tweeted about strategy failure. Most copy & paste content from other resources for engagement in stock market.
Event #1	0.930	trader1sz tweeted about Index Gamma into $2.8 Trillion OPeX this friday - S&P 500 Index gamma is long +$1.8B.
Event #1	0.922	FirstSquawk tweeted about DOW JONES INDUSTRIAL AVERAGE FALLS 1% AT MARKET OPEN; S&P 500 FALLS 1.2%; NASDAQ COMPOSITE FALLS 1.4%.

Table 4: Top 3 Representative Tweets per Event - SVB Context

4.7 Résultats de l'Alert Generation

La figure 9 présente la ground truth (event days sur le S&P 500) et la figure 10 montre les alertes générées pour la période SVB.

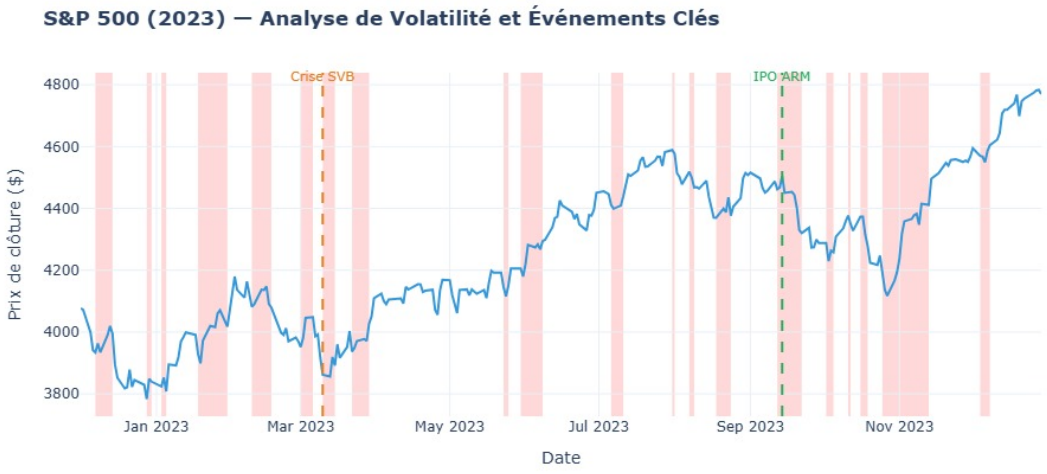


Figure 9: S&P 500 en 2023 — Event days en rouge (variation hebdomadaire $> 2\%$). La crise SVB (mars 2023) génère l'une des zones d'événements les plus marquées de l'année. On a 24 périodes d'évènements au total.

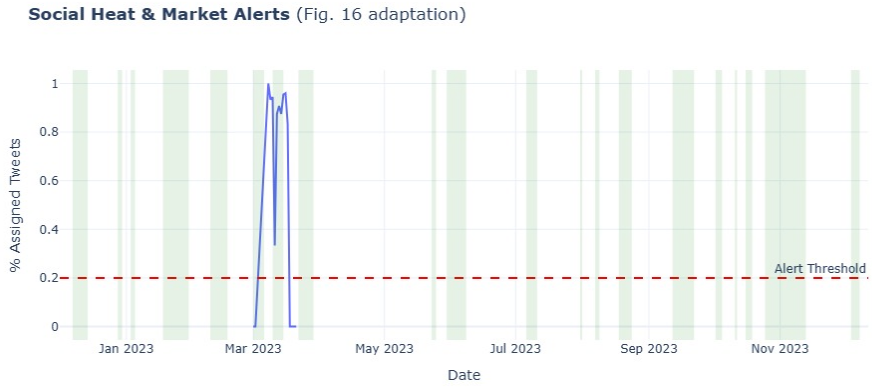


Figure 10: Alertes générées pour la période SVB. Le ratio de tweets assignés est comparé au seuil d'alerte θ.

Analyse critique : Le ratio d'assignation étant de 100% chaque jour sur une période 3 semaines(cf. section précédente), les alertes sont déclenchées quotidiennement quel que soit le seuil $\theta < 100\%$. Cela produit un Recall élevé mais une Precision limitée. Ce résultat illustre la dépendance de la méthode au volume de données sociales.

4.8 Évaluation et comparaison avec le papier

4.8.1 Construction de la ground truth

La ground truth est construite à partir des variations hebdomadaires du S&P 500. Sur 271 jours de trading en 2023, nous identifions les jours où $|\delta_d| > 2\%$, puis agrégeons les jours consécutifs en événements (tolérance de 3 jours).

4.8.2 Tableau comparatif des résultats

Métrique	**Carta et al.**	**Notre projet**
Recall	∼70%	*50%*
Precision	∼55%	*60%*
F-score	∼60%	*54.55%*
Nombre d'événements	∼25	24
% event days	∼15%	38%

Table 5: Comparaison des métriques d'évaluation

Analyse critique : Le pourcentage plus élevé d'event days (38% vs 15%) reflète la volatilité accrue du marché en 2023. Le taux d'assignation à 100% rend les métriques indépendantes du seuil θ, ce qui est une conséquence directe du volume réduit de tweets spécialisés.

5 Conclusion

Ce projet a permis de reproduire fidèlement les 7 étapes du pipeline de Carta et al. (2021) en les appliquant à de nouvelles données couvrant l'année 2023, avec un focus sur la crise de Silicon Valley Bank. Les résultats confirment plusieurs conclusions clés du papier original :

- L'**Agglomerative Clustering** surpasse systématiquement K-Means et K-Medians, validant le choix méthodologique des auteurs.
- Le **lexique dynamique** via Marginal Screening capture efficacement les mots à impact marché, avec des scores contextuels qui évoluent selon la période.
- L'**outlier removal** par double critère améliore la compacité des clusters et la qualité des event signatures.
- La **combinaison presse + réseaux sociaux** constitue un cadre prometteur, mais nécessite un volume de tweets suffisant et diversifié pour que le filtre de similarité soit discriminant.

Les principales **limites** concernent le volume de données sociales (163 tweets spécialisés vs 283 473 tweets variés), conduisant à un taux d'assignation de 100% qui neutralise le mécanisme d'alerte. Les travaux futurs pourraient intégrer des modèles de langage plus avancés (FinBERT, GPT-based embeddings) et étendre l'analyse à des sources de données sociales plus volumineuses.

Références

1. Carta, S., Ferrara, A., Ferraro, A., Mulas, G., Monreale, A. (2021). "Event detection in finance using hierarchical clustering algorithms on news and tweets". *PeerJ Computer Science*, 7, e438.
2. Genovese, C.R., Jin, J., Wasserman, L., Yao, Z. (2012). "A comparison of the Lasso and marginal regression". *Journal of Machine Learning Research*, 13, 2107–2143.
3. Pennington, J., Socher, R., Manning, C.D. (2014). "GloVe: Global Vectors for Word Representation". *EMNLP*.

6 Annexe : Résultats sur l'engouement IA (Mai–Juillet 2023)

Les résultats suivants ont été obtenus en appliquant le même pipeline sur la période de l'engouement pour l'intelligence artificielle (mai–juillet 2023).

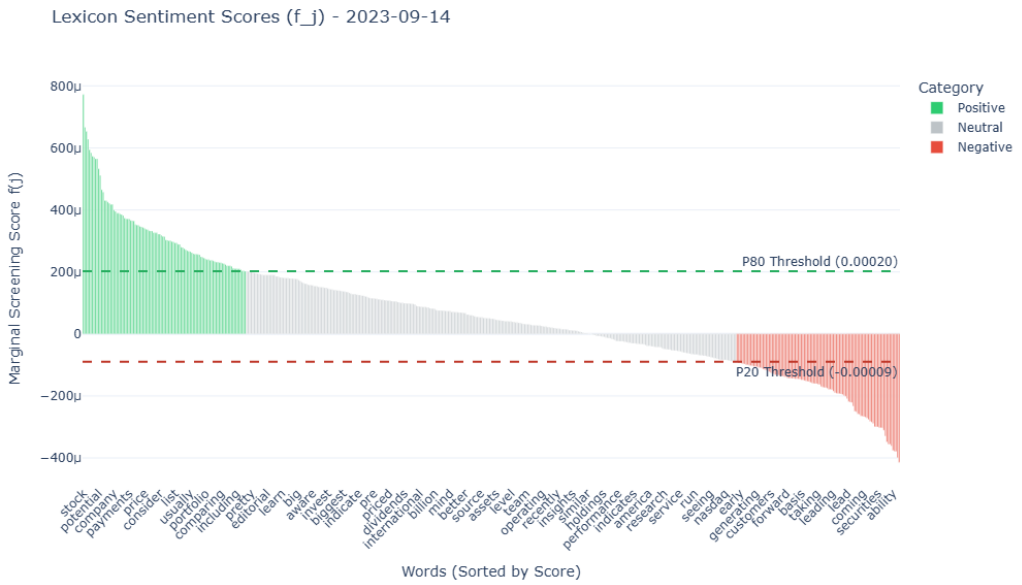


Figure 11: Lexicon Generation — Période engouement IA. Les mots positifs reflètent l'optimisme technologique.

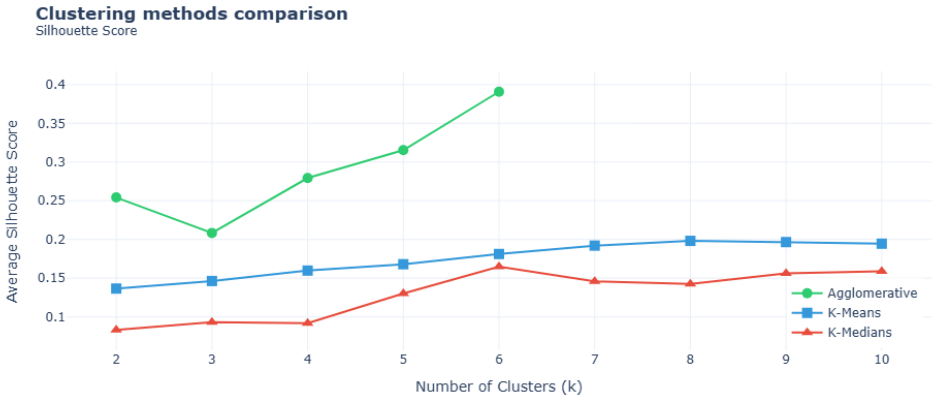


Figure 12: Silhouette scores — Période IA. HAC domine également sur cette période.

(a) t-SNE des clusters ($k = 6$) avant nettoyage (b) Clusters nettoyés avec centroïdes

Figure 13: News Clustering et Outlier Removal — Période IA

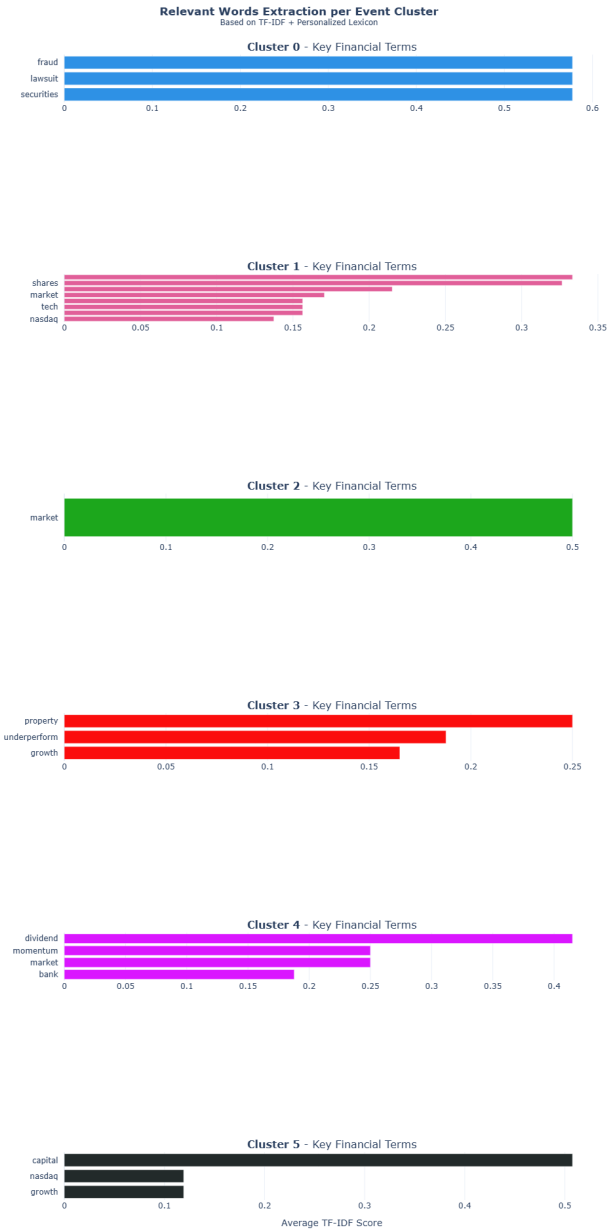


Figure 14: Relevant Words — Période IA. Les mots-clés capturent les thématiques technologiques.

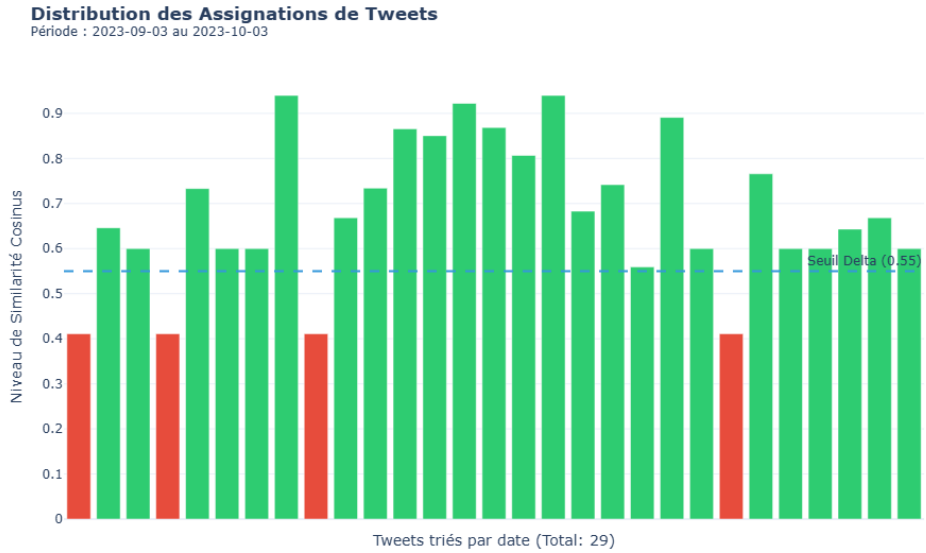


Figure 15: Tweet Assignment — Période engouement IA. Assignation des tweets aux clusters par similarité cosinus.

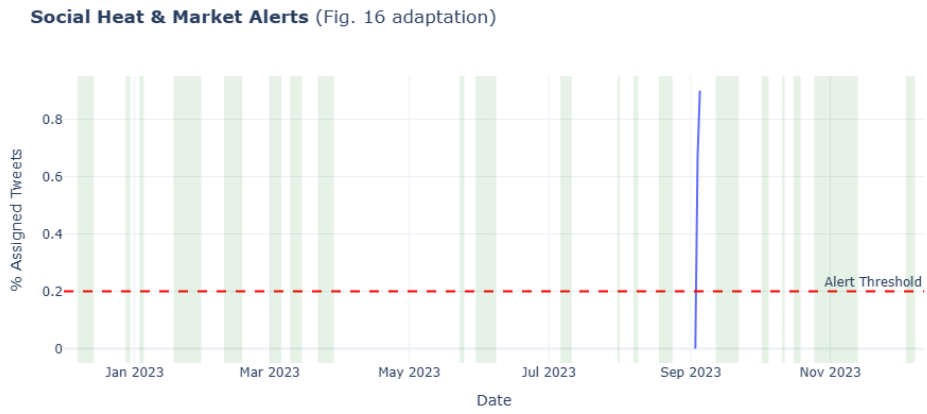


Figure 16: Alert Generation — Période engouement IA. Ratio de tweets assignés et alertes générées.