Documentazione – “Crypto Report”

Gruppo di lavoro:

- Gioele Panico, MAT: 757274, g.panico16@studenti.uniba.it

Link GitHub: Progetto-“CryptoReport”

AA 2024-25

Indice

NOTA: sono evidenziati in grassetto i capitoli contenenti gli argomenti rilevanti per il corso, i capitoli 2 e 6 sono stati inclusi solo per far capire il contesto del progetto.

1. Introduzione

Il progetto "crypto-report" si propone di sviluppare un sistema intelligente per l'analisi automatizzata delle notizie del settore delle criptovalute. L'obiettivo principale è creare un agente in grado di recuperare, elaborare e sintetizzare le informazioni provenienti da fonti affidabili, per generare un report settimanale che evidenzi gli eventi più rilevanti e individui potenziali correlazioni e tendenze nel mondo crypto.

1.1. Descrizione generale del sistema.

Il progetto si propone di sviluppare un sistema intelligente per l'analisi delle notizie del settore delle criptovalute, combinando tecniche di web scraping, machine learning, rappresentazione della conoscenza e ragionamento automatico.

Le principali componenti del sistema sono:

- **Raccolta delle Notizie:**
  - Viene utilizzata l'API di CryptoPanic per ottenere i link degli articoli originali.
  - I contenuti completi degli articoli vengono estratti tramite web scraping con Selenium.
- **Sintesi Automatica degli Articoli:**
  - Un modello LLama 3.2 appositamente addestrato (fine-tuned) viene utilizzato per generare riassunti degli articoli recuperati.
- **Classificazione degli Articoli:**
  - Gli articoli vengono classificati tramite algoritmi di clustering probabilistico nelle seguenti categorie:
    - *News di Mercato, Analisi e Prezzi*
    - *Regolamentazione e Normative*
    - *Adozione e Mainstreaming*
    - *Tecnologia, Innovazione e Nuovi Progetti*
    - *Sicurezza, Hackeraggi e Truffe*
    - *Non Rilevanti*
- **Costruzione di un Knowledge Graph:**
  - Gli articoli della settimana vengono elaborati per estrarre entità nominate (persone, criptovalute, istituzioni finanziarie, aziende).
  - Le relazioni tra entità e articoli vengono rappresentate attraverso un grafo, in cui i nodi sono entità e articoli, e gli archi rappresentano relazioni come *Fondatore, CEO, Presidente* e *Menzioni*.
- **Ragionamento Automatico con Prolog:**
  - Prolog viene utilizzato per interrogare il Knowledge Graph e individuare correlazioni tra gli articoli.
  - Viene fornita la possibilità di identificare articoli rilevanti, articoli correlati e le entità più citate nella settimana.

1.2. Obiettivo del sistema

L'obiettivo del sistema è quello di creare un report settimanale avente il seguente formato:

Crypto Recap – [(YYYY/MM/DD-7)– (YYYY/MM/DD)]

Top 3 articoli:

- [titolo articolo]
  - [riassunto lungo]
  - [categoria] [data]
- [titolo articolo]
  - [riassunto lungo]
  - [categoria] [data]
- [titolo articolo]
  - [riassunto lungo]
  - [categoria] [data]

News di Mercato, Analisi e Prezzi:

- [titolo articolo]
  - [riassunto corto]
  - [categoria] [data]
- [titolo articolo]
  - [riassunto corto]
  - [categoria] [data]

Regolamentazione e Normative:

- [titolo articolo]
  - [riassunto corto]
  - [categoria] [data]
- [titolo articolo]
  - [riassunto corto]
  - [categoria] [data]

….

(stessa struttura per le categorie "Adozione e Mainstreaming", "Tecnologia, Innovazione e Nuovi Progetti" e "Sicurezza, Hackeraggi e Truffe". Ovviamente ogni categoria sarà inclusa solo se ci sono articoli rilevanti per quella categoria)

Trend di Mercato:

(Un elenco delle entità più citate negli articoli)

- [entità più citata]
- [entità più citata]
- [entità più citata]

1.3. Elenco argomenti di Interesse

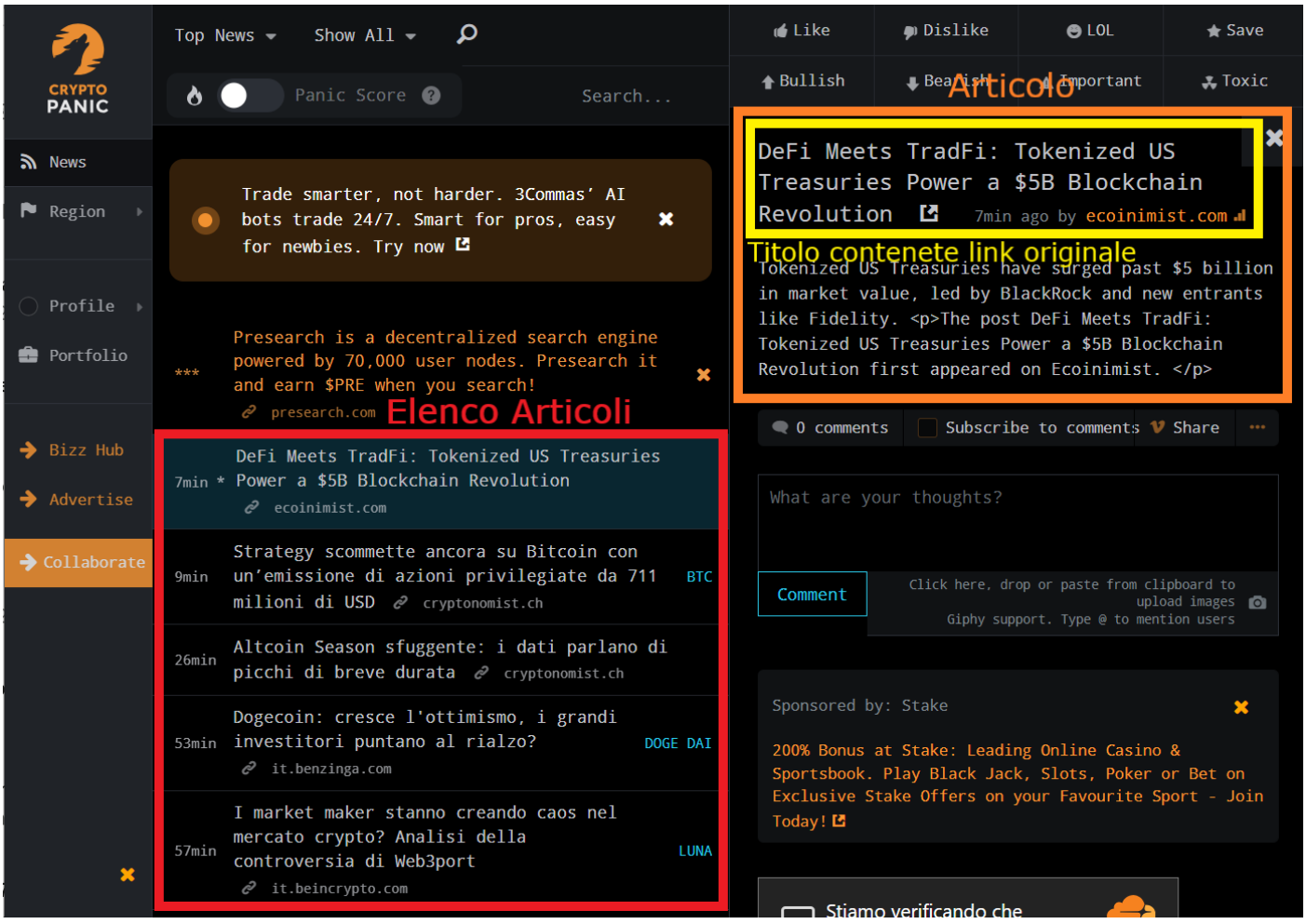
- Apprendimento Supervisionato: fine-tuning della rete neurale LlaMa 3.2 3B di tipo Transformer Networks.
- Classificazione probabilistica semi-supervisionata: chiamata semi-supervisionata perché per identificare le categorie degli articoli viene utilizzato l'algoritmo di cluster K-Means (algoritmo non supervisionato) e i cluster identificati vengono poi puliti manualmente per addestrare il classificatore NaiveBayes (algoritmo supervisionato).
- Rappresentazione della Conoscenza (Knowledge Graph): Creazione e aggiornamento di un Knowledge Graph che rappresenta entità del mondo crypto e le relazioni fra di esse (Founder, CEO, President, ecc.).
- Ragionamento Automatico: Uso di Prolog per eseguire inferenze logiche sui dati del Knowledge Graph, tra cui identificazione degli articoli rilevanti, scoperta di correlazioni e tendenze emergenti.

2. Raccolta e Preprocessing degli Articoli

(Prima di procedere nel dettaglio con gli argomenti del corso un breve capitolo per capire come siamo arrivati alle fasi successive.)

2.1. Recupero informazioni Articoli

Per il recupero degli articoli è stato utilizzato il servizio di **CryptoPanic**, un aggregatore di articoli specifico per il settore delle criptovalute:



Tuttavia, l'API gratuita di CryptoPanic non fornisce direttamente i link agli articoli originali, ma solo i collegamenti alla propria piattaforma. Per poter ottenere i link diretti agli articoli e il loro contenuto completo, è stato necessario implementare una soluzione alternativa tramite **Web Scraping**.

Per questo scopo, è stato utilizzato **Selenium**, una libreria Pyhton usata per il web scraping. Il processo seguito è stato il seguente:

1. **Richiesta API di CryptoPanic:**
   L'API di CryptoPanic viene interrogata per recuperare un elenco di notizie recenti. Ogni notizia viene restituita con diverse informazioni tra cui il titolo, la data e l'URL

che punta alla pagina di CryptoPanic stessa, dove viene mostrato l'articolo con il titolo contenente il link alla fonte originale.

2. **Recupero dei Link Originali:**
   Selenium viene utilizzato per accedere dinamicamente alla pagina di CryptoPanic relativa a ciascun articolo e individuare il link originale contenuto all'interno del titolo dell'articolo. Questo link originale viene estratto per l'uso successivo.

3. **Estrazione del Contenuto dell'Articolo:**
   Una volta ottenuto il link originale, Selenium viene utilizzato per accedere alla pagina web del sito di destinazione e raccogliere il contenuto testuale principale. L'estrazione viene effettuata identificando e salvando tutti i tag HTML più rilevanti per il testo come <h1>, <h2>, e <p>. Viene così generato un contenuto grezzo che sarà ulteriormente elaborato nelle fasi successive del progetto.

Questo processo consente di ottenere un dataset di articoli originali provenienti da svariate fonti giornalistiche che trattano argomenti di interesse nel mondo delle criptovalute.

2.2. Salvataggio delle informazioni degli articoli

Tutte le informazioni estratte sono state salvate nel db chiamato "crypto_news.db" , un db in sqlite avente due tabelle, una chiamata meta_articoli, che contiene tutte le informazioni relative ai meta, e una chiamata "descrizione_articoli" che contiene le informazioni testuali per l'articolo.

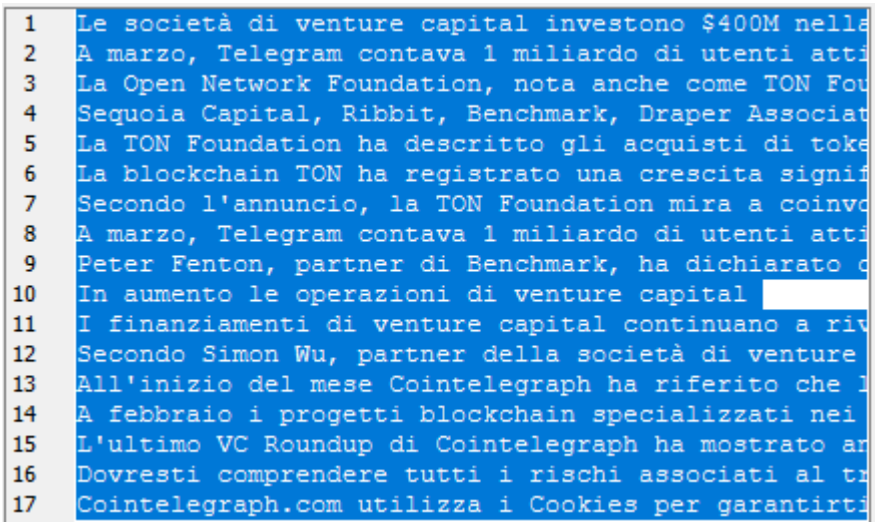
Tabella: meta_articoli

	id	img_url	cryptopanic_url	original_url	data
	Filtro	Filtro	Filtro	Filtro	Filtro
1489	1489	https://en.cryptonomist.ch/wp-…	https://cryptopanic.com/news/…	https://cryptonomist.ch/2025/03/21/…	2025/03/21
1490	1490	https://s44485.pcdn.co/wp-content/…	https://cryptopanic.com/news/…	https://it.beincrypto.com/mercati/…	2025/03/21
1491	1491	https://s44485.pcdn.co/wp-content/…	https://cryptopanic.com/news/…	https://it.beincrypto.com/mercati/…	2025/03/21
1492	1492	https://images.cointelegraph.com/…	https://cryptopanic.com/news/…	https://it.cointelegraph.com/news/75…	2025/03/21
1493	1493	https://s44485.pcdn.co/wp-content/…	https://cryptopanic.com/news/…	https://it.beincrypto.com/mercati/…	2025/03/21
1494	1494	https://images.cointelegraph.com/…	https://cryptopanic.com/news/…	https://it.cointelegraph.com/news/…	2025/03/21
1495	1495	https://www.spaziocrypto.com/content…	https://cryptopanic.com/news/…	https://www.spaziocrypto.com/xrp/…	2025/03/21
1496	1496	https://s44485.pcdn.co/wp-content/…	https://cryptopanic.com/news/…	https://it.beincrypto.com/mercati/…	2025/03/21
1497	1497	https://en.cryptonomist.ch/wp-…	https://cryptopanic.com/news/…	https://cryptonomist.ch/2025/03/21/…	2025/03/21
1498	1498	https://en.cryptonomist.ch/wp-…	https://cryptopanic.com/news/…	https://cryptonomist.ch/…	2025/03/21
1499	1499	https://s44485.pcdn.co/wp-content/…	https://cryptopanic.com/news/…	https://it.beincrypto.com/mercati/…	2025/03/21

Tabella: descrizione_articoli

	id_articolo	titolo	full_article_html
	Filtro	Filtro	Filtro
1487	1487	Le società di venture capital …	Le società di venture capital …
1488	1488	BaFin blocca la vendita del token …	BaFin blocca la vendita del token …
1489	1489	Tornado Cash rimosso dalla lista ner…	Tornado Cash rimosso dalla lista ner…
1490	1490	Le balene crypto hanno acquistato …	Le balene crypto hanno acquistato …
1491	1491	I regolatori tedeschi respingono la …	I regolatori tedeschi respingono la …
1492	1492	Il 75% dei VASP registrati …	Il 75% dei VASP registrati …
1493	1493	Il crollo di 3 mesi di Shiba Inu …	Il crollo di 3 mesi di Shiba Inu …
1494	1494	Hacker ruba $8,4M dal protocollo di …	Hacker ruba $8,4M dal protocollo di …
1495	1495	Vittoria Legale Di Ripple: XRP Non …	Vittoria Legale Di Ripple: XRP Non …
1496	1496	Questa settimana nelle memecoin: SPX…	Questa settimana nelle memecoin: SPX…
1497	1497	Siren sale del 30% in un giorno …	Siren sale del 30% in un giorno …
1498	1498	Le 10 migliori meme coin di BNB da …	Le 10 migliori meme coin di BNB da …
1499	1499	Il volume del DEX di XRP Ledger …	Il volume del DEX di XRP Ledger …

Esempio di full_article_html:



Quindi la tabella descrizione_articoli conterrà un elenco di informazioni testuali degli articoli tra cui anche il contenuto dell' articolo estratto precedentemente con Selenium.

Oltre a queste colonne, la tabella decrizione_articoli contiene anche altre colonne che verranno popolate successivamente come "short_resume", che contiene un breve riassunto dell articolo (circa 50 parole), "long_resume" che contiene un riassunto più dettagliato dell' articolo (circa 200/250 parole) e anche la colonna "category" che conterrà la categoria dell' articolo.

3. Apprendimento Supervisionato – Fine Tuning di LlaMa 3.2

Dopo aver raccolto gli articoli e il loro contenuto, è stato eseguito il fine-tuning del modello LLaMa 3.2 3B per adattarlo alla generazione automatica di riassunti, sia brevi che lunghi, degli articoli recuperati. Il modello LLaMa 3.2 3B è un modello basato su Transformer Networks con 3 miliardi di parametri, progettato per comprendere e generare linguaggio naturale con un buon compromesso tra accuratezza ed efficienza computazionale.
L'obiettivo del fine-tuning è quello di migliorare la capacità del modello di estrarre e sintetizzare le informazioni principali da un articolo, producendo due tipi di riassunti:

- **Riassunti brevi**: Sintesi compatte degli articoli per un'anteprima rapida.
- **Riassunti lunghi**: Sintesi più dettagliate e complete per un'analisi approfondita.

Per ulteriori dettagli sulla struttura e caratteristiche del modello, si rimanda alla documentazione di LLaMa 3.2.

3.1 Strumenti utilizzati

L'addestramento del modello è stato eseguito su Google Colab, una piattaforma che mette a disposizione GPU gratuite.

Le principali librerie utilizzate per l'implementazione del modello e il processo di addestramento sono le seguenti:

- **unsloth** - Caricamento e fine-tuning del modello LLaMa 3.2 3B.
- **torch (PyTorch)** - Calcolo numerico e utilizzo della GPU per l'ottimizzazione del modello.
- **pandas** - Gestione e manipolazione del dataset.
- **datasets (Hugging Face)** - Conversione e gestione del dataset compatibile con i modelli Hugging Face.
- **trl (Transformers Reinforcement Learning)** - Training supervisionato tramite SFTTrainer.
- **transformers (Hugging Face)** - Configurazione dei parametri di addestramento.
- **sklearn (Scikit-Learn)** - Suddivisione del dataset in training e validation set.

Il codice completo utilizzato per l'addestramento è visibile al seguente link.

3.2 Creazione del dataset per il Fine-Tuning

Il dataset utilizzato per il fine-tuning è stato creato manualmente generando, tramite l'utilizzo di ChatGPT, sia un riassunto breve che uno lungo per ogni articolo raccolto.

Il dataset comprende circa 1000 record, divisi come segue:

- 500 record per la generazione di riassunti brevi.
- 500 record per la generazione di riassunti lunghi.

Il dataset è organizzato in un file CSV, con la seguente struttura:

- **User**, contenete il prompt e l'articolo completo
- **Summary**, contenente il riassunto che ci si aspetta di ricevere in output

Il dataset utilizzato è visibile nella cartella Dataset con il nome "summaryDataset.csv".

3.3 Selezione dei parametri e Fine-Tuning

La scelta dei parametri di addestramento è cruciale per ottenere buone prestazioni ed evitare problemi come overfitting o sotto-addestramento. Per ottimizzare il modello, è stata effettuata una ricerca greedy per identificare le migliori combinazioni di parametri.

3.3.1 Suddivisione del dataset in Training e Validation Set

```
from sklearn.model_selection import train_test_split
train_df, val_df = train_test_split(df, test_size=0.2, random_state=42)
```

Il dataset è stato suddiviso in:

- 80% Training Set (utilizzato per l'addestramento del modello)
- 20% Validation Set (usato per la valutazione durante il training)

3.3.2 Configurazione dei parametri per il Training Supervisionato

```
args=TrainingArguments(
        per_device_train_batch_size=2,
        gradient_accumulation_steps=4,
        warmup_steps=5,
        max_steps=100,
        learning_rate=0.0001,
        fp16=not is_bfloat16_supported(),
        bf16=is_bfloat16_supported(),
        optim="adamw_8bit",
        eval_strategy="steps",
        eval_steps=10,
        save_strategy="epoch",
        metric_for_best_model="eval_loss",
        greater_is_better=False,
        output_dir="outputs",
        report_to="none",
  )
```

3.3.3 Parametri principali e loro impatto

Learning Rate (`learning_rate`)

- Controlla la velocità di apprendimento del modello.
  E' un fattore moltiplicativo che regola di quanto vengono modificati i pesi del modello dopo ogni aggiornamento:

$$Weight_{new} = Weight_{old} - learning\ rate \times gradiente$$

  - $Weight$ sono i pesi del modello
  - Il $gradiente$ indica la direzione e la grandezza della modifica necessaria per ridurre la loss.
- Se è troppo alto, il modello impara rapidamente ma potrebbe sfuggire a minimi locali a causa delle grandi oscillazioni.
- Se è troppo basso, il modello potrebbe apprendere lentamente raggiungendo buone performance ma rischia di essere intrappolato in minimi locali.

Batch Size (`per_device_train_batch_size`)

- Definisce quanti campioni vengono processati contemporaneamente dalla rete neurale prima di aggiornare i pesi.
- Se è troppo piccolo aggiorna i pesi più frequentemente ma può causare oscillazioni grandi dei pesi poiché consdiera pochi esempi.
- Se è troppo grande aggiorna i pesi meno frequentemente e considerando più esempi rendendo il training più stabile (meno oscillzazioni) ma rende l'overfitting più probabile imparndo schemi troppo specifici ai dati di training. Inoltre non è applicabile richiede molta VRAM altrimenti può portare a errori di OOM (Out of memory)

Gradient Accumulation (`gradient_accumulation_steps`)

- Il modello accumula i gradienti per più batch prima di aggiornare i pesi.
  Consente di simulare batch più grandi senza aumentare il consumo di memoria.
  E' una tecnica comunemente usata con batch size:
  Esempio:
  - **Senza Gradient Accumulation:** batch_size=8 → Aggiornamento dei pesi dopo ogni 8 campioni.
  - **Con Gradient Accumulation** (steps=4, batch_size=2):
    - Il modello elabora 4 batch da 2 campioni → accumula i gradienti.
    - Solo alla fine dei 4 batch aggiorna i pesi.
    - Il risultato è equivalente a un batch di 8, ma senza caricare 8 campioni in memoria contemporaneamente!

Warmup Steps (`warmup_steps`)

- Controlla il numero di passi iniziali in cui il learning rate cresce gradualmente da zero fino al valore target. Utile perché quando il training inizia i pesi del modello sono casuali, quindi un learning rate alto può causare gradienti instabili.
- Un warmup basso (1-5%) è ideale per il fine-tuning di modelli già pre-addestrati.
- Un warmup alto (10-20%) è utile per modelli addestrati da zero.

Max Steps (`max_steps`)

- Determina la durata dell'addestramento.
- Troppi passi → rischio di overfitting.
- Pochi passi → rischio di sotto-addestramento.

3.4 Strategia di selezione dei parametri

Testiamo diverse combinazioni di Learning Rate, Batch Size e Gradient Accumulation. Mantenendo fissi:

- Max Steps = 100
- Warmup Steps = 5 (5% di Max Steps)

Le configurazioni generate sono le seguenti:

Config ID	Learning Rate (LR)	Batch Size (BS)	Gradient Accumulation (GA)	Effettivo Batch Size (BS * GA)
A	0.0002	8	8	64
B	0.0001	2	4	8
C	0.0005	4	4	16
D	0.0005	8	2	16
E	0.0005	2	2	4
F	0.0003	4	2	8
G	0.0001	2	8	16
H	0.00005	4	2	8
I	0.00005	8	2	16
J	0.0002	2	4	8
K	0.0004	4	4	16
L	0.003	6	3	18

3.5 Addestramento con le configurazioni generate

Config ID = A

Step	Training Loss	Validation Loss
10	0.960700	0.963486
20	0.882100	0.899476
30	0.750700	0.871818
40	0.741200	0.860979
50	0.669500	0.849382
60	0.612500	0.859935
70	0.591200	0.876461
80	0.574900	0.881073
90	0.480000	0.898862
100	0.491000	0.903284

Config ID = B

Step	Training Loss	Validation Loss
10	1.054300	1.008301
20	1.011500	0.938209
30	0.872600	0.919882
40	0.914300	0.900255
50	0.899900	0.888732
60	0.963100	0.878338
70	0.760800	0.870265
80	0.838700	0.864141
90	0.829200	0.860295
100	0.916300	0.858948

Config ID = C

Step	Training Loss	Validation Loss
10	0.951600	0.937523
20	0.861600	0.898710
30	0.946500	0.867338
40	0.764600	0.848834
50	0.853300	0.836384
60	0.762600	0.849179
70	0.637000	0.842751
80	0.685100	0.837282
90	0.646600	0.835706
100	0.618100	0.833356

Config ID = D

Step	Training Loss	Validation Loss
10	0.568200	0.913804
20	0.533700	0.888499
30	0.702400	0.917685
40	0.538300	0.899246
50	0.663200	0.872348
60	0.386900	0.984530
70	0.300900	1.010773
80	0.346300	0.986238
90	0.332300	0.992764
100	0.375000	0.993576

Config ID = E

Step	Training Loss	Validation Loss
10	0.588000	0.942861
20	0.567000	0.914033
30	0.763200	0.908331
40	0.895900	0.890298
50	0.700200	0.876826
60	0.684700	0.875522
70	0.670200	0.858809
80	0.568600	0.853946
90	0.602400	0.853250
100	0.607200	0.850270

Config ID = F

Step	Training Loss	Validation Loss
10	0.347200	1.178109
20	0.441800	1.046015
30	0.442800	1.004608
40	0.598900	0.957539
50	0.721900	0.905466
60	0.763900	0.873436
70	0.570200	0.873127
80	0.623300	0.867449
90	0.644500	0.859795
100	0.698900	0.858345

Config ID = G

Step	Training Loss	Validation Loss
10	1.035100	1.010537
20	0.917200	0.935806
30	0.991200	0.907225
40	0.812800	0.884691
50	0.909300	0.869184
60	0.929700	0.860859
70	0.801700	0.855306
80	0.870900	0.850297
90	0.848900	0.846588
100	0.793300	0.845042

Config ID = H

Step	Training Loss	Validation Loss
10	0.182900	0.920875
20	0.195500	1.036317
30	0.214500	1.106630
40	0.318600	1.096950
50	0.425000	1.063155
60	0.597200	1.025074
70	0.440800	0.997433
80	0.548300	0.985601
90	0.653900	0.979309
100	0.721300	0.978577

Config ID = I

Step	Training Loss	Validation Loss
10	0.112400	1.137540
20	0.216900	1.253228
30	0.605900	1.116398
40	0.540700	1.005123
50	0.725400	0.950795
60	0.485900	0.936062
70	0.320700	0.937768
80	0.394600	0.941178
90	0.222200	0.942834
100	0.385900	0.943293

Config ID = J

Step	Training Loss	Validation Loss
10	0.057700	1.554403
20	0.108100	1.264069
30	0.180200	1.256928
40	0.264400	1.205027
50	0.369500	1.098786
60	0.561100	1.011385
70	0.432900	0.956761
80	0.533500	0.936660
90	0.623100	0.919923
100	0.679200	0.915365

Config ID = K

Step	Training Loss	Validation Loss
10	0.124900	1.676461
20	0.200000	1.336852
30	0.566200	1.136361
40	0.552600	0.968783
50	0.735400	0.885931
60	0.354700	0.957067
70	0.190700	1.132254
80	0.272300	1.197147
90	0.149400	1.181419
100	0.254700	1.176784

Config ID = L

Step	Training Loss	Validation Loss
10	1.003800	0.941303
20	0.819100	0.906181
30	0.916300	0.872000
40	0.809500	0.849360
50	0.631200	0.846028
60	0.670900	0.850816
70	0.696500	0.842790
80	0.618400	0.838349
90	0.680300	0.835731
100	0.530000	0.837541

3.6 Valutazione delle configurazioni dei parametri

Per determinare la migliore configurazione, valutiamo ogni setup in base a:

- Training Loss – Indica quanto bene il modello apprende i dati di training.
- Validation Loss – Indica la capacità del modello di generalizzare su dati nuovi.
- Differenza tra Training e Validation Loss – Se la Training Loss è molto più bassa della Validation Loss, potrebbe esserci overfitting.

Config ID	Training Loss (TL)	Validation Loss (VL)	Valutazione
A	0.491000	0.903284	**Overfitting**: TL troppo più bassa della VL.
B	0.916300	0.858984	**TL alta**: Il modello sta imparando lentamente.
C	0.618100	0.833356	**Bilanciata**: Buona generalizzazione e training loss accettabile.
D	0.375000	0.993567	**Overfitting**: VL troppo alta rispetto alla TL.
E	0.607200	0.850270	**Accettabile,** ma leggermente inferiore a C.
F	0.698900	0.858345	**Accettabile,** ma leggermente peggiore di C.
G	0.793300	0.845042	**TL alta**: Il modello sta imparando lentamente.
H	0.721300	0.978577	**Underfitting**: VL troppo pià alta della TL.
I	0.385900	0.943293	**Overfitting**: TL troppo più bassa della VL.
J	0.679200	0.915365	**Accettabile,** ma VL più alta di C, E ed F.
K	0.254700	1.176784	**Overfitting**: TL troppo più bassa della VL.
L	0.530000	0.837541	**Overfitting**: TL troppo più bassa della VL.

3.7 Selezione della configurazione migliore

Sulla base delle valutazioni effettuate, selezionimao la configurazione C che risulta essere la migliore perché offre il miglior equilibrio tra training loss e validation loss, garantendo una buona capacità di apprendimento senza incorrere in overfitting. Con una training loss di 0.6183 e una validation loss di 0.8345, il modello dimostra di generalizzare bene ai dati non visti, evitando sia un apprendimento eccessivo dei dati di training sia un sotto-addestramento.

NOTA:
Generalmente, la selezione della configurazione ottimale dovrebbe avvenire in due fasi:

1. Fase di test preliminare: valutazione delle configurazioni su 50-100 steps.
2. Fase di validazione definitiva: test della migliore configurazione su 300, 500 o 1000 steps per consolidarne l'efficacia.

Tuttavia, in questo caso, il dataset dispone di circa 1000 esempi, e l'addestramento con un numero elevato di steps su tutte le configurazioni porterebbe a overfitting, poiché gli stessi esempi verrebbero considerati troppe volte.
Quindi, data la dimensione limitata del dataset, procederemo direttamente con la Configurazione C, senza ulteriori test con numeri differenti di steps, per evitare l'overfitting.

3.8 Utilizzo della configurazione migliore

La configurazione selezionata è stata utilizzata per creare il codice su Google Colab che espone l'API per utilizzare il modello addestrato e questa API verrà richiamata all' interno del nostro script principale.
L'API è visibile al seguente link.

4. Classificazione Probabilistica Semi-Supervisionata (Clustering + Classificazione Supervisionata)

IDEA:

L'obiettivo di questa fase del progetto è classificare automaticamente le notizie del mondo crypto in diverse categorie (Regolamentazione, Mercato, Adozione, Innovazione, Sicurezza, ecc.).

Per raggiungere questo scopo, viene adottato un approccio di classificazione semi-supervisionata, combinando due tecniche di classificazione:

- Clustering rigido con K-Means: utilizzato per raggruppare automaticamente gli articoli in cluster senza etichette iniziali.
- Classificazione supervisionata con Naïve Bayes: una volta ottenuti i cluster, questi vengono interpretati e assegnati a categorie significative.
  Successivamente, Naïve Bayes viene addestrato su questi dati per classificare nuovi articoli.

Questa metodologia permette di identificare pattern nascosti nei dati e ridurre la necessità di etichettatura manuale.

4.1 Pre-elaborazione del dataset: conversione in vettori TF-IDF

Poiché gli algoritmi di classificazione lavorano con dati numerici, il testo deve essere convertito in un formato utilizzabile.

A questo scopo viene utilizzata la tecnica TF-IDF (Term Frequency - Inverse Document Frequency), che assegna un peso a ciascuna parola in base alla sua rilevanza nel dataset.

Passaggi chiave della trasformazione TF-IDF:

1) Eliminazione delle stopwords italiane *(es: "il", "la", "con", "per")* per migliorare la qualità dei dati.
2) Calcolo del TF (Term Frequency): Il TF indica quante volte una parola appare in un documento rispetto al numero totale di parole.
3) Calcolo del IDF (Inverse Document Frequency), riduce l'importanza delle parole troppo comuni e aumenta l'importanza delle parole distintive.

$$IDF = log\left(\frac{N}{1 + DF}\right)$$

   Dove:
   - N = numero totale di documenti
   - DF = numero di documenti in cui la parola appare
4) Calcolo di TF-IDF per ogni parola nel documento sarà calcolata TF-IDF che sarà alta se è una parola distintiva, mentre sarà bassa se è una parola comune.

Esempio:
Ipotizzando di aver applicato il metodo TF-IDF descritto precedentemente su degli articoli di esempio otterremo un risultato del genere:

Termini	Articolo 1	Articolo 2	Articolo 3
Bitcoin	0.45	0.50	0.00
Ethereum	0.00	0.30	0.00
Hacker	0.00	0.00	0.80
Dollari	0.35	0.10	0.00

Dall' esempio precedente si capisce che ogni termine rappresenta una feature con un valore numerico e questo permette di rappresentare nello spazio il documento.

4.2 Clustering rigido: K-Means per identificare cluster

L'algoritmo di K-Means è una tecnica di clustering rigido, che suddivide un insieme di dati in K gruppi distinti, minimizzando la distanza tra ogni punto e il centroide del suo cluster.

4.2.1 Strumenti utilizzati

Le principali librerie utilizzate per l'implementazione del modello e il processo di addestramento sono le seguenti:

- **pandas** - Per la gestione e manipolazione del dataset (gestione CSV)
- **matplotlib** - Per la visualizzazione della curva del gomito e l'identificazione del numero ottimale di cluster.
- **sklearn.feature_extraction.text (TfidfVectorizer)** - Per convertire il testo degli articoli in vettori numerici (matrice TF-IDF) utilizzabili dall'algoritmo di clustering.
- **sklearn.cluster (KMeans)** - Per eseguire il clustering degli articoli basato sui vettori TF-IDF generati, e per determinare i centroidi di ogni cluster.
- **kneed (KneeLocator)** - Per identificare automaticamente il "gomito" nella curva dell'inertia e determinare il numero ottimale di cluster.
- **nltk (stopwords)** - Per rimuovere le parole comuni irrilevanti (stopwords) dalla matrice TF-IDF e migliorare l'efficacia del clustering.

4.2.2 Dataset utilizzato

Il dataset utilizzato è "datasetKMenas.csv" formattato nel seguente modo:

id_articolo	long_summary	categoria
34	Uniswap Labs ha lanciato la mainnet di Unichain, un Layer-2 di Ethereum con block time di un secondo...	

35	Il prezzo di ENA, token di Ethena, è sceso del 10% dopo che una balena ha spostato 18 milioni di token su Binance....	
36	Ventiquattro stati USA stanno valutando proposte per creare riserve di Bitcoin....	

Il campo “categoria” è rimasto vuoto perché sarà valorizzato dall’algoritmo Kmeans.

4.2.3 Determinare il numero ottimale di cluster - Curva del Gomito

Uno dei problemi principali di K-Means è la scelta del numero ottimale di cluster K. Per determinarlo, viene utilizzato il metodo della curva del gomito, che segue questi passi:

1. **Eseguire K-Means con diversi valori di K (da 1 a 10).**
2. **Calcolare l'Inertia** , cioè la somma delle distanze quadrate tra ogni punto e il centroide del cluster.
3. **Plottare la curva del gomito**, osservando il punto in cui la riduzione dell'inertia diventa meno significativa.
4. **Selezionare il valore ottimale di K**, che corrisponde al "gomito" della curva.

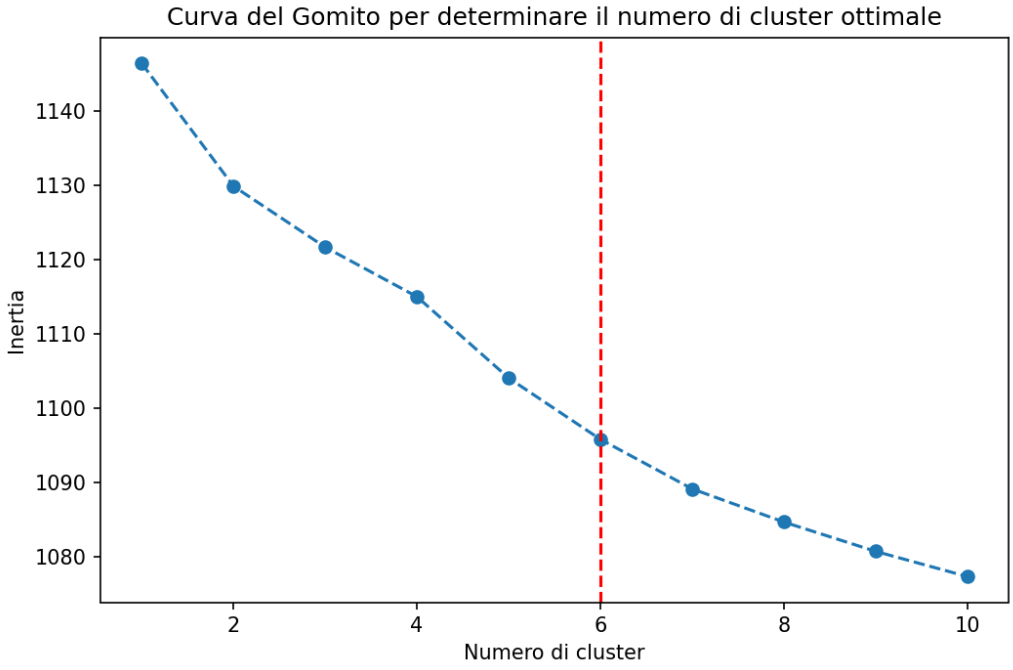
Questa tecnica assicura che il numero di cluster scelto sia né troppo basso (raggruppamento impreciso), né troppo alto (cluster eccessivamente specifici).

L’algoritmo utilizzato è il seguente:

```
# Convertire il testo in numeri con TF-IDF
vectorizer = TfidfVectorizer(stop_words=italian_stopwords, max_features=5000)
X_tfidf = vectorizer.fit_transform(df['titolo'])

# Determinare il numero ottimale di cluster usando la curva del gomito
inertia_values = []
clusters_range = range(1, 11)

for k in clusters_range:
    kmeans = KMeans(n_clusters=k, n_init=5, init='random', random_state=42)
    kmeans.fit(X_tfidf)
    inertia_values.append(kmeans.inertia_)

# Identificare il "gomito" nella curva
knee_locator = KneeLocator(clusters_range, inertia_values, curve="convex",
direction="decreasing")
optimal_clusters = knee_locator.elbow

# Plottare la curva del gomito
plt.figure(figsize=(8, 5))
plt.plot(clusters_range, inertia_values, marker='o', linestyle='--')
```

```
plt.xlabel("Numero di cluster")
plt.ylabel("Inertia")
plt.title("Curva del Gomito per determinare il numero di cluster ottimale")
plt.axvline(optimal_clusters, color='r', linestyle='--')
plt.show()
```

4.2.4 Valutazione dei risultati – Curva del Gomito



Eseguendo l'algoritmo precedentemente verranno testate le classificazioni fatte con un numero diverso di cluster partendo da K=1, fino a K=10.

L'immagine mostra la Curva del Gomito utilizzata per determinare il numero ottimale di cluster. Si può vedere che il numero di cluster ottimale è K=6 perché dopo questa soglia la diminuzione dell' inertia è meno significativa, quindi aggiungere altri cluster non porterebbe a una divisione milgiore.

4.2.5 Esecuzione K-Means con K=6 e dataset aggiornato

```
# Eseguire K-Means con il numero ottimale di cluster
kmeans = KMeans(n_clusters=optimal_clusters, n_init=5, init='random', random_state=42)
df['categoria'] = kmeans.fit_predict(X_tfidf)

# Salvare il dataset aggiornato
output_file = "dataset_KMeans_classificato_long.csv"
df.to_csv(output_file, index=False)
```

Dopo aver analizzato il datest classificato è stato possibile individurare le seguenti categorie:

Cluster	Numero di Articoli	Possibile Categoria
0	219	Adozione e Mainstreaming
1	291	Analisi Tecniche e Sentimenti di Mercato
2	20	Non rilevanti
3	244	Andamento del Mercato e Prezzi
4	342	Innovazione e Nuovi Progetti
5	74	Sicurezza, hackeraggi e Truffe

4.2.6 Affinamento manuale della classificazione

Il Kmeans è stato molto utile per una prima classificazione, ma analizzando i risultati e gli assegnamenti ci sono alcuni articoli e caetgorie molto simili, soprattutto le categorie: "Analisi Tecniche e Sentiment di Mercato" e la categoria "Andamento del Mercato e Prezzi" leggendo alcuni articoli è difficile stabilire effettivamente a quale categoria facciano parte.
Per questo motivo si è preferito accorpare queste due categorie nella categoria "News di Mercatom Analisi e Prezzi" ed aggiungere una nuova categoria più importante chiamata "Regolamentazione e Normative".

Dopo questo affinamento manuale, il dataset è stato riclassificato con le seguenti 6 categorie con questa distribuzione:

Categoria	Numero di Articoli
Adozione e Mainstreaming	135
Innovazione e Nuovi Progetti	146
News di Mercato, Analisi e Prezzi	547
Non rilevanti	36
Regolamentazione e Normative	194
Sicurezza, Hackeraggi e Truffe	134

Ovviamente molte sono categorie di ambito di analisi di mercato e prezzi quindi è normale questa discrepanza di articoli , cioè è normale che esistono molti più articoli nella categoria "News di Marcato,Analisi e Prezzi" rispetto alle altre categorie.

4.3 Apprendimento supervisionato: Naive Bayes

Dopo aver utilizzato Kmeans e una classificazione manuale per la divisione degli articoli in categorie si vuole addestrare un modello in modo tale da utilizzarlo per la classificazione degli articoli successivi.

E' stato scelto l'algoritmo Naive Bayes per la sua semplicità, efficienza e buona accuratezza nei problemi di classificazione di testo, in quanto assume che tutte le feature (parole nel testo) siano indipendenti tra loro, semplificando notevolmente il calcolo della probabilità.

4.3.1 Strumenti utilizzati

Le principali librerie utilizzate per l'implementazione del modello e il processo di addestramento sono le seguenti:

- **pandas** - Per la gestione e manipolazione dei dataset, inclusa la lettura e pulizia dei dati.
- **matplotlib** e **seaborn** - Per la creazione di grafici e visualizzazioni, in particolare la matrice di confusione normalizzata.
- **sklearn (scikit-learn)** - Libreria principale per:
  - **TfidfVectorizer** - Conversione del testo in rappresentazioni numeriche (matrici TF-IDF).
  - **train_test_split** - Suddivisione del dataset in training e test set.
  - **MultinomialNB (Naive Bayes)** - Algoritmo supervisionato per la classificazione testuale.
  - **classification_report, confusion_matrix, accuracy_score** - Metriche per valutare il modello.
- **nltk (Natural Language Toolkit)** - Per l'uso delle stopwords italiane nel preprocessing del testo.
- **numpy** - Per operazioni matematiche, come la normalizzazione della matrice di confusione.
- **pickle** - Per salvare il modello addestrato e il vettorizzatore per futuri utilizzi.

4.3.2 Suddivisione del dataset in Training e Validation Set

```
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2,
random_state=42)
```

Il dataset è stato suddiviso in:

- 80% Training Set (utilizzato per l'addestramento del modello)
- 20% Validation Set (usato per la valutazione durante il training)

4.3.3 Implementazione dell' algoritmo

Per implemenatre l'algoritmo è stato utilizzato il seguente codice:

```
# Inizializzare e addestrare il modello Naive Bayes
model = MultinomialNB()
model.fit(X_train, y_train)

# Prevedere le categorie sul set di test
y_pred = model.predict(X_test)

# Valutare il modello
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuratezza del modello: {accuracy:.2f}")

# Stampare il classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Stampare la matrice di confusione
conf_matrix = confusion_matrix(y_test, y_pred)
labels = sorted(df['category'].unique())  # Ordinare i nomi delle classi

# Normalizzare la matrice per mostrare le proporzioni
conf_matrix_norm = conf_matrix.astype('float') / conf_matrix.sum(axis=1)[:, np.newaxis]

# Plot della matrice di confusione
plt.figure(figsize=(14, 10))  # Aumenta la dimensione della figura
sns.heatmap(conf_matrix_norm, annot=True, cmap='Blues', fmt='.2%', xticklabels=labels,
yticklabels=labels)
plt.xticks(rotation=45, ha='right', fontsize=10)  # Ruota le etichette e riduce il testo
plt.yticks(fontsize=10)  # Riduce il testo delle etichette Y
plt.title('Matrice di Confusione Normalizzata (Proporzioni %)')
plt.xlabel('Previsioni')
plt.ylabel('Vero Valore')
plt.tight_layout()  # Per garantire che tutto rientri nel grafico
plt.show()
```

4.3.4 Valutazione dei Risultati

La valutazione del modello Naive Bayes addestrato è stata effettuata utilizzando diverse metriche standard per problemi di classificazione: Accuratezza, Precisione, Recall e F1-Score. I risultati delle metriche sono stati ottenuti confrontando le predizioni del modello con le etichette effettive del set di test.

Metriche di Valutazione

- Accuracy (Accuratezza): Percentuale di predizioni corrette sul totale degli esempi nel set di test.
- Precision: Percentuale di esempi correttamente classificati rispetto al totale degli esempi previsti per una determinata classe.
- Recall (Sensibilità): Percentuale di esempi correttamente classificati rispetto al totale degli esempi appartenenti a una determinata classe.
- F1-Score: Media armonica tra Precision e Recall, che bilancia l'importanza tra i due.

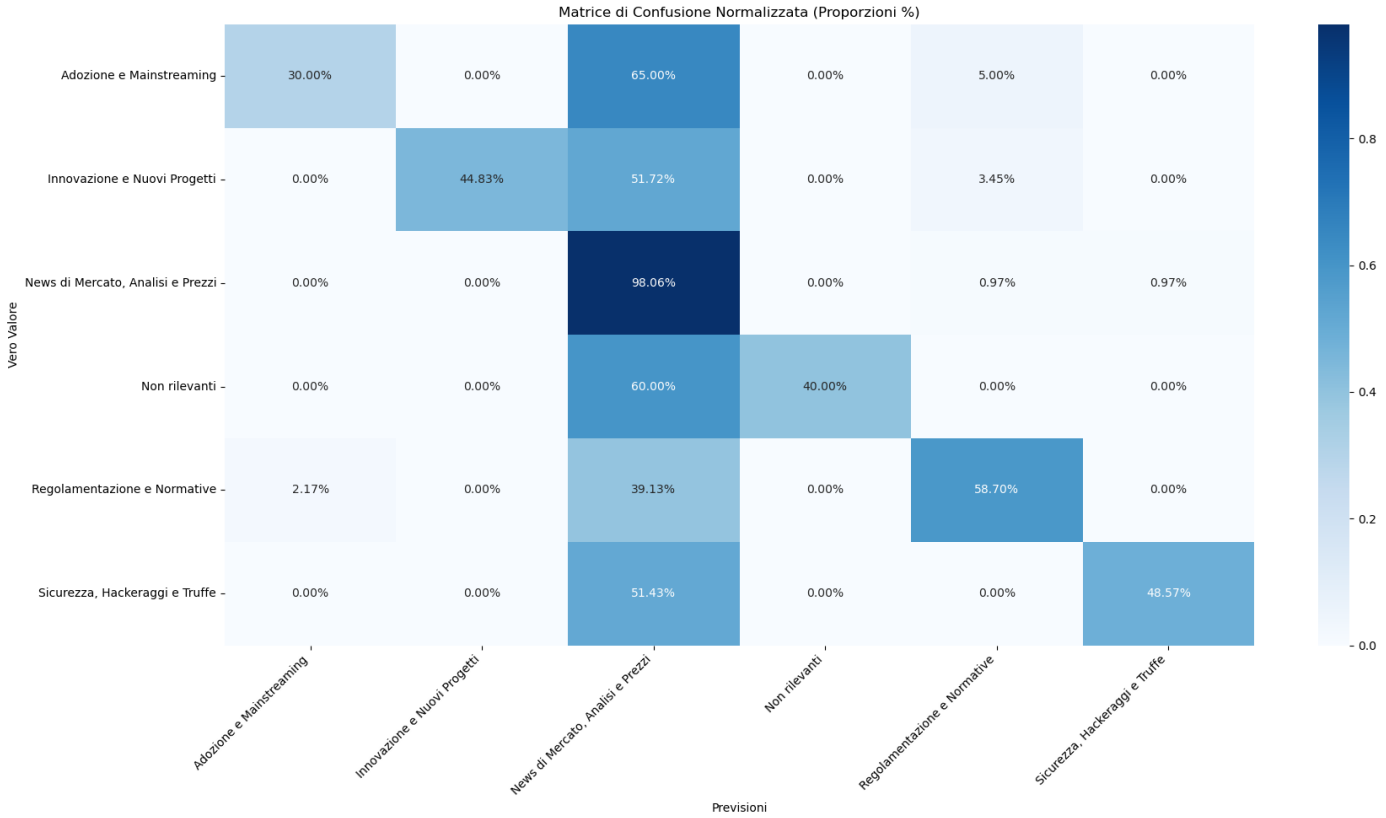
Il modello ha raggiunto un'accuratezza complessiva del 70%. La tabella seguente mostra il Classification Report suddiviso per ogni categoria:

```
Accuratezza del modello: 0.70

Classification Report:
                                precision    recall  f1-score   support

         Adozione e Mainstreaming       0.86      0.30      0.44        20
     Innovazione e Nuovi Progetti       1.00      0.45      0.62        29
News di Mercato, Analisi e Prezzi       0.60      0.98      0.75       103
                  Non rilevanti       1.00      0.40      0.57         5
     Regolamentazione e Normative       0.90      0.59      0.71        46
   Sicurezza, Hackeraggi e Truffe       0.94      0.49      0.64        35

                       accuracy                           0.70       238
                      macro avg       0.88      0.53      0.62       238
                   weighted avg       0.79      0.70      0.68       238
```

Matrice di Confusione

È stata generata anche una Matrice di Confusione Normalizzata per mostrare graficamente le prestazioni del modello, indicando le proporzioni di predizioni corrette e sbagliate per ogni classe (Recall per ogni classe).

Ogni riga rappresenta le etichette effettive (vere categorie), mentre ogni colonna rappresenta le etichette previste (previsioni del modello). Le celle della matrice mostrano quante volte il modello ha assegnato correttamente o erroneamente un'etichetta.



Valutiamo i risultati per ogni categoria:

1. **Adozione e Mainstreaming**:
   - Corretta classificazione: **30.00%**.
   - Viene erroneamente classificato come **News di Mercato, Analisi e Prezzi (65.00%)**.
   - Piccole percentuali di errore anche in altre Regolamentazione e Normatve (5%)
2. **Innovazione e Nuovi Progetti**:
   - Corretta classificazione: **44.83%**.
   - Confusione con **News di Mercato, Analisi e Prezzi (51.72%)**.
   - Un leggero errore anche in **Sicurezza, Hackeraggi e Truffe (3.45%)**.
3. **News di Mercato, Analisi e Prezzi**:
   - Corretta classificazione: **98.06%**. (Ottimo risultato)
   - Errori trascurabili in altre categorie.
4. **Non rilevanti**:
   - Corretta classificazione: **60.00%**.
   - Confusione significativa con **Innovazione e Nuovi Progetti (40.00%)**.
5. **Regolamentazione e Normative**:
   - Corretta classificazione: **58.70%**.
   - Confusione rilevante con **News di Mercato, Analisi e Prezzi (39.13%)**.
6. **Sicurezza, Hackeraggi e Truffe**:
   - Corretta classificazione: **48.57%**.
   - Spesso confuso con **News di Mercato, Analisi e Prezzi (51.43%)**.

4.3.5 Miglioramento con Laplace Smoothing

Analizzando i risultati della matrice di confusione precedente, si può notare che alcune categorie presentano una percentuale di corretta classificazione inferiore al 60%. Questo comportamento è dovuto principalmente allo sbilanciamento dei dati, dove le notizie appartenenti alla categoria "News di Mercato, Analisi e Prezzi" sono predominanti rispetto alle altre categorie.

Per migliorare le prestazioni del modello, è stato utilizzato un metodo chiamato **Laplace Smoothing.**

La Laplace Smoothing è una tecnica utilizzata per gestire il problema dello zero-frequency problem. Questo problema si verifica quando una parola o caratteristica non è presente nei dati di addestramento per una determinata classe. Senza la Laplace Smoothing, la probabilità di quella parola sarebbe zero, e il modello considererebbe impossibile la classificazione corretta.
Il principio della Laplace Smoothing è quello di aggiungere un piccolo valore (detto alpha) a tutte le possibili combinazioni di parole e categorie, evitando così probabilità zero. Questo processo rende il modello più robusto e meno incline ad escludere categorie per mancanza di dati.

Il modello Naive Bayes viene configurato con il parametro **alpha** che controlla l'entità dello smoothing:

- **Alpha = 1.0** (default): Applicazione standard della Laplace Smoothing, aggiunge un conteggio di "1" a tutte le parole.
- **Alpha < 1.0** (es. 0.1): Riduce l'effetto dello smoothing, dando maggiore importanza ai termini rari.
- **Alpha > 1.0** (es. 2.0 o maggiore): Aumenta l'effetto dello smoothing, rendendo il modello più robusto ma meno sensibile ai dettagli.

Nel nostro caso, per dare maggiore peso ai termini rari, migliorando la capacità del modello di distinguere tra categorie con meno esempi di addestramento dobbiamo considerare le configurazioni con alpha < 1

```
model = MultinomialNB(alpha=0.1)  # Regolazione del parametro alpha per Laplace Smoothing()
```

Test con Laplace Smoothing
Generiamo il parametro alpha da 0 a 0.9 con una distanza di 0.1 per ogni configurazione e selezioniamo la configurazione con percentuale di correttezza media migliore:

Test con alpha = 0	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	100,00%
Innovazione e Nuovi Progetti	0%
News di Mercato, Analisi e Prezzi	21,36%
Non rilevanti	40%
Regolamentazione e Normative	0%
Sicurezza, Hackeraggi e Truffe	0%
Media Percentuale Totale	**26,89%**

Test con alpha = 0.1	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	80,00%
Innovazione e Nuovi Progetti	86,21%
News di Mercato, Analisi e Prezzi	82,52%
Non rilevanti	40%
Regolamentazione e Normative	86,96%
Sicurezza, Hackeraggi e Truffe	85,71%
Media Percentuale Totale	**76,9%**

Test con alpha = 0.2	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	75,00%
Innovazione e Nuovi Progetti	86,21%
News di Mercato, Analisi e Prezzi	85,44%
Non rilevanti	40%
Regolamentazione e Normative	91,30%
Sicurezza, Hackeraggi e Truffe	80,00%
Media Percentuale Totale	**76,325%**

Test con alpha = 0.3	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	55,00%
Innovazione e Nuovi Progetti	86,21%
News di Mercato, Analisi e Prezzi	90,29%
Non rilevanti	40%
Regolamentazione e Normative	84,78%
Sicurezza, Hackeraggi e Truffe	77,14%
Media Percentuale Totale	**72,23%**

Test con alpha = 0.4	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	50,00%
Innovazione e Nuovi Progetti	82,76%
News di Mercato, Analisi e Prezzi	93,20%
Non rilevanti	40%
Regolamentazione e Normative	78,26%
Sicurezza, Hackeraggi e Truffe	74,29%
Media Percentuale Totale	**69,75%**

Test con alpha = 0.5	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	55,00%
Innovazione e Nuovi Progetti	65,52%
News di Mercato, Analisi e Prezzi	94,17%
Non rilevanti	40%
Regolamentazione e Normative	76,09%
Sicurezza, Hackeraggi e Truffe	74,29%
Media Percentuale Totale	**67,51%**

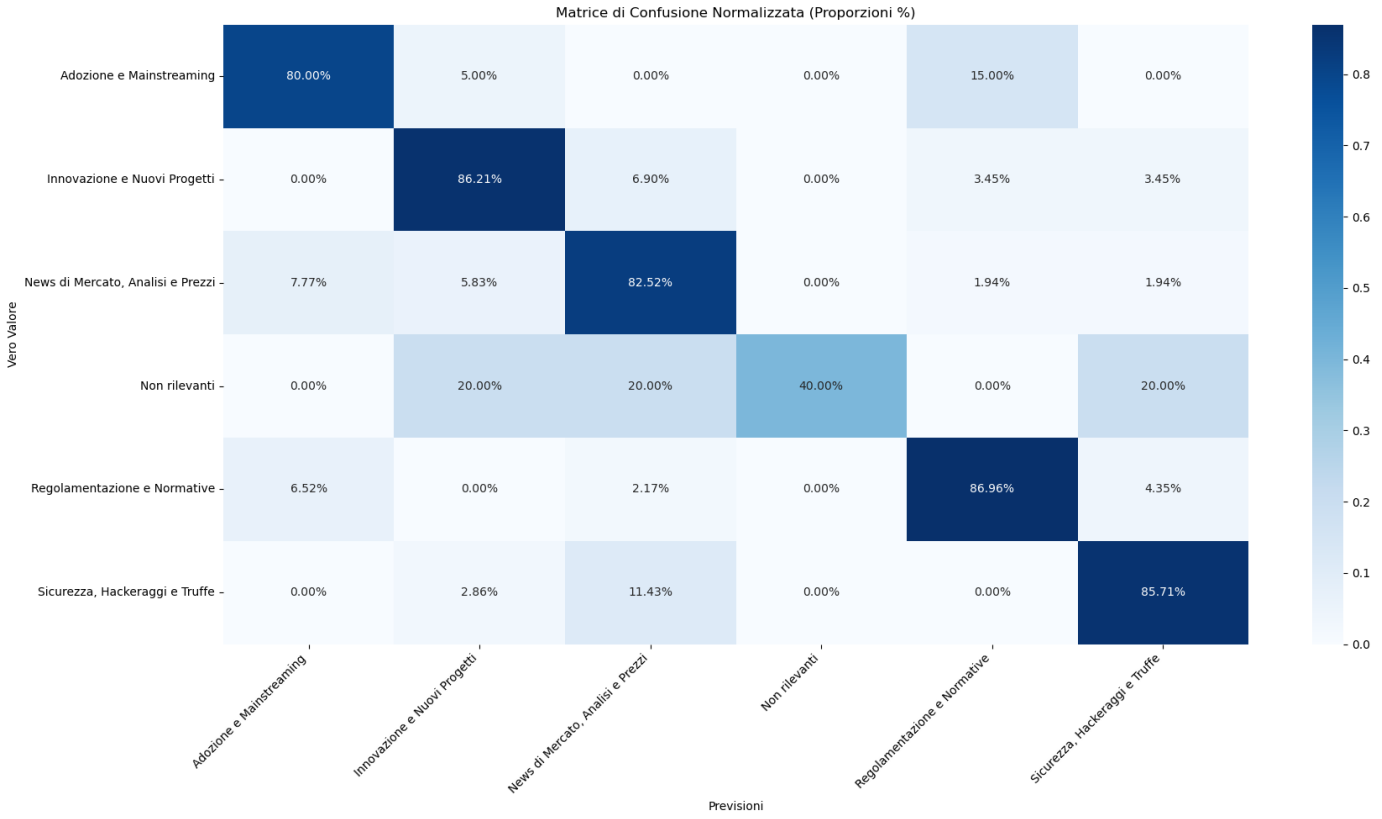
Test con alpha = 0.6	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	50,00%
Innovazione e Nuovi Progetti	62,07%
News di Mercato, Analisi e Prezzi	96,12%
Non rilevanti	40%
Regolamentazione e Normative	71,74%
Sicurezza, Hackeraggi e Truffe	65,71%
Media Percentuale Totale	**64,26%**

Test con alpha = 0.7	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	40,00%
Innovazione e Nuovi Progetti	62,07%
News di Mercato, Analisi e Prezzi	96,12%
Non rilevanti	40%
Regolamentazione e Normative	65,22%
Sicurezza, Hackeraggi e Truffe	65,71%
Media Percentuale Totale	**61,5%**

Test con alpha = 0.8	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	35,00%
Innovazione e Nuovi Progetti	51,72%
News di Mercato, Analisi e Prezzi	97,09%
Non rilevanti	40%
Regolamentazione e Normative	63,04%
Sicurezza, Hackeraggi e Truffe	51,43%
Media Percentuale Totale	**56,38%**

Test con alpha = 0.9	
Categorie	**Percentuale Correttezza Valutazione**
Adozione e Mainstreaming	35,00%
Innovazione e Nuovi Progetti	51,72%
News di Mercato, Analisi e Prezzi	98,06%
Non rilevanti	40,00%
Regolamentazione e Normative	58,70%
Sicurezza, Hackeraggi e Truffe	48,57%
Media Percentuale Totale	**55,34%**

4.3.6 Selezione e valutazione della configurazione migliore (alpha = 0.1)



Valutiamo i risultati per ogni categoria:

1. **Adozione e Mainstreaming**:
   - **Accuratezza migliorata:** 80.00% (prima era circa 30.00%)
   - Molta meno confusione con **News di Mercato, Analisi e Prezzi** (solo 15.00% adesso, contro il precedente 65.00%).
2. **Innovazione e Nuovi Progetti**:
   - **Accuratezza migliorata:** 86.21% (prima era 44.83%).
   - Ridotta la confusione con **News di Mercato, Analisi e Prezzi** (6.90% invece di 51.72%).
   - C'è un piccolo errore anche su **Sicurezza, Hackeraggi e Truffe (3.45%)**.
3. **News di Mercato, Analisi e Prezzi**:
   - **Accuratezza leggermente ridotta:** 82.52% (prima era 98.06%).
   - Alcuni articoli sono ora assegnati erroneamente ad altre categorie, ma l'accuratezza resta comunque molto alta.

4. **Non rilevanti**:
    - **Accuratezza invariata:** 40.00%.
    - Gli articoli sono spesso confusi con **Regolamentazione e Normative (20.00%)** e **News di Mercato, Analisi e Prezzi (20.00%)**.
5. **Regolamentazione e Normative**:
    - **Accuratezza migliorata:** 86.96% (prima era 58.70%).
    - Ridotta la confusione con **News di Mercato, Analisi e Prezzi**.
6. **Sicurezza, Hackeraggi e Truffe**:
    - **Accuratezza migliorata:** 85.71% (prima era 48.57%).
    - Minore confusione con altre categorie, ma rimane una piccola sovrapposizione con **News di Mercato, Analisi e Prezzi (11.43%)**.

Il ridotto valore di alpha ha effettivamente migliorato l'accuratezza complessiva del modello. Le categorie deboli (con meno articoli) sono state migliorate significativamente, come mentre la categoria "News di Mercato, Analisi e Prezzi" ha perso un po' di accuratezza, ma resta comunque molto precisa.

```
Accuratezza del modello: 0.83

Classification Report:
                                precision    recall  f1-score   support

         Adozione e Mainstreaming       0.59      0.80      0.68        20
     Innovazione e Nuovi Progetti       0.74      0.86      0.79        29
News di Mercato, Analisi e Prezzi       0.91      0.83      0.87       103
                  Non rilevanti       1.00      0.40      0.57         5
     Regolamentazione e Normative       0.87      0.87      0.87        46
   Sicurezza, Hackeraggi e Truffe       0.83      0.86      0.85        35

                         accuracy                           0.83       238
                        macro avg       0.82      0.77      0.77       238
                     weighted avg       0.85      0.83      0.83       238
```

L'accuratezza complessiva è: 0.83 (83%)
Questo è un miglioramento significativo rispetto al valore precedente (70%) con alpha=1.0.

4.3.7 Generazione del Modello utilizzato per le classificazioni successive

Salviamo il modello con alpha=0.1 che verrà utilizzato per classificare nuovi articoli:

```python
# Salvare il modello e il vettorizzatore
with open('modello_naive_bayes.pkl', 'wb') as model_file:
    pickle.dump(model, model_file)

with open('vectorizer_tfidf.pkl', 'wb') as vectorizer_file:
    pickle.dump(vectorizer, vectorizer_file)
```

5. Knolwedge Graph e Ragionamento con Prolog

Sommario

In questa sezione viene illustrata la creazione e gestione di un Knowledge Graph per rappresentare le relazioni semantiche tra entità nel mondo delle criptovalute e l'utilizzo del linguaggio Prolog per l'inferenza automatica e il ragionamento logico. L'obiettivo principale è quello di identificare articoli rilevanti, scoprire correlazioni tra essi e rilevare tendenze emergenti basate sulle entità citate.

5.1 Strumenti Utilizzati

Per la costruzione e gestione del Knowledge Graph e per l'inferenza logica sono stati utilizzati i seguenti strumenti:

- **Prolog (PySWIP):** Libreria Python per interfacciarsi con Prolog, utilizzata per eseguire interrogazioni logiche e inferenze sul Knowledge Graph generato.
- **NetworkX:** Libreria Python per la creazione e gestione di grafi, utilizzata per visualizzare la struttura del Knowledge Graph.
- **PyVis:** Libreria per la visualizzazione interattiva di grafi tramite un'interfaccia web, utile per esplorare visivamente le relazioni tra articoli ed entità.
- **SQLite:** Database relazionale utilizzato per memorizzare gli articoli e i rispettivi riassunti prima della generazione del Knowledge Graph.

5.2 Decisioni di Progetto

Il Knowledge Graph viene creato con le seguenti regole logiche definite in Prolog:

- **Entità principali:** Persone, criptovalute, istituzioni finanziarie e aziende.
- **Relazioni principali:** Founder, President, CEO e citazioni da parte degli articoli.
- **File Prolog:** knowledge_base.pl generato dinamicamente ogni settimana con nuovi articoli.
- **Regole di inferenza:**
    - Articoli rilevanti (con almeno 4 entità menzionate).
    - Articoli correlati (che condividono entità o entità connesse tramite relazioni come Founder, President o CEO).
    - Identificazione delle tre entità più menzionate nella settimana.

Il Knowledge Graph viene generato con i nuovi articoli della settimana.
Tutti i dati sono memorizzati in formato Prolog, e successivamente visualizzati tramite NetworkX e PyVis.

5.3 Creazione del Knowledge Graph

Il Knowledge Graph come detto rappresenterà le entità prinicpali del mondo crypto.

Queste sono informazioni statiche sono contenute nel file “entities_and relations.py”.

Definizione delle Entità

```
people = {
    ###CRYPTO - FOUNDER###
    "Satoshi Nakamoto": ["satoshi", "nakamoto", "satoshi nakamoto"],
    "Vitalik Buterin": ["vitalik", "buterin", "vitalik buterin"],
    "Brock Pierce": ["brock pierce"],
    "Reeve Collins": ["reeve collins"],
      …


    ###PRESIDENTI FINANCTIAL INSTITUTION###

    "Jerome Powell": ["jerome powell", "powell"],           #FED
    "Christine Lagarde": ["christine lagarde", "lagarde"],  #BCE
    "Gary Gensler":["gary gensler","gensler"],              #SEC
      …


    ###CEO - Company###

    "Tim Cook": ["tim cook"],               #apple
    "Jensen Huang": ["jensen huang"],       #nvidia
    "Michael Saylor": ["michael saylor"],   #micorstrategy
      …
}

crypto = {
    "Bitcoin": ["btc", "bitcoin"],
    "Ethereum": ["eth", "ethereum"],
    "Tether": ["usdt", "tether"],
      …
}

financial_institutions = {
    "FED": ["fed", "federal reserve", "federal reserve bank"],
    "BCE": ["bce", "banca centrale europea", "european central bank"],
    "SEC": ["sec", "securities and exchange commission", "security exchange commission"],
      …
}

companies = {
    "Apple": ["apple", "aapl"],
    "NVIDIA": ["nvidia", "nvda"],
    "MicroStrategy": ["microstrategy", "mstr"],
      …
}
```

Gli esempi precedentemente mostrati illustrano come le entità principali del mondo crypto siano rappresentate tramite un dizionario strutturato. In questo dizionario, ogni chiave corrisponde a un'entità specifica (ad esempio, persone, criptovalute, istituzioni finanziarie e aziende) e il suo valore è un elenco di possibili alias o varianti testuali con cui quell'entità può essere menzionata all'interno degli articoli.

Questa struttura è fondamentale per l'estrazione delle entità citate: quando il sistema analizza i riassunti degli articoli, confronta il testo con questi alias per identificare la presenza di specifiche entità nel contenuto.

Definizione delle Relazioni

Le relazioni rappresentano i legami noti tra queste entità e sono definite con delle associazioni statiche nel codice. Le principali relazioni sono:

```
Founder_crypto = {
    "Bitcoin": "Satoshi Nakamoto",
    "Ethereum": "Vitalik Buterin",
    "Tether": ["Brock Pierce","Reeve Collins","Craig Sellars"],
      …
}

President_financial_institutions = {
    "FED":"Jerome Powell",
    "BCE":"Christine Lagarde",
    "SEC":"Gary Gensler"
      …
}

CEO_companies = {
    "Apple": "Tim Cook",
    "NVIDIA": "Jensen Huang",
    "MicroStrategy": "Michael Saylor",
      …
}
```

Relazioni Dinamiche

Oltre alle relazioni statiche, viene aggiunta dinamicamente anche la relazione "Article Mentions Entity" quando un articolo cita una o più entità. Queste relazioni vengono generate durante l'elaborazione degli articoli e memorizzate nel Knowledge Graph.

In questo modo, il Knowledge Graph non è solo una rappresentazione statica del dominio crypto, ma un sistema dinamico che si aggiorna continuamente man mano che vengono elaborati nuovi articoli e scoperte nuove relazioni.

Creazione del Knowledge Graph

Dopo aver scandito gli articoli per identificare le entità citate viene creato un file prolog contenente tutte le informazioni estratte e le query.

Il file per la creazione del Knowledge Graph si chiama “generateKnowledgeGraph.py”.

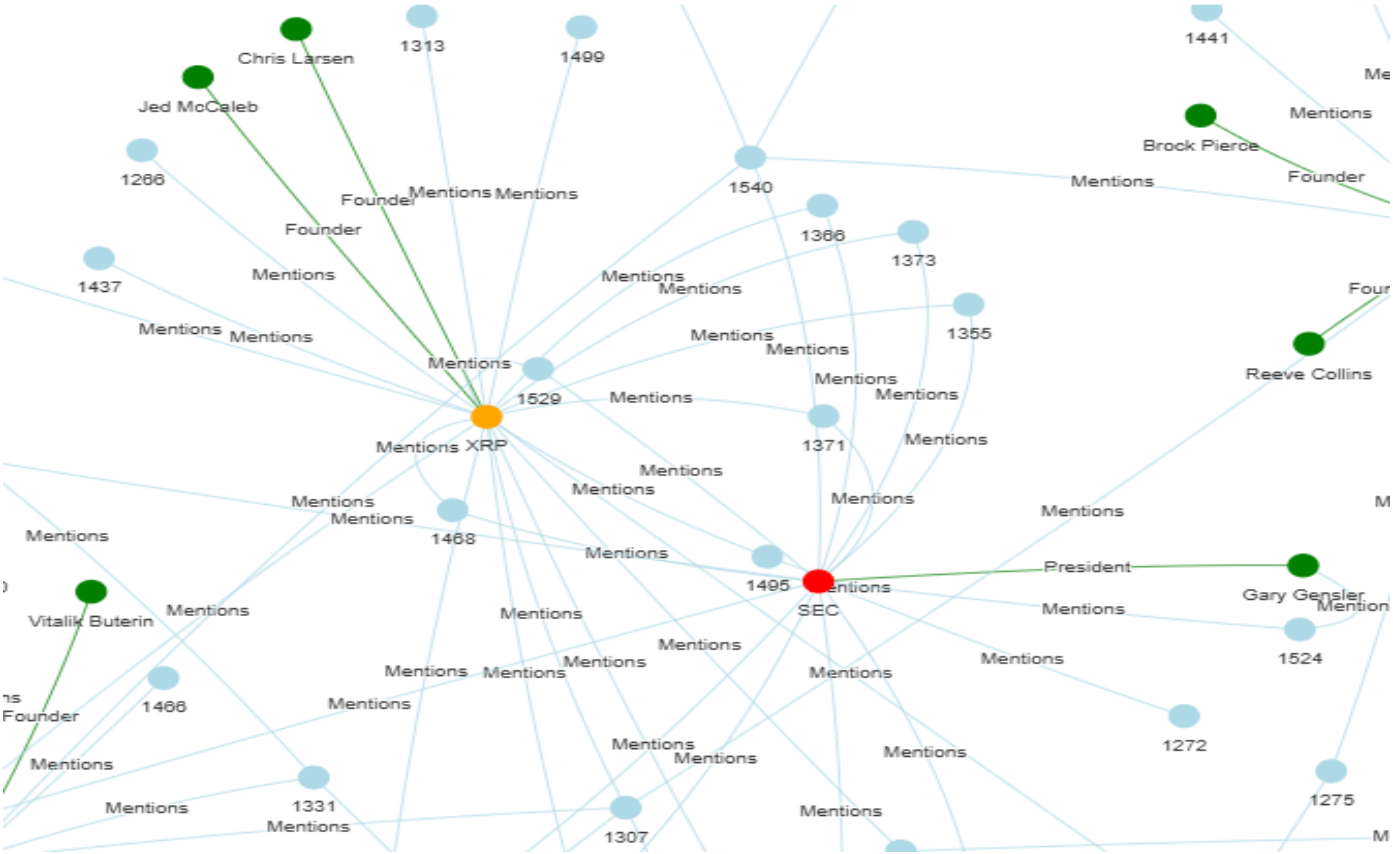
Dopo l’esecuzione del file si può visualizzare il file prolog creato (“knowledg_base.pl”):

```
                                        %Entities
% People
person('Satoshi Nakamoto').
person('Vitalik Buterin').
person('Michael Saylor').
…

% Crypto
crypto('Bitcoin').
crypto('Ethereum').
crypto('Tether').
…

% Financial Institutions
financial_institution('FED').
financial_institution('BCE').
financial_institution('SEC').

% Financial Institutions
financial_institution('FED').
financial_institution('BCE').
financial_institution('SEC').
…

% Companies
company('Apple').
company('Binance').
company('NVIDIA').
…

% Article
article(1269, '2025/03/18').
article(1270, '2025/03/18').
article(1271, '2025/03/18').
…



                                        %Relations

% Founder Relationships
founder('Satoshi Nakamoto', 'Bitcoin').
founder('Vitalik Buterin', 'Ethereum').
founder('Brock Pierce', 'Tether').
```

```
…

% People_PRESIDENT_FinanctialInstitutions
president('Jerome Powell', 'FED').
president('Christine Lagarde', 'BCE').
president('Gary Gensler', 'SEC').
…

% People_CEO_Company
ceo('Tim Cook', 'Apple').
ceo('Jensen Huang', 'NVIDIA').
ceo('Michael Saylor', 'MicroStrategy').
…

% Menzioni
mention(1266, 'XRP').
mention(1267, 'Changpeng Zhao').
mention(1267, 'Binance').
mention(1268, 'Arbitrum').
mention(1268, 'Ethereum').
…
```

Visualizzazione grafica del Knowledge Graph

Il Knowledge Graph può anche essere visto graficamente eseguendo il file “knowledge_graph.html”:



5.4 Logica del Ragionamento e Interrogazioni in Prolog

Una volta creato il Knowledge Graph, il sistema utilizza **Prolog** per eseguire operazioni di ragionamento automatico al fine di identificare articoli rilevanti, correlazioni tra articoli e tendenze emergenti.

Regole Implementate

Il sistema definisce diverse regole Prolog per estrarre conoscenza utile dagli articoli e dal Knowledge Graph:

1. **Identificazione di Articoli Rilevanti**

Questa regola identifica come rilevanti tutti gli articoli che contengono menzioni di almeno 4 entità.

```
% --- Regola per identificare i documenti rilevanti ---
relevant_document(Document) :-
    findall(Entity, mention(Document, Entity), Entities),
    length(Entities, Count),
    Count > 3.
```

2. **Scoperta di Tendenze Emergenti**

Viene calcolata la frequenza di menzione per ogni entità nell'arco di tempo considerato. L'entità è considerata popolare se appare in più di 4 articoli.

```
frequenza_entita(Entita, Count) :-
    setof(ArticoloID, mention(ArticoloID, Entita), ArticoliUnici),
    length(ArticoliUnici, Count).
```

3. **Rilevazione di Articoli Correlati**
   Sono stati definiti diversi modi per identificare articoli correlati:
   - **Correlazione Diretta:** Due articoli citano la stessa entità.

```
articolo_correlato(Articolo1, Articolo2) :-
mention(Articolo1, Entita),
    mention(Articolo2, Entita),
    Articolo1 \= Articolo2.
```

   - **Correlazione per Fondatore:** Due articoli citano entità collegate da relazioni di fondazione.

```
articolo_correlato(Articolo1, Articolo2) :-
    mention(Articolo1, Entita1),
    mention(Articolo2, Entita2),
    (founder(Entita1, Entita2); founder(Entita2, Entita1)),
    Articolo1 \= Articolo2.
```

- **Correlazione per Presidente o CEO:** Due articoli citano entità collegate da relazioni di presidenza o direzione aziendale.

```
% --- Articoli Correlati Indirettamente tramite President ---
articolo_correlato(Articolo1, Articolo2) :-
    mention(Articolo1, Entita1),
    mention(Articolo2, Entita2),
    (president(Entita1, Entita2); president(Entita2, Entita1)),
    Articolo1 \= Articolo2.

% --- Articoli Correlati Indirettamente tramite CEO ---
articolo_correlato(Articolo1, Articolo2) :-
    mention(Articolo1, Entita1),
    mention(Articolo2, Entita2),
    (ceo(Entita1, Entita2); ceo(Entita2, Entita1)),
    Articolo1 \= Articolo2.
```

- **Correlazione Estesa:** Due articoli condividono una delle relazioni sopra definite.

```
articolo_correlato(Articolo1, Articolo2) :-
    mention(Articolo1, Entita1),
    mention(Articolo2, Entita2),
    (   Entita1 = Entita2 ;
        founder(Entita1, Entita2) ;
        founder(Entita2, Entita1) ;
        ceo(Entita1, Entita2) ;
        ceo(Entita2, Entita1) ;
        president(Entita1, Entita2) ;
        president(Entita2, Entita1)
    ),
    Articolo1 \= Articolo2.
```

5. **Estrazione di tutte le Entità con i Conteggi**

   Questa regola recupera tutte le entità presenti nel Knowledge Graph e calcola quante volte ogni entità è stata menzionata.

```
tutte_entita(ListaEntita) :-
    findall([Count, Entita], frequenza_entita(Entita, Count), ListaEntita).
```

5.5 Esecuzione delle interrogazioni in Prolog

Durante l'esecuzione del sistema, vengono effettuate diverse interrogazioni per estrarre conoscenze rilevanti dal Knowledge Graph. Queste query vengono inviate al motore Prolog tramite l'interfaccia PySWIP. Le query eseguite automaticmanete sono

contenute nel file: “queryProlog.py”. Di seguito vengono descritte le principali interrogazioni e i relativi risultati:

1) **Identificazione degli Articoli Rilevanti**
   Questa query è utilizzata per determinare quali articoli devono essere considerati rilevanti. Un articolo è considerato rilevante se contiene più di 3 entità menzionate.

Query:

```
relevant_dcoument(ArticoloID)
```

Risultato (Esempio):

```
Articolo Rilevante: 1459
Articolo Rilevante: 1471
Articolo Rilevante: 1482
Articolo Rilevante: 1497
Articolo Rilevante: 1501
…
```

2. **Ricerca di Articoli Correlati**
   Questa query identifica articoli correlati basandosi su diverse relazioni definite nel Knowledge Graph (correlazione diretta, fondatore, CEO, presidente, e correlazione estesa).

Query:

```
articolo_correlato(ArticoloRilevante,ArticoloCorrelato)
```

Risultato (Esempio):

```
Articolo Rilevante: 1459
Articoli Correlati: 1284, 1287, 1296, 1302, 1434, 1436, 1312, 1318, 1319, 1448, 1321,
1451, 1327, 1329, 1334, 1349, 1351, 1482, 1358, 1497, 1498, 1500, 1509, 1512, 1514, 1515,
1389, 1393, 1521, 1528

Articolo Rilevante: 1471
Articoli Correlati: [1408, 1283, 1541, 1286, 1288, 1291, 1292, 1293, 1294, 1296, 1297,
1424, 1403, 1300, 1301, 1531, 1306, 1307, 1435, 1439, 1314, 1443, 1316, 1318, 1448, 1449,
1450, 1328, 1330, 1331, 1333, 1334, 1464, 1337, 1338, 1467, 1343, 1348, 1349, 1350, 1351,
1481, 1354, 1482, 1484, 1486, 1487, 1365, 1370, 1375, 1376, 1379, 1507, 1513, 1515, 1516,
1517, 1392, 1393, 1522, 1395, 1525, 1271, 1275, 1533, 1278]
…
```

3) Estrazione delle Entità più Menzionate

Questa query viene utilizzata per identificare le entità più citate nell'intervallo di tempo considerato.

Query:

```
findall([Count,Entita], (frequenza_entita(Entita,Count)),ListaEntita)
```

Risultato (Esempio):

```
Articolo Rilevante: 1471
Le entità più citate sono:
- Bitcoin: 58
- Coinbase: 27
- Solana: 24
- Binance: 22
- Etherum: 19
- XRP: 19
```

6. Creazione Report Settimanale

Per questioni di tempo e poiché non è inerente al corso non è stata integrata la creazione del report settimanale, ma per farlo basterebbe utilizzare i risultati delle interrogazioni in Prolog recuperando dagli id degli articoli il contenuto ed includerlo in un file pdf.

7. Conclusioni

Il progetto “Crypto News” ha raggiunto con successo diversi obiettivi prefissati, dimostrando l'efficacia di un sistema intelligente per l’analisi automatizzata delle notizie del settore delle criptovalute. Il sistema è stato in grado di recuperare, sintetizzare e classificare articoli in diverse categorie, e anche di identificare correlazioni e tendenze emergenti tramite un Knowledge Graph.

Possibili sviluppi futuri

Tra i possibili sviluppi futuri, si può considerare l’integrazione di un’analisi del sentiment per ogni articolo raccolto, al fine di comprendere meglio l’impatto delle notizie sull’andamento del mercato. Inoltre, sarebbe utile associare ai report settimanali anche dati storici sui prezzi delle criptovalute e mostrare come notizie positive o negative possano influenzare i prezzi nel breve e medio termine.
Il Knowledge Graph potrebbe essere ulteriormente arricchito con relazioni più complesse e con l’integrazione di fonti aggiuntive, migliorando la capacità del sistema di individuare correlazioni nascoste e generare report più completi e accurati.

Riferimenti Bibliografici

- LLama 3.2 3B - documentazione
- Supervised Learning Foundations
- Overfitting – Laplace Smoothing
- Neural Models for Sequences
- Improved optimization
- Probabilistic Learning
- Bayesian Learning
- Unsupervised Learning
- Individuals and Relations
- Datalog: A Relational Rule Language
- Knowledge Graphs