Sanremo Forecasting: Predizione del Vincitore di Sanremo 2025 per il televoto attraverso l'analisi del sentimento

Marzia De Maina - 0001194461
Enrico Ferraiolo - 0001191698

A.A. 2024/2025

Indice

1 Introduzione

L'idea alla base del nostro progetto nasce dalla volontà di scoprire in anticipo chi fosse il **favorito del pubblico a casa del Festival di Sanremo 2025** attraverso l'**analisi automatica del sentimento** espresso dagli utenti sui social network.
In particolare, ci si è concentrati sull'analisi dei contenuti pubblicati su *X.com* (ex Twitter), piattaforma su cui si registra un'elevata attività durante gli eventi culturali.
Per raggiungere questo obiettivo, abbiamo costruito un dataset contenente tutti i tweet riferiti agli artisti in gara, raccolti durante la live del Festival.
Ciascun post è stato sottoposto a **un'analisi del sentiment** tramite il modello *Twitter-roBERTa-base for Sentiment Analysis* [6], in grado di classificare ciascun post come `positive` o `negative`.

2 Dataset

Il progetto si propone di analizzare i dati relativi al Festival di Sanremo 2025, focalizzandosi sull'**opinione del pubblico a casa** attraverso la piattaforma *X.com*. La creazione del dataset è stata una fase cruciale, volta a garantire l'affidabilità e la completezza delle analisi successive.

2.1 Valutazione delle Metodologie di Raccolta dei Dati

Dopo un'accurata valutazione delle funzionalità offerte dalle API ufficiali del social network *X*, è emerso che la versione gratuita presenta **limitazioni troppo stringenti** in termini di volume dei dati accessibili, frequenza delle richieste e tipo di contenuti ottenibili (ad esempio, tweet recenti). Considerando che l'accesso alle versioni a pagamento delle API fosse economicamente fuori dalla portata del nostro progetto, si è scelto di adottare una soluzione alternativa. È stato quindi realizzato uno ***Scraper***: un bot che simula il comportamento di un utente umano durante la navigazione in una pagina web, capace di estrarre in modo strutturato le informazioni di nostro interesse. L'utilizzo della libreria *Selenium* [3] ha **permesso il controllo** del browser web *Chrome* (selezionato da [2]) interagendo in modo dinamico tra le pagine. Questa scelta ha consentito di raccogliere i dati pubblicamente visibili necessari alla costruzione del dataset in modo più flessibile ed efficiente ai fini didattici.

2.1.1 Implementazione dello Scraper

È stata creata la classe `TwitterScraper` che implementa il bot per recuperare i tweet interessati.
Al fine di evitare la sospensione dell'account utilizzato è stato inserito un **waiting intelligente**: prima di passare alla successiva query di ricerca viene messo in pausa l'algoritmo per un tempo che varia dagli 1 ai 4 minuti. In questo modo è stato possibile aggirare con successo i blocchi di *X*.
Il bot **simula un utente che naviga e compie una ricerca** su *X*. Infatti prima di tutto il bot deve compiere il login sul social network. Le credenziali per l'account sono inserite in un file `.env` e, se mancanti,xx l'esecuzione di scraping si interrompe.
Dopo aver fatto login, si passa alla fase di costruzione della **query di ricerca**, infatti il bot costruirà e cercherà una query del genere:

```
"{Artista}" since:YYYY-MM-DD until:YYYY-MM-DD
```

A questo punto il bot si ritroverà in una schermata dove sono presenti i tweet ricercati. Il bot scrollerà la pagina fino a un massimo di 15 volte (oppure

finché non raccoglie `max_tweets` (variabile che permette di definire un limite massimo di tweet da recuperare) caricando nuovi tweet.
I tweet vengono individuati attraverso la ricerca dell'elemento HTML `<article>` con *data-testid="User-Name"*, per essere sicuri che sia stato postato da un utente. Del tweet trovato vengono salvati:

- Artista della query di ricerca
- Timestamp
- Testo del tweet

Vengono quindi selezionati i tweet contenenti solo testo. Per ogni serata del Festival viene creata una cartella denominata `{NumeroSerata}_serata`. All'interno di essa vengono generate delle sottocartelle, ciascuna rinominata con il nome di uno degli artisti in gara durante la serata selezionata. All'interno di ogni sottocartella viene poi creato un file CSV che contiene le informazioni salvate dei tweet, nominato secondo il formato `YYYY-MM-DD_HH-HH.csv`, dove l'intervallo orario può essere: 21-22, 22-23, 23-00 oppure 00-01. In questo modo, i tweet vengono organizzati per artista e suddivisi per fascia oraria in ciascuna serata.
I tweet estratti e salvati, infatti, sono anche **filtrati in base alla fascia oraria e data** in cui ricadono.
Un esempio della struttura di salvataggio si ha in Figura 1.

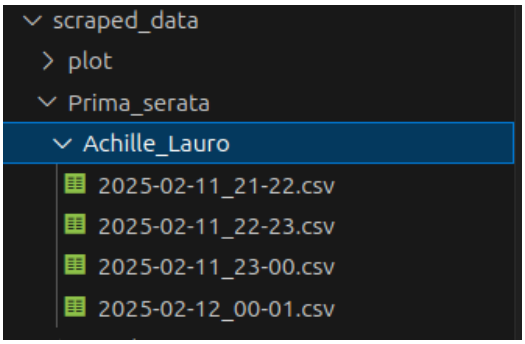

Figura 1: Esempio della struttura di salvataggio dei tweet trovati durante lo scraping

2.1.2 Strategia di ottimizzazione delle query di ricerca

Durante la fase di raccolta dei dati, particolare attenzione è stata posta alla **corretta identificazione dei tweet** riferiti agli artisti partecipanti all'ultima edizione del Festival di Sanremo. Alcuni nomi di concorrenti sono nomi

propri molto diffusi (come Clara, Gaia, Giorgia, Noemi) o nomi comuni (come Modà, che seppur accentato può riferirsi al mondo della moda). Questa condizione ha comportato **un alto rischio di ambiguità** e **il recupero di contenuti non pertinenti al contesto** del Festival.
Per ovviare a questo problema, sono state definite delle query di ricerca specifiche, costruite combinando il nome dell'artista con il termine *Sanremo*: "Clara Sanremo", "Gaia Sanremo", "Giorgia Sanremo", "Modà Sanremo" e "Noemi Sanremo". In questo modo è stato possibile filtrare e affinare in maniera più efficace i tweet realmente riferiti all'evento, migliorando l'accuratezza del dataset finale e riducendo il rumore informativo.

2.1.3 Risultati dello Scraping

Alla fine della sessione di scraping viene eseguita un'operazione di **plotting** attraverso *matplotlib* per ottenere i **grafici complessivi dei tweet** ottenuti per ogni artista e per ogni serata, come si può vedere nelle Figure 2 e 3.

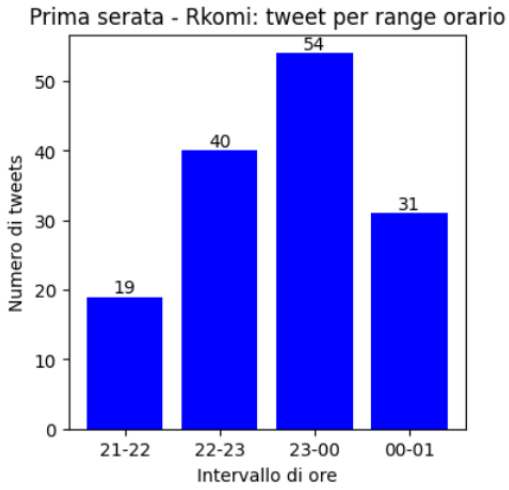

Figura 2: Tweet ottenuti per Rkomi durante lo scraping della prima serata, divisi per fasce orarie

2.2 Pulizia del Dataset

La fase di pulizia del dataset è stata fondamentale per assicurare l'affidabilità e la coerenza dei dati raccolti. In particolare, è stato condotto un importante intervento: **il filtraggio dei tweet non pertinenti**. Sono stati rimossi tutti quei tweet che, pur contenendo *hashtag* riconducibili al Festival di Sanremo,

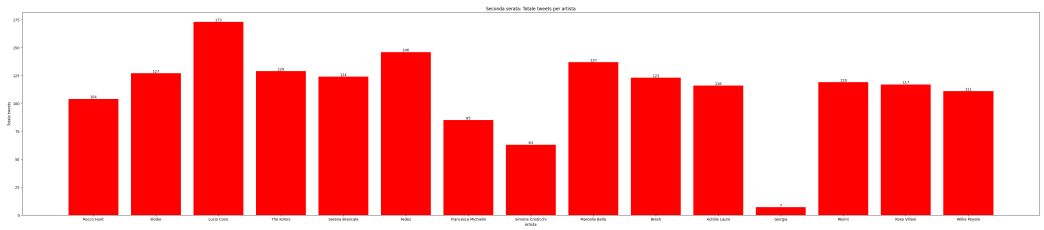

Figura 3: Totale tweet per ciascun artista durante la seconda serata

non erano in alcun modo collegati all'evento. Questo fenomeno si verifica frequentemente a causa dell'uso improprio o strategico degli hashtag da parte degli utenti, i quali li inseriscono per aumentare artificialmente la visibilità dei propri contenuti, anche se non rilevanti. Il filtraggio è stato eseguito interamente in modo manuale.

3 Analisi del Sentimento

Una volta ottenuto il dataset, è stato fondamentale **analizzare il sentimento espresso in ciascun tweet** nei confronti dei singoli artisti, al fine di determinare quale fosse l'**opinione degli utenti**. Questo step si è rivelato essenziale per il raggiungimento dell'obiettivo finale, ovvero la costruzione di un **sistema in grado di valutare il sentiment del pubblico** durante il Festival.
Per l'analisi del sentimento è stato utilizzato un modello pre-addestrato, disponibile pubblicamente e basato su `RoBERTa` [5], una variante ottimizzata di BERT [4] `cardiffnlp/twitter-roberta-base-sentiment-analysis` [6].

3.1 Scelta del Modello

Il modello scelto è in grado di **classificare automaticamente** i testi brevi in base al **tono emotivo predominante**. È stato eseguito il fine-tuning della rete sul dataset [1] - una raccolta di tweet (contenuto testuale) e la rispettiva *label* associata, che ci permette di classificare il testo in diverse categorie di sentimento. Tuttavia, il modello scelto, *cardiffnlp/twitter-roberta-base-sentiment-latest*, dato in input un tweet testuale, restituisce in output la **label di appartenenza più probabile** e la probabilità di appartenere a tale etichetta. Le label in cui il modello classifica del testo sono le seguenti:

- `Negative`

- `Neutral`

- `Positive`

Un esempio di output può essere il seguente:

```
[{'label': 'Negative', 'score': 0.7236}]
```

Queste sono una pool eccessiva di label, infatti delle tre totali vengono trovate di rilievo (per i fini progettuali) solamente le etichette **Negative** e **Positive**. La probabilità totale assegnata dal modello alle label è *1*: questo ci aiuta nella conversione della **classificazione** ternaria in una **binaria** tramite la ridistribuzione paritaria della percentuale da rimuovere. Nello specifico lo score della label **Neutral** va distribuito nelle altre due etichette, permettendo di ottenere risultati non ambigui per una migliore esecuzione delle fasi successive.
A questo punto abbiamo il necessario per **valutare ogni tweet** di cui siamo venuti in possesso attraverso lo scraping. È stato prima caricato il modello, poi eseguita la fase di inferenza per ogni tweet del dataset e infine prodotto un file csv analogo a quello descritto nella sezione precedente. Il file generato è costituito dalle seguenti colonne:

- *Artista*

- *Datetime*

- *Contenuto*

- *Sentimento*

Per un'analisi più approfondita sono stati generati dei **grafici ausiliari** per comprendere l'analisi del sentimento dei tweet, come nelle figure 4, 5 e 6.

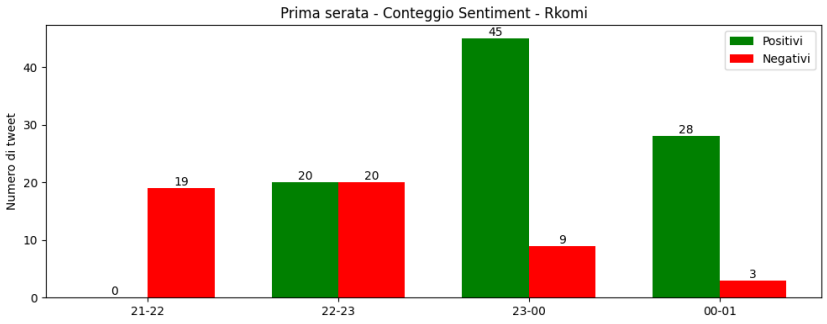

Figura 4: Conteggio per Sentiment dei post dell'artista Rkomi durante la prima serata

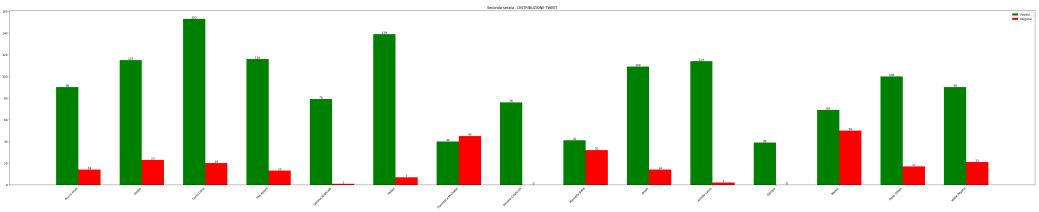

Figura 5: Conteggio per Sentiment dei post di tutti gli artisti in gara durante la seconda serata

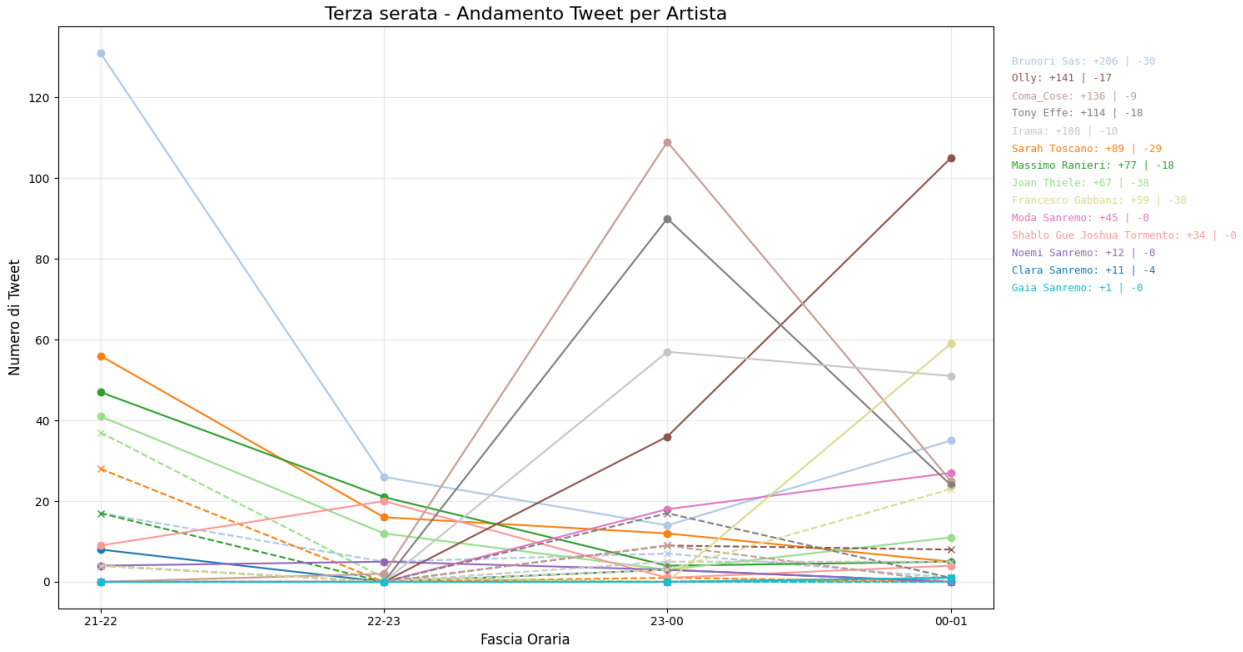

Figura 6: Andamento del Sentiment per gli artisti in gara durante la terza serata

4 Modello di Previsione

L'obiettivo attuale è quello di prevedere il vincitore di Sanremo per il televoto. Prima di tutto è bene definire il problema.

4.1 Definizione del problema

Il progetto vuole creare uno strumento di ***early prediction*** che si basa sui tweet degli utenti su X dopo aver rilevato il sentiment per ogni post del dataset. A questo punto definiamo formalmente la situazione corrente:

1. **Rilevazione del sentiment**: ogni tweet è stato classificato e ne è stato identificato il sentiment.

2. **Aggregazione temporale**: i tweet classificati sono stati raggruppati in quattro **fasce orarie** consecutive (21-22, 22-23, 23-00, 00-01). Per ciascuna serata e per ciascun artista si ottiene così una sequenza di vettori

   $$(p_t, n_t)$$

   dove p_t è il numero di tweet positivi e n_t è il numero di tweet negativi nell'intervallo di tempo attuale t.

3. **Early prediction**: partendo dai conteggi osservati nelle finestre orarie precedenti a quella presente (t) viene costruito un modello in grado di stimare i conteggi futuri per i tweet positivi e negativi partendo dal set

   $$\{(p_1, n_1), ..., (p_k, n_k)\}$$

   e quindi cercando di ottenere:

   $$(p_{t+1}, n_{t+1})$$

   dove:

   - k: rappresenta il numero della **finestra oraria osservata** (ovvero di tutti i tweet già conteggiati). Quindi se $k = 1$ sto cercando di prevedere i dati della seconda finestra oraria usando come dati di input quelli della prima.
   - t: rappresenta la **fascia oraria attuale**, facendo un piccolo abuso di notazione possiamo definire $t+1$ come la successiva fascia oraria.

   È quindi chiara la correlazione tra t e k: esse convivono insieme e ci permettono di definire matematicamente le finestre orarie da valutare e l'istante di tempo in cui siamo, ovvero la **finestra oraria corrente**.

4.1.1 Previsione al Tempo t

Uno degli aspetti più interessanti del lavoro è la capacità di effettuare una **previsione in tempo reale** (o in differita, impostando manualmente il tempo t) sull'esito della gara, simulando scenari in cui il modello cerca di **anticipare il vincitore** in un determinato momento di una qualsiasi serata, basandosi solo sui dati disponibili fino a quell'istante (t). Tutti i dati successivi a tale istante vengono esclusi, rendendo la previsione coerente con un contesto di **previsione parziale**. È bene notare che, avvicinandosi alla fine della gara, i dati raccolti riflettono con maggiore esattezza l'esito reale del pubblico, considerando che il modello utilizza come dati di allenamento gli n istanti di tempo precedenti al tempo t (`lookback`). Di conseguenza, la qualità e l'affidabilità della previsione migliorano progressivamente, fino a convergere in molti casi con l'effettivo risultato della competizione. Questo approccio consente non solo di valutare la performance del modello nel predire il vincitore finale, ma anche di esplorare la dinamica della percezione del pubblico nel corso della serata, identificando eventuali cambiamenti improvvisi o trend significativi.

4.1.2 Implementazione

L'istante di tempo t viene definito dalla funzione `get_time_to_predict()`:

```
TIME_TO_PREDICT = get_time_to_predict(SERATE[-1], "23:53:00")
```

che prende in input due parametri:

1. La **serata** su cui effettuare la previsione (in questo caso l'ultima).
2. L'**orario** preciso in cui effettuare la previsione (datetime).

La funzione `load_data_night()` **carica tutti i dati** dei post raccolti fino alla serata e all'orario scelti. Infine viene utilizzata la funzione `aggregate_data()` per il raggruppamento dei dati: calcola per ogni artista il positivi e negativi. Questi dati vengono filtrati ulteriormente escludendo le fasce orarie e le serate non rilevanti e preparati per il modello tramite la funzione `prepare_lstm_data_with_labels()` che **normalizza i dati** e **codifica le etichette** degli artisti per renderli compatibili con l'apprendimento del modello.

4.2 Architettura del modello

È stato costruito un **early predictor** basato su *LSTM*, col fine di catturare le dinamiche temporali dei conteggi di tweet positivi e negativi restituendo

così una *previsione doppia* (positivi e negativi) dei conteggi futuri.
Il modello è costruito come segue:

- **Layer LSTM** da 128 unità
- **Dropout layer** - per evitare overfitting
- **Layer LSTM** da 64 unità
- **Dropout layer**
- **Layer denso** da 64 unità intermedio che utilizza `ReLU`
- **Dropout layer**
- **Layer denso** da 2 unità per ottenere la previsione del conteggio di tweet positivi e negativi

Come funzione di loss è stata scelta la **MSE** poiché quello che stiamo affrontando è un problema di regressione. Come ottimizzatore è stato scelto **Adam**.

4.3 Training

4.3.1 Preparazione dei Dati

Prima di tutto è fondamentale **aggregare i dati** per ciascun artista e intervallo orario fino a quello attuale t, ottenendo così una serie temporale bidimensionale da 2 feature. Per ogni artista è stata inoltre normalizzata tale serie tramite un *MiniMaxScaler* sui valori $[0, 1]$, favorendo la compatibilità di comprensione e allenamento del modello.
Dopodiché è stata definita una **lookback window**, che serve a definire la quantità di passi temporali precedenti a quello attuale.
Per preservare l'ordine cronologico e la valutazione di forecasting del modello, lo split dei dati di addestramento è stato eseguito **senza shuffle** e con una frazione di dati di **validazione pari al 20%**.
Per evitare l'overfitting è stata applicata la callback *early stopping* sulla metrica *validation loss* con una pazienza pari a 5 epoche.

5 Analisi Inferenza Finale

Dopo l'addestramento del modello si è passati alla **fase finale di inferenza** e di **valutazione dei risultati**. Abbiamo allenato il modello con i seguenti parametri per la variabile `TIME_TO_PREDICT = get_time_to_predict(SERATE[-1], "23:53:00")`. In questo modo riusciamo a collocarci nell'ultima fascia oraria disponibile nell'ultima serata utile sia per l'inferenza che per il training del modello, ottenendo, sperabilmente, il vincitore di Sanremo per il televoto.
Alla fine dell'allenamento sui dati corrispondenti alla nostra posizione temporale, è stata eseguita inferenza ottenendo i risultati mostrati all'interno della Tabella 1 dove:

- **Positivo**: numero dei post predetti con sentiment positivo
- **Negativo**: numero dei post predetti con sentiment negativo
- **Totale**: calcolato come $Positivo - Negativo$

L'inferenza del modello produce **due output in floating point** che vengono troncati per simulare dei valori interi. Di conseguenza, la colonna *Totale* può non coincidere esattamente con la somma dei post positivi e negativi, perché il calcolo avviene in **virgola mobile**, ma i risultati vengono mostrati come numeri interi troncati.

Posizione	Artista	Totale	Positivi	Negativi
1	Brunori Sas	16	17	2
2	Olly	15	17	1
3	Lucio Corsi	15	16	2
4	Fedez	15	15	1
5	Irama	12	14	2
6	Achille Lauro	11	12	0
7	Bresh	11	13	2
8	Giorgia	11	11	1
9	Coma_Cose	10	13	2
10	Rocco Hunt	10	13	3
11	Rose Villain	9	12	2
12	Simone Cristicchi	9	11	2
13	Tony Effe	9	11	2
14	Serena Brancale	9	9	0
15	The Kolors	7	12	5
16	Elodie	6	11	5
17	Shablo Gue & Joshua Tormento	6	6	0
18	Sarah Toscano	5	8	3
19	Modà	5	5	0
20	Francesco Gabbani	5	7	2
21	Francesca Michielin	5	9	4
22	Willie Peyote	4	9	5
23	Massimo Ranieri	4	6	2
24	Noemi	4	5	1
25	Rkomi	4	8	5
26	Clara	3	3	0
27	Joan Thiele	1	5	3
28	Gaia	1	1	0
29	Marcella Bella	1	5	4

Tabella 1: Predizioni del numero di post per artista

I risultati ottenuti rispecchiano, in modo quasi del tutto preciso, i risultati della *CLASSIFICA GIURIA DEL TELEVOTO* di *5° Serata - Sabato - Prima fase* pubblicato ufficialmente dalla RAI [7] e mostrata nella Figura 7.

5° Serata - Sabato - Prima fase

Campioni (29 artisti)

CLASSIFICA GIURIA DEL TELEVOTO

Posizione	Artista	Percentuale
1	BRUNORI SAS	17,4402
2	OLLY	16,2643
3	LUCIO CORSI	12,3138
4	FEDEZ	9,6502
5	ACHILLE LAURO	8,5419
6	GIORGIA	8,0316
7	SIMONE CRISTICCHI	3,7542
8	IRAMA	2,8916
9	FRANCESCO GABBANI	2,3763
10	BRESH	2,1521
11	TONY EFFE	1,8629
12	ROCCO HUNT	1,55
13	MODÀ	1,2908
14	COMA_COSE	1,2409
15	ELODIE	1,1444
16	WILLIE PEYOTE	1,0907
17	ROSE VILLAIN	1,0567
18	SARAH TOSCANO	0,9880
19	SHABLO	0,8968
20	SERENA BRANCALE	0,8728
21	NOEMI	0,7909
22	THE KOLORS	0,7109
23	FRANCESCA MICHIELIN	0,5758
24	RKOMI	0,5365
25	CLARA	0,4944
26	MASSIMO RANIERI	0,4035
27	JOAN THIELE	0,3847
28	MARCELLA BELLA	0,3657
29	GAIA	0,3276

Figura 7: *CLASSIFICA GIURIA DEL TELEVOTO* di *5° Serata - Sabato - Prima fase* pubblicato ufficialmente dalla RAI [7]

5.1 Confronto dei Risultati

In questa sezione vengono comparati i risultati ottenuti dal nostro modello con i risultati ufficiali. In primo luogo, l'analisi è stata effettuata **sull'intera graduatoria**, poi, in modo più accurato, **sulle prime ed ultime cinque posizioni**.

5.1.1 Classifica Totale

Di seguito viene riportata la Tabella 2 che evidenzia le **posizioni esatte e inesatte predette dal modello** e confrontate con quelle reali in Figura 7. Per comprendere a pieno i risultati del modello è stata aggiunta la colonna *Errore* che evidenzia di quante posizioni si discostano i risultati ottenuti rispetto a quelli reali. Infatti, per ogni artista viene calcolato come segue:

$$Errore = PosizioneReale - PosizionePredetta$$

Si ottiene un valore *positivo* se il nostro modello ha posizionato il concorrente più in alto nella tabella, *negativo* se invece è più in basso. È possibile anche studiare l'errore medio della classificazione sull'intera tabella:

$$mediaErroreTotale = \frac{80}{29} = 2,75$$

L'errore medio per ogni posizione è quindi poco più di 2 posizioni.

Posizione	Errore	Artista	Totale	Positivi	Negativi
1	0	Brunori Sas	16	17	2
2	0	Olly	15	17	1
3	0	Lucio Corsi	15	16	2
4	0	Fedez	15	15	1
5	+3	Irama	12	14	2
6	-1	Achille Lauro	11	12	0
7	+3	Bresh	11	13	2
8	-2	Giorgia	11	11	1
9	+5	Coma_Cose	10	13	2
10	+2	Rocco Hunt	10	13	3
11	+6	Rose Villain	9	12	2
12	-5	Simone Cristicchi	9	11	2
13	-2	Tony Effe	9	11	2
14	+6	Serena Brancale	9	9	0
15	+7	The Kolors	7	12	5
16	-1	Elodie	6	11	5
17	+2	Shablo Gue & Joshua Tormento	6	6	0
18	0	Sarah Toscano	5	8	3
19	-6	Modà	5	5	0
20	-11	Francesco Gabbani	5	7	2
21	+2	Francesca Michielin	5	9	4
22	-6	Willie Peyote	4	9	5
23	+3	Massimo Ranieri	4	6	2
24	-3	Noemi	4	5	1
25	-1	Rkomi	4	8	5
26	-1	Clara	3	3	0
27	0	Joan Thiele	1	5	3
28	+1	Gaia	1	1	0
29	-1	Marcella Bella	1	5	4

Tabella 2: Classifica predizioni post con righe in evidenza e colonna Errore

È bene sottolineare che i risultati del modello sono ovviamente più precisi nella **Top 5** e nella **Bottom 5** della classifica, questo perché le posizioni intermedie, come si può notare dalla classifica ufficiale, hanno una ridotta differenza percentuale.

5.1.2 Top 5 e Bottom 5

Le seguenti tabelle riportano una sola porzione di ogni classifica precedente: più precisamente vengono visualizzate le rispettive *Top 5* (Figure 3 e 4) e *Bottom 5* (Figure 5 e 6).

Posizione	Artista	Totale	Positivi	Negativi
1	Brunori Sas	16	17	2
2	Olly	15	17	1
3	Lucio Corsi	15	16	2
4	Fedez	15	15	1
5	Irama	12	14	2

Tabella 3: Top 5 delle predizioni effettuate dal modello

Posizione	Artista	Percentuale (%)
1	Brunori Sas	17,4402
2	Olly	16,2643
3	Lucio Corsi	12,3138
4	Fedez	9,6502
5	Achille Lauro	8,5419

Tabella 4: Top 5 ufficiale

Posizione	Artista	Totale	Positivi	Negativi
25	Rkomi	16	17	2
26	Clara	15	17	1
27	Joan Thiele	15	16	2
28	Gaia	15	15	1
29	Marcella Bella	12	14	2

Tabella 5: Bottom 5 delle predizioni effettuate dal modello

Posizione	Artista	Percentuale (%)
25	Clara	0,4944
26	Massimo Ranieri	0,4035
27	Joan Thiele	0,3837
28	Marcella Bella	0,3657
29	Gaia	0,3276

Tabella 6: Bottom 5 ufficiale

Le tabelle evidenziano *in verde i risultati uguali* e *in rosso quelli sbagliati*, notiamo infatti che il nostro modello ha predetto in modo corretto **4/5 della Top 5** ufficiale del televoto, portando a un ottimo risultato.
Per capire a pieno i risultati del modello possiamo calcolare l'errore medio:

$$mediaErroreTop5 = \frac{3}{5} = 0,6$$

$$mediaErroreBottom5 = \frac{4}{5} = 0,8$$

Capiamo quindi che il modello nelle due *mini-classifiche* sbaglia la classificazione per poco meno di una posizione.

5.2 Grafici delle Previsioni

Di seguito, nelle Figure 8 e 9, vengono riportati i grafici per **l'andamento dei tweet predetti** durante l'ultima serata del Festival di Sanremo 2025.

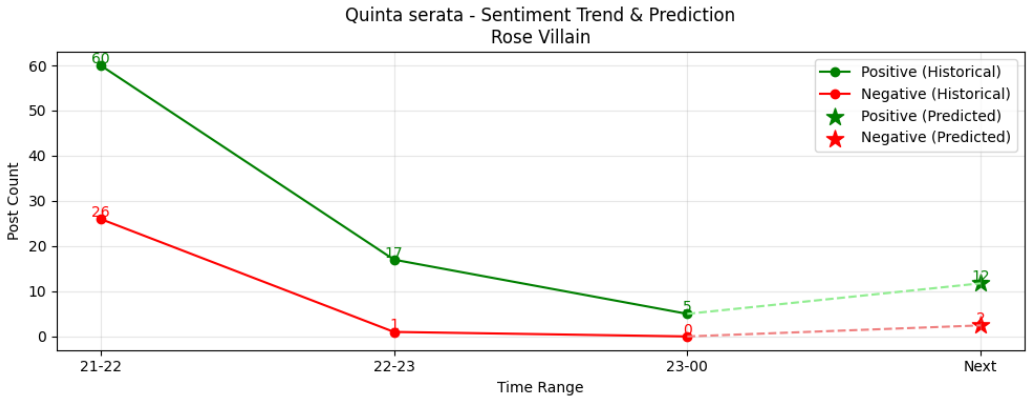

Figura 8: Predizione del Sentiment per Rose Villain durante la 5° serata

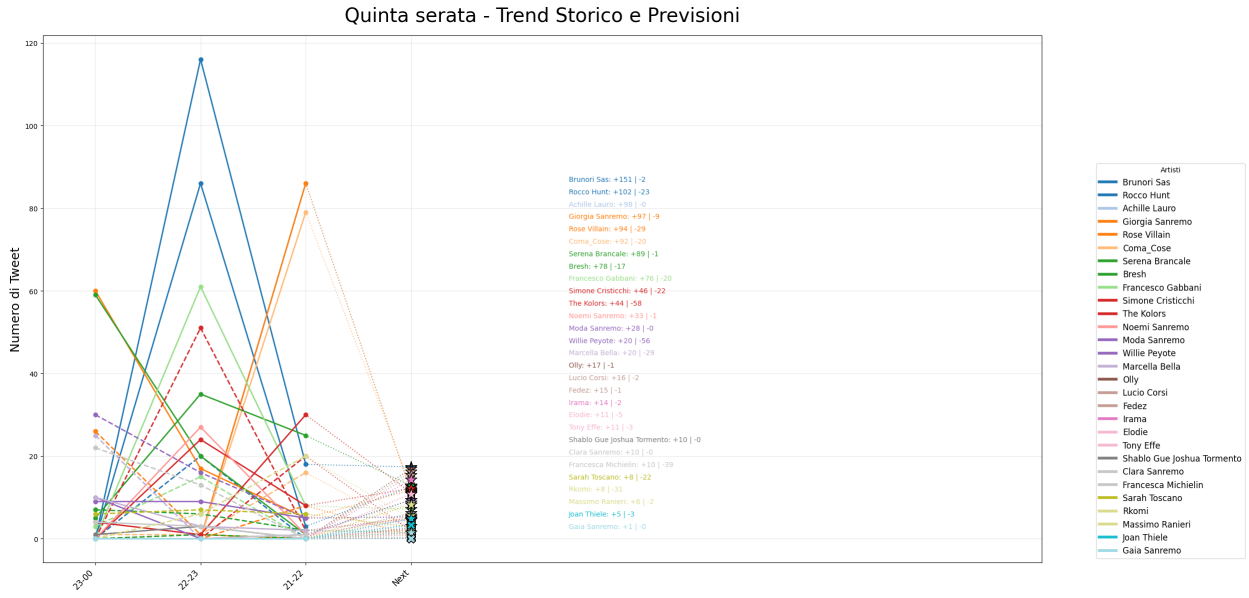

Figura 9: Predizione del Sentiment per ogni artista in gara durante la 5° serata

Riferimenti bibliografici

[1] Francesco Barbieri et al. "TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification". In: *Proceedings of Findings of EMNLP*. 2020.

[2] Software Freedom Conservancy. *Supported Browsers*. `https://www.selenium.dev/documentation/webdriver/browsers/`. Visitato il: 26/05/2025. 2025.

[3] Software Freedom Conservancy. *The Selenium Browser Automation Project*. `https://www.selenium.dev/documentation/`. Visitato il: 21/05/2025. 2025.

[4] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: `1810.04805 [cs.CL]`. URL: `https://arxiv.org/abs/1810.04805`.

[5] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: `1907.11692 [cs.CL]`. URL: `https://arxiv.org/abs/1907.11692`.

[6] Daniel Loureiro et al. "TimeLMs: Diachronic Language Models from Twitter". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Dublin, Ireland: Association for Computational Linguistics, mag. 2022, pp. 251–260. DOI: `10.18653/v1/2022.acl-demo.25`. URL: `https://aclanthology.org/2022.acl-demo.25`.

[7] RAI. *RAI*. `https://www.rai.it/dl/doc/2025/02/16/1739717811627_sanremo-2025-risultati-televoto-campioni.pdf`. Visitato il: 26/05/2025. 2025.