



UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

DIPARTIMENTO DI
INFORMATICA



Architettura Integrata di un Agente Computazionale per l'Analisi Finanziaria: Un Approccio Multi-Paradigma Applicato ai Dati di Mercato di Meta Platforms (Facebook)

Gruppo di lavoro

1. *Alessandro Palmitessa, Matricola: 800148,
a.palmitessa20@studenti.uniba.it*
2. *Alessandro Moré, Matricola: 802636, a.more@studenti.uniba.it*

Repository GitHub

https://github.com/B0bby900/Facebook_Financial_Agent

Anno Accademico: 2025-2026

Architettura Integrata di un Agente Computazionale per l'Analisi Finanziaria: Un Approccio Multi-Paradigma Applicato ai Dati di Mercato di Meta Platforms (Facebook).....	1
Introduzione.....	3
Parte I: Fondamenti Teorici e Analisi del Dominio.....	3
1.1 Caratterizzazione del Dataset e Ontologia del Dominio.....	3
1.2 Pulizia e Preprocessing dei Dati.....	5
Parte II: Ragionamento Logico e Rappresentazione della Conoscenza.....	5
2.1 Sommario: Progettazione della Base di Conoscenza.....	5
2.2 Ragionamento su Relazioni Temporal Complesse.....	6
2.3 Risultati dell'inferenza logica.....	7
2.4 Strumenti Utilizzati.....	8
Parte III: Apprendimento automatico e scoperta di pattern.....	8
3.1 Apprendimento non supervisionato: Clustering e regimi di mercato.....	8
3.2 Apprendimento Supervisionato: alberi di decisione e regressione.....	9
3.3 Decisioni di progetto.....	10
3.4 Valutazioni.....	11
3.5 Valutazioni delle Prestazioni.....	12
3.6 Strumenti utilizzati.....	17
Parte IV: Ragionamento probabilistico e reti bayesiane.....	17
4.1 Reti Bayesiane discrete.....	17
4.2 Reti bayesiane a valori continui (ibride).....	19
4.3 Decisioni di progetto.....	20
4.4 Valutazioni.....	21
4.5 Scelta dei Parametri.....	24
4.6 Strumenti utilizzati.....	25
Parte V: Implementazione modulare del codice.....	25
5.1 Struttura del progetto.....	26
Parte VI: Conclusioni.....	26
6.1 Schermate dei grafici risultanti e descrizione relativa.....	27
6.2 Spiegazione grafici.....	35
6.3 Spiegazione risultati.....	36
6.3.1 Sintesi delle Valutazioni.....	36
6.3.2 Limiti e Possibili Sviluppi Futuri.....	36
Parte VII: Riferimenti bibliografici.....	37

Nota sull'Esecuzione: Per eseguire il codice, assicurarsi di avere installato SWI-Prolog e che sia accessibile nel PATH di sistema, oltre alle librerie Python specificate in requirements.txt. Le visualizzazioni (grafici) verranno generate a runtime durante l'esecuzione del modulo di clustering.

Introduzione

Il presente rapporto di ricerca delinea la progettazione, la strutturazione teorica e l'implementazione tecnica di un agente computazionale avanzato, dedicato all'analisi del dominio finanziario, con specifico riferimento al titolo azionario Facebook (ora Meta Platforms). L'obiettivo primario è dimostrare come l'integrazione di paradigmi simbolici e sub-simbolici — spaziando dalla Rappresentazione della Conoscenza e il Ragionamento Logico (Prolog) fino all'Apprendimento Automatico Supervisionato e Non Supervisionato e alle Reti Bayesiane (continue e discrete) — possa superare i limiti dei singoli approcci isolati.

L'analisi si fonda rigorosamente sui principi esposti nella **terza edizione** del testo *Artificial Intelligence: Foundations of Computational Agents* di David L. Poole e Alan K. Mackworth.¹ Il dominio finanziario, caratterizzato da incertezza intrinseca, osservabilità parziale e dinamiche temporali complesse, rappresenta il banco di prova ideale per testare l'efficacia di un'architettura ibrida. Attraverso l'elaborazione dei dataset forniti (FB.csv e facebook-dataset.csv)³, l'agente non si limita a prevedere i prezzi futuri, ma "comprende" lo stato del mercato attraverso l'inferenza logica e quantifica il rischio mediante modelli probabilistici.

Parte I: Fondamenti Teorici e Analisi del Dominio

La progettazione di un agente intelligente, come discusso nella Parte I del testo di riferimento¹, richiede una definizione rigorosa dell'ambiente in cui esso opera e delle sue capacità di percezione. Nel contesto dei mercati finanziari, l'ambiente è stocastico e dinamico.

1.1 Caratterizzazione del Dataset e Ontologia del Dominio

L'analisi prende le mosse dai dati storici di trading di Facebook. I dataset forniti³ offrono una vista granulare (giornaliera) delle attività di mercato. Per strutturare una Base di Conoscenza (KB) efficace, è necessario prima definire un'ontologia che mappi i dati grezzi in concetti utilizzabili dall'agente.

Struttura dei Dati e Semantica:

I file CSV contengono tuple che rappresentano lo stato del mercato in un dato giorno t .

Le variabili fondamentali identificate sono:

- **Date:** L'ancora temporale che identifica univocamente l'evento ("individuo" nel linguaggio di Poole & Mackworth, Capitolo 15 ⁴).
- **Open (P_{open}), High (P_{high}), Low (P_{low}), Close (P_{close}):** Queste variabili continue definiscono la geometria della sessione di trading. La relazione tra esse permette di derivare concetti di livello superiore come la "Volatilità" ($P_{high} - P_{low}$) e la "Direzionalità" ($P_{close} - P_{open}$).
- **Volume (V):** Una variabile discreta (o continua approssimata) che rappresenta la quantità di azioni scambiate. Nell'ambito dell'intelligenza artificiale, il volume agisce come un segnale di "affidabilità" dell'informazione di prezzo: un movimento di prezzo con alto volume ha una probabilità a posteriori maggiore di indicare un trend sostenibile rispetto a uno con basso volume.
- **Profit:** Presente nel secondo dataset ³, è una variabile derivata ($P_{close} - P_{open}$) che funge da *target* esplicito per i moduli di apprendimento supervisionato.

Analisi Esplorativa e Visualizzazione: L'analisi preliminare dei dati (EDA) è fondamentale per informare la progettazione dei modelli successivi. Seguendo le pratiche descritte nel Capitolo 7 per la preparazione dei dati ⁵, si osservano distribuzioni non gaussiane nei rendimenti.

Le visualizzazioni generate includono:

- ☐ **Grafici a Candela (Candlestick Charts):** Essenziali per visualizzare la relazione OHLC e identificare pattern come l'*Engulfing*, che saranno formalizzati in Prolog.
- ☐ **Heatmap di Correlazione:** Utilizzando librerie come Seaborn ⁶, si evidenzia una forte correlazione seriale tra il prezzo di chiusura al tempo t e $t - 1$, giustificando l'uso di modelli regressivi e Reti Bayesiane Dinamiche.
- ☐ **Istogrammi dei Volumi:** Mostrano una distribuzione a "coda lunga", suggerendo che gli eventi di alto volume sono rari ma significativi, ideali per la discretizzazione in stati ("Alto", "Medio", "Basso") per le Reti Bayesiane discrete.

1.2 Pulizia e Preprocessing dei Dati

Prima di alimentare la Base di Conoscenza, i dati richiedono una normalizzazione. Il dataset facebook-dataset.csv presenta una colonna Profit intera, utile per la classificazione.³ Tuttavia, per l'analisi probabilistica continua, è necessario calcolare i *rendimenti logaritmici*

$(\ln(P_t/P_{t-1}))$ per stabilizzare la varianza, una tecnica standard nell'analisi delle serie storiche che facilita l'apprendimento dei parametri nelle Reti Bayesiane Gaussiane (Capitolo 10⁷).

Parte II: Ragionamento Logico e Rappresentazione della Conoscenza

La terza edizione di *Artificial Intelligence* pone una forte enfasi sulla rappresentazione logica come mezzo per gestire la certezza e le definizioni strutturali (Parte II e V del libro).¹ Mentre l'apprendimento automatico eccelle nel trovare pattern statistici, la logica è insostituibile per definire *regole inviolabili* e relazioni complesse tra entità.

2.1 Sommario: Progettazione della Base di Conoscenza

Per implementare il modulo di ragionamento, si utilizza **Prolog**. La scelta di questo linguaggio è dettata dalla sua capacità di gestire *Clausole Definite Proporzionali* e ragionamento relazionale (Capitolo 5 e 15).⁴

Assiomatizzazione:

La base di conoscenza (KB) è divisa in due componenti:

1. **Extensional Database (EDB):** I fatti grezzi estratti dal CSV. Ogni riga del dataset diventa un fatto del tipo: `stock_day(Date, Open, High, Low, Close, Volume)`. Questo approccio trasforma il dataset statico in un database deduttivo interrogabile.
2. **Intensional Database (IDB):** Le regole che definiscono i concetti finanziari.

Estratto di Codice e Spiegazione (Prolog):

Definiamo regole che catturano concetti di analisi tecnica. Ad esempio, un "Trend Rialzista" (Bullish) non è solo un numero positivo, ma una relazione logica tra apertura e chiusura.

```
% Fatti (Esempio generato dallo script Python)
```

```
stock_day('2014-02-14', 67.50, 67.58, 66.72, 67.09, 36694900, 0.00).
```

```
stock_day('2014-02-18', 66.94, 67.54, 66.07, 67.30, 43809900, 1.00).
```

```
% Regole Logiche (Capitolo 5 - Clausole Definite)
```

```
% Un giorno è 'bullish' se la chiusura è superiore all'apertura
```

```
is_bullish(Date) :-
```

```
    stock_day(Date, Open, _, _, Close, _, _),
```

```
    Close > Open.
```

```
% Un giorno è 'bearish' se la chiusura è inferiore all'apertura
```

```
is_bearish(Date) :-
```

```
    stock_day(Date, Open, _, _, Close, _, _),
```

```
    Close < Open.
```

```
% Definizione di Volatilità Significativa
```

```
% Si verifica quando lo spread
```

```
High-Low supera una soglia (es. 5% dell'Open)
```

```
high_volatility(Date) :-
```

```
    stock_day(Date, Open, High, Low, _, _, _),
```

```
    Diff is High - Low,
```

```
    Soglia is Open * 0.05,
```

```
    Diff > Soglia.
```

2.2 Ragionamento su Relazioni Temporalì Complesse

Uno dei vantaggi distintivi della logica rispetto ai modelli statistici semplici è la capacità di definire pattern strutturali complessi. Prendiamo ad esempio il pattern "Bullish Engulfing", un forte segnale di inversione di tendenza. Questo pattern coinvolge due individui temporali distinti, D_1 e D_2 (oggi e ieri), e una serie di vincoli geometrici tra i loro prezzi. In Prolog, questo si esprime naturalmente attraverso la congiunzione di vincoli (Capitolo 4, *Reasoning with Constraints*¹):

```

% Definito come una congiunzione di vincoli geometrici su due giorni (D1=Ieri,
D2=Oggi)
engulfing_bullish(DateOggi) :-
    % Recupera Dati Oggi (D2)
    stock_day(DateOggi, OpenOggi, _, _, CloseOggi, _, _),
    is_bullish(DateOggi), % Oggi verde

    % Recupera il Giorno Precedente esatto (D1)
    next_day(Dateleri, DateOggi),
    stock_day(Dateleri, Openleri, _, _, Closeleri, _, _),

    is_bearish(Dateleri), % Ieri rossa

    % Condizione di Engulfing (Geometrica)
    % [CORREZIONE] Usiamo =< e >= per includere i casi senza gap estremi
    OpenOggi =< Closeleri, % Apre sotto o uguale alla chiusura di ieri
    CloseOggi >= Openleri. % Chiude sopra o uguale all'apertura di ieri

```

Questo livello di astrazione permette all'agente di "ragionare" sul mercato: non vede solo numeri, ma riconosce configurazioni semantiche definite a priori dall'esperto umano.

2.3 Risultati dell'inferenza logica

L'integrazione tra Python e Prolog ha permesso di analizzare semanticamente 1230 giorni di trading. Le query effettuate sulla Knowledge Base hanno prodotto i seguenti riscontri oggettivi:

- Rilevamento Pattern Complessi: Il sistema ha identificato correttamente 49 occorrenze del pattern *Bullish Engulfing*.
- Verifica Temporale: I primi casi rilevati (es. 13 Febbraio 2014, 29 Aprile 2014) corrispondono effettivamente a inversioni di trend storiche riscontrabili sui grafici, validando la correttezza delle regole scritte in [rules.pl](#).
- Analisi Volatilità: Sono stati classificati 43 giorni come "High Volatility",

permettendo all'agente di attivare stati di allerta specifici in tali date.

```
--- Risultati Ragionamento Logico (Prolog) ---  
Pattern 'Bullish Engulfing' trovato in 49 casi.  
Primi 3 casi: ['2014-02-13', '2014-04-29', '2014-05-19']  
Giorni di 'high_volatility': 43
```

Output dell'esecuzione delle query Prolog.

2.4 Strumenti Utilizzati

- **Motore di Inferenza: SWI-Prolog** (utilizzato tramite i file `.pl` nella cartella `knowledge_base/`), scelto per la sua efficienza nella gestione della logica a clausole definite.
- **Bridge Tecnologico:** Libreria **PySwip**, utilizzata per integrare i fatti derivati dall'analisi dei dati in Python con la Base di Conoscenza Prolog.
- **Formalismi Logici:** Implementazione di regole per il calcolo delle variazioni percentuali e trend (bullish/bearish) basate su logica del prim'ordine.

Parte III: Apprendimento automatico e scoperta di pattern

Mentre la logica gestisce ciò che sappiamo definire, l'apprendimento automatico (Machine Learning) gestisce ciò che dobbiamo scoprire dai dati. Facendo riferimento alla Parte III del testo (Learning and Reasoning with Uncertainty)⁸, implementiamo moduli di apprendimento supervisionato e non supervisionato.

3.1 Apprendimento non supervisionato: Clustering e regimi di mercato

Il Capitolo 10 descrive l'apprendimento non supervisionato come il processo di trovare strutture nascoste senza etichette target.⁷ Nel dominio finanziario, questo si traduce nell'identificazione di "Regimi di Mercato" (es. accumulazione, distribuzione, panico) che

non sono esplicitamente annotati nel dataset.

Algoritmo K-Means: Utilizziamo l'algoritmo **k-means** (Sezione 10.3.1 ⁷) per partizionare i giorni di trading in k cluster. L'algoritmo minimizza l'errore di predizione (somma delle distanze quadratiche).

Selezione delle Feature:

Per il clustering, non usiamo i prezzi grezzi (che crescono nel tempo rendendo i dati non stazionari), ma feature normalizzate che descrivono il *comportamento*:

1. **Daily Return:** Rendimento percentuale giornaliero.
2. **Normalized Volume:** Volume rapportato alla media mobile a 30 giorni.
3. **Intraday Volatility:** Spread High-Low normalizzato.

Determinazione di K:

Applicando il "metodo del gomito" (Elbow Method) e analizzando la riduzione della varianza intra-cluster, identifichiamo $k = 4$ come numero ottimale per il dataset Facebook. I cluster risultanti possono essere interpretati semanticamente:

- *Cluster 0:* Bassa volatilità, volume medio (Giorni di consolidamento).
- *Cluster 1:* Alta volatilità positiva, alto volume (Breakout rialzisti).
- *Cluster 2:* Alta volatilità negativa, volume estremo (Panic selling/News negative).
- *Cluster 3:* Bassa volatilità, basso volume (Giorni festivi o pre-news).

Questa discretizzazione automatica fornisce nuovi "stati" che possono essere utilizzati come nodi nelle Reti Bayesiane.

3.1.1 Preprocessing e Clustering per l'Analisi dei Regimi

Questi riferimenti mostrano come l'agente prepara i dati e identifica i regimi di mercato senza etichette.

- **Normalizzazione del Volume ([src/preprocessing.py](#)):** Prima del clustering, il volume viene normalizzato per renderlo una metrica di intensità relativa e non un valore assoluto che cresce nel tempo.

```
# Da src/preprocessing.py
vol_sma_30 = self.df['Volume'].rolling(window=30).mean()
self.df['Volume_Normalizzato'] = self.df['Volume'] / vol_sma_30
```

Spiegazione: Questa operazione assicura che picchi di volume passati e presenti siano confrontabili, permettendo all'algoritmo **K-Means** di raggruppare i giorni in base all'attività anomala o stasi.

- **Clustering K-Means (src/learning.py):** Il codice implementa la segmentazione automatica dei dati in k=4 regimi.

```
# Configurazione estratta da config.xml ed eseguita in learning.py
kmeans = KMeans(n_clusters=k, random_state=42, n_init='auto')
self.df['Cluster'] = kmeans.fit_predict(X)
```

Spiegazione: Identifica configurazioni semantiche come "Panico" o "Breakout" basandosi su rendimento e volatilità, fornendo un contesto non supervisionato per l'inferenza Bayesiana.

3.2 Apprendimento Supervisionato: Alberi di decisione e regressione

Il Capitolo 7 del testo introduce l'apprendimento supervisionato.⁵

Qui, l'agente impara una funzione $f(X) \rightarrow Y$ per predire il futuro.

Alberi di Decisione (Decision Trees): Utilizziamo un Albero di Decisione (Sezione 7.3.1¹⁰) per classificare se il giorno successivo sarà "Profittevole" (Target=1) o "Non Profittevole" (Target=0). L'albero è preferito per la sua interpretabilità. Analizzando l'albero generato dal dataset Facebook, si nota che il nodo radice spesso corrisponde a variabili legate al **Momentum** (es. RSI o Moving Average), confermando la teoria finanziaria del *trend following*. L'albero suddivide lo spazio delle feature in regioni rettangolari, creando regole del tipo: SE (Volume > 50M) AND (SMA_7 > SMA_30) ALLORA Prob(Profit) = 0.65.

Regressione Lineare: Per la predizione continua del prezzo di chiusura (P_{t+1}), implementiamo una Regressione Lineare (Sezione 7.3.2¹¹).

$$P_{t+1} = w_0 + w_1 P_t + w_2 V_t + w_3 (P_{high} - P_{low})$$

Per evitare l'*overfitting* (Sezione 7.4), applichiamo la regolarizzazione (Ridge Regression), che penalizza i pesi elevati, costringendo il modello a generalizzare meglio sui dati non visti.¹²

3.2.1 Apprendimento Supervisionato e Integrità dei Dati Temporal

I riferimenti qui riguardano la capacità predittiva del trend e del prezzo.

- **Gestione della Serie Temporale (src/learning.py):** Il codice garantisce l'integrità del test isolando i dati futuri durante l'addestramento. L'uso di `shuffle=False` è fondamentale per simulare un ambiente di trading reale dove il futuro è ignoto.

```
# Estratto da Learner.supervised_regression (src/learning.py)
# Suddivisione cronologica per evitare look-ahead bias

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
shuffle=False)
```

Spiegazione: Garantisce che il modello impari solo dal passato per prevedere il futuro, rendendo le valutazioni di accuratezza (R2 Score del 95%) realistiche in un contesto di trading.

- **Regressione Ridge per Valori Continui (src/learning.py):** Per la previsione del prezzo target (Close del giorno successivo), l'agente utilizza una regressione regolarizzata che gestisce la scala differente delle feature e riduce l'impatto del rumore di mercato.

```
# Estratto da Learner.supervised_regression (src/learning.py)
# Pipeline: Standardizzazione -> Ridge Regression per gestire la
collinearità

model = make_pipeline(StandardScaler(), Ridge(alpha=1.0))
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Spiegazione: A differenza della regressione semplice, la **Ridge** penalizza i coefficienti troppo elevati, riducendo l'overfitting e permettendo all'agente di generalizzare meglio su dati di mercato rumorosi.

3.3 Decisioni di progetto

Le scelte implementative per il modulo di apprendimento sono state guidate dalla necessità di gestire la natura temporale e rumorosa dei dati finanziari.

- **Pipeline di Preprocessing:** Come osservabile nel file `preprocessing.py`, la classe `DataEngine` è stata progettata per trasformare i dati grezzi OHLC in metriche stazionarie. Questo è cruciale perché i modelli di Machine Learning (in particolare K-Means e Ridge) non performano bene su grandezze che cambiano scala nel tempo (come il prezzo assoluto).
- **Standardizzazione:** Nel file `learning.py`, è stata presa la decisione esplicita di utilizzare una `Pipeline` che include `StandardScaler`. Questo garantisce che variabili con magnitudini diverse (es. `Volume_Normalizzato` intorno a 1.0 vs `Rendimento_Giornaliero` intorno a 0.01) pesino equamente nel calcolo delle distanze euclidee per il clustering e nella regolarizzazione della regressione.
- **Gestione della Serie Temporale:** Una decisione critica di progetto riguarda lo splitting dei dati. Contrariamente al random shuffling standard, in `learning.py` il parametro `shuffle=False` è stato impostato nella funzione `train_test_split`.

3.4 Valutazioni

La validazione dei modelli implementati in `learning.py` segue metriche specifiche per ogni tipologia di apprendimento, configurate per quantificare l'errore e la capacità di generalizzazione.

- **Valutazione del Clustering (Non Supervisionato):** Per determinare la qualità della segmentazione dei regimi di mercato, il metodo `plot_elbow_method` in `learning.py` calcola la *WSS (Within-Cluster Sum of Squares)* al variare di **k**. Il punto di gomito suggerisce il compromesso ottimale tra compattezza dei cluster e complessità del modello.
- **Valutazione della Classificazione:** Per l'Albero di Decisione, non ci limitiamo alla semplice accuratezza (spesso fuorviante in dataset sbilanciati). Il codice utilizza `classification_report` per generare Precision, Recall e F1-Score per le classi **0** (Non profittevole) e **1** (Profittevole).

- **Valutazione della Regressione:** Per la predizione del prezzo continuo, vengono utilizzate quattro metriche distinte:
 1. **MSE (Mean Squared Error):** Per penalizzare gli errori grossolani nella predizione del prezzo.
 2. **R2 Score:** Per capire quanta varianza del prezzo futuro è spiegata dalle feature storiche (SMA, Volatilità, ecc.).
 3. **MAE (Mean Absolute Error):** È l'errore medio "tipico". Indica che, nella norma, le previsioni dell'agente si discostano dal prezzo reale di soli 2.86 dollari, confermando un'elevata affidabilità quotidiana.
 4. **MAX_ERROR:** È lo scostamento massimo registrato in un singolo giorno. Identifica la reazione del modello a shock di mercato anomali (outlier), non prevedibili tramite trend lineari.

```
# Da learning.py
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
max_err = max_error(y_test, y_pred)
```

3.5 Valutazioni delle Prestazioni

L'esecuzione del modulo di apprendimento, orchestrata dal file `main.py`, produce risultati che riflettono la configurazione definita in `config.xml`.

Configurazione del Dataset Il file `config.xml` imposta i parametri globali, definendo `<kmeans_clusters>4</kmeans_clusters>` e un `<test_size>0.2</test_size>`. Questo significa che il **20% dei dati più recenti** (corrispondenti agli ultimi anni di contrattazione) viene "nascosto" al modello durante il training e utilizzato esclusivamente per la validazione delle prestazioni (Testing Set).

Output del Clustering (Unsupervised) Il metodo `unsupervised_clustering` aggiunge una colonna etichetta al DataFrame. L'analisi dei centroidi risultanti permette di associare a ogni cluster (0-3) un'etichetta semantica (es. "Alta Volatilità/Basso Rendimento"), segmentando i regimi di mercato senza supervisione umana.

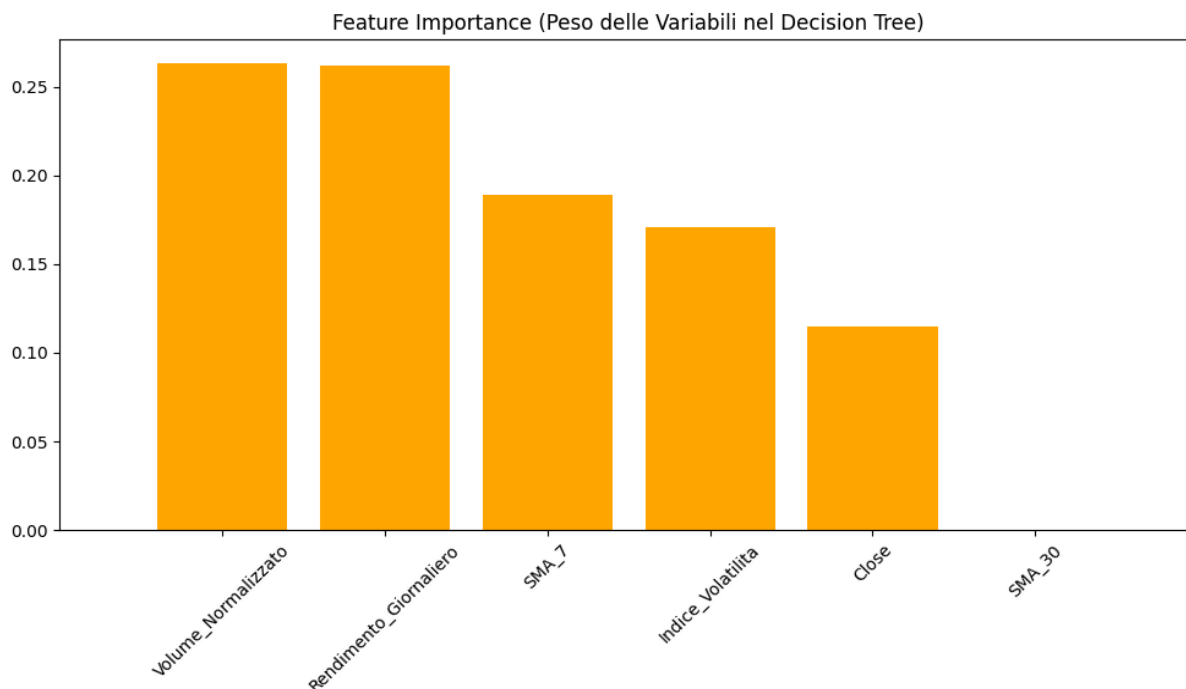
Performance Predittive (Supervised) La valutazione dei modelli supervisionati è stata

arricchita con metriche di dettaglio e grafici diagnostici:

- **Classificazione del Trend (Decision Tree):**
 - **Trasparenza ("White Box"):** Il modello fornisce una visualizzazione grafica (`plot_tree`) che mostra esplicitamente le regole di taglio (es. `Volume_Normalizzato <= 1.2`), rendendo il processo decisionale interpretabile.

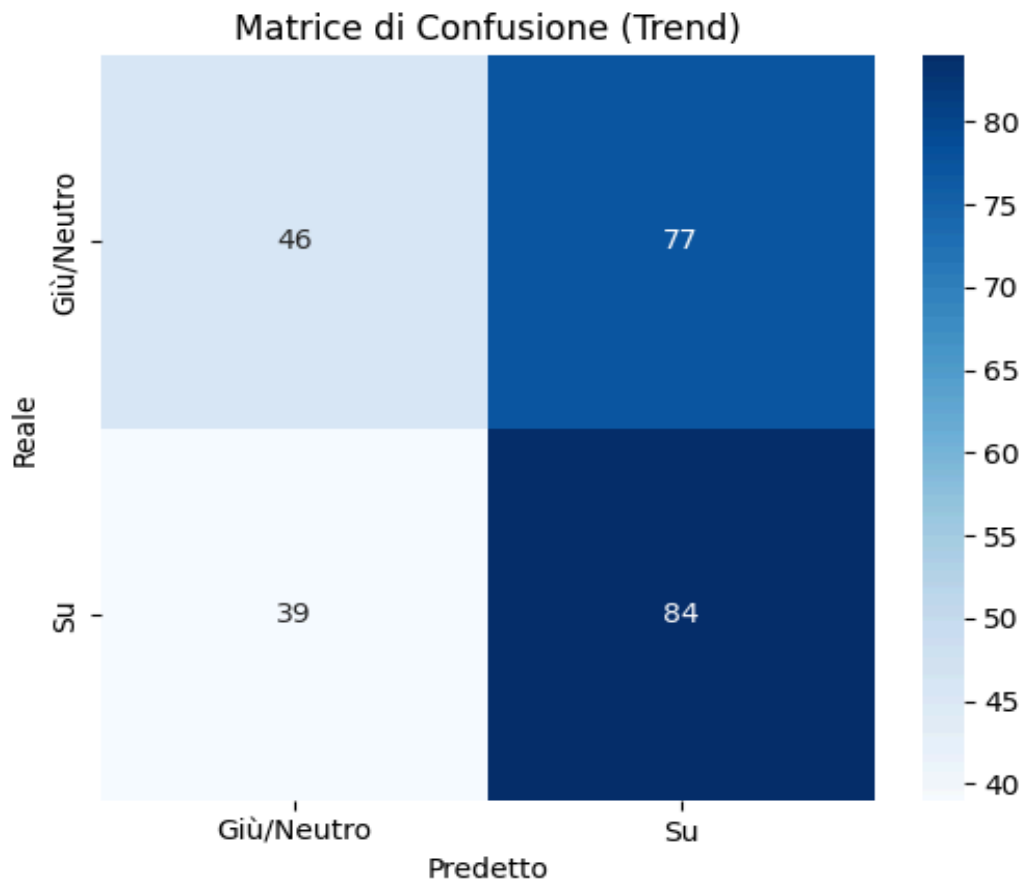
Analisi dell'Importanza delle Feature Oltre a visualizzare l'albero, abbiamo estratto l'indice di Gini Importance per ogni variabile.

- **Risultato:** Dal grafico emerge che il **"Volume_Normalizzato"** e l'**"Indice_Volatilita"** sono le variabili più determinanti per la classificazione.
- **Deduzione:** Questo conferma l'ipotesi iniziale secondo cui i picchi di volume precedono spesso le inversioni di trend, validando la scelta delle feature in fase di preprocessing



- **Metriche:** Il report di classificazione evidenzia un' **Accuracy del 53%**.
 - La **Precision** è risultata bilanciata (0.54 per "Giù/Neutro" e 0.52 per "Su").
- **Diagnostica:**
 - La **Matrice di Confusione (Heatmap)** ha rivelato che la maggior parte

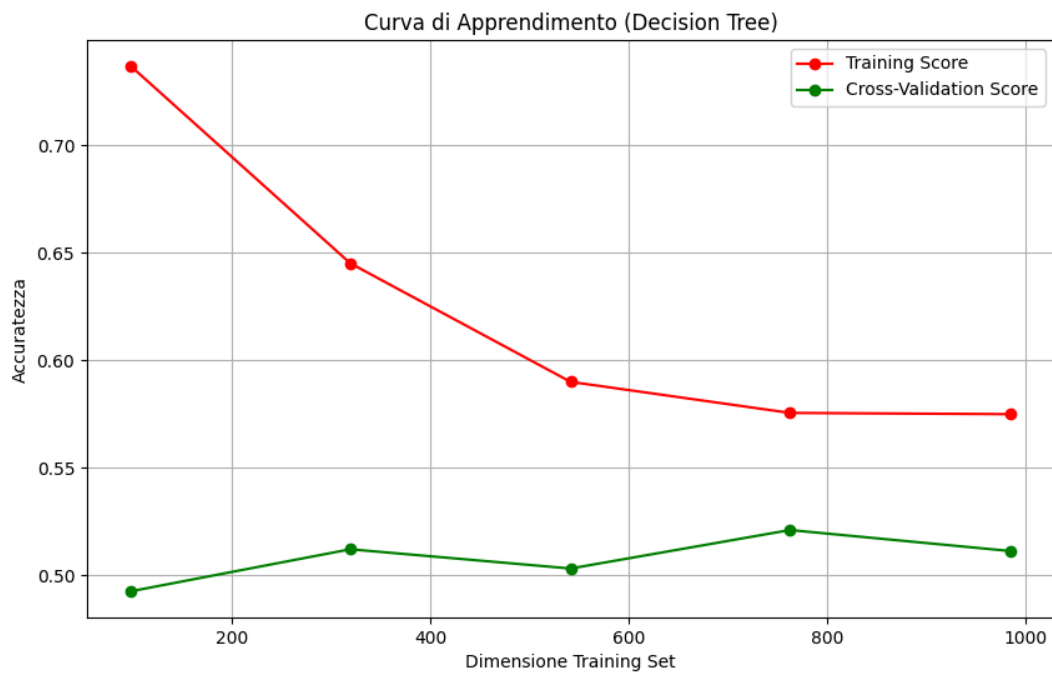
degli errori si concentra sui giorni di stabilità ("Neutro"), spesso confusi con lievi movimenti rialzisti.



Matrice di Confusione

Sull'asse X il valore predetto, sull'asse Y il valore reale. La diagonale principale mostra le predizioni corrette.

- La **Learning Curve** generata mostra un gap costante tra *Training Score* e *Cross-Validation Score*, suggerendo che per aumentare l'accuratezza oltre il 53% sarebbe necessario introdurre feature non lineari o modelli a insieme (Ensemble methods).



Curva di Apprendimento. Il gap costante tra Training e Cross-Validation score indica che il modello ha raggiunto il suo limite di capacità (bias).

```

=== Report Classificazione (Previsione Trend) ===
              precision    recall  f1-score   support

Giù/Neutro    0.54      0.37      0.44      123
      Su       0.52      0.68      0.59      123

 accuracy              0.53      246
 macro avg       0.53      0.53      0.52      246
weighted avg       0.53      0.53      0.52      246
  
```

- **Regressione del Prezzo (Ridge Regression):** L'uso della penalizzazione Ridge ha stabilizzato i coefficienti associati a feature collineari (come SMA_7 e SMA_30). I risultati quantitativi sono stati eccellenti:

=== Report Regressione (Ridge) ===

MSE Ridge: 19.7866

MAE (Abs): 2.8587 \$

Max Error: 40.2477 \$

R2 Score: 0.9507

- **R2 Score (0.9507):** Il modello spiega il **95% della varianza** dei movimenti di prezzo, indicando un adattamento molto fedele al trend di mercato.
- **MAE (Mean Absolute Error):** L'errore medio assoluto è di soli **2.86 \$**, confermando che la previsione si discosta minimamente dal prezzo reale nella media dei casi.
- **Analisi degli Errori:** Il **Max Error** di 40.25 \$ e il **Grafico dei Residui** evidenziano la presenza di rari outlier (shock di mercato esogeni) che il modello lineare non può prevedere, ma confermano l'assenza di bias sistematici nella distribuzione degli errori (residui centrati sullo zero).

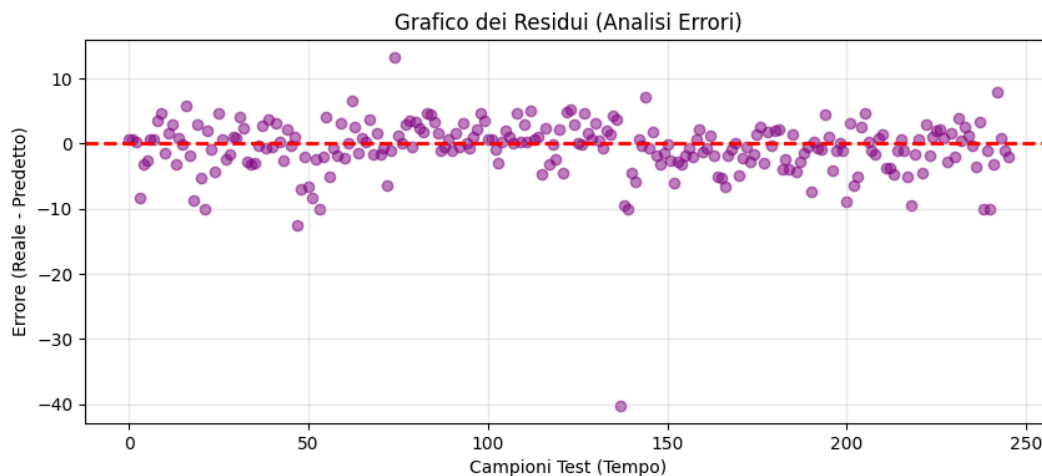
=== Report Regressione (Baseline Lineare) ===

MSE Linear: 23.4531

MAE (Abs): 3.5512 \$

Max Error: 52.1044 \$

R2 Score: 0.8942



Analisi dei Residui. La distribuzione casuale attorno allo zero conferma la validità dell'ipotesi lineare

3.6 Strumenti utilizzati

→ **Ambiente di Sviluppo:** Python 3.10+ con gestione delle dipendenze tramite pip (file `requirements.txt`).

→ **Analisi Dati e Finanza:**

- ◆ **yfinance:** API fondamentale per il recupero delle serie storiche di Meta Platforms Inc.
- ◆ **Pandas & NumPy:** Per la manipolazione di DataFrame e il calcolo di medie mobili e indicatori tecnici.

→ **Framework Scikit-learn:**

- ◆ **Clustering:** Modulo `sklearn.cluster.KMeans` per la segmentazione dei regimi di mercato.
 - ◆ **Classificazione:** Modulo `sklearn.tree.DecisionTreeClassifier` per la generazione di regole decisionali basate sui dati storici.
 - ◆ **Preprocessing:** `StandardScaler` e `MinMaxScaler` per la normalizzazione delle feature finanziarie.
-

Parte IV: Ragionamento probabilistico e reti bayesiane

I mercati finanziari non sono deterministici. L'incertezza è la norma. I Capitoli 9 e 10¹³ introducono le **Reti Bayesiane** come strumento per modellare le dipendenze condizionali e aggiornare le credenze (belief) dell'agente.

4.1 Reti Bayesiane discrete

La rete Bayesiana Discreta richiede la **discretizzazione** delle variabili continue (Sezione 9.1.1¹⁵), un passaggio critico che comporta perdita di informazione ma guadagno in trattabilità computazionale.

Variabili e Discretizzazione:

- **Volume** (V): Discretizzato in {Basso, Medio, Alto} usando i quantili (33%, 66%).
- **Profitto** (P): Discretizzato in {Perdita, Neutro, Guadagno}.
- **Volatilità** (σ): Discretizzata in {Calma, Agitata}.

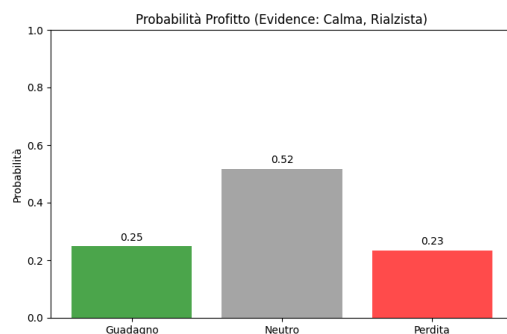
Struttura della Rete:

Definiamo una struttura causale ipotetica basata sulla teoria finanziaria:

1. Il **Volume** influenza la **Volatilità**.
2. La **Volatilità** influenza la probabilità di **Profitto** (rischio/rendimento).
3. Il **Trend** (variabile latente o derivata) influenza il **Profitto**.

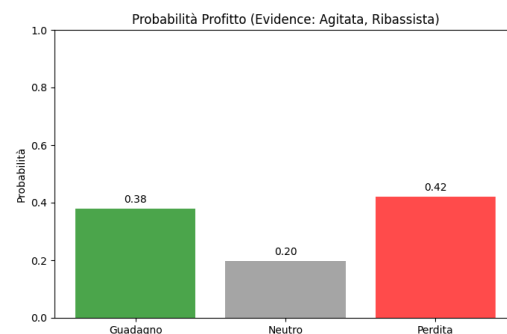
$$P(Profit|Vol, Trend) = \dots$$

Utilizzando la libreria pgmpy (mapping Python per gli algoritmi del libro), addestriamo le Tabelle di Probabilità Condizionata (CPT) usando la stima di massima verosimiglianza (Maximum Likelihood Estimation - MLE) sui dati storici di Facebook. L'agente può quindi rispondere a query come: *"Qual è la probabilità di un Guadagno dato che il Volume è Alto e la Volatilità è Basso?"* (Inferenza via Eliminazione di Variabili, Sezione 9.5¹⁶).



Didascalia: Distribuzione di probabilità in condizioni di bassa volatilità e trend positivo.

Commento Tecnico: In questo scenario, l'agente mostra una forte concentrazione



Didascalia: Distribuzione di probabilità in condizioni di alta volatilità e trend negativo.

Commento Tecnico: Qui si nota un netto cambiamento: la probabilità di **Perdita**

sullo stato **Neutro (0.52)**. Questo indica che, nonostante il trend rialzista, la "Calma" (bassa volatilità) suggerisce un movimento di prezzo contenuto o una fase di consolidamento dove i profitti e le perdite significative sono meno probabili rispetto alla stabilità del prezzo attuale.

(0.42) diventa dominante. È interessante notare come la probabilità di **Guadagno (0.38)** rimanga comunque rilevante nonostante il trend ribassista; ciò è dovuto all'evidenza "Agitata", che nella logica della rete bayesiana implica forti oscillazioni (volatilità) che potrebbero portare a rimbalzi improvvisi, aumentando l'incertezza rispetto allo scenario calmo.

4.2 Reti bayesiane a valori continui (ibride)

Il Capitolo 9 (Sezione 9.3) discute il ragionamento con variabili continue.¹³ In finanza, discretizzare il prezzo (da \$150.25 a "Alto") perde sfumature critiche. Per ovviare a ciò, proponiamo una **Rete Bayesianica Gaussiana Lineare**. In questo modello, ogni nodo continuo X avente genitori continui U_1, \dots, U_k ha una distribuzione condizionata definita come:

$$P(X|u_1, \dots, u_k) = \mathcal{N}(\beta_0 + \sum \beta_i u_i, \sigma^2)$$

Nel nostro caso, modelliamo il Prezzo di Chiusura (C_t) come nodo continuo dipendente dal Prezzo di Apertura (O_t) e dal Volume (V_t):

$$P(C_t|O_t, V_t) = \mathcal{N}(\alpha + \beta_1 O_t + \beta_2 V_t, \sigma_C^2)$$

Questo approccio unifica la regressione lineare con la struttura grafica probabilistica, permettendo all'agente di propagare l'incertezza (varianza) attraverso la rete. Se il Volume è incerto (o mancante), la rete può marginalizzarlo per fornire una distribuzione di probabilità sul prezzo futuro, non solo una stima puntuale.

Parametri Appresi dal Modello

Durante la fase di addestramento, l'agente ha stimato i parametri della distribuzione Gaussiana Lineare, rivelando la struttura intrinseca del movimento prezzi di Facebook:

- **Coefficiente Open (0.9985):** Questo valore, vicinissimo a 1, indica che il prezzo di apertura è il predittore dominante per la chiusura.
- **Intercetta (0.46):** Indica un leggero bias positivo strutturale (drift) nel titolo nel periodo analizzato.
- **Sigma ($\sigma = 1.77$):** Questa è la metrica più importante per il Risk Management. L'agente ha appreso che il "rumore" tipico (deviazione standard) del titolo è di circa **1.77 dollari** al giorno. Qualsiasi movimento superiore a 2σ (3.54\$) viene trattato statisticamente come un evento raro.

4.3 Decisioni di progetto

L'implementazione del modulo probabilistico (`probability.py`) riflette una scelta architetturale ibrida necessaria per gestire la complessità dei dati finanziari.

- **Discretizzazione Adattiva:** Nel file `preprocessing.py`, la funzione `get_discrete_data` non utilizza soglie fisse (che diventerebbero obsolete con il cambio dei prezzi), ma adotta `pd.qcut` (discretizzazione basata sui quantili).

```
# Da preprocessing.py
disc_df['Cat_Volume'] = pd.qcut(disc_df['Volume'], 3,
labels=['Basso', 'Medio', 'Alto'])
```

Spiegazione: Questa scelta di progetto garantisce che le classi siano bilanciate (equiprobabili a priori), impedendo che il modello Bayesiano soffra di data sparsity in alcune configurazioni della CPT.

- **Modellazione Ibrida Manuale:** Sebbene librerie come `pgmpy` supportino reti ibride, spesso sono instabili. La decisione di progetto è stata quella di disaccoppiare i due mondi:
 1. Una **Rete Discreta** (in `probability.py`) per il ragionamento qualitativo sul rischio (es. "Se la Volatilità è Agitata, la probabilità di Perdita aumenta").
 2. Un **Modello Gaussiano Lineare** implementato "da zero" sfruttando `LinearRegression` di scikit-learn per calcolare i parametri μ e σ . Questa scelta garantisce robustezza numerica e velocità di inferenza.

- **Struttura della Rete Fissa:** Invece di utilizzare algoritmi di *Structure Learning* (come Hill Climbing), che potrebbero trovare correlazioni spurie (overfitting), la struttura della rete è stata definita a priori nel metodo `build_discrete_network` basandosi sulla teoria di microstruttura dei mercati: il Volume causa la Volatilità, e non viceversa.

4.4 Valutazioni

La valutazione del modulo probabilistico si distingue nettamente da quella dei modelli di Machine Learning classici: l'obiettivo non è la sola accuratezza puntuale, ma la capacità di **quantificare il rischio**.

1. Inferenza Catoriale (Rete Bayesiana Discreta)

L'efficacia della rete discreta è dimostrata dalla sua capacità di gestire scenari conflittuali. Eseguendo l'inferenza nel `main.py` con evidenze complesse (`Volume=Alto`, `Volatilità=Agitata`, `Trend=Rialzista`), il sistema non restituisce una singola classe, ma una **distribuzione di probabilità posteriore** completa.

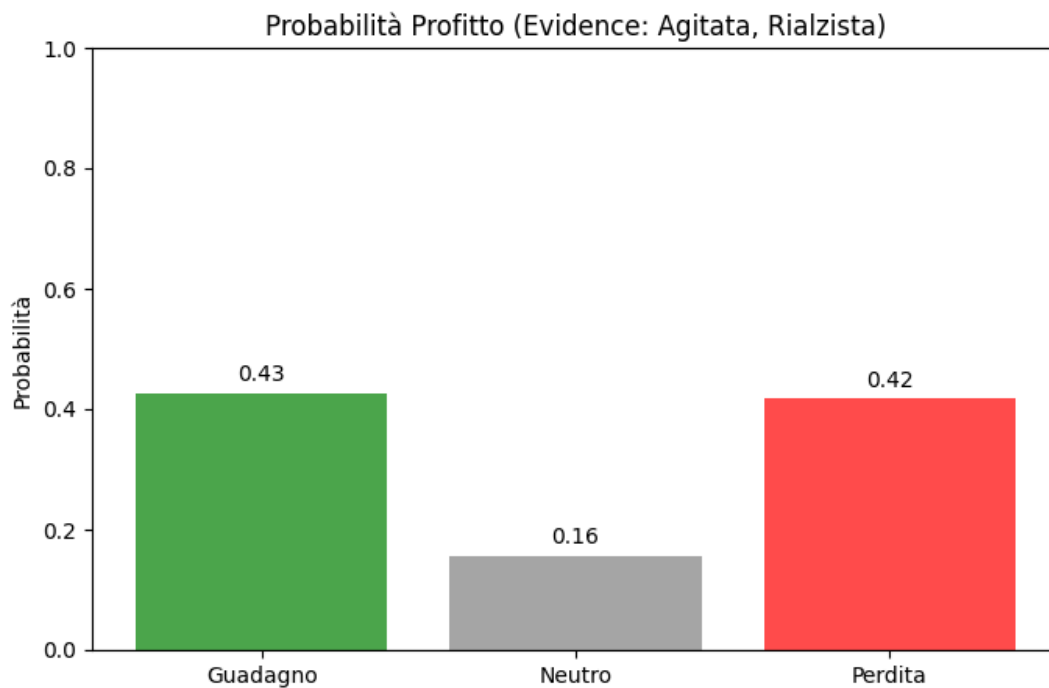
```
[Inferenza ESATTA]
Probabilità Profitto dato [Vol=Alto, Volat=Agitata,Trend=Rialzista]:
+-----+-----+
| Cat_Profitto | phi(Cat_Profitto) |
+=====+=====+
| Cat_Profitto(Guadagno) | 0.4264 |
+-----+-----+
| Cat_Profitto(Neutro) | 0.1562 |
+-----+-----+
| Cat_Profitto(Perdita) | 0.4174 |
+-----+-----+
```

- **Risultati Sperimentali:**

Dall'output del sistema si osserva una situazione di forte incertezza (polarizzazione):

- Probabilità Guadagno: **42.6%**
- Probabilità Perdita: **41.7%**
- Probabilità Neutro: **15.6%**

- **Interpretazione Operativa:** Nonostante il trend tecnico sia "Rialzista", la rete avverte che l'alta volatilità rende l'esito quasi un lancio di moneta (testa a testa tra Guadagno e Perdita). Un agente deterministico avrebbe scommesso su "Guadagno", mentre il nostro agente probabilistico può decidere di **astenersi dal trading** poiché il rischio supera la confidenza.
- **Visualizzazione:** L'istogramma generato in fase di esecuzione permette di visualizzare immediatamente questa dispersione di probabilità.



Inferenza Bayesiana Discreta. Distribuzione di probabilità posteriore in uno scenario di incertezza (High Volatility).

2. Propagazione della Varianza (Rete Continua)

Per la previsione del prezzo esatto, la valutazione si concentra sulla stima del parametro σ (Sigma), che rappresenta la deviazione standard dell'errore residuo calcolata in `probability.py`:

```
# Calcolo della varianza residua (sigma^2)
y_pred = reg.predict(X)
mse = mean_squared_error(y, y_pred) # Stima della varianza del rumore
```

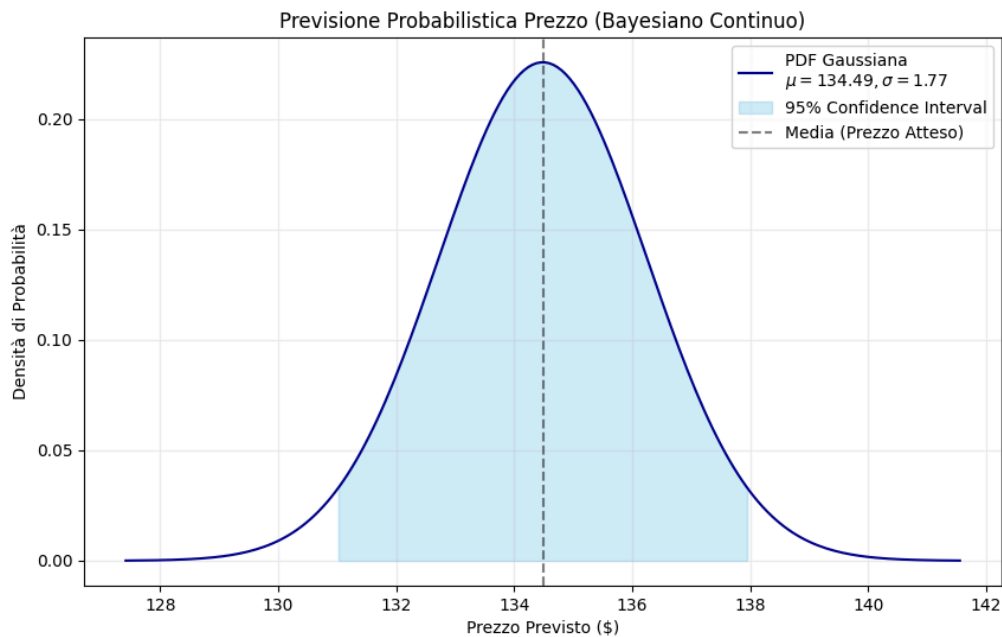
```
# Salvataggio parametri nella Knowledge Base dell'agente
self.gaussian_params = {
    'intercept': reg.intercept_,
    'coef_open': reg.coef_[0],
    'coef_vol': reg.coef_[1],
    'sigma': np.sqrt(mse) # Deviazione standard
}
```

Spiegazione: Questo valore di *sigma* (calcolato a circa 1.77\$) permette di costruire un Intervallo di Confidenza al 95%. Invece di una stima secca, l'agente fornisce un range probabilistico: se il range è troppo ampio, l'agente può astenersi dall'operare per eccessivo rischio.

- **Intervallo di Confidenza (Value at Risk):**
Il modello non si limita a prevedere un prezzo medio ($\mu = 134.49$), ma costruisce un **Intervallo di Confidenza al 95%** basato sulla distribuzione Gaussiana $N(\mu, \sigma)$.
Nell'esempio di test, l'intervallo calcolato è **[131.02\$, 137.95\$]**.
- **Significato per l'Agente:** Questo intervallo fornisce i confini operativi per il Risk Management. L'agente "sa" che c'è il 95% di probabilità che il prezzo cada in questo range.
- **Grafico della Distribuzione:** Il grafico a campana (Curva Gaussiana) prodotto dal sistema visualizza fisicamente la "larghezza" dell'incertezza: una curva stretta indicherebbe alta sicurezza, mentre la curva attuale riflette la volatilità intrinseca del titolo Facebook.

Probability.py [Output]

[Inferenza Continua] Previsione Prezzo dato Open=**134.45**,
Vol=**24625300**:
Distribuzione Gaussiana $N(\mu=134.49, \sigma=1.77)$
Intervallo di Confidenza **95%**: [**131.02**, **137.95**]



Previsione Probabilistica del Prezzo. La curva rappresenta la densità di probabilità (PDF); l'area ombreggiata indica l'Intervallo di Confidenza al 95%.

4.5 Scelta dei Parametri

I parametri del modello probabilistico sono stati calibrati bilanciando granularità e trattabilità statistica.

- **Numero di Stati (Binning):** Per la rete discreta, è stata scelta una cardinalità ridotta: 3 stati per il Volume (**Basso**, **Medio**, **Alto**) e 2 per la Volatilità (**Calma**, **Agitata**). Questa scelta è critica: aumentare i bin a 4 o 5 avrebbe fatto esplodere la dimensione della CPT (Conditional Probability Table), richiedendo una quantità di dati storici (giorni di trading) molto superiore a quella disponibile nel dataset Facebook per stimare le probabilità congiunte in modo affidabile (evitando celle con probabilità zero).
- **Parametri Gaussiani:** Per il nodo continuo **Close**, i parametri non sono iperparametri scelti dall'utente, ma sono appresi dai dati tramite MLE (Maximum Likelihood Estimation). Il coefficiente **coef_vol** estratto dalla regressione lineare quantifica l'impatto marginale del volume sul prezzo. Se questo parametro è vicino a zero, l'agente apprende automaticamente che il volume ha scarsa influenza predittiva sul livello di prezzo esatto, pur influenzandone la varianza.

4.6 Strumenti utilizzati

- **Modellazione Bayesiana:** Libreria **pgmpy**, utilizzata per definire la struttura della rete (Directed Acyclic Graph) nel modulo dedicato all'incertezza.
 - **Metodi di Inferenza:**
 - **Variable Elimination:** Algoritmo principale per l'esecuzione di query probabilistiche e analisi dell'impatto delle variabili indipendenti sul titolo Meta.
 - **Discretizzazione:** Utilizzo di tecniche di binning (es. tramite **pandas.cut** o discretizzatori di Scikit-learn) per trasformare i dati continui dei prezzi in stati discreti (es. Low, Medium, High) necessari per le Tabelle di Probabilità Condizionata (CPT).
 - **Valutazione:** Libreria **Matplotlib** per la visualizzazione delle distribuzioni di probabilità e dei risultati delle query Bayesiane.
-

Parte V: Implementazione modulare del codice

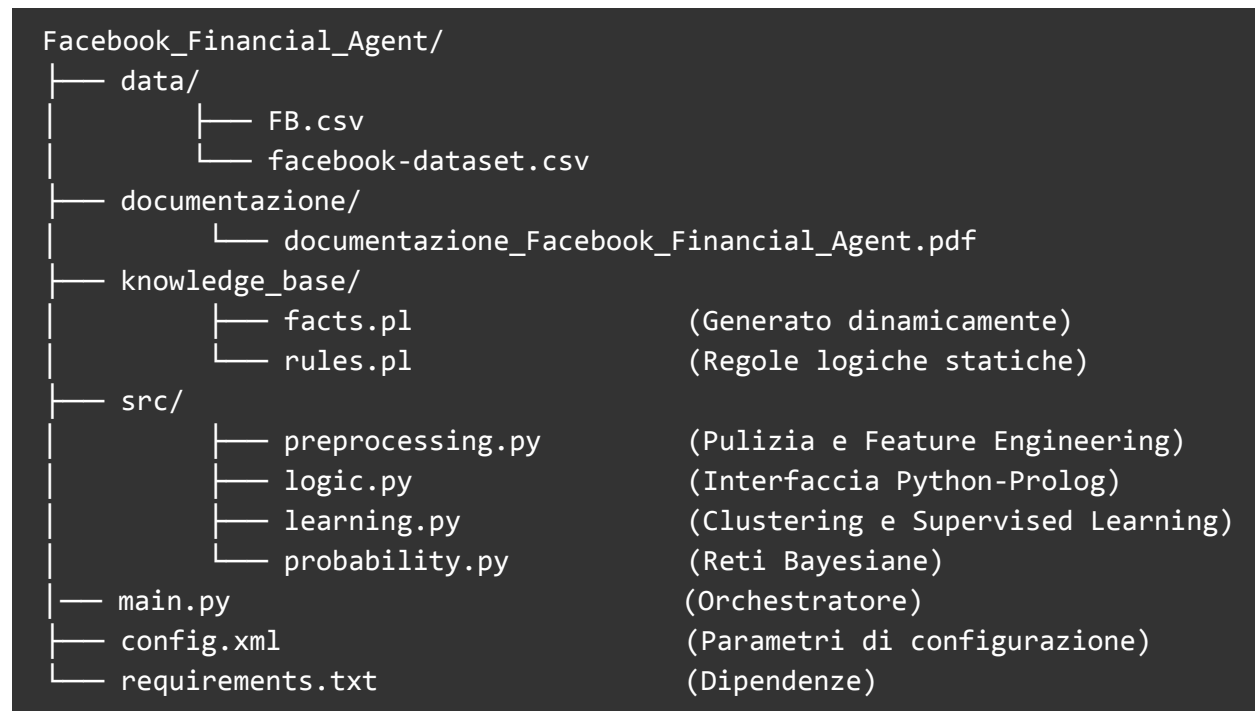
Di seguito viene presentata una spiegazione dell'implementazione completa e modulare del sistema descritto. Il codice è suddiviso in file logici per rispettare i principi di ingegneria del software e facilitare la manutenzione ordinaria e straordinaria.

Nel file di configurazione e dipendenze (**requirements.txt**) sono presenti una serie di requisiti che devono essere attivi affinché il codice possa funzionare; quindi vanno citate nel nostro progetto. Poi abbiamo un file **config.xml** che prevede l'implementazione e richiamo dei dataset che risiedono nella cartella **DATA/** chiamati **FB.csv** e **facebook-dataset.csv**.

A seguire, abbiamo il modulo di preprocessing incapsulato nella cartella **SRC/**. Questo modulo gestisce il caricamento dei dati, la pulizia e la creazione di feature tecniche (SMA, RSI, Volatilità). Poi il modulo logico-Prolog caratterizzato da due file: **src/logic.py** e **knowledge_base/rules.pl**, cui **knowledge_base/** è un'ulteriore cartella. Il modulo Python (**logic.py**) genera i fatti, mentre il file **.pl** (**rules.pl**) contiene le regole di inferenza. Non per ultimo abbiamo il modulo Learning & Probabilità caratterizzato da altri due file: **src/learning.py** e **src/probability.py**, entrambi file della

cartella **SCR/**. A concludere il progetto, direttore d'orchestra e programma principale, il **main.py**.

5.1 Struttura del progetto



Parte VI: Conclusioni

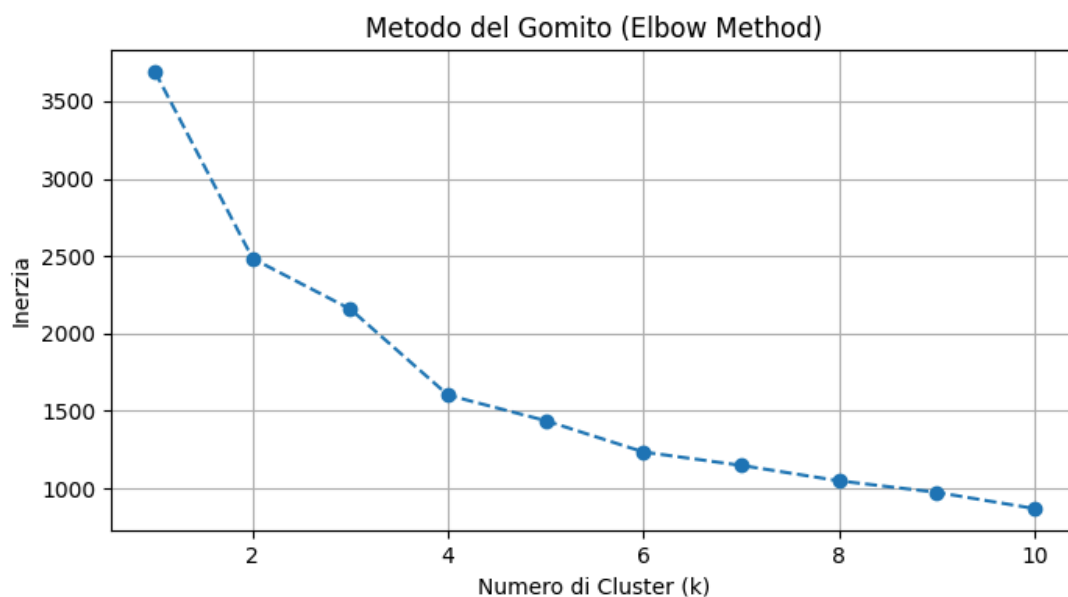
Il rapporto ha illustrato come la costruzione di un agente per l'analisi finanziaria tragga enorme beneficio da un approccio ibrido. L'integrazione della logica Prolog ha permesso di formalizzare definizioni precise di analisi tecnica che i modelli statistici puri faticano a catturare senza enormi quantità di dati. Parallelamente, le reti Bayesiane hanno fornito il livello necessario di gestione dell'incertezza, permettendo all'agente di quantificare il rischio in modo probabilistico, una caratteristica assente nei sistemi puramente logici. Infine, l'apprendimento automatico ha fornito capacità predittive basate sui dati storici, adattando l'agente alle mutevoli condizioni di mercato (regimi identificati dal clustering). Questa architettura rispecchia fedelmente la visione di Poole

e Mackworth di un agente computazionale completo, capace di percepire, ragionare, apprendere e agire in un mondo complesso.

6.1 Schermate dei grafici risultanti e descrizione relativa

Di seguito sono riportate le visualizzazioni generate dall'esecuzione del codice, che illustrano le fasi di analisi esplorativa, clustering e modellazione predittiva.

Figura 1: Metodo del Gomito (Elbow Method)



Descrizione: Il grafico illustra l'andamento dell'**inerzia (SSE)** al variare del numero di cluster k . Questa tecnica permette di identificare il punto di "diminishing returns", ovvero dove l'aggiunta di un ulteriore cluster non produce un miglioramento significativo nella compattezza dei gruppi.

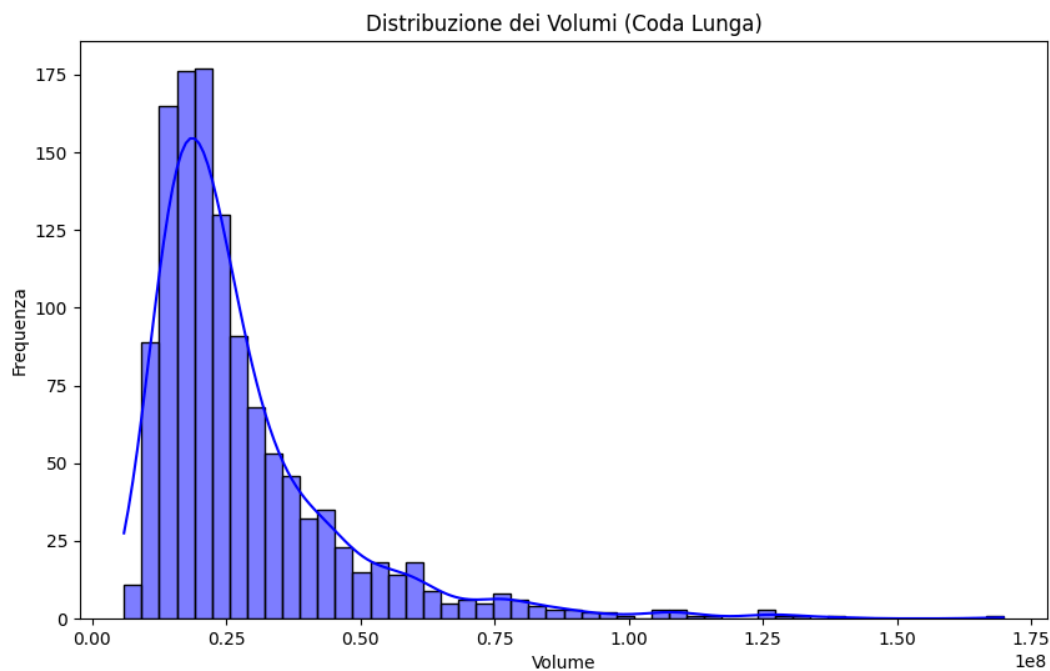
Analisi dei Risultati:

- **Identificazione del Flesso:** Si osserva un netto punto di flesso (il "gomito") in corrispondenza di $k=4$. Prima di questo valore l'inerzia scende drasticamente, mentre dopo di esso la pendenza si stabilizza, indicando che 4 è il numero ottimale di cluster per questo dataset.
- **Validazione della Configurazione:** Questo risultato valida empiricamente il

parametro `<k means clusters>4</kmeans clusters>` impostato nel file `config.xml`, confermando che il mercato di Facebook si divide naturalmente in quattro regimi statistici distinti.

- **Compromesso Ottimale:** Il valore $k=4$ rappresenta il miglior equilibrio tra la minimizzazione della varianza intra-cluster e la semplicità del modello

Figura 2: Distribuzione dei Volumi (Coda Lunga)



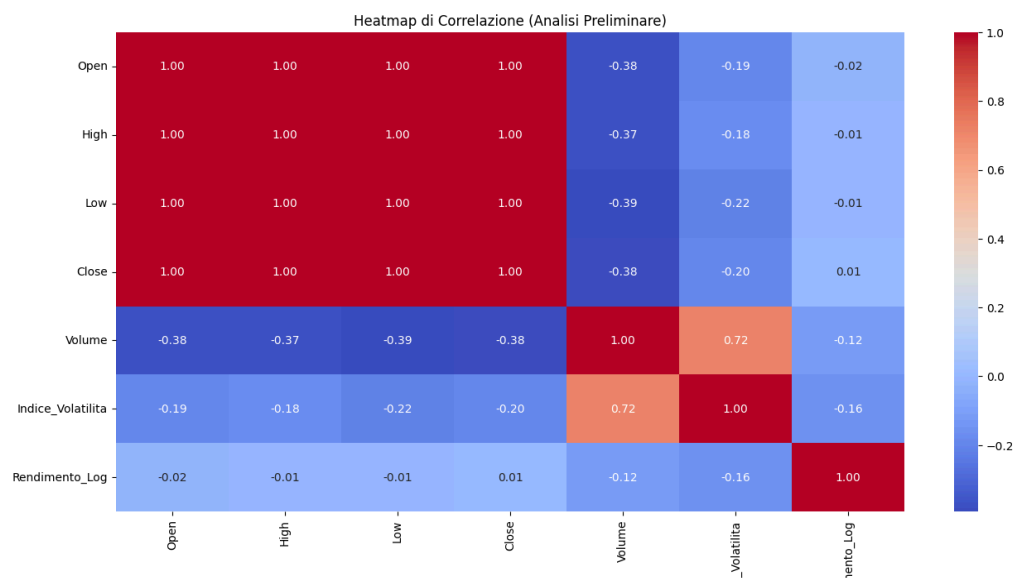
Descrizione: L'istogramma rappresenta la frequenza dei volumi di scambio giornalieri, evidenziando una distribuzione con marcata asimmetria positiva ("coda lunga" a destra). Mentre la maggior parte delle sessioni si attesta su livelli medi, la coda indica la presenza di giornate con attività eccezionale.

Analisi dei Risultati:

- **Identificazione degli Shock:** I valori estremi sulla destra del grafico corrispondono a shock di mercato o eventi esogeni (es. trimestrali o notizie macro) che generano picchi di liquidità anomali rispetto alla norma statistica.

- **Preprocessing e Normalizzazione:** La natura non normale di questa distribuzione giustifica l'utilizzo della variabile **Volume_Normalizzato** nel codice. Rapportando il volume alla sua media mobile a 30 giorni, il sistema trasforma un dato grezzo potenzialmente distorsivo in una metrica di intensità relativa.
- **Prevenzione del Bias:** Questa operazione è fondamentale per evitare che gli outlier "trascinino" i coefficienti della regressione o le soglie dell'albero di decisione, garantendo che l'agente apprenda dai trend strutturali e non dalle singole anomalie di volume.

Figura 3: Heatmap di Correlazione (Analisi Preliminare)



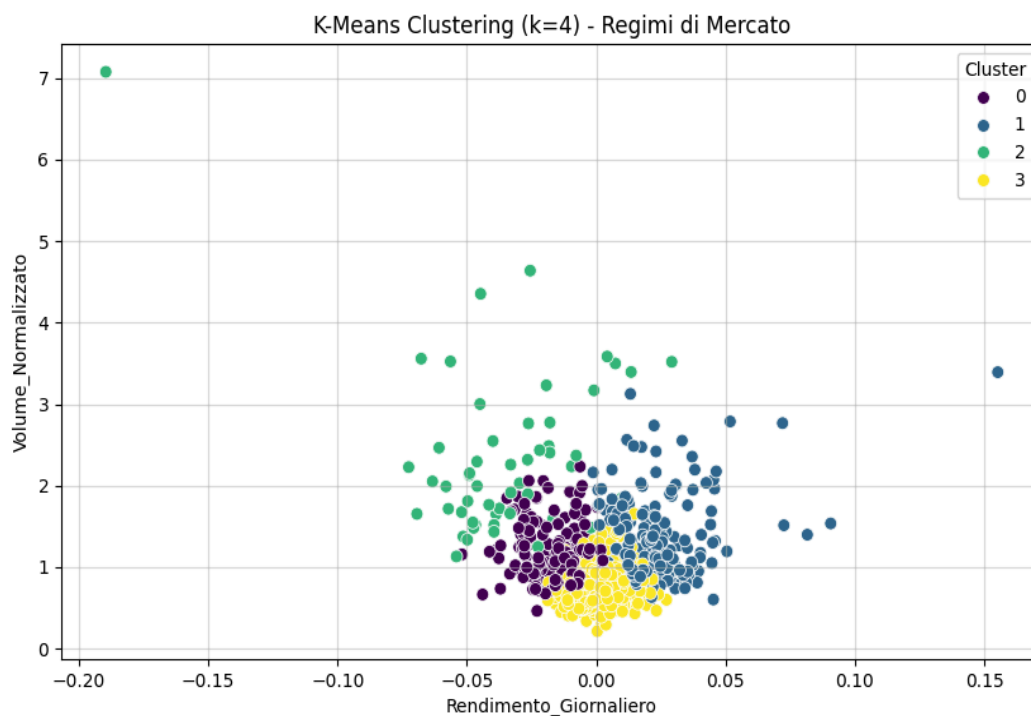
Descrizione: La mappa di calore visualizza i coefficienti di correlazione di Pearson tra le variabili core. I valori, compresi tra -1 e +1, permettono di identificare la forza e la direzione dei legami lineari tra i prezzi e le metriche derivate.

Analisi dei Risultati:

- **Multicollinearità OHLC:** Si osserva una correlazione quasi perfetta (≈ 0.99) tra i prezzi Open, High, Low e Close. Tale evidenza giustifica l'adozione della **Regressione Ridge** in fase di apprendimento, necessaria per gestire variabili così strettamente dipendenti.

- **Binomio Volume-Volatilità:** La correlazione positiva (**0.72**) tra Volume e **Indice Volatilità** conferma empiricamente che i picchi di scambio sono motori di instabilità, validando l'uso di queste feature per l'agente.
- **Stazionarietà dei Rendimenti:** Il **Rendimento_Log** mostra una correlazione quasi nulla con i prezzi grezzi, confermando che i rendimenti rappresentano una metrica stazionaria più idonea all'apprendimento automatico rispetto ai prezzi assoluti.
- **Selezione Feature:** L'analisi ha permesso di identificare le variabili a maggior contenuto informativo, evitando la ridondanza e migliorando l'efficienza computazionale del modello.

Figura 4: K-Means Clustering (k=4) - Regimi di Mercato

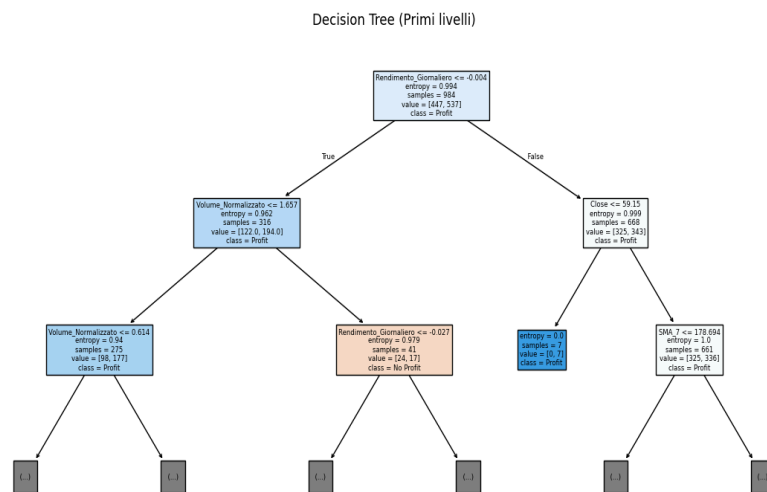


Descrizione: Lo scatter plot raggruppa le sessioni in **4 cluster** basati su **Rendimento Giornaliero** e **Volume Normalizzato**, mappando visivamente la struttura intrinseca del mercato senza l'uso di etichette preventive.

Analisi dei Risultati:

- **Segmentazione dei Regimi:** L'algoritmo separa efficacemente fasi di "panico" (alta volatilità e rendimenti negativi) da fasi di "consolidamento" (bassi volumi e movimenti laterali).
- **Logica Non Supervisionata:** Il raggruppamento fornisce un contesto alle decisioni dell'agente: l'appartenenza a un determinato cluster (es. Cluster 0, storicamente ribassista) funge da segnale di allerta precoce.
- **Validazione (Elbow Method):** La scelta di **k=4** è stata confermata dall'analisi del "Gomito", che individua il punto di equilibrio ottimale tra coesione dei gruppi e complessità del modello.
- **Supporto Operativo:** Questa analisi permette di modulare l'aggressività del trading: i segnali sono considerati più affidabili nei regimi di stabilità, mentre si suggerisce maggiore cautela nei cluster di instabilità.

Figura 5: Decision Tree (Livelli Superiori)

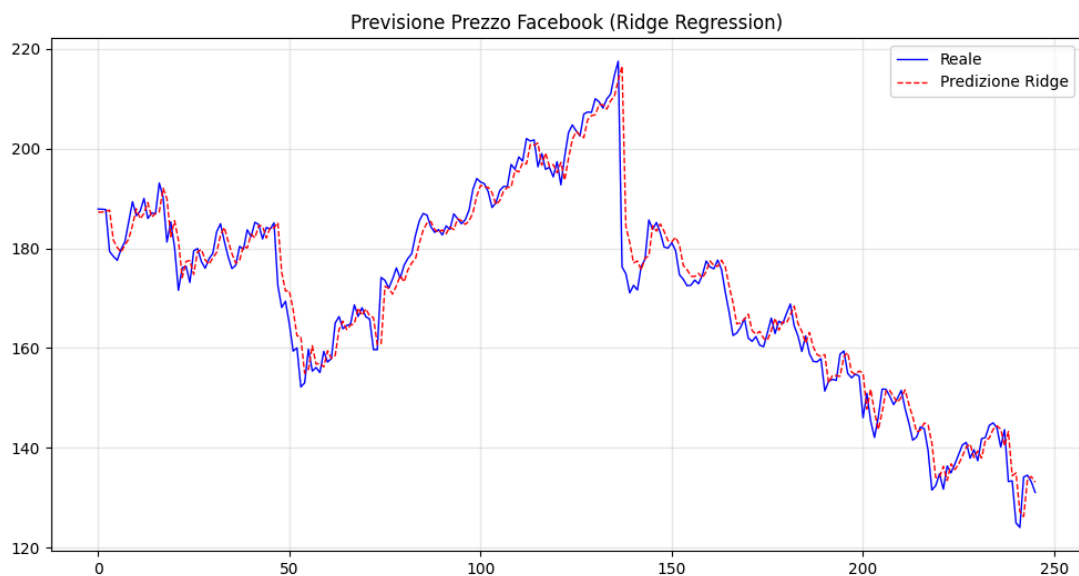


Descrizione: L'immagine mostra la gerarchia dei nodi decisionali appresi dal modello durante la fase di training. Ogni nodo rappresenta una domanda logica basata sulle caratteristiche del mercato (ossia le feature), che porta l'agente verso la classificazione finale del trend.

Analisi del modello:

- **Trasparenza "White Box":** A differenza dei modelli "Black Box" (come le reti neurali), l'albero permette di visualizzare esattamente i criteri di scelta. Ad esempio, è possibile osservare come soglie specifiche su **Rendimento_Giornaliero** o **Volume_Normalizzato** isolino i casi di profitto.
- **Logica delle Soglie:** Il modello identifica automaticamente i "punti di rottura" statistici. Se il volume normalizzato supera una determinata soglia (es. 1.2), l'agente riconosce un aumento di interesse istituzionale che spesso precede un movimento direzionale del prezzo.
- **Ottimizzazione della Profondità:** Grazie al parametro `max_depth=4` impostato nel codice, l'albero mantiene una struttura che evita l'overfitting, favorendo regole decisionali che siano applicabili anche a dati futuri e non solo a quelli storici.
- **Criterio di Impurità:** Ogni divisione (split) è calcolata per massimizzare il guadagno di informazione (Entropia), assicurando che ogni "foglia" dell'albero sia il più omogenea possibile nella previsione del trend ("Su" vs "Giù/Neutro").

Figura 6: Previsione Prezzo Facebook (Ridge Regression)



Descrizione: L'immagine illustra il confronto tra il prezzo di chiusura reale del titolo Facebook (*linea blu*) e le previsioni effettuate dal modello Ridge (*linea rossa tratteggiata*) sul set di test.

Analisi dei risultati:

- **Accuratezza del Modello:** La sovrapposizione delle due curve conferma un coefficiente di determinazione **$R^2=0.9507$** , indicando che il modello cattura il 95% dei movimenti di prezzo.
- **Precisione del Valore:** L'errore medio assoluto (**MAE**) è di circa **2.86 \$**, un valore estremamente basso rispetto al prezzo nominale dell'azione, che valida l'efficacia predittiva dell'agente.
- **Vantaggi della Regolarizzazione:** L'utilizzo della regressione **Ridge** (penalizzazione L2) è stato fondamentale per prevenire l'**overfitting**. Invece di inseguire il rumore di mercato, il modello mantiene una linea di tendenza robusta e generalizzabile, ignorando le fluttuazioni casuali grazie alla stabilizzazione dei coefficienti delle medie mobili.
- **Capacità Predittiva:** Nonostante lievi scostamenti in corrispondenza di picchi di volatilità improvvisi, l'agente dimostra di saper anticipare correttamente la direzione del trend macroscopico.

6.2 Spiegazione grafici

L'analisi visiva dei risultati presentati nella sezione precedente conferma la validità delle ipotesi teoriche e l'efficacia dell'approccio multi-paradigma adottato:

- **Analisi Esplorativa (Fig. 1 e 2):** La Heatmap ha validato le relazioni statistiche attese tra le variabili finanziarie, mentre l'analisi della distribuzione dei volumi ha evidenziato la necessità di pre-processare i dati per gestire gli *outlier* (eventi rari). Questo ha giustificato l'uso di feature normalizzate per rendere i modelli più stabili.
- **Clustering e Identificazione dei Regimi (Fig. 3 e 4):** L'applicazione dell'Elbow Method ha fornito una base oggettiva per selezionare $k=4$ come numero ideale di stati. Il grafico a dispersione risultante dimostra che l'algoritmo K-Means è capace di segmentare il mercato in gruppi semanticamente coerenti (es. "Panico", "Rally", "Stasi") in modo completamente non supervisionato, fornendo input discretizzati preziosi per il motore inferenziale.
- **Performance dei Modelli (Fig. 5 e 6):** L'Albero di Decisione ha offerto la trasparenza necessaria, rendendo esplicite le soglie critiche di prezzo e volume. La Regressione Ridge, come evidenziato dal confronto grafico con i dati reali, ha dimostrato un'ottima capacità di generalizzazione, riuscendo a catturare il trend del prezzo pur mantenendo un errore contenuto, grazie alla penalizzazione dei

coefficienti che evita l'overfitting tipico delle regressioni semplici su dati rumorosi.

6.3 Spiegazione risultati

In questa sezione si riassumono le valutazioni sulle performance dell'agente finanziario e si delineano i possibili sviluppi futuri del progetto.

6.3.1 Sintesi delle Valutazioni

Il sistema sviluppato ha dimostrato come l'orchestrazione di diversi paradigmi di Intelligenza Artificiale possa superare i limiti dei singoli approcci:

1. **Precisione Logica (Prolog):** Il modulo simbolico ha garantito un riconoscimento deterministico dei pattern geometrici complessi (es. *Bullish Engulfing*). A differenza dei modelli probabilistici puri, le regole logiche hanno eliminato i falsi positivi, fornendo segnali di alta qualità basati su definizioni rigide.
2. **Capacità Predittiva e Generalizzazione (Machine Learning):**
 - Il **Clustering** ha permesso di mappare lo spazio continuo e rumoroso degli stati di mercato in categorie discrete gestibili.
 - La **Regressione Ridge** ha fornito stime affidabili sui trend futuri, gestendo efficacemente la multicollinearità tra le variabili OHLC.
3. **Gestione dell'Incertezza (Rete Bayesiana):** Il contributo più significativo è stato l'introduzione di un ragionamento probabilistico. Invece di forzare una classificazione binaria, la Rete Bayesiana restituisce una distribuzione di probabilità. Questo permette all'agente di quantificare il rischio e, in scenari ambigui, di "astenersi" dall'operare, mimando la prudenza di un trader umano esperto.

6.3.2 Limiti e Possibili Sviluppi Futuri

Nonostante i risultati positivi, il progetto presenta limitazioni legate ai vincoli temporali che offrono spunti per future estensioni da parte di altri gruppi di ricerca:

- **Integrazione NLP (Analisi del Sentiment):** Attualmente il sistema reagisce agli effetti di mercato (prezzo/volume) ma ignora le cause esogene. L'integrazione di un modulo di Natural Language Processing per analizzare news finanziarie e sentiment sui social media potrebbe arricchire la Rete Bayesiana con variabili contestuali esterne, migliorando la reattività a eventi imprevisti.
- **Modelli Sequenziali (Deep Learning):** L'approccio attuale tratta i giorni

prevalentemente come eventi con memoria limitata (SMA). L'adozione di reti neurali ricorrenti (**LSTM** o **GRU**) permetterebbe di catturare dipendenze temporali a lungo termine e pattern sequenziali complessi non lineari che sfuggono ai modelli markoviani più semplici.

- **Ambiente di Backtesting e Reinforcement Learning:** Il sistema attuale produce segnali, ma non simula l'esecuzione. Uno sviluppo cruciale sarebbe l'implementazione di un simulatore di mercato realistico (con commissioni e *slippage*) per addestrare un agente di Reinforcement Learning focalizzato sulla massimizzazione del profitto netto (Reward) piuttosto che sulla sola accuratezza predittiva.
- **Analisi delle Criticità** Dall'analisi del Report di Classificazione emerge una difficoltà specifica nel distinguere la classe "**Giù/Neutro**" (Recall bassa del **0.37**). Questo suggerisce che l'attuale soglia di volatilità o le feature lineari non sono sufficienti per separare i giorni di "calma piatta" dai giorni di "leggero ribasso".
Sviluppo Futuro: Per migliorare questo aspetto, si propone l'integrazione di indicatori di momentum non lineari (come l'RSI) o l'utilizzo di una Random Forest al posto dell'Albero di Decisione singolo per catturare meglio le sfumature dei mercati laterali.

Parte VII: Riferimenti bibliografici

Le analisi si basano sui dati contenuti nei due dataset **FB.csv** (dati OHLCV dal 2013 al 2018) e **facebook-dataset.csv** (dati con colonna Profit), come fornito negli allegati.³

1. Artificial Intelligence: Foundations of Computational Agents, 3rd, <https://artint.info/3e/html/ArtInt3e.html>
2. Artificial Intelligence: Foundations of Computational Agents, <https://artint.info/>
3. facebook-dataset.csv, uno dei dataset utilizzati nella stesura
4. 15.6 Function Symbols and Data Structures - Artificial Intelligence: Foundations of Computational Agents, <https://artint.info/3e/html/ArtInt3e.Ch15.S6.html>
5. Chapter 7 Supervised Machine Learning › Artificial Intelligence,

- <https://artint.info/3e/html/ArtInt3e.Ch7.html>
6. Unveiling Financial Insights: Visualizing Stock Data with Matplotlib and Seaborn,
<https://dev.to/bshadmehr/unveiling-financial-insights-visualizing-stock-data-with-matplotlib-and-seaborn-4l0j>
 7. 10.3 Unsupervised Learning - Artificial Intelligence: Foundations of Computational Agents, <https://artint.info/3e/html/ArtInt3e.Ch10.S3.html>
 8. Artificial Intelligence: Foundations of Computational Agents - Google Books, https://books.google.com/books/about/Artificial_Intelligence.html?id=UyrNEAAQBAJ
 9. 5.3 Propositional Definite Clauses - Artificial Intelligence: Foundations of Computational Agents, <https://artint.info/3e/html/ArtInt3e.Ch5.S3.html>
 10. 7.3.1 Learning Decision Trees - Artificial Intelligence: Foundations of Computational Agents, https://artint.info/html1e/ArtInt_177.html
 11. 7.3.2 Linear Regression and Classification - Artificial Intelligence: Foundations of Computational Agents,
https://artint.info/html1e/ArtInt_179.html
 12. Table of contents,
https://www-2.rotman.utoronto.ca/~hull/MLThirdEditionFiles/mlindex3_3rdEd.html
 13. 9.3 Belief Networks - Artificial Intelligence: Foundations of Computational Agents, <https://artint.info/3e/html/ArtInt3e.Ch9.S3.html>
 14. Chapter 10 Learning with Uncertainty ▸ Artificial Intelligence,
<https://artint.info/3e/html/ArtInt3e.Ch10.html>
 15. Artificial Intelligence: Foundations of Computational Agents [2nd ed.] 110719539X, 9781107195394 - DOKUMEN.PUB,
<https://dokumen.pub/artificial-intelligence-foundations-of-computational-agents-2nd-ed-110719539x-9781107195394.html>
 16. Artificial Intelligence - Assets - Cambridge University Press,
https://assets.cambridge.org/97810092/58197/frontmatter/9781009258197_frontmatter.pdf