Università degli Studi di Milano - Bicocca
Scuola di Scienze
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di laurea in Informatica

Analisi comparativa di modelli di previsione di Serie Storiche

Relatore: Prof.ssa Enza Messina

Correlatore: Marco Piazza

Relazione della prova finale di:
Andrea Vasciminno
Matricola 904899

Anno Accademico 2024-2025

Ringraziamenti

Probabilmente non sono la persona migliore nel fare questo genere di cose. Non lo sono mai stata e, quasi sicuramente, mai lo sarò. Per una volta però, ci tengo a dedicare personalmente i miei più sentiti ringraziamenti alle persone che maggiormente mi sono state vicine nel corso di questi tre anni e, più in generale, nella mia vita.
Un enorme ringraziamento, che non sarà mai abbastanza, va a tutta la mia famiglia.
Ai miei genitori, Roberto e Monica, che da sempre siete i miei pilastri. Tu, papà, sei il mio modello d'ispirazione. Nessuno come te mi ha mai dimostrato quanto sia importante il duro lavoro e il non abbattersi mai. Sei, tutt'oggi, l'unico uomo al mondo da cui ho la certezza che non smetterò mai di imparare abbastanza. E tu, mamma, sei stata la persona che più di tutte mi è stata vicina nei miei anni di istruzione. Anche se abbiamo modi di fare spesso differenti, so perfettamente quanto mi vuoi bene e non riesco neanche a quantificare quanto ne voglia io a te. Spero che un giorno, al di là di quello che il futuro mi riserverà, di poter essere almeno un decimo delle persone e dei genitori straordinari che siete stati per me. Grazie di tutto.
E poi c'è mia sorella, Sara, la persona per cui farei qualsiasi cosa al mondo. Anche se non lo sai, sei sempre stata la mia fonte d'ispirazione. Vederti raggiungere tutti gli obiettivi che ti sei sempre prefissata mi ha sempre spronato a dare il massimo. Se oggi sono una persona che raggiunge i propri traguardi, la maggior parte del merito è tuo, che solamente essendo te stessa mi hai sempre ispirato in ogni fase della mia vita.
Un grazie a tutte le persone che mi sono state vicine in questo percorso universitario, ai miei compagni di corso Felice, Simone e Vincenzo.
Grazie a tutti i miei amici che mi sono sempre stati accanto. Un grazie speciale a Davide, Gabriele e Giammarco, amici di una vita, tra le poche persone che, da sempre, mi affiancano nella mia vita. Persone vere, capaci di dirmi sempre la verità, anche quando sembra difficile farlo. Grazie per le risate, le litigate e tutti i momenti magnifici passati insieme in questi anni. Sono felice di avervi al mio fianco in un traguardo così importante per me. Spero che siate parte di tutto ciò che verrà da oggi in poi.
L'ultimo grande ringraziamento ci tengo a farlo a una persona fondamentale nel mio percorso di studi, la prof.ssa Elisa Russo. Grazie per avermi insegnato che chiunque a questo mondo possiede qualcosa da insegnarmi. L'ho conosciuta che ero un bambino e lei, grazie ai suoi insegnamenti, ha cambiato il mio carattere e mi ha reso un uomo. Le

sono grato non solo per l'istruzione che mi ha fornito ma, soprattutto, per quello che mi ha insegnato dal punto di vista umano. Una vera e propria maestra di vita.

Introduzione

Lo studio delle serie storiche è un tema centrale in molti ambiti della ricerca e del mondo applicativo. Ogni volta che si raccolgono dati nel tempo, come i prezzi di mercato, i consumi energetici o l'andamento di fenomeni sociali, sorge la necessità di comprenderne l'evoluzione e, soprattutto, di riuscire a prevedere come si comporteranno in futuro.

Questa esigenza nasce sia da motivazioni pratiche, come prendere decisioni più consapevoli o ridurre i rischi legati all'incertezza, sia da un interesse scientifico verso la comprensione delle dinamiche che regolano sistemi complessi.

Il problema, tuttavia, non è banale: le serie storiche possono mostrare andamenti irregolari, cambiamenti improvvisi o tendenze di lungo periodo difficili da catturare con un unico approccio. Proprio per questo, nel corso del tempo sono stati sviluppati diversi strumenti di analisi e previsione, che vanno dai modelli statistici più tradizionali fino a metodi moderni di apprendimento automatico.

Questa tesi si inserisce in tale contesto e si propone di studiare alcune di queste metodologie, mettendole a confronto sullo stesso insieme di dati per valutarne potenzialità e limiti.

Questo lavoro si propone di analizzare diverse metodologie di rappresentazione delle serie storiche, con l'obiettivo di confrontare modelli statistici e modelli di Machine Learning, in particolare reti di Deep Learning di tipo LSTM.

Il confronto viene condotto su un dataset di benchmark, applicando a ciascun modello un preprocessing dei dati coerente con le sue specifiche esigenze. Successivamente, vengono costruiti i modelli e svolte procedure di fine-tuning finalizzate a ottimizzarne le prestazioni.

I risultati ottenuti vengono infine confrontati sia dal punto di vista grafico sia tramite le principali metriche di valutazione, in modo da evidenziare punti di forza e limiti dei diversi approcci.

Struttura della tesi

La presente tesi è articolata in cinque capitoli principali, che guidano il lettore lungo il percorso di analisi e confronto tra modelli statistici e di deep learning applicati a serie storiche.

Il Capitolo 1 introduce il contesto dello studio, le motivazioni alla base della ricerca e gli obiettivi perseguiti, fornendo inoltre una panoramica generale sui principali metodi di approccio.

Il Capitolo 2 descrive nel dettaglio il dataset utilizzato, illustrandone le caratteristiche principali, la provenienza e le operazioni preliminari di pulizia e preparazione dei dati. Definisce inoltre le metriche utilizzate per l'analisi dei risultati.

Il Capitolo 3 è dedicato all'analisi del modello ARIMA. Viene fornita una presentazione dei fondamenti teorici, una discussione riguardo alle procedure di identificazione dei parametri, le tecniche di stima e i risultati sperimentali ottenuti.

Il Capitolo 4 approfondisce l'utilizzo delle reti neurali LSTM. Viene introdotta l'architettura di riferimento, spiegati i criteri di addestramento e ottimizzazione, e vengono presentati i risultati derivanti dall'applicazione del modello ai dati oggetto di studio.

Il Capitolo 5 propone un confronto tra i modelli ARIMA e LSTM, sia dal punto di vista delle metriche quantitative che dell'analisi qualitativa delle previsioni. Infine, vengono discusse le potenziali linee di sviluppo futuro della ricerca.

Indice

Capitolo 1

Approcci alla modellazione delle serie temporali

La previsione di serie temporali (Time Series Forecasting), come discusso in [16] *Introduction to time series analysis and forecasting*, è un settore dell'analisi predittiva che si occupa di stimare futuri valori di una variabile sulla base dei dati storici osservati nel tempo.

In una serie storica, i dati vengono raccolti e registrati in sequenza cronologica lungo un determinato intervallo temporale, rendendo possibile l'identificazione di pattern e tendenze che possono essere sfruttate per effettuare previsioni affidabili.

Lo studio delle serie storiche è di fondamentale importanza in numerosi ambiti applicativi, tra cui la finanza, l'economia, la meteorologia, il settore energetico, i trasporti, la sanità e molti altri. Ad esempio, prevedere l'andamento dei prezzi azionari, il consumo di energia o la domanda di servizi sanitari consente di ottimizzare le risorse, pianificare strategie e mitigare i rischi.

Una serie storica è una sequenza di osservazioni ordinate temporalmente, le cui dinamiche sono influenzate da diverse componenti fondamentali. Tra queste, il trend descrive l'andamento a lungo termine, come una crescita o decrescita progressiva. A questo si aggiunge la stagionalità, che introduce pattern ricorrenti a intervalli regolari, spesso legati a fattori ciclici come le stagioni o i giorni della settimana. Infine, il rumore comprende tutte le variazioni casuali e imprevedibili che non seguono uno schema specifico, poiché influenzate da fattori esterni non misurabili.

1.1 Metodologie per la rappresentazione delle serie storiche

La previsione delle serie storiche può essere affrontata attraverso una varietà di modelli, che differiscono per complessità, interpretabilità e capacità predittiva. Tali modelli si suddividono generalmente in due grandi categorie: modelli statistici tradizionali e modelli basati su tecniche di apprendimento automatico (Machine Learning)[10].

I modelli statistici, come ARIMA, ETS e VAR, si fondano su assunzioni teoriche ben definite riguardo la struttura della serie temporale e offrono strumenti interpretativi chiari. Sono particolarmente efficaci quando i dati mostrano pattern lineari e ben strutturati. Allo stesso tempo, però, non risultano particolarmente utili in contesti con grandi quantità di dati e feature da utilizzare

I modelli di Machine Learning, invece, fanno uso di algoritmi più flessibili e spesso non richiedono ipotesi forti sulla natura del dato. Questi approcci si sono rivelati particolarmente utili nel catturare relazioni complesse e non lineari, soprattutto in presenza di grandi quantità di dati.

1.1.1 Metodi Statistici

I metodi statistici [1] rappresentano l'approccio classico e consolidato per l'analisi e la previsione delle serie temporali. A differenza dei modelli di Machine Learning, questi si basano su formulazioni matematiche esplicite che cercano di descrivere la struttura stocastica sottostante ai dati. L'obiettivo è quello di identificare e modellare componenti come il trend, la stagionalità e il rumore residuo, assumendo che le dinamiche passate possano essere estrapolate per prevedere il futuro.

Tra i modelli statistici più diffusi e studiati vi è la famiglia ARIMA (Autoregressive Integrated Moving Average). Questi modelli esprimono il valore futuro di una serie come una combinazione lineare dei suoi valori passati e degli errori di previsione passati. Un presupposto fondamentale per molti di questi metodi è la stazionarietà della serie temporale, ovvero la costanza delle sue proprietà statistiche (come media e varianza) nel tempo. Qualora la serie non fosse stazionaria, è spesso necessario applicare trasformazioni, come la differenziazione, per renderla tale.

Il principale punto di forza dei modelli statistici risiede nella loro elevata interpretabilità. I parametri del modello hanno un significato statistico preciso, che permette di comprendere chiaramente la relazione tra le variabili e le dinamiche della serie. Tuttavia, il loro limite principale è l'assunzione di linearità nelle relazioni tra i dati. Questa caratteristica li rende meno adatti a catturare le complesse dinamiche non lineari che spesso caratterizzano le serie temporali reali, ambito in cui i modelli di Machine Learning dimostrano invece una maggiore efficacia.

1.1.2 Modelli di Machine Learning

I modelli più avanzati per la previsione delle serie temporali si basano prevalentemente su algoritmi di Machine Learning [23], oppure su approcci ibridi che combinano modelli statistici classici con tecniche di apprendimento automatico. La motivazione di questa evoluzione è dettata dall'esigenza di modellare relazioni altamente non lineari tra i dati. Questi modelli ottengono generalmente una precisione superiore, anche su orizzonti previsivi di lungo termine.

Un aspetto fondamentale da sottolineare è che la maggior parte dei modelli di Machine Learning per la previsione avanzata si fonda sull'impiego di reti neurali, ovvero strutture computazionali ispirate al funzionamento del cervello umano. Tra queste, le reti neurali ricorrenti (RNN) e le loro varianti più evolute come le LSTM (Long Short-Term Memory) e le GRU (Gated Recurrent Unit) si sono dimostrate particolarmente efficaci nell'analisi di dati sequenziali [22], come le serie temporali, grazie alla loro capacità di mantenere una memoria del contesto passato.

Tuttavia, tra gli svantaggi principali dei modelli basati su Machine Learning vi è la scarsa interpretabilità. A differenza dei modelli statistici, che forniscono parametri facilmente leggibili e relazioni dirette tra le variabili, i modelli di Machine Learning sono spesso considerati delle "scatole nere", il che può costituire un limite in contesti dove la trasparenza del processo decisionale è fondamentale.

Capitolo 2

Setup dell'esperimento

2.1 Dataset Ethereum Historical Prices

L'analisi dei mercati finanziari, in particolare di quelli legati alle criptovalute, richiede la disponibilità di dati storici accurati e strutturati. Ethereum (ETH), seconda criptovaluta per capitalizzazione di mercato dopo Bitcoin, è caratterizzata da un'elevata volatilità, influenzata da fattori tecnologici, economici e regolatori[18]. La possibilità di monitorare e prevedere l'andamento del suo prezzo riveste un ruolo centrale nello sviluppo di strategie di investimento e nella valutazione della dinamica del mercato delle criptovalute.

Il dataset [26] oggetto di studio raccoglie le quotazioni giornaliere di Ethereum dal 1° gennaio 2018 al 26 settembre 2024, fornendo una serie temporale completa e di lunga durata. Ogni osservazione corrisponde a una singola sessione di trading e include informazioni fondamentali per descrivere l'andamento del mercato: prezzo di apertura (*Open*), massimo giornaliero (*High*), minimo giornaliero (*Low*), prezzo di chiusura (*Close*) e volume scambiato (*Volume*).

La variabile principale di interesse è il **prezzo di chiusura**, frequentemente utilizzato negli studi di analisi tecnica e nella previsione dei prezzi[6]. Tuttavia, la disponibilità delle altre metriche consente di integrare approcci più sofisticati, che considerano la volatilità nell'arco della giornata, le dinamiche di domanda e offerta e i segnali derivanti dai volumi di scambio.

La struttura giornaliera del dataset lo rende adatto a diversi ambiti applicativi. Dallo sviluppo di modelli di forecasting per serie storiche, alla progettazione e valutazione di algoritmi di trading automatico, fino all'analisi esplorativa dei trend di lungo periodo. Grazie alla ricchezza informativa e all'ampiezza temporale, questo dataset rappresenta una risorsa preziosa per ricercatori, analisti e investitori interessati a comprendere le dinamiche di Ethereum e del più ampio ecosistema delle criptovalute.

Analisi esplorativa

La Tabella 2.1 riporta le principali statistiche descrittive delle variabili considerate nel dataset, incluse media, varianza, deviazione standard, mediana e valore minimo. Queste misure forniscono una prima panoramica della distribuzione dei dati e permettono di valutare la presenza di eventuali differenze di scala o di variabilità tra le variabili analizzate

Variabile	**Mean**	**Variance**	**Std. Dev**	**Median**	**Min**
Open	1,448.708	1,465,906.0	1,210.746	1,319.290	84.12
High	1,489.955	1,548,302.0	1,244.308	1,353.285	84.53
Low	1,402.849	1,373,834.0	1,172.107	1,271.875	82.08
Close	1,449.480	1,466,272.0	1,210.897	1,320.270	84.12
Volume	1,003,126.0	1.241976×10^{12}	1,114,440.0	524,569.395	29,811.67

Tabella 2.1: Statistiche descrittive dataset

La variabile di riferimento su cui sono stati condotti gli esperimenti è, come precedentemente indicato, Close, che rappresenta il valore di chiusura giornaliero. In Figura 2.1 è mostrato l'andamento temporale di tale variabile sull'intero periodo di osservazione.

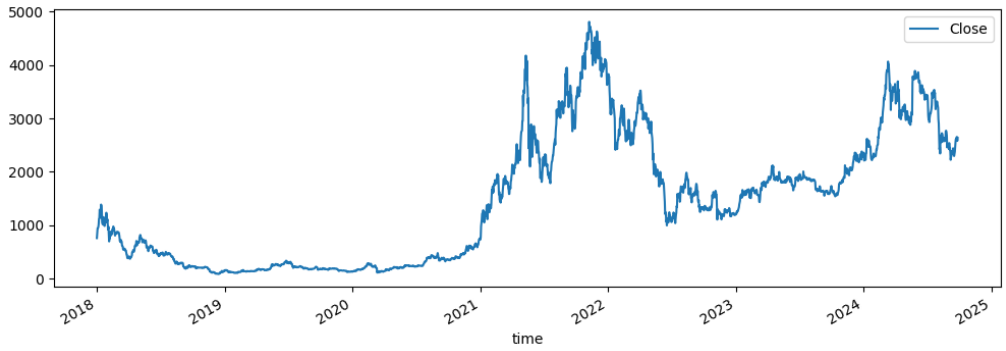

Figura 2.1: Andamento temporale della variabile *Close*, utilizzata come target di previsione.

Dall'analisi del grafico emerge che la variabile ha mantenuto un andamento relativamente stabile nella fase iniziale delle rilevazioni, fino al 2021, per poi mostrare un incremento della volatilità con oscillazioni più marcate e la presenza di picchi significativi sia verso l'alto che verso il basso. Tale comportamento riflette la natura dinamica e non stazionaria del mercato, rendendo il problema di previsione particolarmente complesso per i modelli di apprendimento automatico.

Per valutare la stazionarietà della serie temporale relativa alla variabile Close, è stata calcolata la *media mobile* e la *deviazione standard mobile*[12] utilizzando una finestra temporale di 30 giorni. I risultati sono illustrati in Figura 2.2.

Dall'osservazione del grafico si nota che, nel corso del periodo analizzato, sia la media che la deviazione standard variano in modo significativo. In particolare, nella prima parte

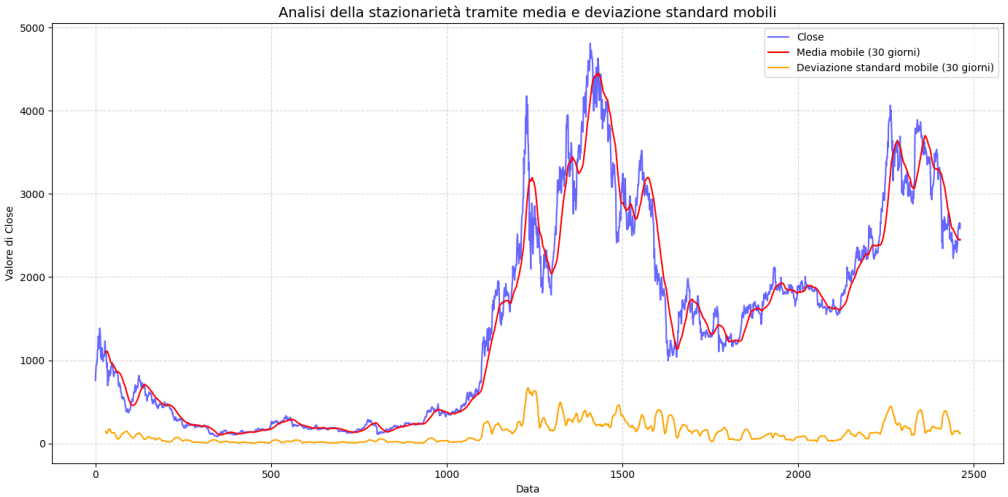

Figura 2.2: Andamento della variabile *Close* con sovrapposizione della media e della deviazione standard mobili

della serie la media mobile presenta valori contenuti e relativamente stabili, mentre nelle fasi successive, corrispondenti ai periodi di forte crescita e successivo calo del prezzo, si evidenziano oscillazioni ampie e irregolari. Analogamente, la deviazione standard mobile mostra un incremento in corrispondenza dei picchi di prezzo, segno di una maggiore volatilità del mercato.

Queste variazioni indicano che la serie non può essere considerata stazionaria, poiché le sue proprietà statistiche (media e varianza) non restano costanti nel tempo. Tale comportamento riflette la natura dinamica del mercato delle criptovalute e suggerisce la necessità di applicare tecniche di trasformazione o differenziazione per stabilizzare la serie prima dell'addestramento, specialmente in modelli statistici come ARIMA.

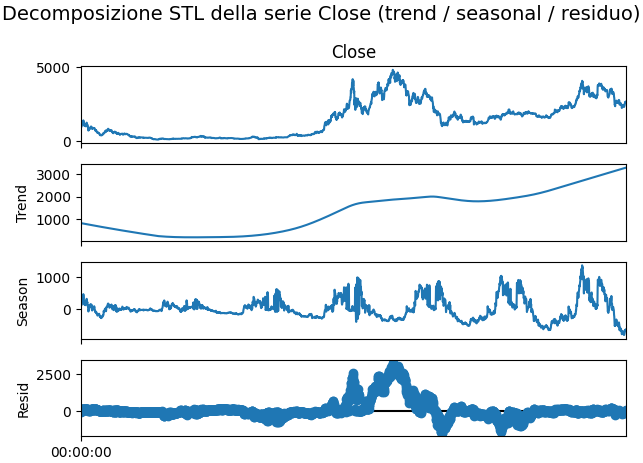

Figura 2.3: Decomposizione STL della serie temporale *Close* nelle componenti di trend, stagionalità e residuo.

Per approfondire la struttura temporale della variabile Close, è stata applicata una decomposizione STL [3], che consente di scomporre la serie originale nelle tre componenti fondamentali: trend, stagionalità e residuo.

I risultati dell'analisi sono riportati in Figura 2.3. L'analisi di stagionalità e delle tendenze rappresenta un passaggio fondamentale nell'esplorazione delle serie temporali, in quanto consente di comprendere la natura dei dati prima di procedere con la fase di modellazione predittiva. La componente di Trend rappresenta la tendenza di fondo della serie temporale, cioè l'andamento di lungo periodo che non dipende da fluttuazioni stagionali o da rumore. Oltre a evidenziare la non stazionarietà della serie, la componente di trend fornisce indicazioni preziose sulla struttura di lungo periodo del dataset. In particolare, permette di individuare la direzione complessiva del fenomeno, eventuali punti di svolta che segnalano cambi di regime e il grado di persistenza temporale della serie. L'andamento regolare del trend fino al 2021, seguito da irregolarità suggerisce una presenza di memoria a lungo termine durante la fase di crescita, che va a sparire nelle fasi successive. Questo implica che i modelli di previsione dovranno essere in grado di gestire dipendenze temporali di diversa scala, combinando la capacità di cogliere pattern di lungo periodo con la flessibilità nel rappresentare variazioni improvvise.

La componente stagionale presenta oscillazioni periodiche che si ripetono nel tempo, con ampiezza variabile. Questi pattern suggeriscono la presenza di dinamiche cicliche a breve termine, determinate da eventi costanti di mercato ma evidenziando soprattutto irregolarità nella stagionalità, implicando assenza di ciclicità dei dati. Infine, la componente residua rappresenta le variazioni della serie che non vengono spiegate dal trend o dalla stagionalità. Essa mostra fluttuazioni più marcate in corrispondenza dei periodi di maggiore volatilità del mercato. Pur contenendo una quota di rumore casuale, il residuo evidenzia anche movimenti locali che potrebbero derivare da fattori esterni al modello, come variazioni improvvise della domanda. Questa componente costituisce la parte più complessa da prevedere, poiché non segue pattern regolari o ricorrenti.

2.2 Metriche di valutazione

Per valutare le prestazioni dei modelli di previsione, sono state utilizzate tre metriche comunemente adottate nel campo del time series forecasting[11]: la Mean Absolute Error (MAE), la Root Mean Squared Error (RMSE) e la Mean Absolute Percentage Error (MAPE). Queste metriche consentono di misurare la distanza tra i valori previsti e quelli osservati, fornendo una valutazione quantitativa dell'accuratezza del modello.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad \text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad \text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

dove:

- y_i rappresenta il valore reale osservato al tempo i;
- $\hat{y}_i$ è il valore previsto dal modello;
- n indica il numero totale di osservazioni.

Queste metriche sono tra le metriche più comunemente utilizzate per valutare le prestazioni di un modello predittivo.

Il MAE misura l'errore medio in valore assoluto, fornendo un'indicazione intuitiva della deviazione media tra i valori osservati e quelli previsti. Si tratta di una metrica robusta rispetto alla presenza di outlier, poiché ogni errore contribuisce in modo lineare al calcolo complessivo. Tuttavia, essa non distingue tra errori piccoli e grandi, attribuendo loro lo stesso peso, e può quindi risultare meno sensibile alle deviazioni più marcate.

L'RMSE, invece, penalizza maggiormente gli errori di grande entità grazie all'elevazione al quadrato delle differenze tra valori osservati e stimati. Di conseguenza, risulta più sensibile agli outlier rispetto alla MAE ed è particolarmente utile quando si desidera enfatizzare la presenza di grandi deviazioni nelle previsioni. D'altro canto, in presenza di pochi valori anomali, il suo valore può risultare distorto, fornendo una percezione eccessiva dell'errore medio complessivo.

Infine, il MAPE esprime l'errore medio in termini percentuali, consentendo di confrontare modelli o serie temporali con scale differenti. Pur essendo una misura di facile interpretazione, presenta tuttavia due principali limitazioni: in primo luogo, può divergere in presenza di valori osservati y_i prossimi allo zero, poiché il denominatore amplifica l'errore relativo; in secondo luogo, tende a penalizzare in misura maggiore gli errori commessi su valori di piccola entità rispetto a quelli su valori elevati, introducendo così una possibile distorsione nella valutazione complessiva dell'accuratezza del modello.

In sintesi, l'utilizzo congiunto di MAE, RMSE e MAPE consente di ottenere una valutazione completa delle prestazioni del modello, favorendo il confronto anche con prestazioni offerte da altri modelli.

Capitolo 3

ARIMA

Sviluppato negli anni '70, il modello ARIMA[15] rappresenta uno dei veri e propri pilastri nella previsione delle serie temporali e continua ancora oggi a essere ampiamente utilizzato in contesti molto diversi tra loro. Anche nei modelli più avanzati di tipo ibrido[9] o ensemble basati sul Machine Learning, ARIMA viene spesso incluso come componente fondamentale.

3.1 Struttura del modello

Il modello ARIMA si basa sull'assunto, comune a molte famiglie di modelli statistici, che i valori futuri di una variabile possano essere espressi come una combinazione lineare dei valori passati. Tale ipotesi costituisce al contempo il principale punto di forza e il limite intrinseco del modello: da un lato garantisce una struttura semplice e interpretabile; dall'altro riduce la capacità di catturare dinamiche non lineari e ne limita l'efficacia predittiva nel lungo periodo, a causa della forte dipendenza dai dati storici.

L'acronimo ARIMA (AutoRegressive Integrated Moving Average) sintetizza le tre componenti fondamentali del modello:

- **AutoRegressive (AR)**: i valori futuri della serie dipendono linearmente da un numero p di osservazioni passate. Ad esempio, per $p = 1$, il valore al tempo t viene stimato in funzione del valore osservato al tempo $t - 1$.
- **Integrated (I)**: la serie viene differenziata d volte al fine di renderla stazionaria. L'operazione di differenziazione consente di eliminare trend e variazioni stagionali non stazionarie.
- **Moving Average (MA)**: i valori futuri dipendono linearmente da un numero q di errori commessi nelle previsioni precedenti (residui).

Nell'applicazione, i modelli ARIMA differenziano l'utilizzo dei parametri. I parametri che devono essere impostati sono p, d e q.

Il parametro p rappresenta la parte autoregressiva, ossia quanti ritardi del valore osservato vengono usati come regressori. Ad esempio, se si imposta $p = 2$, il valore attuale è funzione lineare dei valori osservati ai tempi $t - 1$ e $t - 2$.

Il parametro d rappresenta quante volte differenziare la serie originale per renderla stazionaria. Se $d = 1$, si utilizza la prima differenza: $y_t - y_{t-1}$, mentre se $d = 2$ si differenzia due volte, e così via.

Il parametro q rappresenta la componente di media mobile, cioè quanti ritardi dell'errore sono utilizzati nel modello. Pertanto, se $q = 1$, il valore predetto y_t attuale dipende anche dall'errore associato a y_{t-1}.

3.2 Selezione dei parametri

Un aspetto centrale nell'implementazione di un modello statistico come l'ARIMA riguarda la definizione dei parametri ottimali [25]. Il primo requisito è che la serie storica analizzata sia stazionaria. In assenza di stazionarietà, infatti, il modello tende a fornire previsioni poco affidabili. Dalle analisi condotte nel Capitolo 2 la serie mostra evidenze di non stazionarietà. Come riscontro numerico di questa osservazione, è stato utilizzato il test di Dickey-Fuller Aumentato (ADF) [13].

Statistica	**Valore**
ADF Statistic	-1.397600
p-value	0.583426

Tabella 3.1: Risultati del test di Dickey-Fuller Aumentato (ADF)

Dai valori riportati in Tabella 3.1, il *p-value* risulta superiore alla soglia comunemente adottata di 0.05, confermando la non stazionarietà della serie originale. Per ovviare a tale condizione, si ricorre alla differenziazione, che consente di eliminare eventuali trend e rendere la serie più adatta alla modellizzazione.

Il processo di ricerca del grado ottimale di differenziazione (d) si articola nei seguenti passaggi:

- **Analisi della serie originale:** valutazione del *p-value* tramite il test ADF, precedentemente descritto. Come evidenziato, il valore ottenuto indica la non stazionarietà della serie.
- **Iterazione della differenziazione:** si applica ripetutamente la differenziazione, valutando il *p-value* al termine di ogni iterazione. Quando il *p-value* scende al di sotto della soglia di significatività, il corrispondente valore del parametro d viene considerato ottimale. Applicando tale procedura al nostro dataset, l'algoritmo ha individuato un valore ottimale pari a $d = 1$.

Il risultato della prima differenziazione è illustrato in Figura 3.1

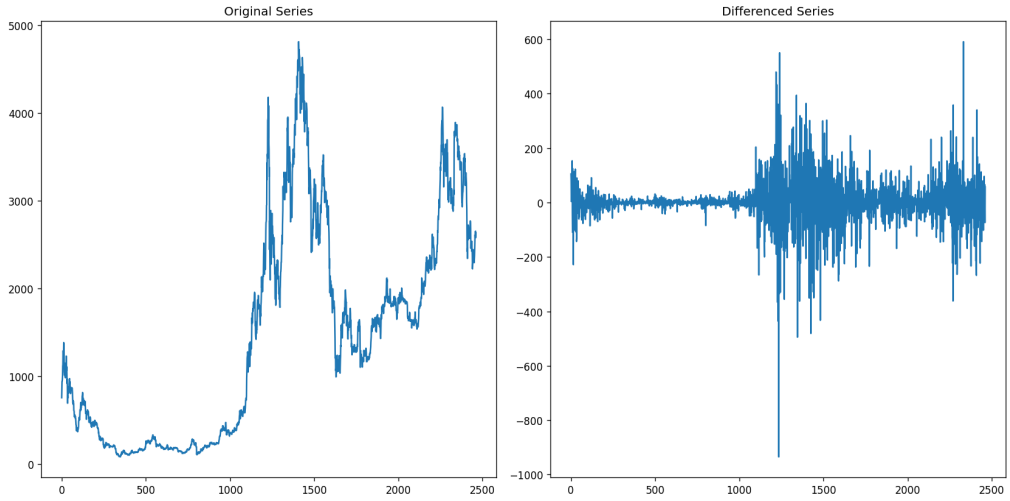

Figura 3.1: Confronto tra serie originale e a seguito di differenziazione di primo ordine

Per identificare i parametri p e q del modello ARIMA, è fondamentale analizzare le caratteristiche della funzione di autocorrelazione (ACF) e della funzione di autocorrelazione parziale (PACF)[15].

La funzione di autocorrelazione misura il grado di dipendenza lineare tra i valori della serie temporale y_t e i suoi valori ritardati y_{t-k}. Essa è definita come:

$$\rho_k = \frac{\mathrm{Cov}(y_t, y_{t-k})}{\sqrt{\mathrm{Var}(y_t)\,\mathrm{Var}(y_{t-k})}}$$

La funzione di autocorrelazione parziale rappresenta invece la correlazione tra y_t e y_{t-k} una volta rimossa l'influenza dei ritardi intermedi $y_{t-1}, \ldots, y_{t-k+1}$. Essa consente di individuare la componente autoregressiva del modello.

$$\phi_{kk} = \frac{\rho_k - \sum_{j=1}^{k-1} \phi_{k-1,j}\rho_{k-j}}{1 - \sum_{j=1}^{k-1} \phi_{k-1,j}\rho_j}$$

L'analisi congiunta di ACF e PACF permette di identificare i parametri p e q:

- un taglio netto dell'ACF dopo il lag q indica una componente MA(q);
- un taglio netto della PACF dopo il lag p indica una componente AR(p).

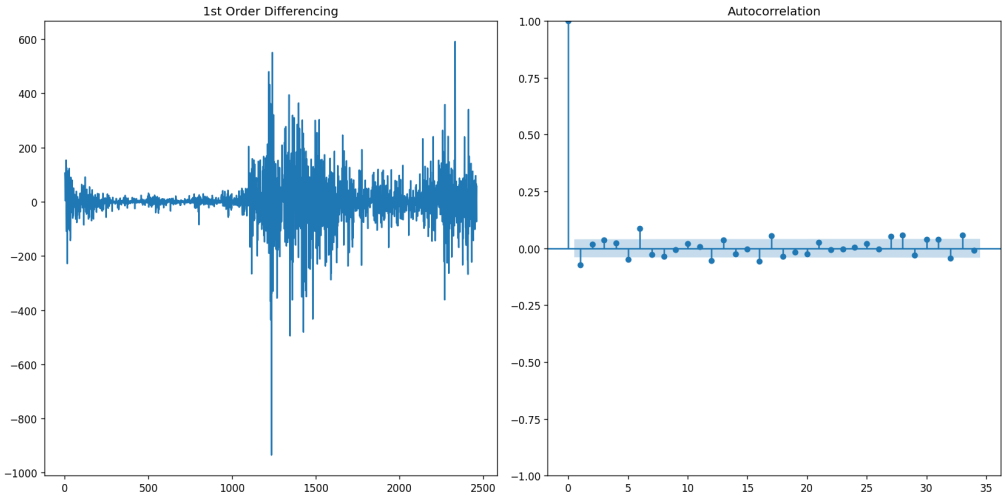

Figura 3.2: Funzione di autocorrelazione (ACF).

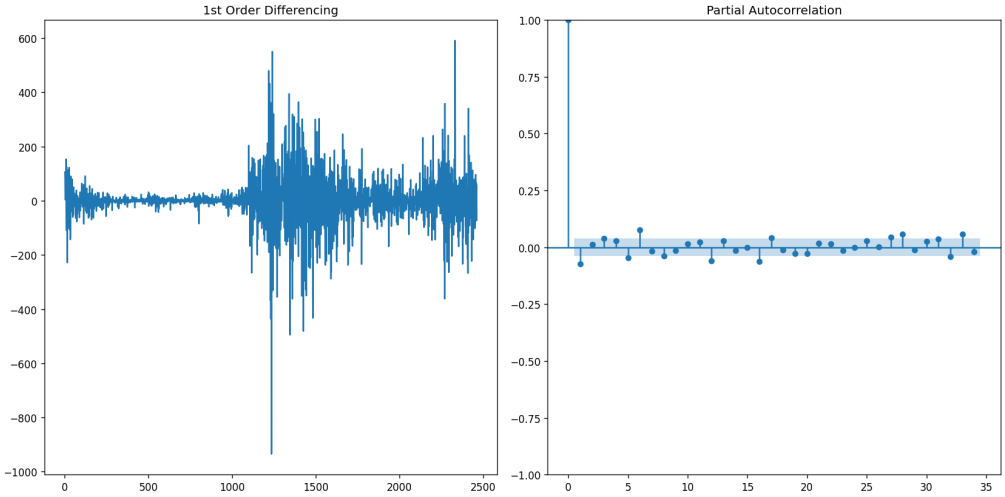

Figura 3.3: Funzione di autocorrelazione parziale (PACF).

L'analisi dei grafici permette di individuare in modo chiaro i parametri ottimali per il modello. La serie differenziata oscilla attorno allo zero, senza mostrare trend evidenti, il che conferma la necessità di una singola differenziazione, con $d = 1$(Figura 3.1). La funzione di autocorrelazione parziale (PACF) mostra una netta variazione dopo il primo lag, suggerendo l'inclusione di un termine autoregressivo di primo ordine, ovvero $p = 1$. (Figura 3.3). In modo analogo, la funzione di autocorrelazione (ACF) evidenzia anch'essa una brusca variazione dopo il primo lag, indicando la necessità di un termine di media mobile di ordine uno, $q = 1$ (Figura 3.2).

Le stime ottenute hanno condotto all'identificazione di un modello ARIMA(1,1,1). Tuttavia, considerata la forte somiglianza dei parametri e con l'obiettivo di semplificare la struttura del modello, si è deciso di adottare un modello ARIMA(1,1,0).

Questa scelta è giustificata non solo da ragioni di parsimonia, ma anche da considerazioni interpretative. In particolare, il modello ARIMA(1,1,0) può essere interpretato come un modello AR(1) applicato alla serie differenziata. L'operazione di differenziazione di primo ordine consente infatti di eliminare eventuali componenti di trend presenti nella serie originaria, permettendo di analizzare la relazione autoregressiva tra la serie differenziata e il suo valore ritardato. In altri termini, l'applicazione di un modello ARIMA(1,1,0) alla serie originale è equivalente all'utilizzo di un modello AR(1) sulla serie delle differenze. Tale formulazione consente di cogliere sia la tendenza di lungo periodo, attraverso la componente di integrazione, sia la dipendenza di breve periodo, rappresentata dalla componente autoregressiva.

In conclusione, il modello ARIMA(1,1,0) non introduce una struttura sostanzialmente nuova rispetto all'AR(1), ma ne estende l'applicabilità a serie temporali non stazionarie, risultando quindi particolarmente adeguato in presenza di trend che renderebbero meno appropriato un modello autoregressivo semplice.

3.3 Generazione delle predizioni

Una volta eseguite le analisi sui parametri del modello, la generazione delle predizioni risulta relativamente semplice. Una delle peculiarità dei modelli statistici come l'ARIMA è infatti la possibilità di lavorare direttamente sui dati, senza la necessità di procedure di normalizzazione o preprocessing particolarmente complessi.

Dopo aver stimato il modello con i parametri ottimali, le prestazioni ottenute sono riassunte in Tabella 3.2.

Metrica	**Valore**
RMSE	80.64
MAE	43.91
MAPE	3.23%

Tabella 3.2: Prestazioni del modello ARIMA con parametri ottimizzati.

La Figura 3.4 mostra la capacità predittiva del modello, evidenziando il confronto tra i valori stimati e i valori reali della serie storica.

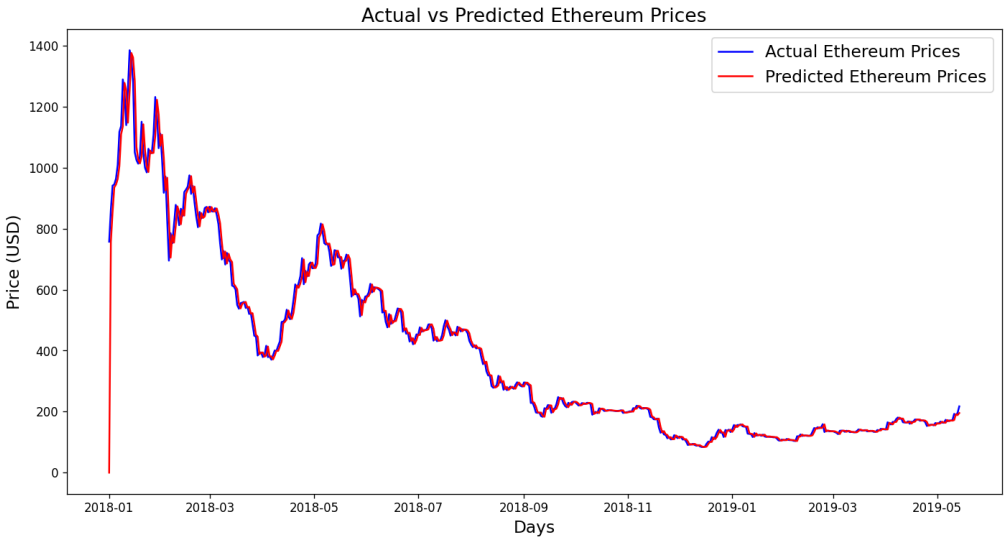

Figura 3.4: Valori predetti rispetto ai valori reali con modello ARIMA.

Dal grafico si osserva che il modello riesce apparentemente a seguire il trend generale della serie, mostrando valori delle metriche di errore complessivamente soddisfacenti. Tuttavia, un'analisi più approfondita evidenzia come le previsioni tendano a ricalcare strettamente l'andamento passato, comportandosi più come una riproduzione dei valori precedenti che come una vera e propria previsione.

In particolare, nelle prime 100 osservazioni (Figura 3.5) si nota maggiormente questo *shift* temporale, segno che il modello reagisce con un certo ritardo alle variazioni della serie. Tale scostamento tende progressivamente a ridursi con l'aumentare del numero di osservazioni, senza andare mai però a svanire, confermando la limitata capacità del modello di anticipare effettivamente i movimenti futuri.

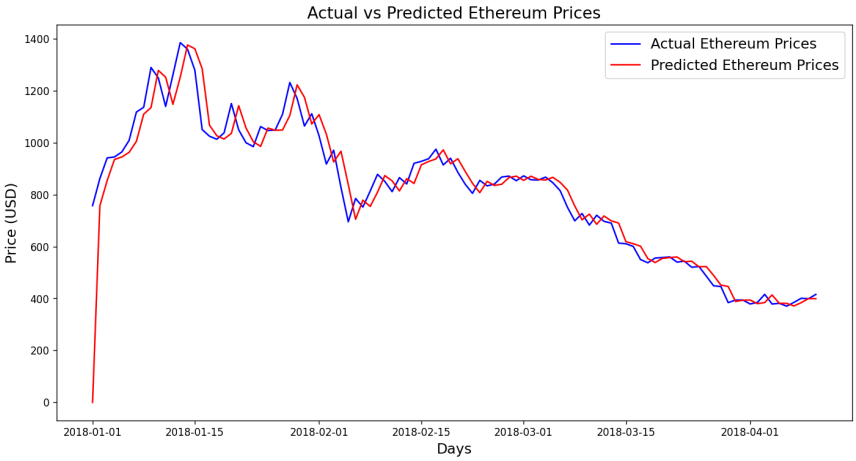

Figura 3.5: Focus sui primi 100 punti temporali predetti

Questo comportamento può essere attribuito anche alla semplicità del modello. Infatti, se da un lato la simulazione di un modello AR(1) garantisce una certa facilità interpretativa, dall'altro la presenza di un solo termine autoregressivo non consente al modello predittivo di cogliere eventuali ritardi che si generano nella serie temporale. Di conseguenza, il modello tende ad apprendere con un ritardo di un punto temporale e a propagare tale errore nelle previsioni successive. Si può quindi dedurre che il modello non stia semplicemente replicando i valori passati, ma stia piuttosto apprendendo un trend ritardato, reiterando questo comportamento nelle previsioni future.

3.4 Generazione della predizione one-step ahead

A differenza del caso precedente, per la valutazione delle prestazioni del modello è stata adottata una procedura di tipo **walk-forward**. In questo approccio, il dataset è stato suddiviso in due sottoinsiemi: un insieme di *training* e uno di *test*.

Ad ogni passo temporale, il modello viene stimato sui dati disponibili fino a quel momento e viene quindi generata una previsione *one-step ahead*. Tale metodologia, sebbene più onerosa dal punto di vista computazionale, dovrebbe riprodurre più fedelmente il contesto operativo.

Utilizzando lo stesso dataset impiegato nella fase precedente, è stato mantenuto invariato l'ordine dei parametri del modello ARIMA. In particolare, il dataset è stato inizialmente suddiviso in un primo insieme di osservazioni, utilizzato come *training set* per stimare l'andamento complessivo della serie, e successivamente è stata implementata la procedura *walk-forward one-step ahead*, che prevede il riaddestramento del modello a ogni nuovo punto temporale disponibile.

Le previsioni ottenute sono riportate in Figura 3.6, mentre le principali metriche di valutazione del modello sono presentate nella Tabella 3.3.

Metrica	**Valore**
RMSE	6.72
MAE	4.55
MAPE	2.99%

Tabella 3.3: Prestazioni del modello ARIMA con metodologia walk-forward

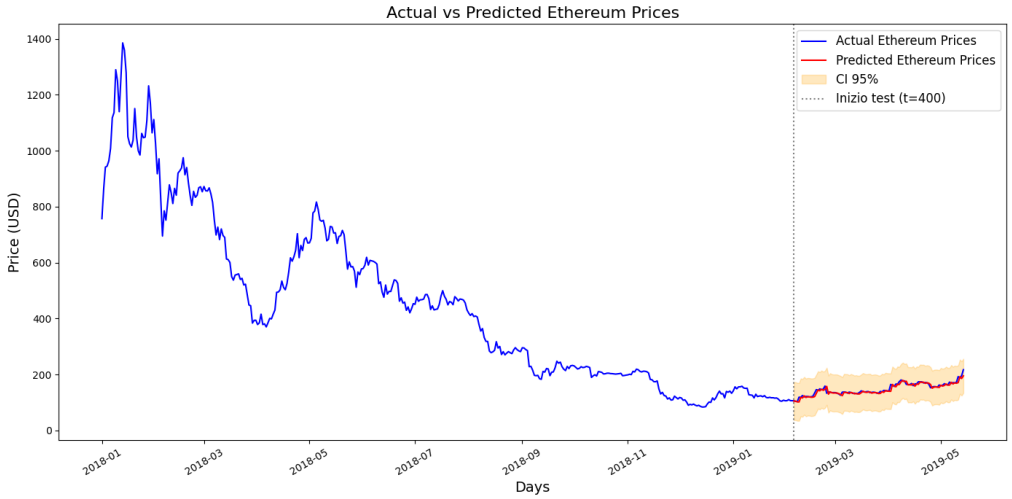

Figura 3.6: Predizioni ottenute con la metodologia walk-forward

I risultati ottenuti mediante questa tipologia di validazione risultano significativamente migliori rispetto a quelli precedentemente ottenuti. Dal confronto delle tabelle contenenti le metriche di valutazione, si osserva infatti una diminuzione di tutti i parametri che misurano la bontà delle predizioni. Ciò suggerisce che, pur mantenendo la semplicità del modello, il metodo con cui tale semplicità viene applicata può avere un impatto rilevante sulle prestazioni complessive.

Inoltre, è stato generato un intervallo di confidenza, particolarmente utile in un contesto finanziario come quello considerato[20]. Nel caso di dati finanziari, infatti, l'elevata volatilità e la presenza di componenti aleatorie rendono le previsioni soggette a un elevato grado di incertezza. L'intervallo di confidenza consente di quantificare tale incertezza, fornendo non soltanto una stima puntuale, ma anche una misura della variabilità attesa delle previsioni. Questo aspetto risulta di fondamentale importanza per l'analisi del rischio e per la definizione di strategie decisionali più consapevoli: ad esempio, un investitore o un analista può utilizzare l'ampiezza dell'intervallo di confidenza per valutare la robustezza delle previsioni e determinare la propria propensione al rischio in relazione ai possibili scenari futuri.

Capitolo 4

Long Short Term Memory

4.1 Introduzione alle Recurrent Neural Networks

Le Recurrent Neural Networks (RNNs) [8] rappresentano una classe di reti neurali progettate per l'elaborazione di dati sequenziali, come serie temporali, testo o segnali audio. La loro peculiarità risiede nella capacità di mantenere una memoria interna che consente di modellare dipendenze temporali tra gli elementi della sequenza.

A differenza delle reti feedforward, dove l'informazione fluisce in maniera unidirezionale dagli input verso l'output, le RNNs introducono collegamenti ricorrenti che permettono di riutilizzare lo stato nascosto calcolato a un passo temporale precedente. [2] In questo modo, l'output in un istante t dipende non solo dall'input corrente x_t, ma anche dallo stato nascosto h_{t-1}:

$$h_t = \phi(W_h x_t + U_h h_{t-1} + b_h)$$

dove ϕ rappresenta una funzione di attivazione non lineare, W_h e U_h sono matrici di pesi, e b_h è il termine di bias.

Questa struttura rende le RNNs adatte a catturare relazioni sequenziali, come l'ordine delle parole in una frase o l'andamento di una variabile nel tempo. Tuttavia, le RNNs tradizionali presentano alcune criticità[14]:

- **Vanishing gradient**: durante l'addestramento, i gradienti possono diventare estremamente piccoli, limitando la capacità della rete di apprendere dipendenze a lungo termine;
- **Exploding gradient**: al contrario, in alcuni casi i gradienti possono crescere eccessivamente, rendendo instabile il processo di ottimizzazione;
- Difficoltà a modellare **dipendenze a lungo termine**, in quanto lo stato nascosto tende a privilegiare informazioni recenti rispetto a quelle più lontane nel tempo.

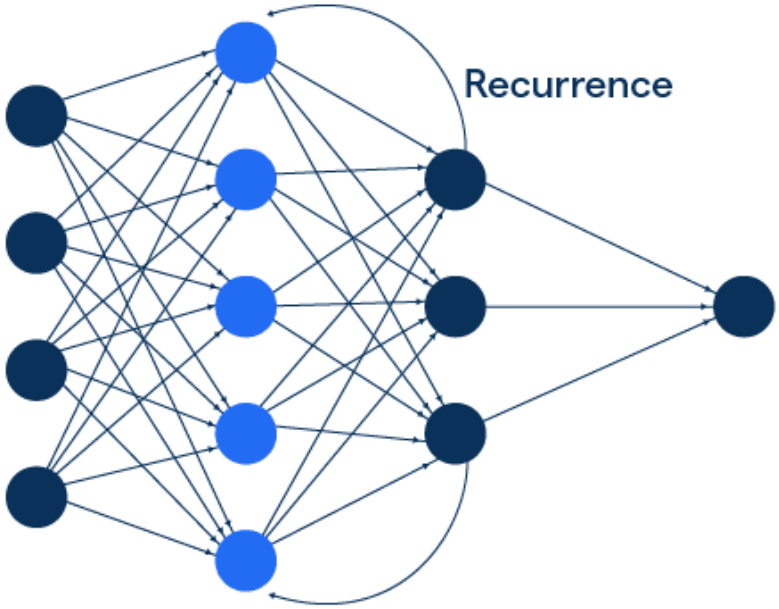

Figura 4.1: Rappresentazione di una Recurrent Neural Network

4.1.1 Il problema del Vanishing Gradient

Uno dei principali limiti delle RNNs [5] riguarda la difficoltà nell'apprendimento di dipendenze a lungo termine, causata dal cosiddetto problema del vanishing gradient.

Se la sequenza è molto lunga, i gradienti tendono a ridursi esponenzialmente man mano che si procede verso i primi passi temporali. In altre parole, i contributi degli input più lontani vengono progressivamente annullati, rendendo la rete incapace di apprendere relazioni di lungo periodo.

Formalmente, questo fenomeno si manifesta perché l'aggiornamento dei pesi dipende da una moltiplicazione ripetuta delle derivate della funzione di attivazione.

Ad esempio, utilizzando la funzione sigmoide, la derivata assume valori compresi tra 0 e 0.25:

$$\sigma'(x) = \sigma(x)(1 - \sigma(x)) \in (0, 0.25)$$

Pertanto, la moltiplicazione iterata di valori inferiori a 1 porta a una decrescita esponenziale del gradiente. Il risultato è che la rete riesce ad apprendere relazioni a breve termine, poiché i gradienti dei passi temporali più vicini non si annullano completamente. Inoltre, le relazioni a lungo termine vengono perse, perché l'informazione non riesce a fluire in maniera efficace fino ai primi stati nascosti della sequenza.

Questo problema ha rappresentato per anni un ostacolo significativo all'uso efficace delle RNNs su compiti complessi di modellazione sequenziale. Per affrontare questa limitazione sono state introdotte architetture più avanzate, come le **Long Short-Term Memory** (LSTM)[4], progettate specificamente per preservare e gestire le informazioni a lungo termine.

4.2 Struttura dell'architettura

Una rete LSTM è un'evoluzione di una RNN. L'obiettivo principale, rispetto alle RNNs tradizionali, è creare un percorso che consenta alle informazioni passate di essere utilizzate direttamente, senza subire modifiche che potrebbero comprometterne l'accuratezza. Questo approccio mira a evitare il problema del vanishing gradient. Questo tipo di rete neurale utilizza due differenti funzioni di attivazione:

- Sigmoid function, la seguente funzione prende ogni coordinata sull'asse delle x e la trasforma in una coordinata sull'asse y con valore compreso fra 0 e 1.

$$f(x) = \frac{e^x}{e^x + 1}$$

- Tanh function, la seguente funzione prende ogni coordinata sull'asse delle x e la trasforma in una coordinata sull'asse y compresa tra -1 e 1

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

LSTM si basa sulle RNNs, arricchendole con la capacità di conservare e recuperare informazioni provenienti da momenti passati, che potrebbero risultare utili in fasi successive della predizione.

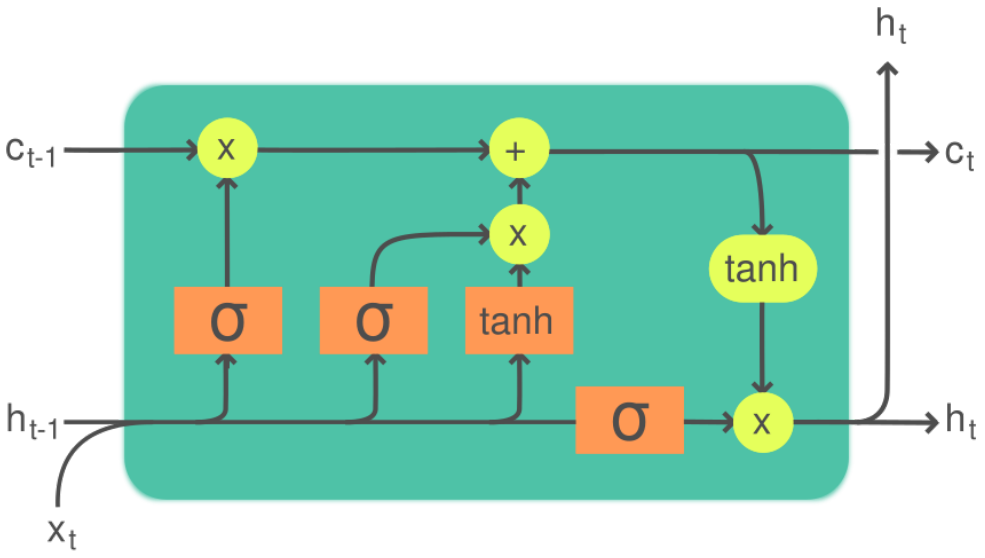

Figura 4.2: Rappresentazione di una rete LSTM

Per permettere ciò, all'interno di un'unità LSTM si distinguono due componenti fondamentali: la cell state e lo hidden state. La cell state rappresenta la memoria a lungo termine del modello. Questo valore viene aggiornato attraverso semplici operazioni di moltiplicazione e somma, poiché non è influenzato direttamente da pesi o bias. Tale caratteristica consente alla memoria a lungo termine di fluire lungo la sequenza di unità senza che il gradiente svanisca, permettendo così al modello di conservare informazioni utili anche a distanza di numerosi passi temporali. L'obiettivo principale della cell state è quindi quello di mantenere informazioni rilevanti nel tempo, in modo da poterle riutilizzare quando risultano necessarie per la previsione di eventi futuri.

Lo hidden state, invece, rappresenta la memoria a breve termine. A differenza della cell state, che immagazzina conoscenze più stabili e persistenti, lo hidden state viene aggiornato a ogni passo temporale ed è fortemente influenzato dagli input più recenti. Questa proprietà lo rende particolarmente efficace nel catturare contesti locali e dinamiche di breve periodo all'interno della sequenza di dati. Inoltre, lo hidden state funge anche da output dell'unità LSTM per ciascun passo temporale, trasmettendo le informazioni rilevanti al livello successivo della rete o direttamente all'output finale, a seconda della specifica architettura adottata.
Per ogni passo temporale t le informazioni si compongono di:

- x_t, rappresenta il nuovo pezzo di informazione che si sta considerando
- h_{t-1}, rappresenta l'hidden state all'istante temporale precedente
- c_{t-1}, rappresenta il cell state all'istante temporale precedente

Il primo passo in una rete LSTM è proporre un nuovo candidato per la cell state, calcolato tramite la formula:

$$\tilde{C}_t = \tanh(U_c x_t + V_c h_{t-1} + b_c)$$

Successivamente, viene introdotto il **Forget Gate $\mathbf{f_t}$** [7], il cui scopo è determinare quali informazioni conservare e quali dimenticare dallo stato di memoria precedente:

$$f_t = \sigma(U_f x_t + V_f h_{t-1} + b_f)$$

Poiché la funzione sigmoide restringe i valori tra 0 e 1, il forget gate agisce come un filtro: valori vicini a 1 indicano informazioni da mantenere, mentre valori prossimi a 0 segnalano informazioni da scartare.

Analogamente, si definisce l'**Input Gate $\mathbf{i_t}$**, responsabile di stabilire quali componenti del nuovo candidato $\tilde{C}_t$ debbano essere aggiunte alla memoria:

$$i_t = \sigma(U_i x_t + V_i h_{t-1} + b_i)$$

A questo punto, il nuovo cell state viene aggiornato combinando le informazioni rilevanti del passato con quelle più recenti:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

L'uso del prodotto elemento per elemento permette di pesare ogni componente in base alla sua importanza, determinata dai rispettivi gate.

Dopo aver aggiornato la memoria, è necessario decidere quali informazioni utilizzare per produrre l'output. A tal fine, si impiega l'**Output Gate $\mathbf{o_t}$**:

$$o_t = \sigma(U_o x_t + V_o h_{t-1} + b_o)$$

Questo gate seleziona le informazioni rilevanti dalla cell state da trasmettere all'output. Lo hidden state finale, ovvero l'effettivo output della cella LSTM, si ottiene tramite:

$$h_t = o_t \odot \tanh(C_t)$$

4.3 Applicazione nel Time Series Forecasting

Le reti LSTM si sono affermate come una delle architetture più efficaci per la previsione di serie temporali grazie alla loro capacità di modellare dipendenze a lungo termine e catturare dinamiche temporali complesse nei dati[19]. L'obiettivo principale nel time series forecasting è prevedere il valore futuro di una variabile osservata, basandosi su valori passati. Nel contesto delle serie temporali, l'input tipico di una rete LSTM è una finestra mobile di osservazioni passate. Ogni finestra è un sottoinsieme della sequenza storica, ad esempio:

$$X = x_{t-n}, ..., x_{t-1}$$

dove n rappresenta l'ampiezza della finestra, e x_t è il valore della serie al tempo t. La rete viene addestrata a predire x_t, il valore successivo nella sequenza.
Gli output della rete possono essere generati in due modalità differenti. Nel primo caso si parla di previsioni a *passo singolo*, in cui la rete fornisce una sola predizione futura a partire da una finestra di dati passati. Nel secondo caso si considerano invece le previsioni *multi-step*, in cui viene stimata una sequenza di valori futuri. Quest'ultima modalità può essere implementata secondo due approcci principali: nell'approccio autoregressivo, il valore predetto viene riutilizzato come input per generare le previsioni successive, consentendo al modello di estendere progressivamente l'orizzonte temporale della previsione; nell'approccio diretto, invece, la rete viene addestrata per produrre simultaneamente l'intera sequenza dei valori futuri, evitando la propagazione degli errori accumulati nei passaggi iterativi tipici dell'approccio autoregressivo.

4.3.1 Vantaggi e svantaggi

Come già detto, le reti LSTM rappresentano una delle architetture più utilizzate nel campo del deep learning per l'analisi e la previsione di sequenze temporali. Tuttavia, come ogni tecnologia, anche le LSTM presentano punti di forza e criticità che è fondamentale considerare nella fase di progettazione di un modello predittivo.

I principali vantaggi di questo tipo di rete risiedono nella sua capacità di gestire efficacemente le dipendenze a lungo termine, nella flessibilità nell'elaborazione di dati sequenziali e nella possibilità di modellare dinamiche complesse. Le reti LSTM sono infatti progettate per superare il problema del gradiente che svanisce, tipico delle reti neurali ricorrenti tradizionali, grazie a una struttura a porte che consente di mantenere informazioni rilevanti per lunghi intervalli temporali. Inoltre, a differenza di altri modelli, le LSTM non richiedono ipotesi stringenti sulla stazionarietà dei dati o sulla presenza di una struttura temporale rigidamente definita, risultando quindi adatte a una vasta gamma di serie temporali. Infine, la loro architettura permette di catturare pattern non lineari e interazioni complesse nei dati, rendendole particolarmente efficaci in contesti applicativi quali la previsione meteorologica o finanziaria.

Per eseguire un'analisi accurata nella scelta di una rete, è necessario considerare non solo i vantaggi, ma anche gli svantaggi associati all'utilizzo di architetture di questo tipo. Un primo limite è rappresentato dall'elevata complessità computazionale: l'addestramento di modelli basati su LSTM può risultare estremamente oneroso in termini di risorse, specialmente quando si lavora con dataset di grandi dimensioni o con architetture particolarmente profonde. Un ulteriore aspetto critico riguarda la difficoltà di interpretazione, poiché le LSTM, come molte altre reti di deep learning, vengono spesso considerate modelli black-box[21]. Il processo attraverso il quale tali reti giungono a una determinata previsione non è infatti facilmente interpretabile, a differenza di modelli statistici più semplici che offrono una chiara visibilità sui parametri e sulle relazioni tra variabili. Le LSTM, invece, operano tramite una complessa rete di pesi, attivazioni e aggiornamenti interni, difficili da tradurre in regole esplicite, il che può costituire un limite in contesti in cui è fondamentale comprendere e giustificare le decisioni del modello.

Un ulteriore svantaggio è rappresentato dalla forte sensibilità delle prestazioni alla scelta degli iperparametri, quali il numero di unità, il tasso di apprendimento e la lunghezza delle sequenze. La definizione ottimale di questi valori richiede spesso un processo di tuning complesso e dispendioso. Infine, in presenza di dataset di dimensioni limitate, le LSTM possono incorrere in fenomeni di overfitting[17], adattandosi eccessivamente ai dati di addestramento e riducendo così la loro capacità di generalizzazione.

4.4 Modello univariato one-step ahead

Un primo approccio esplorato nello sviluppo del sistema di previsione ha riguardato la costruzione di un modello **univariato**, incentrato esclusivamente sulla variabile *Close*, ovvero il prezzo di chiusura giornaliero di Ethereum. L'idea di base è quella di impiegare una rete ricorrente di tipo LSTM per effettuare una predizione *one-step ahead*, ossia stimare il valore successivo della serie storica a partire da una finestra temporale di osservazioni passate.

Preprocessing dei dati

La fase preliminare ha previsto un flusso di preprocessing piuttosto canonico ma necessario per assicurare la qualità del processo di addestramento. In particolare, sono stati svolti i seguenti passaggi:

- **Lettura e ispezione del dataset**: il dataset è stato caricato e sottoposto ad una prima analisi esplorativa, così da verificare la coerenza temporale e la presenza di eventuali anomalie. La Figura 4.3 mostra l'andamento complessivo del prezzo di chiusura di Ethereum nel periodo considerato.

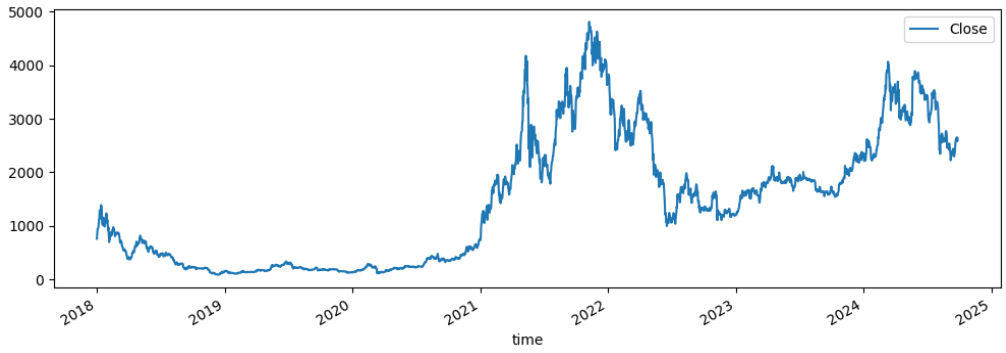

Figura 4.3: Andamento della variabile target di Previsione

- **Identificazione della variabile target**: tra le variabili presenti nel dataset è stata selezionata come target la colonna Close, ritenuta la più significativa ai fini previsionali.
- **Normalizzazione**: per migliorare la stabilità numerica e favorire la convergenza del modello, i valori della variabile target sono stati scalati tramite una trasformazione di tipo MinMax nell'intervallo $[0, 1]$.

Dopo questa fase, sono state generate le **finestre di predizione**. In particolare, il modello è stato configurato per leggere sequenze di **5** punti temporali consecutivi e stimare il valore immediatamente successivo, realizzando così una previsione *one-step ahead*.

Il dataset è stato successivamente suddiviso mediante uno **split temporale** 70/15/15 nei set di training, validation e test[20]. Tale scelta consente di mantenere la natura sequenziale della serie e di valutare le prestazioni del modello su dati cronologicamente successivi a quelli utilizzati in fase di addestramento. Seguendo questa logica il dataset è stato suddiviso nelle finestre di predizione.

Architettura del modello

L'architettura della rete LSTM sviluppata è riportata in Tabella 4.1. Essa è composta da uno strato LSTM principale con 64 unità, seguito da due strati fully connected di cui l'ultimo con singolo neurone di output, adatto ad una previsione scalare.

Layer (type)	**Output Shape**	**Param #**
LSTM (lstm)	(None, 64)	16,896
Dense	(None, 8)	520
Dense (dense)	(None, 1)	9
Total params		**17,425**
Trainable params		**17,425**
Non-trainable params		**0**

Tabella 4.1: Struttura del modello LSTM a finestra breve

Prestazioni del modello

Al termine della fase di training e validazione, sono state generate le predizioni sul set di test, con l'obiettivo di valutare le capacità di generalizzazione del modello. I risultati ottenuti sono riportati in Tabella 4.2.

Metric	**Value**
RMSE	484.95
MAE	394.98
MAPE	93.72%

Tabella 4.2: Prestazioni del modello LSTM one-step ahead a finestra breve

L'analisi dei valori assunti dalle metriche RMSE e MAE mette in evidenza criticità nelle capacità previsionali del modello, risultato ulteriormente confermato dal confronto grafico tra i valori osservati e quelli stimati.

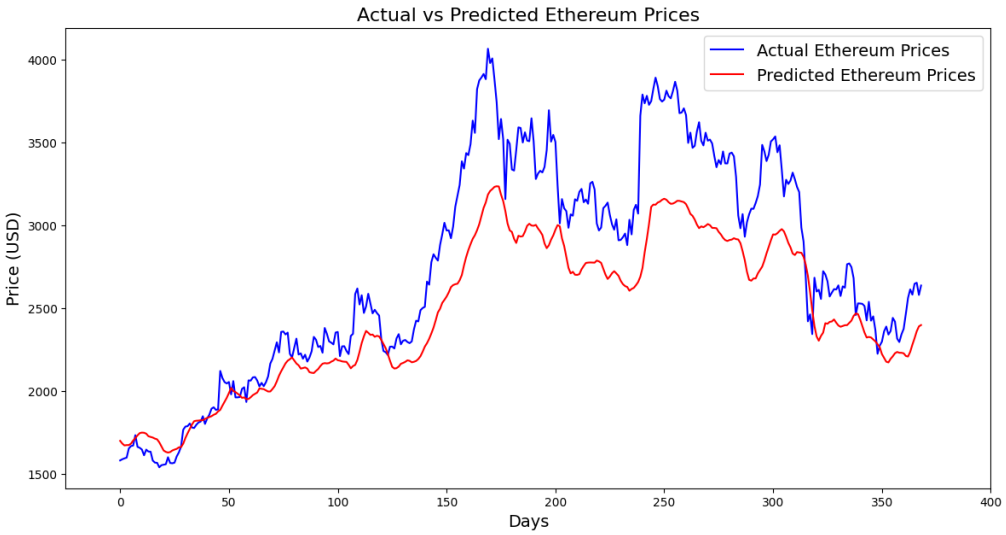

Figura 4.4: Visualizzazione dei valori predetti rispetto ai valori reali con implementazione univariata a finestra breve

Discussione dei risultati

Dall'analisi della Figura 4.4 emergono alcune criticità legate alla qualità delle previsioni. La curva reale e quella predetta mostrano andamenti di crescita e decrescita solo parzialmente simili, ma caratterizzati, nella maggior parte delle osservazioni, da errori di sottostima. Tale sottostima si mantiene costante anche in corrispondenza dei picchi: il modello, infatti, non è in grado di prevederli né di avvicinarsi in modo soddisfacente ai loro valori effettivi.

Si osserva inoltre che i valori predetti presentano una volatilità significativamente inferiore rispetto a quelli reali. Questo comportamento suggerisce che il modello tenda ad approssimare verso la media, piuttosto che fornire previsioni aderenti alle variazioni effettive della serie. Una previsione di questo tipo risulta quindi poco utile sia per individuare l'andamento del trend generale, sia per cogliere eventuali oscillazioni improvvise della serie temporale.

4.5 Modello univariato one-step ahead con finestra lunga

Dopo aver valutato un primo modello *one-step ahead* basato su finestre temporali ridotte (5 osservazioni), si è deciso di sviluppare una seconda implementazione mantenendo la stessa logica di previsione, ma ampliando la finestra di input a **60 valori consecutivi**.

L'idea alla base di questa scelta risiede nella volontà di fornire alla rete un maggior contesto storico: sebbene il primo modello sia riuscito a catturare il trend complessivo della serie, ha mostrato difficoltà nel rappresentare le oscillazioni locali e la volatilità del prezzo di Ethereum. Un'osservazione attenta del dataset, infatti, suggerisce che le dinamiche di medio periodo possano essere meglio comprese da un modello che dispone di un orizzonte di input più esteso.

Preprocessing dei dati

La fase di preprocessing relativa al modello con finestra lunga riprende le stesse operazioni di base già illustrate per il modello *one-step ahead* con finestra breve, ma introduce alcune differenze mirate a migliorare la qualità delle sequenze fornite in input alla rete.

- **Variabile target:** risulta essere nuovamente il prezzo di chiusura giornaliero di Ethereum, essendo la variabile più utile a questo tipo di predizione e per favorire un confronto tra i modelli
- **Normalizzazione:** in questo caso si è scelto di adottare un `MinMaxScaler` con range $(0, 1)$, al posto dello `StandardScaler` utilizzato nella prima implementazione. La motivazione di questa scelta risiede nel fatto che le reti LSTM tendono a convergere più stabilmente quando le feature sono comprese in un intervallo limitato e privo di valori negativi: questo risulta particolarmente utile in presenza di una finestra di input più estesa, in cui l'accumulo di varianza potrebbe altrimenti rallentare l'apprendimento
- **Lunghezza della finestra:** se nel modello precedente si utilizzavano 5 osservazioni per predire il valore successivo, qui la sequenza di input è stata portata a 60 osservazioni. Tale scelta nasce dall'analisi del dataset, che mostra una dinamica caratterizzata da oscillazioni non sempre catturate nel brevissimo periodo: fornire al modello un orizzonte temporale più lungo consente di apprendere pattern di medio periodo e di avere un contesto storico più ricco a supporto della previsione
- **Suddivisione del dataset:** la logica di suddivisione del dataset è stata adattata a questo scenario: mentre il primo modello adottava uno split 70/15/15 (train/-validation/test), nel modello a finestra lunga è stato utilizzato uno **split 80/20** tra training e test, senza introdurre un validation set separato. Questa decisione è stata presa per massimizzare il numero di sequenze a disposizione in fase di

addestramento, compensando il fatto che l'uso di finestre da 60 riduce il numero totale di campioni effettivamente utilizzabili

Architettura del modello

L'architettura adottata per il modello con finestra lunga riprende la struttura generale già impiegata nella prima implementazione, ma viene arricchita per sfruttare al meglio il maggior numero di osservazioni disponibili in input.

Il modello è infatti costruito come una rete LSTM sequenziale, composta da due strati ricorrenti con 50 unità ciascuno: il primo restituisce l'intera sequenza di stati nascosti, il secondo produce un'unica rappresentazione compatta della sequenza, poi elaborata da due livelli `Dense` completamente connessi, l'ultimo dei quali fornisce la predizione finale del prezzo di chiusura.

Questa architettura a due livelli ricorrenti è stata scelta in quanto la finestra di input a 60 osservazioni fornisce un contesto informativo molto più ampio rispetto al caso precedente. L'utilizzo di due strati LSTM consecutivi consente di catturare sia le dipendenze di breve periodo che quelle di medio periodo.

Layer (type)	**Output Shape**	**Param #**
Input Layer	(None, 60, 1)	0
LSTM (50 units, return seq.)	(None, 60, 50)	10,400
Dropout (0.2)	(None, 60, 50)	0
LSTM (50 units)	(None, 50)	20,200
Dropout (0.2)	(None, 50)	0
Dense (25)	(None, 25)	1,275
Dense (1)	(None, 1)	26
Total params		**31,901**
Trainable params		**31,901**
Non-trainable params		**0**

Tabella 4.3: Struttura del modello LSTM con finestra pari a 60 passi temporali

Prestazioni del modello

Dopo l'addestramento, sono state generate le predizioni sui dati di test. Le prestazioni sono state valutate tramite la metrica MSE e con un'analisi sui valori reali e predetti. I risultati principali sono riportati in Tabella 4.4.

Metric	Value
RMSE	102.75
MAE	68.36
MAPE	2.54%

Tabella 4.4: Prestazioni del modello LSTM one-step ahead a finestra lunga

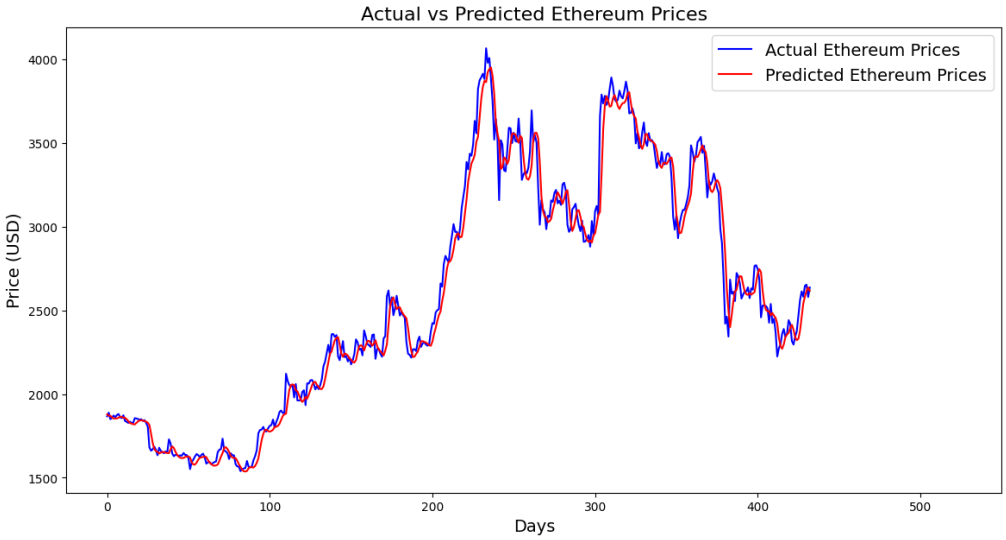

Figura 4.5: Visualizzazione dei valori predetti rispetto ai valori reali con implementazione univariata a finestra lunga

Discussione dei risultati

Dall'analisi della Figura 4.5 emerge invece come l'utilizzo di una finestra di previsione più lunga abbia portato a un miglioramento significativo delle prestazioni del modello. Tale configurazione ha consentito di riprodurre in modo più convincente l'andamento generale del prezzo di chiusura, evidenziando al contempo una buona capacità di generalizzazione. Rispetto al modello basato su una finestra più breve, l'impiego di una finestra di lungo periodo ha permesso di cogliere con maggiore efficacia i movimenti di medio termine della serie. Permangono tuttavia piccoli errori locali in corrispondenza dei picchi di crescita e di discesa, con una tendenza del modello a smussare la volatilità. Nel complesso, le previsioni ottenute risultano utili per descrivere la dinamica di fondo del processo, pur continuando a mostrare alcune limitazioni nella cattura di variazioni improvvise.

4.6 Modello univariato one-step ahead con integrazione di stagionalità

Dopo aver condotto una prima fase di valutazione con modelli univariati, che hanno fornito risultati discreti, si è deciso di proseguire le sperimentazioni introducendo variabili aggiuntive al modello. In particolare, l'attenzione si è concentrata sull'inclusione di feature legate alla stagionalità, al fine di esplorare l'impatto di tali componenti sul miglioramento delle previsioni [24]. Sebbene dall'analisi preliminare del dataset, presentata nel Capitolo 2, non emerga una stagionalità marcata, l'integrazione di variabili stagionali è stata comunque ritenuta utile. Tale scelta si basa sull'ipotesi che anche deboli ricorrenze periodiche o schemi temporali impliciti possano influenzare l'andamento della serie, e che il modello possa trarre beneficio da queste informazioni per una migliore generalizzazione e stabilità predittiva.

Preprocessing dei dati

La fase di preprocessing mantiene parte delle procedure già adottate per il modello one-step ahead a finestra lunga, ma presenta alcune modifiche significative rese necessarie dall'introduzione delle nuove feature stagionali:

- **Variabile target:** anche in questo caso la variabile target è rappresentata dal prezzo di chiusura giornaliero di Ethereum. Tuttavia, poiché il modello dispone ora di ulteriori variabili esplicative, risulta fondamentale definire con precisione la variabile dipendente al fine di garantire la correttezza del processo di addestramento.

- **Aggiunta di feature stagionali:** per incorporare informazioni di natura stagionale, si è fatto riferimento al tempo trascorso dal momento iniziale delle misurazioni dell'andamento del prezzo di Ethereum. Le componenti giorno e mese sono state successivamente trasformate mediante funzioni trigonometriche di seno e coseno, al fine di rappresentare in modo continuo e ciclico la periodicità temporale.

Time	Close	Day sin	Day cos	Year sin	Year cos
2018-01-01	757.22	-2.39×10^{-12}	1.000	0.006193	0.999981
2018-01-02	861.97	-1.26×10^{-11}	1.000	0.023394	0.999726
2018-01-03	941.10	-8.34×10^{-12}	1.000	0.040587	0.999176
2018-01-04	944.83	-4.05×10^{-12}	1.000	0.057769	0.998330
2018-01-05	963.88	2.52×10^{-13}	1.000	0.074934	0.997189

Tabella 4.5: Esempio di porzione del dataframe dopo l'aggiunta delle feature stagionali

- **Normalizzazione:** è stata eseguita solamente sulla variabile target, poichè che *seno* e *coseno* sono valori che per definizione matematica risultano compresi in intervalli $(-1, 1)$ che non necessitano di normalizzazione

- **Generazione delle finestre di previsione:** per consentire l'addestramento del modello, la serie temporale è stata trasformata in un insieme di finestre mobili di lunghezza prefissata. Ogni finestra raccoglie un segmento di osservazioni consecutive e rappresenta l'input del modello, mentre il valore immediatamente successivo costituisce la variabile da prevedere. Poiché i dati comprendono più caratteristiche simultanee (valori di chiusura e indicatori stagionali), ciascuna finestra contiene, per ogni istante temporale, l'intero vettore delle feature disponibili. In questo modo, ogni esempio è formato da una sequenza di vettori come input e da un singolo valore target come output, rendendo possibile la cattura delle dipendenze temporali sia nel tempo sia tra le diverse variabili considerate. La lunghezza delle sequenze è sempre di 60 punti temporali
- **Suddivisione del dataset:** si è adottata nuovamente una suddivisione train/test con rapporto 80/20

Architettura del modello

Il modello previsionale adottato è di tipo sequenziale e si basa sempre su una rete LSTM. L'input del modello è costituito da sequenze di 60 punti temporali e 5 variabili per ciascun passo.

La prima unità LSTM, composta da 50 neuroni e configurata per restituire l'intera sequenza di stati nascosti, consente di catturare le dipendenze temporali a lungo raggio. Per ridurre il rischio di overfitting viene inserito uno strato di dropout con probabilità pari a 0.2.

Segue una seconda unità LSTM, anch'essa con 50 neuroni, che restituisce soltanto lo stato finale, sintetizzando le informazioni più rilevanti della sequenza. Un ulteriore strato di dropout viene applicato per rafforzare la regolarizzazione.

La rete prosegue con uno strato denso intermedio di 25 neuroni, utile per trasformare lo spazio delle rappresentazioni, e termina con uno strato denso a singola unità che produce il valore finale di previsione.

Questa architettura dovrebbe consentire al modello di apprendere sia le dinamiche temporali della serie storica sia le relazioni tra le diverse variabili in ingresso.

Layer (type)	**Output Shape**	**Param #**
LSTM	(None, 60, 50)	11,200
Dropout	(None, 60, 50)	0
LSTM	(None, 50)	20,200
Dropout	(None, 50)	0
Dense	(None, 25)	1,275
Dense	(None, 1)	26
Total params		**32,701**
Trainable params		**32,701**
Non-trainable params		**0**

Tabella 4.6: Architettura del modello LSTM per addestramento con features stagionali

Prestazioni del modello

Dopo l'addestramento, sono state generate le predizioni sui dati di test. Le prestazioni sono state valutate tramite la metrica MSE e con un'analisi sui valori reali e predetti. I risultati principali sono riportati in Tabella 4.7.

Metric	Value
RMSE	132.19
MAE	94.29
MAPE	3.45%

Tabella 4.7: Prestazioni del modello LSTM con features stagionali

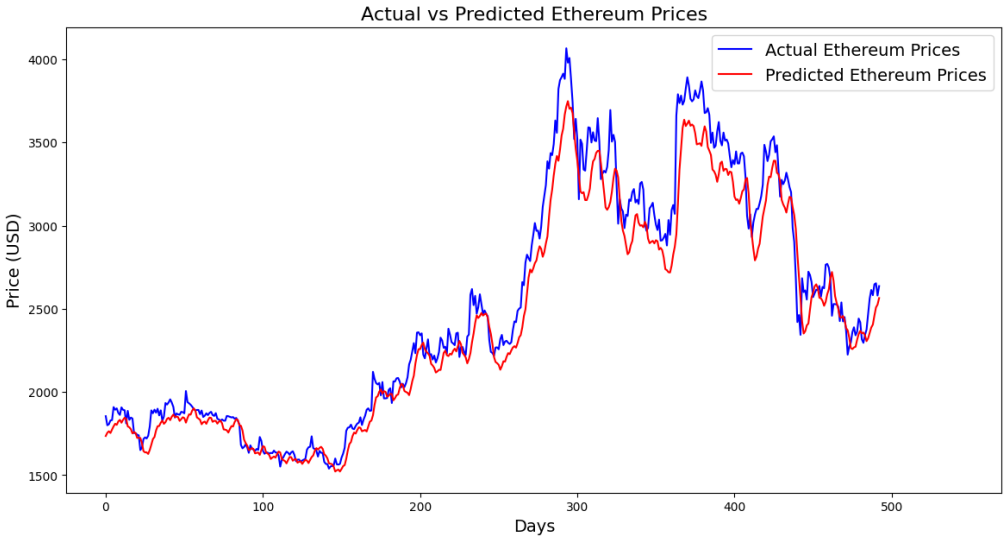

Figura 4.6: Visualizzazione dei valori predetti rispetto ai valori reali con implementazione di features stagionali

Discussione dei risultati

Dall'analisi della Figura 4.6 e dalla valutazione delle metriche riportate in Tabella 4.7 emergono alcune considerazioni significative. Il modello riesce a riprodurre l'andamento del prezzo di chiusura in maniera complessivamente soddisfacente, mostrando una buona capacità di seguire il trend generale della serie. Tuttavia, rispetto al modello univariato con finestra lunga, manifesta maggiori difficoltà nella previsione dei picchi, che tendono a essere sistematicamente sottostimati. Si nota inoltre una riduzione della sovrastima nelle prime cento osservazioni rispetto al modello precedente, segno di un miglior bilanciamento iniziale nella fase predittiva. Nonostante ciò, le previsioni risultano comunque più efficaci nel cogliere la dinamica di fondo del processo piuttosto che le variazioni improvvise. Nel complesso, questo esperimento suggerisce che le prestazioni del modello siano leggermente inferiori rispetto a quelle ottenute con l'approccio precedentemente sviluppato. L'inclusione delle variabili stagionali, in questo caso, non sembra aver apportato miglioramenti significativi, probabilmente a causa della natura stessa del fenomeno analizzato.

Come discusso nel Capitolo 2, infatti, il mercato delle criptovalute difficilmente presenta una stagionalità annua marcata. Di conseguenza, le feature aggiunte hanno avuto un'utilità limitata in questo contesto e, in alcuni casi, potrebbero aver introdotto rumore, riducendo la capacità predittiva complessiva del modello. I risultati ottenuti contraddicono, quindi, l'ipotesi iniziale secondo cui anche deboli ricorrenze temporali, presenti in dataset privi di una marcata stagionalità, potessero contribuire al miglioramento delle prestazioni del modello. Al contrario, l'esperimento ha mostrato che l'inclusione delle feature stagionali non ha apportato benefici significativi, ma anzi, per via della già discussa introduzione del rumore, è andata a ridurre la capacità predittiva complessiva.

Capitolo 5

Conclusioni

Il presente capitolo raccoglie le considerazioni conclusive emerse dal lavoro svolto e propone alcune riflessioni sui possibili sviluppi futuri della ricerca. Dopo aver analizzato e confrontato le prestazioni dei modelli implementati, vengono discusse le implicazioni dei risultati ottenuti. Vengono poi suggerite alcune direzioni di approfondimento che potrebbero ampliare e consolidare il contributo di questo studio nell'ambito della previsione di serie temporali finanziarie.

5.1 Confronto dei risultati

Il confronto tra i modelli sviluppati ha evidenziato differenze significative sia nella natura degli approcci adottati sia nelle prestazioni ottenute. I modelli ARIMA e LSTM si fondano su principi metodologici profondamente diversi: il primo su una formulazione statistica lineare, il secondo su un paradigma di apprendimento automatico in grado di cogliere relazioni non lineari e complesse. Nonostante tali differenze, entrambi i modelli sono stati applicati al medesimo dataset e hanno generato risultati che offrono spunti di riflessione interessanti sull'efficacia dei rispettivi approcci.

Il modello ARIMA ha mostrato ottime prestazioni dal punto di vista quantitativo, in particolare nella versione basata sull'approccio walk-forward, che ha restituito le misure di errore più contenute tra tutti i modelli analizzati. Inoltre, l'implementazione di questo tipo di modello consente di generare, con relativa semplicità, intervalli di confidenza utili per quantificare l'incertezza associata alle previsioni, un aspetto particolarmente rilevante nel contesto finanziario. Inoltre, l'analisi grafica ha evidenziato che, nonostante le metriche risultino favorevoli, il modello tende in alcune occasioni a seguire la serie storica piuttosto che anticiparne i cambiamenti futuri. Le previsioni, come nel caso della configurazione senza particolari metodi di validazione, risultano spesso allineate all'andamento passato, mostrando ritardi nelle variazioni improvvise e una capacità limitata di catturare la volatilità tipica del mercato di Ethereum. Ciò conferma che valori

numerici di errore ridotti non sempre si traducono in previsioni qualitativamente migliori. Nonostante questo, va sottolineato che le predizioni ARIMA possono essere facilmente contestualizzate. Infatti, la semplicità interpretativa del modello rappresenta uno dei suoi principali punti di forza. Nel caso dell'approccio walk-forward, una volta compreso e quantificato il ritardo sistematico delle previsioni, queste possono essere comunque utilizzate in modo efficace, valorizzando l'autoregressività che caratterizza la struttura del modello.

I modelli LSTM, pur mostrando inizialmente prestazioni meno soddisfacenti nella configurazione a finestra breve, hanno evidenziato miglioramenti significativi con l'aumento dell'ampiezza della finestra temporale. L'uso di una finestra lunga ha permesso alla rete di apprendere dipendenze di medio periodo, producendo risultati più coerenti con l'andamento reale dei dati. Questo tipo di implementazione richiede, però, una maggiore quantità di risorse computazionali e un'attenta progettazione architetturale. Difatti, la definizione di un modello coerente con il dataset disponibile risulta un processo complesso. A ciò si aggiunge la scarsa interpretabilità tipica dei modelli basati su deep learning, che rende difficile comprendere in modo diretto il funzionamento interno della rete e il contributo delle singole variabili, soprattutto se confrontato con la trasparenza di un modello statistico come l'ARIMA.

L'integrazione di informazioni stagionali ha inoltre permesso di confutare una delle ipotesi iniziali, secondo cui anche deboli componenti stagionali avrebbero potuto migliorare la qualità delle previsioni. In realtà, tali componenti si sono rivelate fonte di rumore, peggiorando la stabilità delle predizioni e suggerendo che, nel caso specifico del dataset analizzato, la stagionalità non costituisce una variabile significativa. Nonostante ciò, i modelli LSTM hanno raggiunto un buon compromesso tra accuratezza numerica e coerenza grafica delle previsioni, offrendo risultati complessivamente più equilibrati.

Nel complesso, il confronto tra i modelli suggerisce che la valutazione delle prestazioni non possa basarsi esclusivamente sull'analisi numerica delle metriche di errore. Queste ultime rappresentano un indicatore utile ma parziale, poiché non tengono conto della capacità del modello di rappresentare correttamente la forma, la direzione e la coerenza temporale della serie. È quindi opportuno considerare anche aspetti qualitativi, quali la stabilità, la reattività alle variazioni e la robustezza delle previsioni, che nel caso dei modelli LSTM risultano complessivamente più convincenti.

5.2 Sviluppi futuri

Questo lavoro apre a diverse possibilità di approfondimento e miglioramento. Un primo sviluppo naturale riguarda l'integrazione di ulteriori feature esplicative, come indicatori tecnici (media mobile, RSI, volatilità implicita) o variabili esogene legate a fattori macroeconomici e notizie di mercato. Tali informazioni aggiuntive potrebbero fornire al modello una rappresentazione più completa del contesto, migliorando la qualità

delle previsioni. Tutti i modelli sviluppati hanno infatti mostrato alcune difficoltà nel comprendere e gestire la volatilità del dataset, fenomeno tipico dei mercati finanziari e amplificato nel contesto delle criptovalute, dove le variazioni di prezzo risultano spesso repentine e difficilmente prevedibili.

Un secondo ambito di evoluzione riguarda l'ottimizzazione delle architetture di rete. In particolare, l'impiego di modelli più recenti, come le GRU (Gated Recurrent Unit) o le architetture Transformer, potrebbe consentire una gestione più efficiente della memoria a lungo termine e un miglioramento sostanziale della qualità delle previsioni, sebbene a fronte di costi computazionali di addestramento significativamente più elevati, soprattutto nel caso delle architetture Transformer. Queste ultime si stanno affermando come una valida alternativa alle reti LSTM anche nell'ambito delle serie temporali, grazie alla loro capacità di modellare dipendenze globali attraverso meccanismi di attenzione, riducendo al contempo la necessità di processare le sequenze in modo strettamente ricorrente.

Un ulteriore sviluppo riguarda la metodologia di previsione multivariata e multi-step, già accennata nel capitolo Capitolo 1. Un approccio multivariato permetterebbe di stimare simultaneamente l'evoluzione di più variabili correlate, come ad esempio il prezzo massimo e minimo giornaliero, migliorando così anche le previsioni su una singola variabile target. Allo stesso modo, una strategia multi-step consentirebbe di predire più punti temporali futuri in un'unica fase di addestramento, riducendo la tendenza osservata in alcuni modelli a ripetere i valori passati invece di anticiparli. Tali approcci, se opportunamente calibrati, potrebbero incrementare la robustezza del modello e renderlo più adatto a scenari operativi complessi.

Sarebbe inoltre interessante replicare le analisi condotte sulle features di stagionalità utilizzando un dataset che presenti una componente stagionale effettivamente significativa. In tal modo, sarebbe possibile quantificare con maggiore precisione l'impatto di tali componenti sulla qualità delle previsioni e valutare se l'inclusione di variabili stagionali porti effettivamente a un miglioramento delle prestazioni.

Infine, un aspetto di grande interesse riguarda la valutazione economica delle previsioni. Simulare strategie di trading basate sui segnali generati dai modelli predittivi permetterebbe di tradurre i risultati quantitativi delle metriche di errore in indicatori di performance economica, offrendo una prospettiva più concreta e applicativa del valore dei modelli sviluppati. In questo modo, il lavoro potrebbe trovare un riscontro pratico nel complesso e dinamico contesto del mercato delle criptovalute, fornendo strumenti utili non solo per l'analisi teorica ma anche per il supporto decisionale in ambito finanziario.

In conclusione, il lavoro svolto ha mostrato come la combinazione di modelli statistici e tecniche di apprendimento automatico possa offrire un quadro completo per l'analisi e la previsione di serie temporali complesse. L'approccio comparativo adottato ha permesso di individuare punti di forza e limiti di entrambe le famiglie di modelli, ponendo le basi per futuri sviluppi orientati verso sistemi ibridi capaci di coniugare la solidità teorica dei modelli classici con la flessibilità e la potenza rappresentativa del deep learning.

Bibliografia

[1] B. Abraham e J. Ledolter, *Statistical Methods for Forecasting*. John Wiley & Sons, 1983.

[2] D. E. Rumelhart, G. E. Hinton e R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, n. 6088, pp. 533–536, 1986.

[3] R. B. Cleveland, W. S. Cleveland, J. E. McRae, I. Terpenning et al., "STL: A seasonal-trend decomposition," *J. off. Stat*, vol. 6, n. 1, pp. 3–73, 1990.

[4] S. Hochreiter e J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, n. 8, pp. 1735–1780, 1997.

[5] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, n. 02, pp. 107–116, 1998.

[6] J. J. Murphy, *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. Penguin, 1999.

[7] F. A. Gers, J. Schmidhuber e F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural computation*, vol. 12, n. 10, pp. 2451–2471, 2000.

[8] L. R. Medsker, L. Jain et al., "Recurrent neural networks," *Design and applications*, vol. 5, n. 64-67, p. 2, 2001.

[9] P.-F. Pai e C.-S. Lin, "A hybrid ARIMA and support vector machines model in stock price forecasting," *Omega*, vol. 33, n. 6, pp. 497–505, 2005.

[10] C. M. Bishop e N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.

[11] R. J. Hyndman e A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, n. 4, pp. 679–688, 2006.

[12] R. H. Shumway e D. S. Stoffer, *Time series analysis and its applications: with R examples*. Springer, 2006.

[13] R. Mushtaq, *Augmented Dickey–Fuller Test*, Working Paper, 2011.

[14] R. Pascanu, T. Mikolov e Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, Pmlr, 2013, pp. 1310–1318.

[15] G. E. Box, G. M. Jenkins, G. C. Reinsel e G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[16] D. C. Montgomery, C. L. Jennings e M. Kulahci, *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[17] I. Goodfellow, Y. Bengio, A. Courville e Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[18] S. Corbet, A. Meegan, C. Larkin, B. Lucey e L. Yarovaya, "Exploring the dynamic relationships between cryptocurrencies and other financial assets," *Economics letters*, vol. 165, pp. 28–34, 2018.

[19] T. Fischer e C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European journal of operational research*, vol. 270, n. 2, pp. 654–669, 2018.

[20] R. J. Hyndman e G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[21] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.," *Queue*, vol. 16, n. 3, pp. 31–57, 2018.

[22] Y. Yu, X. Si, C. Hu e J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural computation*, vol. 31, n. 7, pp. 1235–1270, 2019.

[23] B. Lim e S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, n. 2194, p. 20 200 209, 2021.

[24] A. Dokumentov e R. J. Hyndman, "STR: Seasonal-trend decomposition using regression," *INFORMS Journal on Data Science*, vol. 1, n. 1, pp. 50–62, 2022.

[25] V. Romanuke, "Arima model optimal selection for time series forecasting," *Maritime Technical Journal*, 2022.

[26] H. M. Sadati, *Ethereum Historical Data 2018–2024*, 2024.