Università degli studi di Padova
Dipartimento di Scienze Statistiche
Corso di Laurea Triennale in

Statistica per le Tecnologie e le Scienze

Relazione Finale

Analisi della volatilità delle principali criptovalute

Relatrice: Prof.ssa Luisa Bisaglia
Dipartimento di Scienze Statistiche

Laureando: Bruno A. Innocente
Matricola N. 1187710

Anno Accademico 2021/2022

Indice

INTRODUZIONE

Il mercato delle criptovalute è sempre più al centro dell'attenzione mondiale e attrae un numero crescente di risorse e investitori. Questo successo è in parte legato all'interesse per la tecnologia sui cui si basano le cripto ovvero la *blockchain*. La "catena di blocchi" ha trovato applicazione nel mondo delle aziende in quattro ambiti: scambio di valore, verificabilità dei dati, coordinamento dei dati e realizzazione di processi affidabili (Portale 2021).

Tuttavia, il principale utilizzo che viene fatto delle criptovalute è di tipo speculativo. Questi asset innovativi sono oggetto di un crescente interesse finanziario e necessitano di strumenti per comprendere appieno le loro caratteristiche, soprattutto quelle non direttamente osservabili come la volatilità. Lo studio di queste attività finanziarie è importante per gli investitori considerato anche che il rischio di bolla finanziaria nel settore del *Web3* è molto concreto. Gli ultimi cinque anni hanno rivoluzionato il mondo delle criptovalute. In letteratura sono stati fatti diversi tentativi per modellare le serie finanziarie delle cripto, ad esempio Zia-Rehman et al. (2020). Tuttavia, eventi importanti hanno influenzato ampiamente il mercato delle monete digitali. Perciò è opportuno verificare se i risultati dei modelli ottenuti in passato siano validi anche ora, dopo questi cambiamenti.
In questa relazione vengono prese in considerazione le tre più importanti criptovalute per capitalizzazione di mercato: bitcoin, ether e binance coin. Il primo capitolo contiene l'analisi descrittiva e preliminare delle monete digitali, la descrizione dei principali eventi storici e di attualità legati al mercato delle cripto. Inoltre, viene riportata una breve riflessione riguardo la correlazione tra bitcoin, oro fisico e i principali indici della volatilità del mercato azionario americano (S&P500 e VIX). Nel secon-

do capitolo vengono esposti i principali modelli parametrici utilizzati per modellare la media e la volatilità dei rendimenti. In particolare, per la volatilità, si considerano i modelli della famiglia GARCH quali: il GARCH standard, l'EGARCH, il TGARCH e il PGARCH. Per la selezione dei modelli, vengono utilizzate diverse distribuzioni di probabilità per identificare quella che meglio descrive la volatilità delle criptovalute. Viene dimostrato che i migliori modelli ottenuti sono quelli che utilizzano distribuzioni a code pesanti come la t di Student. I modelli selezionati vengono testati, nel capitolo 3, attraverso previsioni e stime del VaR (*Value-at-Risk*) utilizzando come *benchmark* di riferimento il modello *RiskMetrics*. Nel quarto e ultimo capitolo vengono riassunti i risultati principali, prestando particolare attenzione ai modelli che hanno ottenuto i risultati migliori.

Capitolo 1

Analisi preliminari

1.1 Descrizione delle criptovalute e principali eventi storici

La storia delle criptovalute inizia nel 2008, quando una personalità anonima, sotto lo pseudonimo di Satoshi Nakamoto, pubblica un articolo in cui viene descritta la più famosa e celebre valuta digitale: il bitcoin (Nakamoto 2008). Questa moneta virtuale, rivoluzionaria secondo il suo creatore, si basa su una tecnologia in grado di gestire pagamenti elettronici senza la necessità di alcun intermediario finanziario. La nascita di bitcoin avviene subito dopo lo scoppio della crisi dei mutui *sub-prime* e nel pieno della 'Grande Recessione'. Questi due eventi hanno creato un clima di sfiducia nelle istituzioni finanziarie favorendo l'ascesa delle criptovalute fino ad oggi.

Una definizione esaustiva di criptovaluta viene fornita dalla Banca d'Italia ed è riportata di seguito: «Le "cripto-attività" tipo bitcoin, dette anche "valute virtuali" sono "una rappresentazione di valore digitale che non è emessa o garantita da una banca centrale o da un ente pubblico, non è necessariamente legata a una valuta legalmente istituita, non possiede lo status giuridico di valuta o moneta, ma è accettata da persone fisiche e giuridiche come mezzo di scambio e può essere trasferita, memorizzata e scambiata elettronicamente"» (Caponera e Gola 2019) .
Sebbene le cripto nascano come alternativa alla moneta tradizionale per

pagamenti elettronici, il loro ruolo è stato principalmente speculativo.

Il processo per creare alcune monete digitali è chiamato *mining* e, dal punto di vista energetico, è molto costoso. Alcuni paesi, avendo prezzi dell'energia relativamente bassi, sono diventati dei riferimenti per "minatori di criptovalute". Inizialmente la Cina era uno di questi paesi. Le operazioni di estrazione erano così avanzate da favorire attività illegali e mettere in difficoltà il settore energetico cinese (*China declares all crypto-currency transactions illegal* 2021). Dopo che il governo di Pechino dichiarò illegali le cripto e ogni attività ad esse connessa, molti minatori si spostarono in Kazakhstan. Anche qui i prezzi dell'energia aumentarono a tal punto da favorire lo scoppio delle rivolte per la crisi energetica (Corinto 2022). Questi eventi hanno provocato un aumento della volatilità del prezzo di molte criptovalute. Prima dello scoppio della guerra, l'Ucraina si era offerta di diventare il paese leader per lo sviluppo delle criptovalute (*L'Ucraina vuole puntare sulle criptovalute* 2021). Tuttavia, a causa del conflitto attualmente in corso con la Russia, utilizza monete digitali per finanziare sia le sue attività belliche che quelle di soccorso agli sfollati (*Crypto war aid for Ukraine: Are donations in Bitcoin an innovation or just a sideshow?* 2022, Winkie 2022). Il colosso americano PayPal non ha potuto ignorare la crescita delle cripto e ha iniziato ad accettare pagamenti in valuta digitale (Soldavini 2020). Nel 2021, per la prima volta nella Storia, Coinbase, un'azienda *exchange* di criptovalute, viene quotata in borsa (Soldavini 2020). Grandi istituzioni finanziarie come la BCE e il governo cinese studiano le criptovalute per adottare una versione alternativa di moneta digitale delle valute legali (*Un euro digitale* 2022, *Cina: la grande crescita silenziosa dello Yuan digitale* 2022).

1.2 Criptovalute analizzate

Bitcoin (BTC) è la valuta digitale più famosa in assoluto. Fu ideata nel 2008 da uno o più individui sotto lo pseudonimo di Satoshi Nakamoto e non è legato ad alcuna banca. Per tenere traccia di tutte le sue transazioni viene utilizzata la tecnologia *blockchain*. Bitcoin nasce come sistema di pagamento alternativo, sebbene gli esperti finanziari lo considerino un

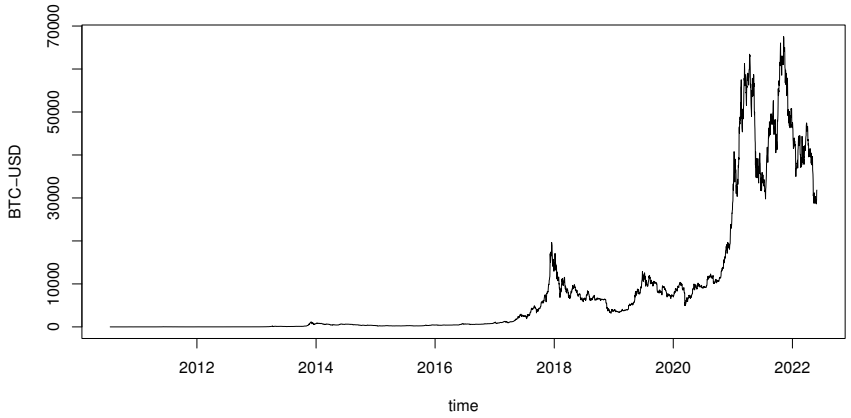
asset di riserva di valore come l'oro, ma ad alta volatilità.

Ethereum è un sistema di *blockchain open-source* decentralizzato dotato di una propria criptovaluta, ether (ETH). Ethereum funziona come piattaforma per numerose altre criptovalute e per l'esecuzione di contratti automatizzati decentralizzati (*Che cos'è Ethereum?* 2022).

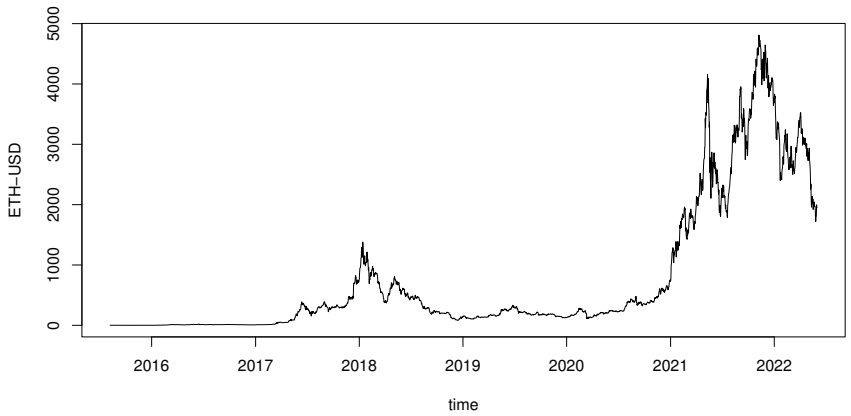
Binance Coin (BNB) è una criptovaluta creata dall'omonimo *exchange* cinese Binance che ad oggi è anche l'*exchange* più grande al mondo per volume di transazioni giornaliero. BNB può essere utilizzato sia come metodo di pagamento, sia come *token* di utilità per pagare le commissioni sullo scambio di Binance che per la partecipazione alle vendite di *token* come lancio iniziale di Binance. BNB alimenta anche il Binance DEX (scambio decentralizzato) (*Che cos'è Binance Coin?* 2022).

Le tre criptovalute, prese in considerazione in questa relazione, sono state scelte in base alla capitalizzazione di mercato al 04/03/2021. Al primo e al secondo posto si trovano rispettivamente bitcoin ed ether. Al terzo posto c'è binance coin. In realtà ci sarebbe dovuta essere lo *stablecoin* tether. Gli *stablecoin* sono *crypto asset* (in particolare sono *token*) che hanno due caratteristiche importanti: la prima, in quanto *token* e quindi *asset* digitali, i loro scambi sono registrati su una *blockchain*, esattamente come accade per gli scambi di bitcoin. La seconda e principale particolarità, da cui prendono il nome, è il fatto che il loro valore sia "ancorato" ad un *asset* o un bene reale, come il dollaro, l'euro oppure l'oro. Per esempio, un *token* vale un dollaro (Marazzina 2022). Proprio a causa della sua natura "stabile" si è deciso di non includere tether nello studio, in quanto la sua volatilità è per natura ridotta. Le serie dei rendimenti considerate iniziano a maggio 2017, e terminano a fine maggio 2022 (Periodo di riferimento delle serie dei rendimenti: dal 01/05/2017 al 31/05/2022). Questo intervallo temporale è stato scelto per diversi motivi. Da maggio 2017 la capitalizzazione del mercato delle criptovalute è cresciuta rapidamente causando cambiamenti sostanziali nella maggior parte delle cripto più importanti, in particolare per bitcoin ed ether (Hardle et al. 2020). Questo cambiamento nel comportamento

delle serie si può osservare nel grafico 1.1, figure a) e b). A questo periodo risale anche la pandemia di Sars-Cov-2 che ha causato un generale aumento della volatilità dei principali mercati finanziari. La terza criptovaluta (BNB) è nata nel giugno 2017, ma i primi dati aprezzabili partono da settembre 2017. Anche considerando la serie di binance coin dall'origine, la sua capitalizzazione di mercato è aumentata così rapidamente che la serie stessa può essere studiata assieme ad ether e bitcoin.



(a) *Andamento del tasso BTC-USD*



(b) *Andamento del tasso ETH-USD*

Figura 1.1: Andamento dei tassi di cambio con USD delle due principali criptovalute.

1.3 Analisi della letteratura

Le questioni più dibattute nella letteratura degli ultimi cinque anni sulle criptovalute riguardano: la similiarità tra bitcoin e oro, lo stabilire se bitcoin può essere considerato un bene rifugio e la ricerca dei migliori modelli per la gestione del rischio e per le previsioni delle più importanti criptovalute. Chu et al. (2017) hanno adattato diversi modelli della famiglia GARCH per sette criptovalute: bitcoin, dash, dogecoin, litecoin, maidsafecoin, monero e ripple. Basandosi sui criteri d'informazione automatici, come per esempio quello di Akaike e sui risultati del *Value-at-Risk* (VaR), gli autori dimostrano che, sui dati da loro utilizzati, il miglior modello per le cripto sia il GJRGARCH. Inoltre, gli autori sostengono che le cripto monete siano da considerarsi investimenti rischiosi e non mezzi di pagamento. Zia-Rehman et al. (2020) hanno adattato i modelli GARCH, EGARCH, TGARCH e APGARCH con distribuzione normale e t di Student alle otto più importanti cripto (comprese bitcoin ed ether). Il miglior modello che hanno identificato per ether è l'EGARCH, mentre per il bitcoin è l'APGARCH. In entrambi i casi con distribuzione t di Student. Gyamerah (2019), con uno studio incentrato solo sul bitcoin, conferma l'impatto asimmetrico degli shock sulla volatilità. Il miglior modello che trova per descrivere la volatilità di bitcoin è il TGARCH con distribuzione NIG. Diversi studi riguardano, invece, il legame tra oro e bitcoin. Gkillas e Longin (2019) sostengono che il bitcoin sia effettivamente il nuovo oro digitale. Il metallo prezioso - affermano - è ancora una buona risorsa nella gestione del rischio. Se si dovesse scegliere tra oro e bitcoin essi spiegano che una combinazione di entrambe si rivelerebbe la migliore soluzione per la gestione del rischio in un portafoglio. Mentre Kyriazis (2020) afferma che bitcoin non ha ancora le caratteristiche per essere un bene rifugio come l'oro. Le proprietà di copertura (*hedging*) del bitcoin, contro i mercati del petrolio e delle azioni, sono buone, anche se inferiori all'oro. Con il metallo prezioso, bitcoin ha una correlazione bassa o negativa. Approssimativamente il bitcoin può essere considerato un *asset* via di mezzo tra l'oro e il dollaro americano. López-Cabarcos et al. 2021 hanno cercato di trovare una variabile esogena per descrivere la volatilità di bitcoin usando *proxy* della volatilità dei mercati come: S&P500, VIX

e un indicatore del sentimento degli investitori utilizzando i messaggi del *social network* Stocktwits. Dalle loro analisi emerge che il bitcoin è un titolo *safe heaven* e che la variabile esogena migliore per la volatilità di bitcoin è il sentimento degli investitori riguardo al mercato azionario, piuttosto che la volatilità dello stesso mercato. Negli utlimi anni si sta diffondendo la previsione della volatilità tramite metodi di *machine learning*. Shen et al. (2021) hanno confrontato tramite previsioni e stime del VaR, modelli della classe GARCH e una rete neurale RNN, utilizzando come *benchmark* il modello *RiskMetrics*. Viene confermato che i migliori modelli per la gestione del rischio sono quelli di tipo GARCH.

1.4 Statistiche descrittive

I dati utilizzati in questo studio sono i tassi di cambio presi in chiusura per BTC-USD, ETH-USD e BNB-USD. Per ETH e BTC si sono utilizzati i dati disponibili sul sito Coinmetrics (*Coinmetrics* 2022), mentre per ottenere i tassi di BNB, si è fatto uso dei dati di Yahoo Finance (*Yahoo Finance* 2022) scaricabili tramite il *software* R (*The R Project for Statistical Computing* 2022). La serie di interesse è quella dei rendimenti logaritmici, ottenibile dai prezzi in chiusura tramite la seguente formula:

$$r_t = log\left(\frac{P_t}{P_{t-1}}\right)$$

Osservando la tabella 1.1, si può notare come il rendimento medio considerato abbia un valore maggiore per la criptovaluta più giovane BNB. Da questi dati, si può anche intuire l'allontanamento dall'ipotesi di gaussianità osservando la curtosi. Tenendo conto dell'eccesso di curtosi e della deviazione standard, la criptovaluta più volatile sembra essere BNB.

	nobs	Min	Max	Mean	Median	St. dev	Skewness	Kurtosis
BTC	1856	-0.47056	0.22405	0.00167	0.00214	0.04217	-0.66976	10.4903
ETH	1856	-0.56561	0.30062	0.00173	0.00177	0.05459	-0.56153	8.6789
BNB	1664	-0.54308	0.52922	0.00305	0.00115	0.06099	0.38553	14.0762

Tabella 1.1: Statistiche descrittive delle principali criptovalute.

Questo viene confermato, anche intuitivamente, osservando i grafici delle serie dei tassi di cambio e dei rendimenti in figura 1.2. Un'altra indicazione di un comportamento non gaussiano sulle code della distribuzione si osserva nei QQ-plot in figura 1.4 e nella rappresentazione della densità empirica confrontata con la normale in figura 1.3.

Per quanto riguarda i test statistici, il test di Kolmogorov-Smirnov conferma che i rendimenti non si distribuiscono secondo una distribuzione normale (i risultati del test in tabella 1.2).

I grafici dell'ACF e PACF dei rendimenti al quadrato mostrano la presenza di effetti ARCH in tutte tre le serie considerate assieme ai test di Ljung Box e dei Moltiplicatori di Lagrange (LM) riportati rispettivamente nelle tabelle 1.3 e 1.4.

	BTC	ETH	BNB
Statistic	0.4476	0.4346	0.4333
n°obs	1856	1856	1664
p-value	0	0	0

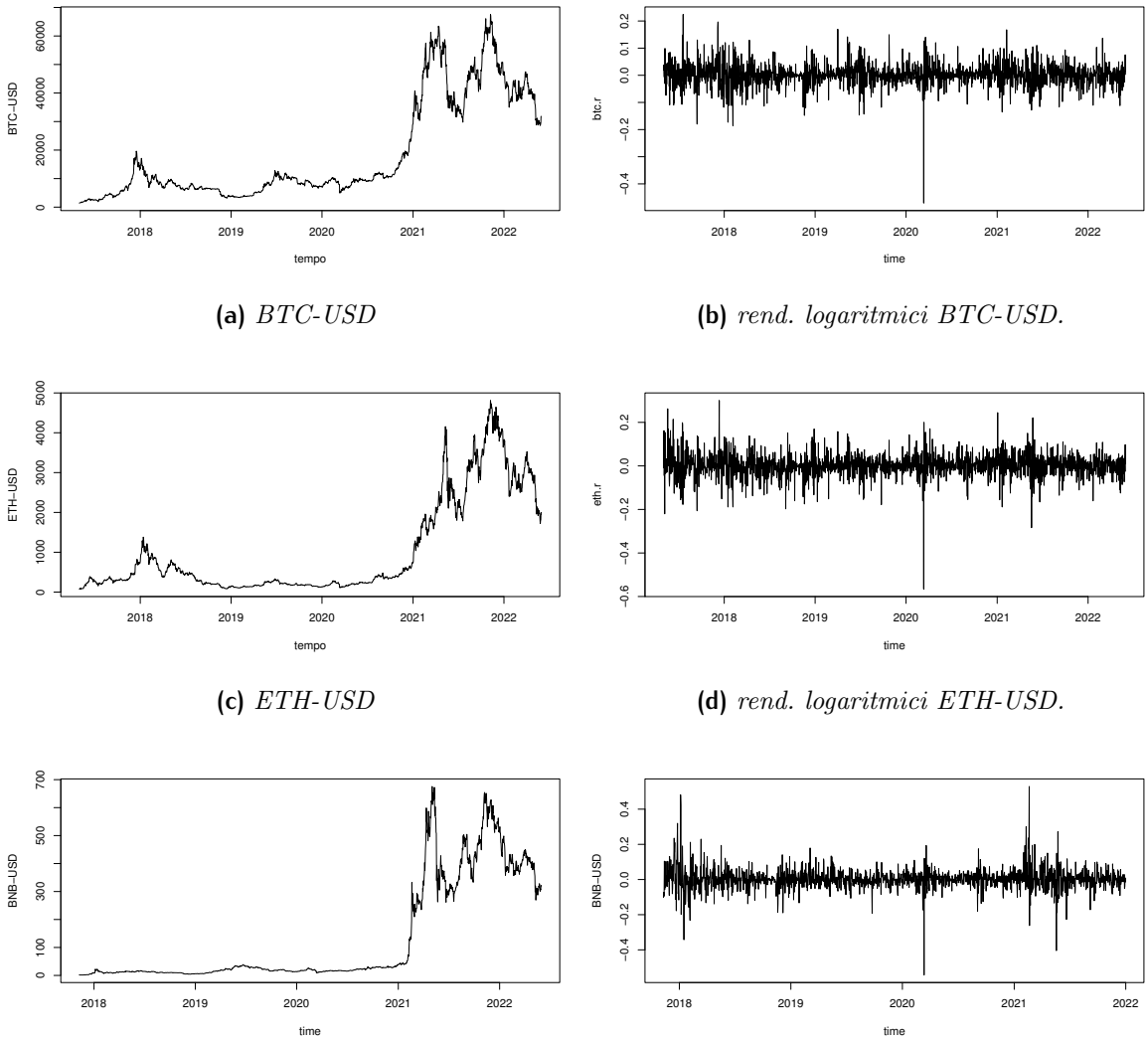
Tabella 1.2: Test di Kolmogorov-Smirnov per la verifica della normalità dei rendimenti.

	BTC	ETH	BNB
Statistic	38.364	65.733	180.88
lag	8	8	8
p-value	6.449e-06	3.457e-11	0

Tabella 1.3: Test di Ljung-Box sui rendimenti al quadrato.

	BTC	ETH	BNB
Statistic	33.699	56.246	231.3
df	12	12	12
p-value	0.0007524	1.081e-07	0

Tabella 1.4: ARCH LM-test. Ipotesi nulla: no effetti ARCH.



(a) *BTC-USD*

(b) *rend. logaritmici BTC-USD.*

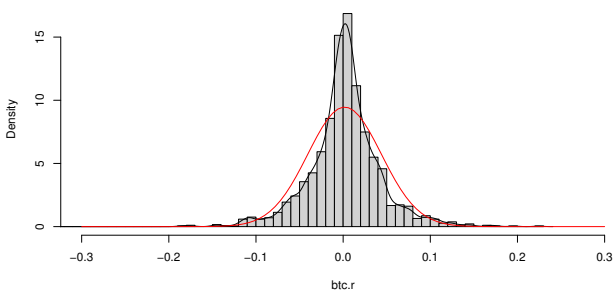
(c) *ETH-USD*

(d) *rend. logaritmici ETH-USD.*

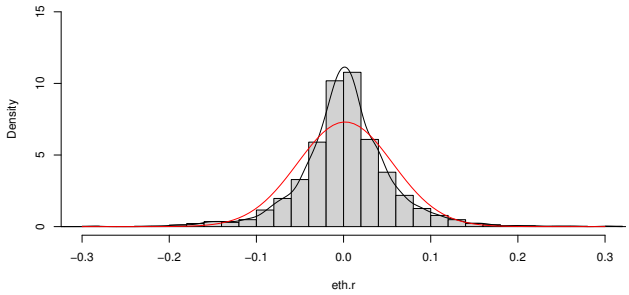
(e) *BNB-USD*

(f) *rend. logaritmici BNB-USD.*

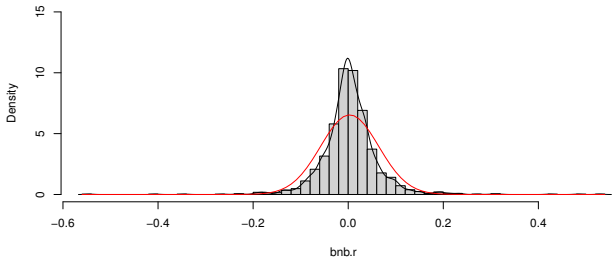
Figura 1.2: Grafici dell'andamento del tasso di cambio e dei relativi rendimenti (da sinistra a destra) per BTC, ETH e BNB (dall'alto verso il basso).



(a) *BTC-USD*

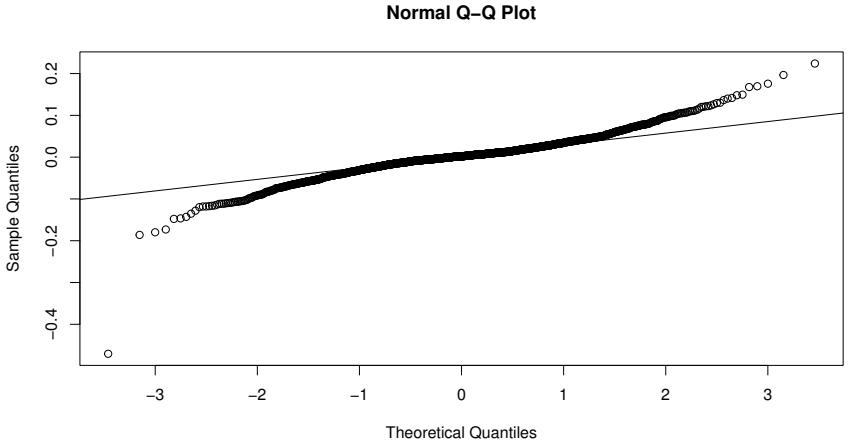


(b) *ETH-USD*

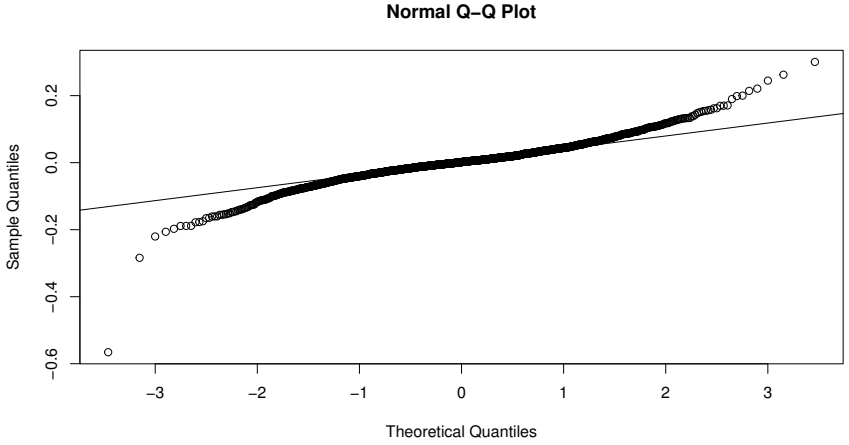


(c) *BNB-USD*

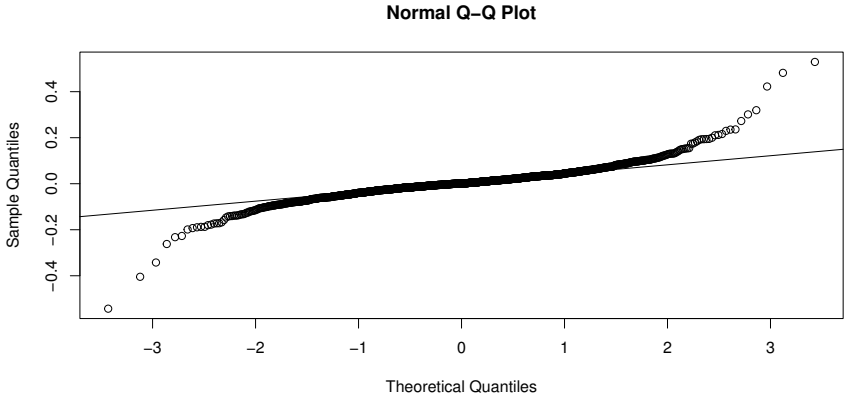
Figura 1.3: Grafici delle densità empiriche: la curva in **rosso** rappresenta la densità della normale, la curva in **nero** rappresenta la stima kernel della densità.



(a) *BTC qqplot.*

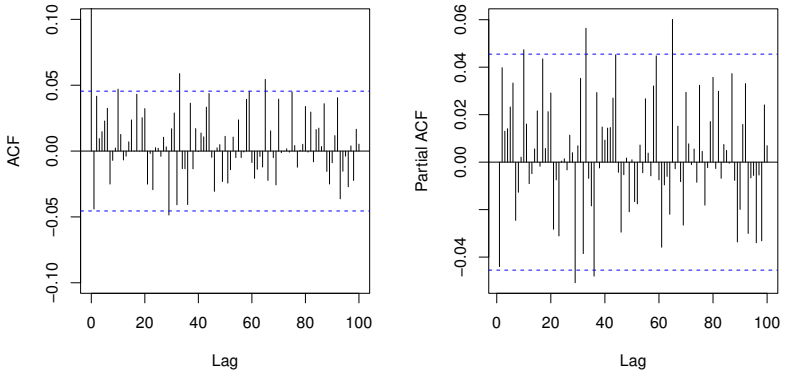


(b) *ETH qqplot.*

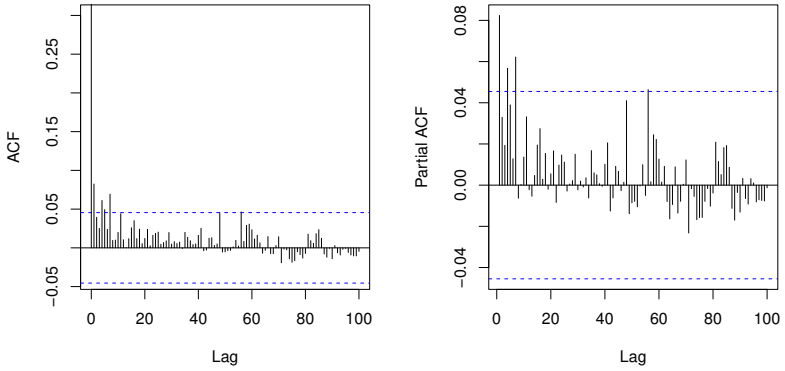


(c) *BNB qqplot.*

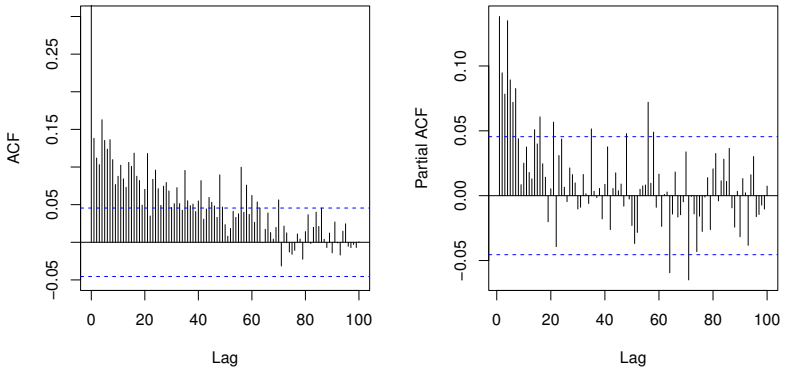
Figura 1.4: QQ-plot dei rendimenti logaritmici.



(a) *ACF e PACF dei rendimenti di BTC-USD.*

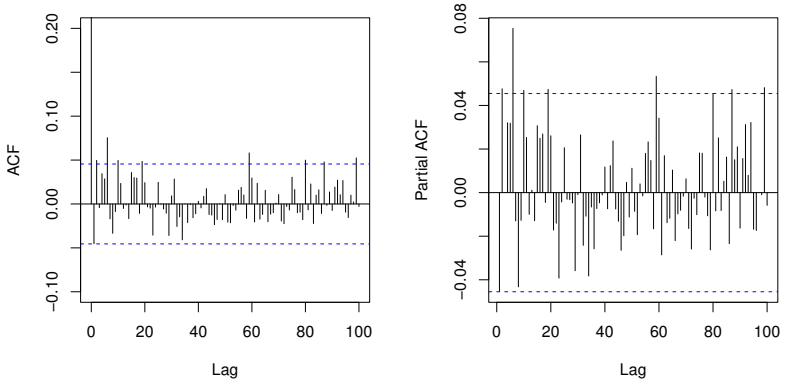


(b) *ACF e PACF dei rendimenti al quadrato di BTC-USD.*

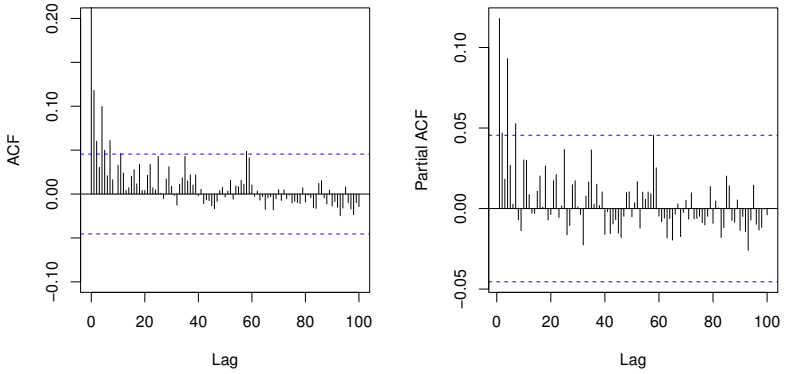


(c) *ACF e PACF dei rendimenti in valore assoluto di BTC-USD.*

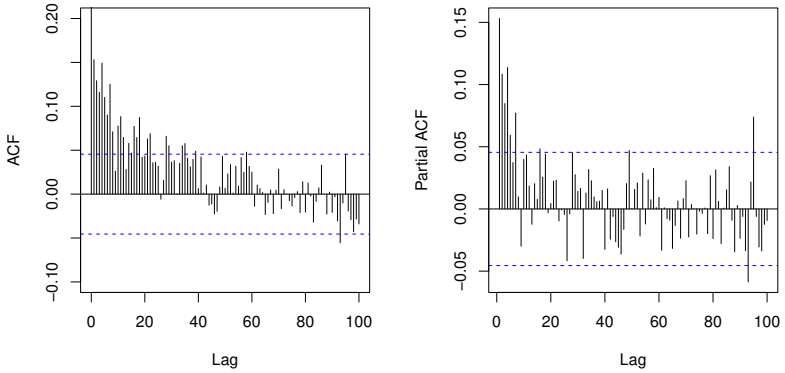
Figura 1.5: ACF e PACF dei rendimenti di BTC-USD.



(a) *ACF e PACF dei rendimenti di ETH-USD*

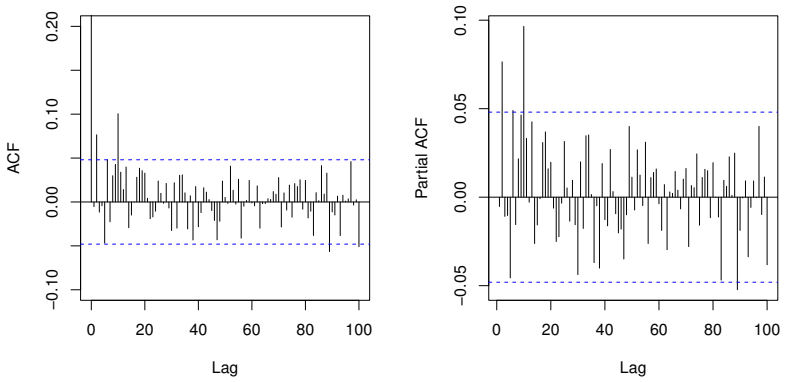


(b) *ACF e PACF dei rendimenti al quadrato di ETH-USD.*

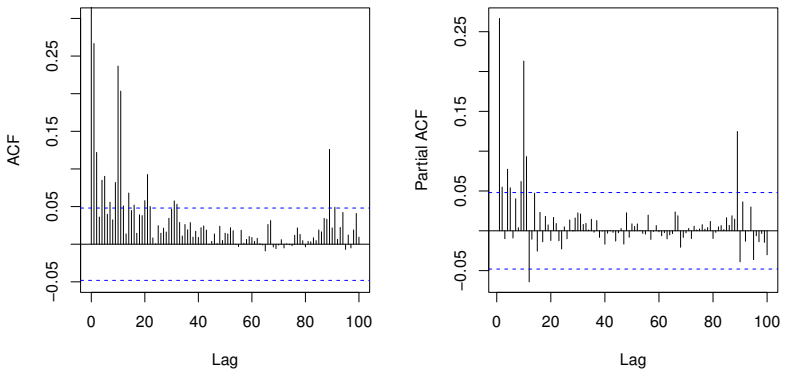


(c) *ACF e PACF dei rendimenti in valore assoluto di ETH-USD.*

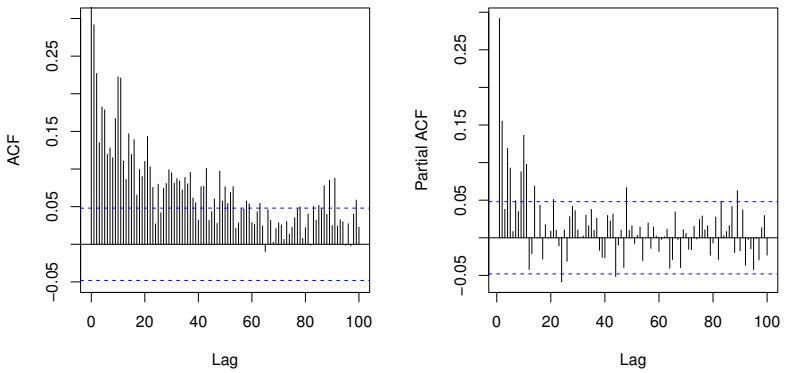
Figura 1.6: ACF e PACF dei rendimenti di ETH-USD.



(a) *ACF e PACF dei rendimenti di BNB-USD.*



(b) *ACF e PACF dei rendimenti al quadrato di BNB-USD.*



(c) *ACF e PACF dei rendimenti in valore assoluto di BNB-USD.*

Figura 1.7: ACF e PACF dei rendimenti di BNB-USD.

1.5 Bitcoin un caso particolare

Bitcoin è la cripto-attività più studiata. Per capire il suo comportamento, molti esperti hanno provato a paragonarlo a diversi strumenti finanziari o indici. Viene spesso accostato all'oro per le sue caratteristiche intrinseche: difficile da estrarre, non può essere prodotto arbitrariamente, ed è disponibile in quantità limitata (Hardle et al. 2020). Anche psicologicamente il bitcoin viene considerato da una parte degli investitori come una riserva di valore in periodi di grande volatilità dei mercati. La letteratura si è dimostrata molto divisa sotto questo punto di vista con alcuni studi che sostengono la similarità con il metallo prezioso (Gkillas e Longin 2019), mentre altri no (Hardle et al. 2020). In questo studio, per il periodo considerato, si è analizzata la correlazione tra i rendimenti di BTC, VIX, oro e SP500. I grafici 1.8 e 1.9 dimostrano che bitcoin ha una bassa correlazione con gli *asset* presi in considerazione nel periodo di riferimento. Questo indica che bitcoin e oro sono strumenti complementari e la loro presenza combinata può essere molto utile per la diversificazione di un portafoglio di titoli.

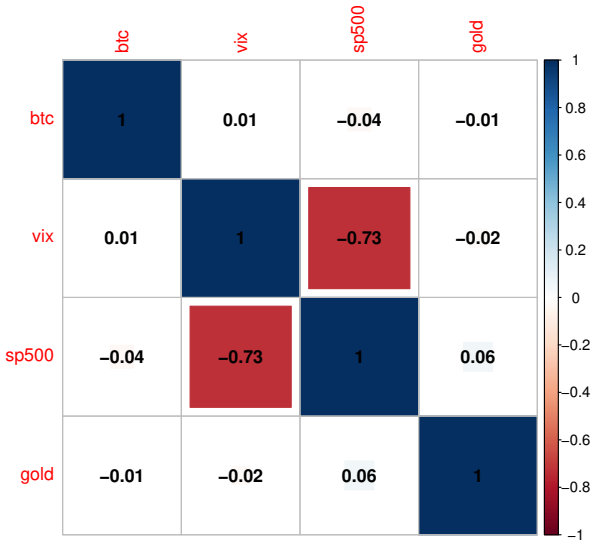


Figura 1.8: Correlazione del bitcoin con altri *asset* nel periodo di riferimento.

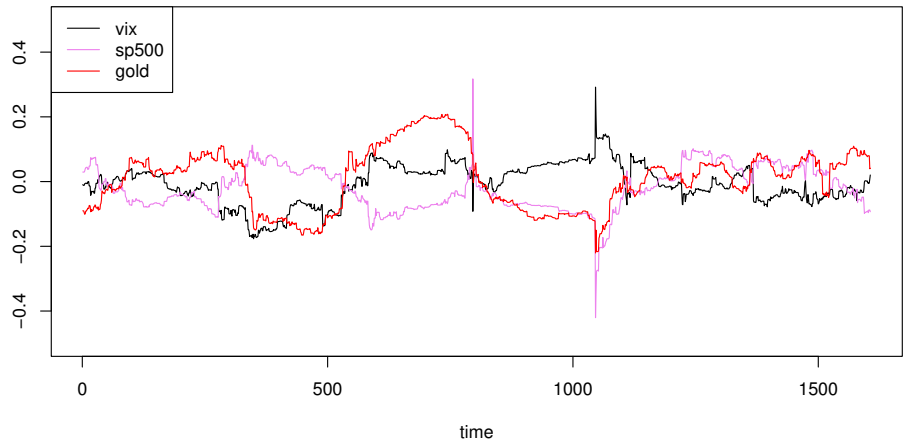


Figura 1.9: Correlazione giornaliera del rendimento BTC-USD con S&P500, oro e indice VIX. *Rolling window* di 250 giorni.

Capitolo 2

Modellazione con modelli di tipo GARCH

2.1 Modelli GARCH

2.1.1 Equazione della media

Per modellare la media condizionata si sono utilizzati i modelli della classe ARMA, rappresentabili con la seguente equazione generale:

$$\mu_t = \phi_0 + \sum_{i=1}^{p} \phi_i r_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j}$$

Si sono presi in considerazione modelli ARMA(p, q) per la media selezionati in base alla bontà di adattamento (AIC e BIC) e alla significatività dei parametri (con p e q al massimo pari a 3). Per bitcoin, il modello migliore si è rivelato essere un AR(1), per ether ARMA(3,3) e per binance MA(2).

2.1.2 GARCH

Il modello GARCH è stato proposto da Bollerslev (1986). La formulazione del modello è così specificata:

$$r_t = \mu_t + \varepsilon_t = \mu_t + \sigma_t \epsilon_t$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2$$

con le seguenti restrizioni sui parametri:
$\omega > 0$, $\alpha_i \geq 0$, $\beta_j \geq 0$ e $\sum \alpha_i + \sum \beta_j < 1$

Il modello GARCH(1,1) è caratterizzato dalla seguente equazione della volatilità:

$$\sigma_t^2 = \omega + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2$$

con $\omega > 0$ e $\alpha_1, \beta_1 \geq 0$

2.1.3 EGARCH

Il modello Exponential GARCH è stato introdotto da Nelson (1991). Una sua possibile specificazione, come riportata in Brooks (2008), è la seguente:

$$\ln(\sigma_t^2) = \omega + \alpha \left(\frac{|\varepsilon_{t-1}|}{\sqrt{\sigma_{t-1}^2}} - \sqrt{\frac{2}{\pi}} \right) + \beta \ln(\sigma_{t-1}^2) + \gamma^* \frac{\varepsilon_{t-1}}{\sqrt{\sigma_{t-1}^2}}$$

dove $\gamma^* = \alpha\gamma$

Il modello EGARCH ha principalmente due vantaggi. Il primo riguarda l'equazione della volatilità che, essendo specificata in termini di logaritmo della varianza condizionata, fa in modo che ai parametri non debba essere imposto alcun vincolo. Infatti, la trasformazione esponenziale garantisce la non negatività della varianza. Inoltre, questo modello permette di considerare le assimetrie tramite il parametro γ^*. Se $\gamma^* \neq 0$, l'EGARCH coglie l'asimmetria degli effetti sulla volatilità, provocati da shock con segni differenti.

2.1.4 TGARCH

Introdotto da Zakoian (1994), questo modello permette di tenere in considerazione l'effetto *leverage*, (effetto leva). Quest'ultimo sostiene che ci sia un impatto asimmetrico degli shock sulla volatilità di un titolo e che, a parità di grandezza, gli shock negativi abbiano un impatto maggiore degli shock positivi sulla volatilità. L'equazione della volatilità nel modello Threshold GARCH (1,1) è così specificata:

$$\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \gamma I_{t-1} \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2$$

dove

$$I_{t-1} = \begin{cases} 1 & \text{se } \varepsilon_{t-1} < 0 \\ 0 & \text{se } \varepsilon_{t-1} \geq 0 \end{cases}$$

quindi l'impatto degli shock negativi $(\alpha + \gamma)$ è maggiore dell'impatto degli shock positivi (α). Se $\gamma > 0$ si osserva l'asimmetria, mentre se $\gamma = 0$ shock positivi e negativi hanno lo stesso impatto sulla volatilità.

2.1.5 PGARCH

Il modello PGARCH è stato introdotto da Ding et al. (1993). L'equazione della volatilità del modello è così specificata:

$$\sigma_t^\delta = \omega + \sum_j^q \alpha_j (|\varepsilon_{t-j}| - \gamma_j \varepsilon_{t-j})^\delta + \sum_i^p \beta_i \sigma_{t-i}^\delta$$

con $\omega > 0$, $\alpha_j \geq 0$, $\beta_i \geq 0$, $\gamma_j \in (-1, 1)$ parametro di asimmetria e $\delta > 0$ parametro di potenza. Al variare di δ si ottiene un comportamento diverso del modello:

- se $\delta = 2$ si ha un GARCH standard, con l'aggiunta dell'effetto leva (TGARCH);
- se $\delta = 1$ si ha un modello per σ_t, e, se $\gamma = 0$, il modello è riconducibile al TS-GARCH (Schwert 1989);
- δ può essere fissato a priori oppure può essere stimato attraverso la massima verosimiglianza.

2.1.6 Scelta della distribuzione degli errori

Per decidere la distribuzione degli errori nei vari modelli di tipo GARCH, si sono adattati tutti i modelli sopra elencati con varie distribuzioni tipiche della finanza (Normale, Skew Normal, t di Student, Skew Student, Generalized Error Distribution, Skew GED, Normal-Inverse Gaussian e Generalized Hyperbolic). La scelta è ricaduta sulle due distribuzioni generalmente in grado di fornire i migliori risultati in base ai criteri di informazione AIC (Akaike Information Criteria) e BIC (Bayesian

Information Criteria). Questi due criteri sono così definiti:

$$AIC = 2d - 2l(\hat{\theta})$$

$$BIC = d\ log(n) - 2l(\hat{\theta})$$

dove n rappresenta il numero di osservazioni, d il numero di parametri del modello, $\hat{\theta}$ il vettore dei parametri stimati del modello e $l(\hat{\theta})$ la funzione di log-verosimiglianza massimizzata.

I risultati degli AIC dei vari modelli sono stati riportati nella tabella 2.1, mentre quelli del BIC nella tabella 2.2. Come risulta dalle analisi esplorative, la distribuzione normale non è in grado di modellare serie con code pesanti. La distribuzione t di Student sembra essere la migliore per modellare i rendimenti di bitcoin, anche se la NIG dimostra un buon adattamento per il GARCH standard. Per ether si considerano i modelli con distribuzione NIG e GED, mentre per binance quelli con distribuzione t di Student e NIG. Dati i valori dei criteri di informazione, è interessante notare come, indipendentemente dalla distribuzione, i migliori modelli della classe GARCH per ogni criptovaluta siano l'EGARCH e, a seguire, il GARCH standard.

Tabella 2.1: Valori di AIC per i modelli GARCH a seconda della distribuzione delle innovazioni. In grassetto il valore di AIC migliore in base alla distribuzione per ogni modello adattato.

Bitcoin	norm	snorm	std	sstd	ged	sged	nig	ghyp
GARCH	-3.5370	-3.5408	-3.7774	-3.7765	-3.7747	-3.7744	**-3.7788**	-3.7778
EGARCH	-3.5447	-3.5463	**-3.7903**	-3.7897	-3.7800	-3.7799	-3.7865	-3.7886
TGARCH	-3.5457	-3.5475	**-3.7893**	-3.7885	-3.7634	-3.7769	-3.7860	-3.7877
PGARCH	-3.5455	-3.5473	**-3.7880**	-3.7873	-3.7792	-3.7790	-3.7848	-3.7865
Ether	norm	snorm	std	sstd	ged	sged	nig	ghyp
GARCH	-3.0221	-3.0240	-3.1995	-3.1984	-3.2018	-3.2006	-3.2040	**-3.2043**
EGARCH	-3.0219	-3.0223	-3.2003	-3.1992	**-3.2098**	-3.2021	-3.2035	-3.2024
TGARCH	-3.0120	-3.0104	-3.1984	-3.1976	-3.1756	-3.1752	**-3.2023**	-3.2015
PGARCH	-3.0224	-3.0235	-3.1986	-3.1975	-3.2018	-3.1965	**-3.2027**	-3.2026
Binance	norm	snorm	std	sstd	ged	sged	nig	ghyp
GARCH	-2.9904	-2.9939	-3.1700	-3.1689	-3.1705	-3.1692	**-3.1706**	-3.1693
EGARCH	-2.9894	-2.9918	**-3.1713**	-3.1702	-3.1698	-3.1686	-3.1709	-3.169
TGARCH	-2.9871	-2.9896	-3.1658	-3.1646	-3.1664	-3.1651	**-3.1666**	-3.1653
PGARCH	-2.9887	-2.9916	**-3.1687**	-3.1676	-3.1683	-3.1670	**-3.1687**	-3.1675

Tabella 2.2: Valori di BIC per i modelli GARCH a seconda della distribuzione delle innovazioni. In grassetto il valore di BIC migliore in base alla distribuzione per ogni modello adattato.

Bitcoin	norm	snorm	std	sstd	ged	sged	nig	ghyp
GARCH	-3.5206	-3.5211	**-3.7577**	-3.7535	-3.7550	-3.7514	-3.7558	-3.7515
EGARCH	-3.5250	-3.5233	**-3.7673**	-3.7634	-3.7570	-3.7536	-3.7602	-3.7590
TGARCH	-3.5259	-3.5245	**-3.7662**	-3.7622	-3.7404	-3.7506	-3.7597	-3.7581
PGARCH	-3.5225	-3.5210	**-3.7617**	-3.7577	-3.7529	-3.7495	-3.7552	-3.7536
Ether	norm	snorm	std	sstd	ged	sged	nig	ghyp
GARCH	-2.9892	-2.9878	-3.1633	-3.1589	**3.1657**	-3.1612	-3.1646	-3.1616
EGARCH	-2.9858	-2.9828	-3.1608	-3.1565	**-3.1704**	-3.1593	-3.1607	-3.1564
TGARCH	-2.9759	-2.9710	-3.1590	-3.1548	-3.1361	-3.1325	**-3.1596**	-3.1554
PGARCH	-2.9830	-2.9808	-3.1559	-3.1515	**-3.1590**	-3.1505	-3.1567	-3.1533
Binance	norm	snorm	std	sstd	ged	sged	nig	ghyp
GARCH	-2.9686	-2.9684	-3.1445	-3.1398	**-3.1450**	-3.1401	-3.1415	-3.1365
EGARCH	-2.9639	-2.9627	**-3.1422**	-3.1375	-3.1407	-3.1358	-3.1381	-3.1336
TGARCH	-2.9616	-2.9605	-3.1367	-3.1319	**-3.1373**	-3.1324	-3.1339	-3.1290
PGARCH	-2.9596	-2.9589	**-3.1360**	-3.1313	-3.1356	-3.1307	-3.1323	-3.1275

2.2 Bitcoin

Tabella 2.3: Modelli con distribuzione delle innovazioni t di Student per bitcoin, livelli di significatività ***,**,* rispettivamente all'1%, 5%, e 10%. Gli *standard error* sono riportati tra parentesi. In grassetto i migliori risultati ottenuti di AIC e BIC.

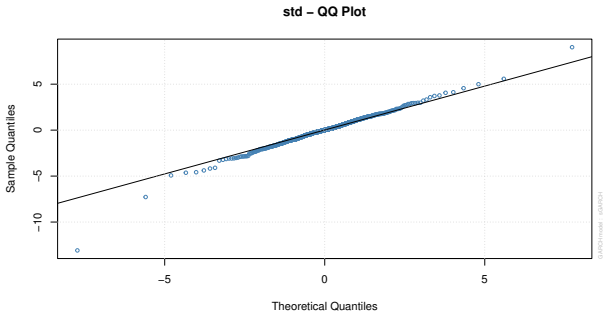
	GARCH	EGARCH	TGARCH	PGARCH
μ	0.001885***	0.001738***	0.001739***	0.001742***
	(0.000652)	(0.000656)	(0.000527)	(0.000597)
ϕ_1	-0.045580**	-0.055028**	-0.055342***	-0.055241**
	(0.022252)	(0.021580)	(0.021287)	(0.021863)
ω	0.000012	-0.035758**	0.000296	0.000286
	(0.000010)	(0.015341)	(0.000215)	(0.000390)
α	0.074810***	0.013485	0.115819***	0.115814***
	(0.011986)	(0.014288)	(0.018477)	(0.018536)
β	0.924190***	0.994252***	0.921558***	0.921691***
	(0.014885)	(0.002757)	(0.012101)	(0.012667)
γ	-	0.194016***	-0.063777	-0.063960
	-	(0.004922)	(0.081817)	(0.081787)
δ	-	-	-	1.00710***
	-	-	-	(0.237285)
LL	3111.03	3122.67	3121.76	3121.77
AIC	-3.77741	**-3.79035**	-3.78925	-3.78804
BIC	-3.75768	**-3.76733**	-3.76624	-3.76174

In tutti i modelli per bitcoin il parametro media μ, il parametro AR ϕ_1 e il parametro GARCH β sono significativi. Il miglior modello per bitcoin sembra essere l'AR(1)-EGARCH, sia in termini di bontà di adattamento che di significatività dei parametri. Il parametro γ^* in questo modello indica la presenza di asimmetria con shock positivi che hanno un impatto maggiore di quelli negativi. Questo viene confermato dalla curva NIC in figura A.4. Osservando i residui standardizzati dei modelli non emergono

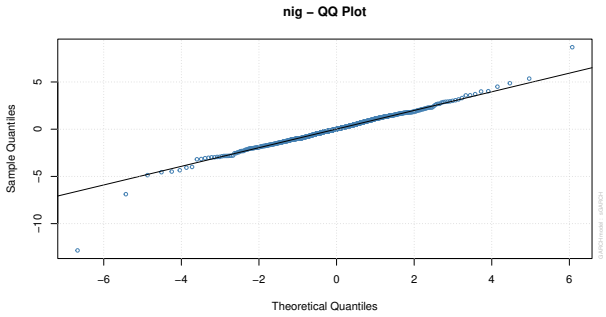
Tabella 2.4: Modelli con distribuzione delle innovazioni NIG per bitcoin, livelli di significatività ***,**,* rispettivamente all'1%, 5%, e 10% . Gli *standard error* sono riportati tra parentesi. In grassetto i migliori risultati ottenuti di AIC e BIC.

	GARCH	EGARCH	TGARCH	PGARCH
μ	0.001261	0.001052	0.001093	0.001096*
	(0.000816)	(0.000761)	(0.000790)	(0.000648)
ϕ_1	-0.052928**	-0.060303***	-0.061078***	-0.061042***
	(0.021274)	(0.019859)	(0.021137)	(0.021362)
ω	0.000016**	-0.079434	0.000556**	0.000550
	(0.000008)	(0.050242)	(0.000248)	(0.000623)
α	0.079870***	0.007718	0.110214***	0.110186***
	(0.014493)	(0.014872)	(0.017989)	(0.017982)
β	0.919130***	0.987573***	0.913580***	0.913635***
	(0.014392)	(0.008006)	(0.014406)	(0.014994)
γ	-	0.185130***	-0.040447	-0.040482
	-	(0.038834)	(0.086269)	(0.086317)
δ	-	-	-	1.419777***
	-	-	-	(0.340501)
LL	3113.17	3120.48	3120.06	3120.065
AIC	-3.778798	**-3.786477**	-3.785967	-3.784750
BIC	-3.755784	**-3.760175**	-3.759665	-3.755162

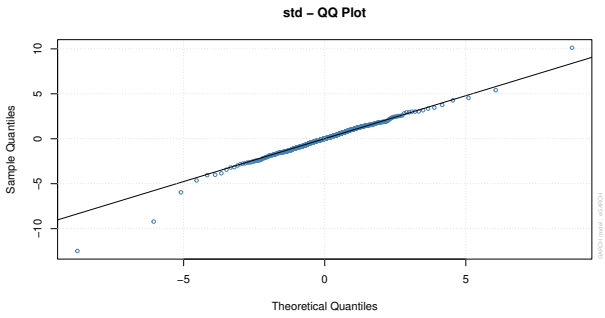
grosse differenze per le diverse distribuzioni (figure 2.1 e 2.2). Pertanto i modelli che verranno analizzati in fase di previsione saranno l'EGARCH e il GARCH standard con distribuzione t di Student.



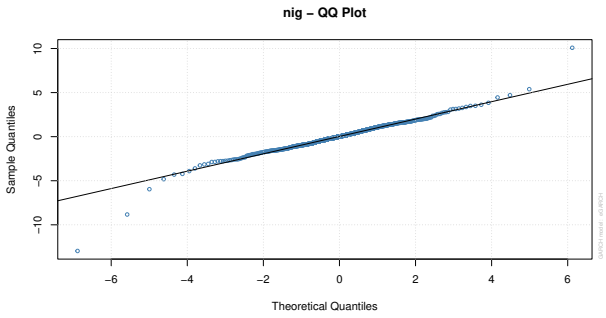
(a) *AR(1)-GARCH(1,1) std*



(b) *AR(1)-GARCH(1,1) nig*

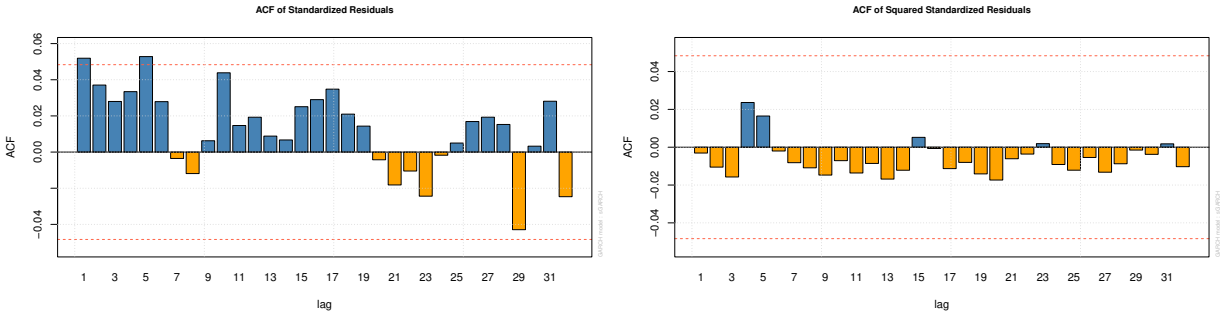


(c) *AR(1)-EGARCH(1,1) std*

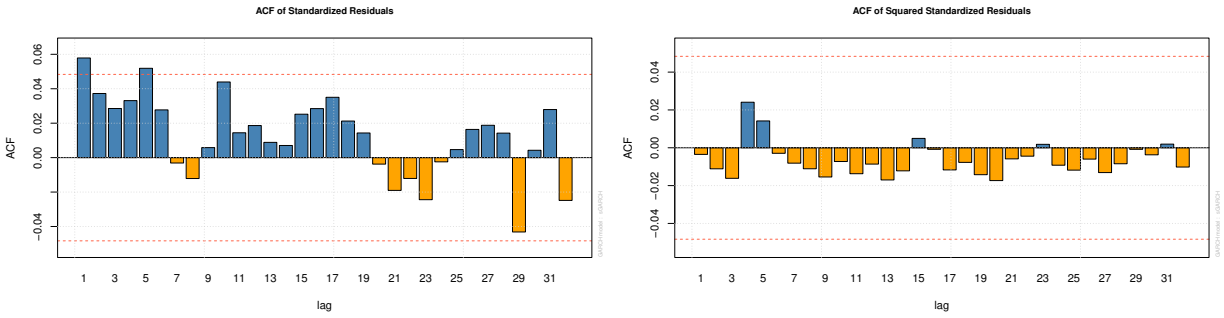


(d) *AR(1)-EGARCH(1,1) nig*

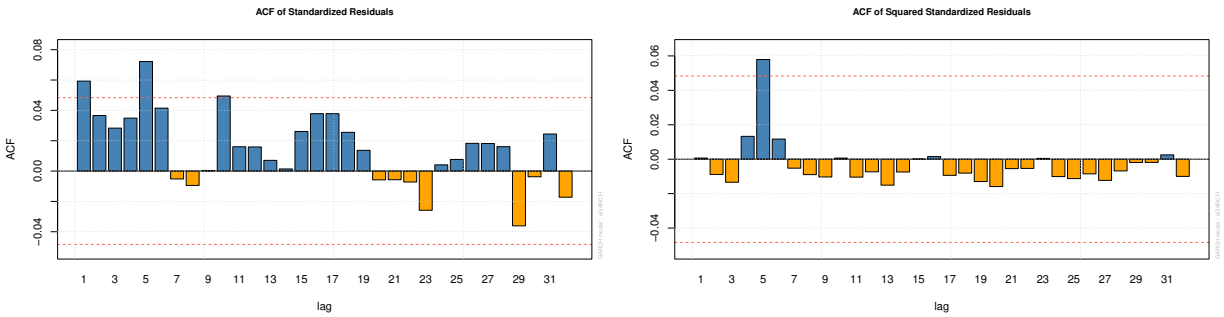
Figura 2.1: QQ-plot dei residui standardizzati per i migliori modelli stimati per bitcoin.



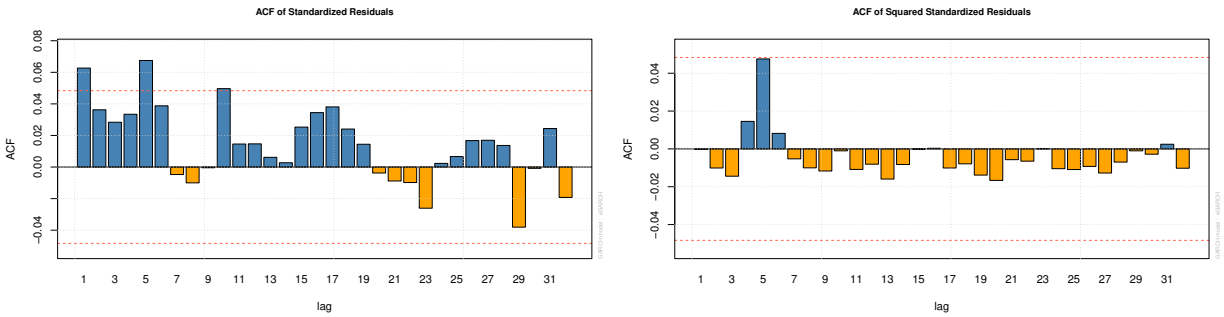
(a) *AR(1)-GARCH(1,1) std*



(b) *AR(1)-GARCH(1,1) nig*



(c) *AR(1)-EGARCH(1,1) std*



(d) *AR(1)-EGARCH(1,1) nig*

Figura 2.2: ACF dei residui standardizzati (a sinistra) e dei residui standardizzati al quadrato (a destra) dei migliori modelli stimati per bitcoin.

2.3 Ether

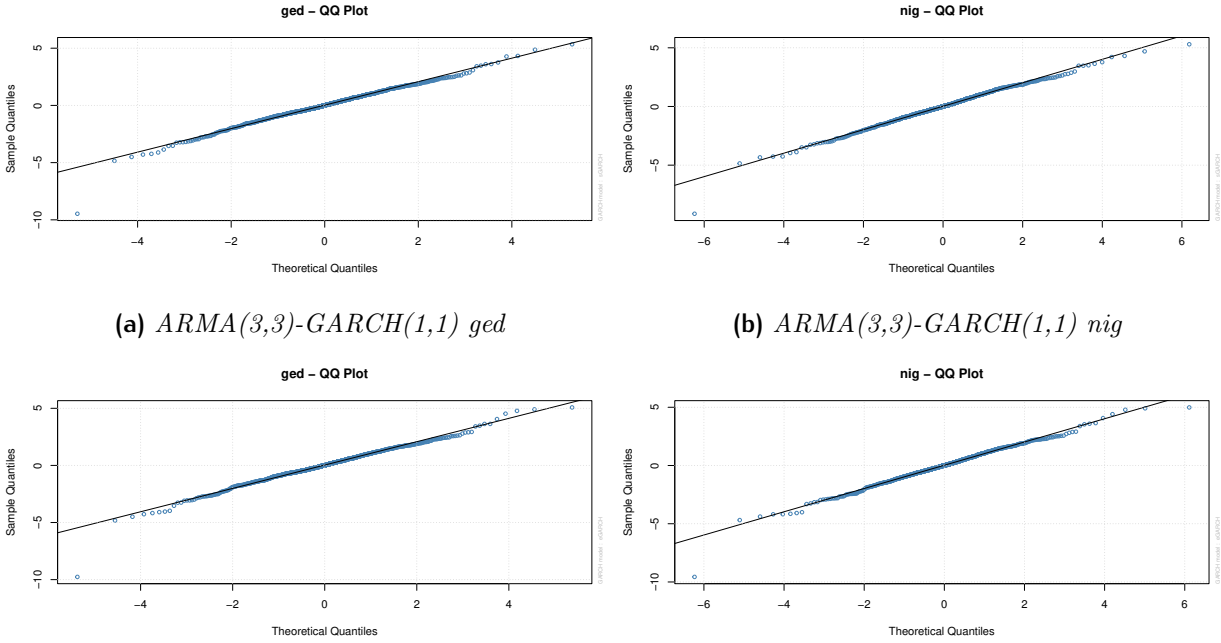
Nel caso di ether, le due distribuzioni che sembrano fornire i migliori risultati in termini di bontà di adattamento sono la GED e la NIG. Anche in questo caso, analizzando i residui dei modelli più accurati, (figure 2.3 e 2.4) non emergono grosse differenze. Il miglior modello sotto la distribuzione GED è l'EGARCH(1,1). Mentre, se le innovazioni si distribuiscono come NIG, è il GARCH(1,1) standard. Contrariamente al bitcoin, i rendimenti di ether sembrano aumentare la loro volatilità in presenza di shock negativi (si veda figura A.4 in appendice A) in accordo con lo studio di Zia-Rehman et al. (2020). Per le successive analisi, sia previsive che del VaR, si prediligono i modelli con la distribuzione GED perchè essi hanno un numero maggiore di parametri significativi e mostrano dei valori inferiori di AIC e BIC.

Tabella 2.5: Modelli con distribuzione delle innovazioni GED per ether, livelli di significatività ***,**,* rispettivamente all'1%, 5%, e 10%. Gli *standard error* sono riportati tra parentesi. In grassetto i migliori risultati ottenuti di AIC e BIC.

	GARCH	EGARCH	TGARCH	PGARCH
μ	0.001744***	0.001218***	0.001791***	0.001645***
	(0.000219)	(0.000188)	(0.000669)	(0.000204)
ϕ_1	0.234444***	0.042862***	-0.423427***	0.114050***
	(0.003403)	(0.001051)	(0.025524)	(0.004194)
ϕ_2	-0.800184***	-0.801899***	-0.326918***	-0.830608***
	(0.011372)	(0.001140)	(0.035318)	(0.003549)
ϕ_3	-0.216361***	-0.421973***	-0.093816***	-0.333442***
	(0.005216)	(0.000300)	(0.017021)	(0.011558)
θ_1	-0.319424***	-0.134873***	0.337630***	-0.197295***
	(0.003755)	(0.000522)	(0.027073)	(0.007801)
θ_2	0.857183***	0.851034***	0.351075***	0.878939***
	(0.011203)	(0.000111)	(0.032757)	(0.000142)
θ_3	0.116123***	0.325802***	0.080864***	0.244983***
	(0.001358)	(0.000177)	(0.019550)	(0.000933)
ω	0.000171***	-0.293182***	0.000002	0.000405
	(0.000058)	(0.064572)	(0.000020)	(0.000419)
α	0.103762***	-0.007023	0.078673***	0.113401***
	(0.024525)	(0.019052)	(0.013092)	(0.025292)
β	0.843149***	0.950431***	0.944848***	0.850526***
	(0.034941)	(0.010996)	(0.009004)	(0.035971)
γ	-	0.201587***	-0.066812	0.021501
	-	(0.033524)	(0.088665)	(0.079833)
δ	-	-	-	1.688576***
	-	-	-	(0.359221)
LL	2642.90	2650.46	2622.34	2644.85
AIC	-3.20183	**-3.20981**	-3.17559	-3.20177
BIC	-3.16566	**-3.17035**	-3.13614	-3.15903

Tabella 2.6: Modelli con distribuzione delle innovazioni NIG per ether, livelli di significatività ***,**,* rispettivamente all'1%, 5%, e 10%. Gli *standard error* sono riportati tra parentesi. In grassetto i migliori risultati ottenuti di AIC e BIC.

	GARCH	EGARCH	TGARCH	PGARCH
μ	0.001721	0.001636**	0.001799	0.001714
	(0.001154)	(0.000742)	(0.001129)	(0.001150)
ϕ_1	0.16559***	0.195268***	0.148830***	0.158213**
	(0.061406)	(0.058511)	(0.056287)	(0.066524)
ϕ_2	-0.802806***	-0.744230***	-0.798465***	-0.800507***
	(0.019843)	(0.013165)	(0.087998)	(0.012654)
ϕ_3	-0.269710***	-0.378780***	-0.276921***	-0.273626***
	(0.045700)	(0.016756)	(0.027932)	(0.043129)
θ_1	-0.251408***	-0.286794***	-0.238792***	-0.244935***
	(0.060094)	(0.0366899	(0.062433)	(0.065614)
θ_2	0.861452***	0.798971***	0.860597***	0.860364***
	(0.016193)	(0.001119)	(0.079977)	(0.006017)
θ_3	0.168404***	0.273880***	0.172387***	0.171577***
	(0.041418)	(0.000152)	(0.023977)	(0.037726)
ω	0.000165***	-0.265020***	0.002572***	0.000660
	(0.000055)	(0.070233)	(0.000939)	(0.000640)
α	0.117845***	0.000868	0.122542***	0.124761***
	(0.026471)	(0.019158)	(0.023190)	(0.025570)
β	0.839939***	0.954942***	0.868814***	0.858351***
	(0.032500)	(0.011898)	(0.028115)	(0.031693)
γ	-	0.212474***	0.022855	0.007870
	-	(0.034864)	(0.098725)	(0.082972)
δ	-	-	-	1.484213***
	-	-	-	(0.329664)
LL	2645.67	2646.25	2645.32	2646.65
AIC	**-3.20401**	-3.20347	-3.20234	-3.20274
BIC	**-3.16456**	-3.16073	-3.15960	-3.15671



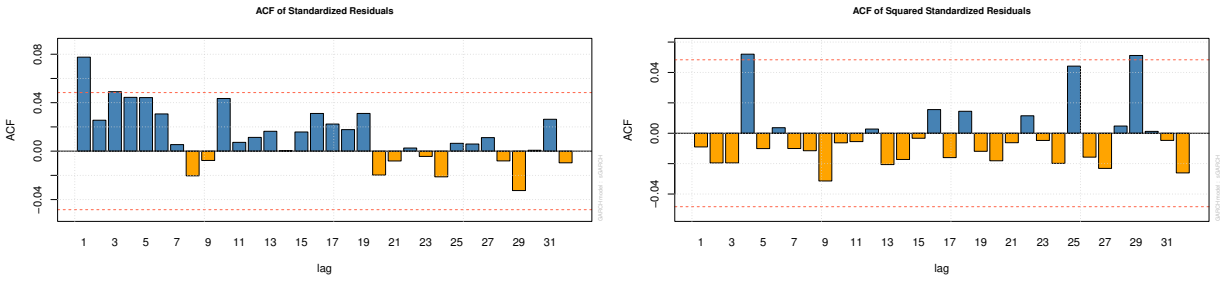
(a) *ARMA(3,3)-GARCH(1,1) ged*

(b) *ARMA(3,3)-GARCH(1,1) nig*

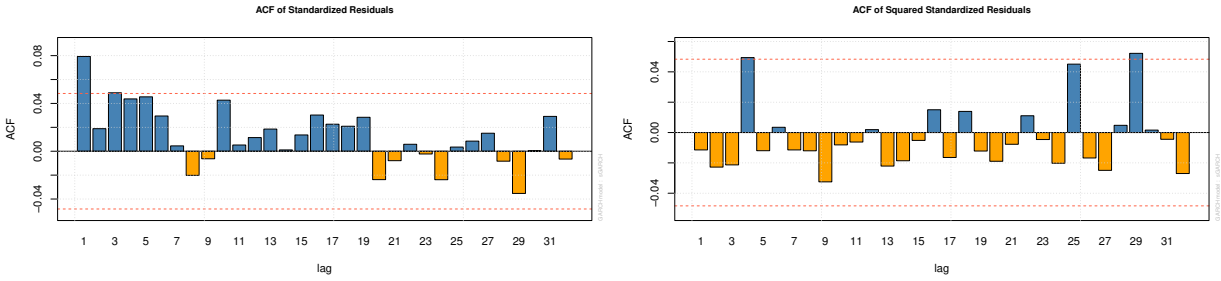
(c) *ARMA(3,3)-EGARCH(1,1) ged*

(d) *ARMA(3,3)-EGARCH(1,1) nig*

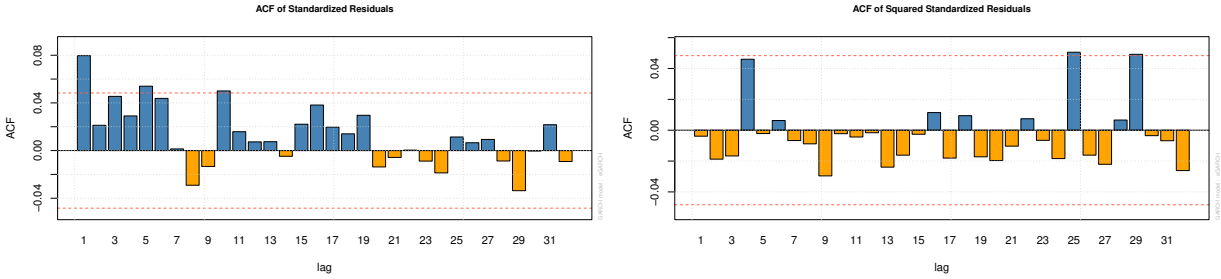
Figura 2.3: QQ-plot dei residui standardizzati dei migliori modelli stimati per ether.



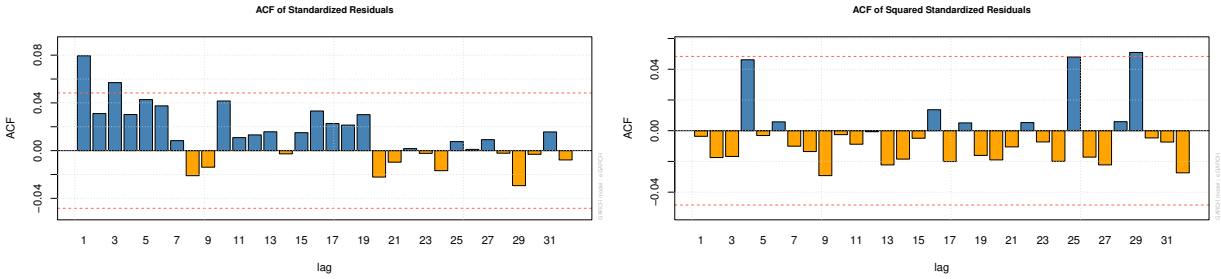
(a) *ARMA(3,3)-GARCH(1,1) GED*



(b) *ARMA(3,3)-GARCH(1,1)* NIG



(c) *ARMA(3,3)-EGARCH(1,1) GED*



(d) *ARMA(3,3)-EGARCH(1,1) NIG*

Figura 2.4: ACF dei residui standardizzati (a sinistra) e dei residui standardizzati al quadrato (a destra) dei migliori modelli stimati per ether.

2.4 Binance Coin

Tabella 2.7: Modelli con distribuzione delle innovazioni std per binance, livelli di significatività ***,**,* rispettivamente all'1%, 5%, e 10%. Gli *standard error* sono riportati tra parentesi. In grassetto i migliori risultati ottenuti di AIC e BIC.

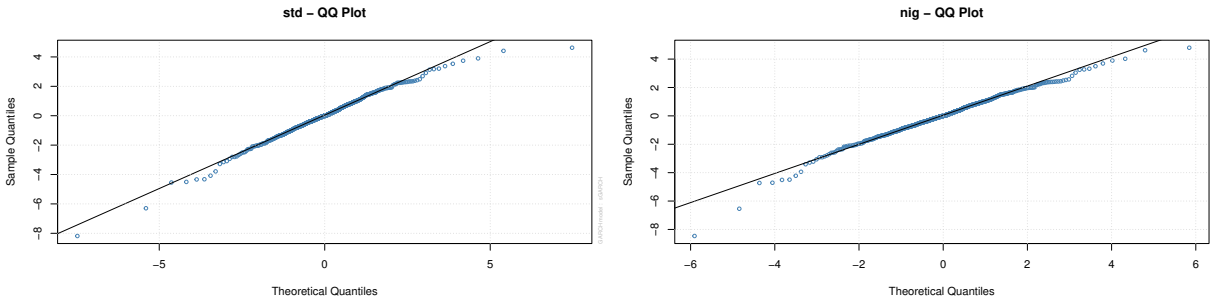
	GARCH	EGARCH	TGARCH	PGARCH
μ	0.001917**	0.002001**	0.001998**	0.002101**
	(0.000970)	(0.000804)	(0.000814)	(0.000983)
θ_1	-0.066111***	-0.067460**	-0.068854***	-0.066599**
	(0.025617)	(0.030227)	(0.025319)	(0.025423)
θ_2	0.017630	0.019857	0.020901	0.019398
	(0.025707)	(0.017743)	(0.022623)	(0.025608)
ω	0.000139**	-0.189104***	0.001821**	0.000199
	(0.000055)	(0.050094)	(0.000721)	(0.000231)
α	0.140924***	0.019005	0.137032***	0.132051***
	(0.037998)	(0.020349)	(0.027112)	(0.037135)
β	0.840506***	0.967228***	0.876610***	0.856598***
	(0.037151)	(0.008593)	(0.025719)	(0.037501)
γ	-	0.245785***	-0.052938	-0.092349
	-	(0.041035)	(0.092267)	(0.072751)
δ	-	-	-	1.832615***
	-	-	-	(0.414553)
LL	2308.41	2310.37	2306.34	2309.46
AIC	-3.16999	**-3.17131**	-3.16575	-3.16868
BIC	**-3.14453**	-3.14221	-3.13666	-3.13595

Osservando i QQ-plot dei residui (figura 2.5), emerge un migliore adattamento del modello EGARCH per quanto concerne la parte destra della distribuzione, sia per la NIG che per la t di Student. Come per il bitcoin, anche per binance gli shock positivi sembrano avere un impatto maggiore sulla volatilità rispetto agli shock negativi (figura A.4). Data

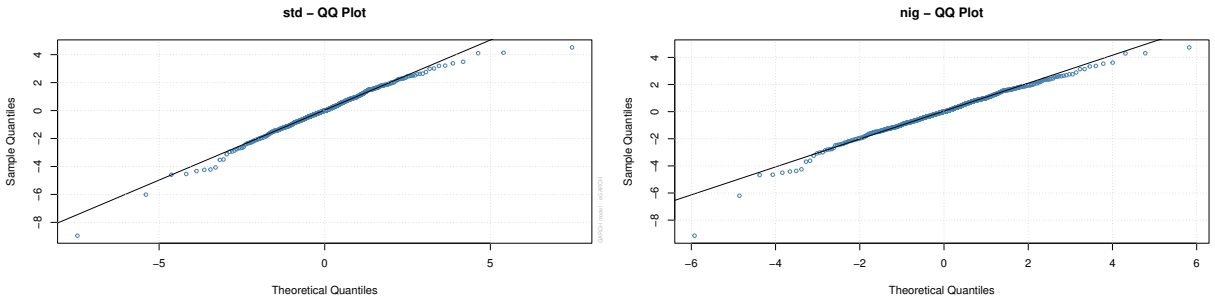
Tabella 2.8: Modelli con distribuzione delle innovazioni NIG per binance, livelli di significatività ***,**,* rispettivamente all'1%, 5%, e 10%. Gli *standard error* sono riportati tra parentesi. In grassetto i migliori risultati ottenuti di AIC e BIC.

	GARCH	EGARCH	TGARCH	PGARCH
μ	0.001577	0.001590**	0.001549	0.001643
	(0.001172)	(0.000782)	(0.000958)	(0.001176)
θ_1	-0.028041***	-0.075398***	-0.076495***	-0.074009***
	(0.025232)	(0.022227)	(0.022659)	(0.025151)
θ_2	0.031429	0.018914	0.020113	0.017752
	(0.025183)	(0.015857)	(0.019603)	(0.025066)
ω	0.000106***	-0.197703***	0.001795**	0.000226
	(0.000048)	(0.057401)	(0.000684)	(0.000270)
α	0.136717***	0.013136	0.136134***	0.132111***
	(0.033224)	(0.020129)	(0.025849)	(0.033295)
β	0.848063***	0.966323***	0.873193***	0.851913***
	(0.035890)	(0.009746)	(0.025614)	(0.037226)
γ	-	0.243979***	-0.031052	-0.064434
	-	(0.040721)	(0.091798)	(0.072152)
δ	-	-	-	1.775258**
	-	-	-	0.430062
LL	2309.85	2311.05	2307.96	2310.48
AIC	-3.17059	**-3.17087**	-3.16661	-3.16871
BIC	**-3.14149**	-3.13814	-3.13388	-3.13234

la differenza nei criteri di informazione si prediligono i modelli GARCH e EGARCH con distribuzione t di Student per le successive analisi.

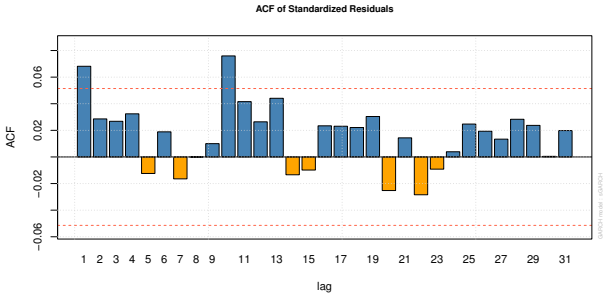


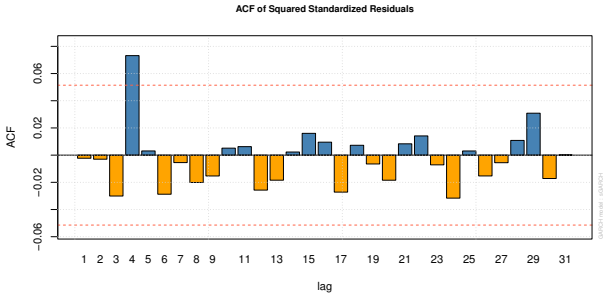
(a) *MA(2)-GARCH(1,1) std* **(b)** *MA(2)-GARCH(1,1) nig*



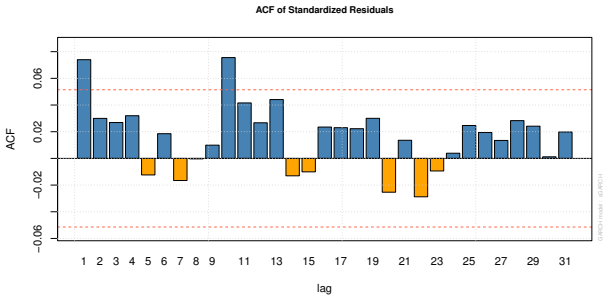
(c) *MA(2)-EGARCH(1,1) std* **(d)** *MA(2)-EGARCH(1,1) nig*

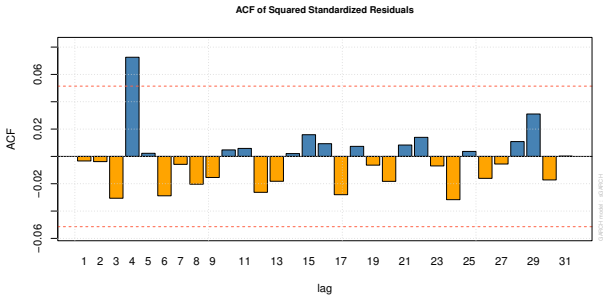
Figura 2.5: QQ-plot dei residui standardizzati dei migliori modelli stimati per binance.



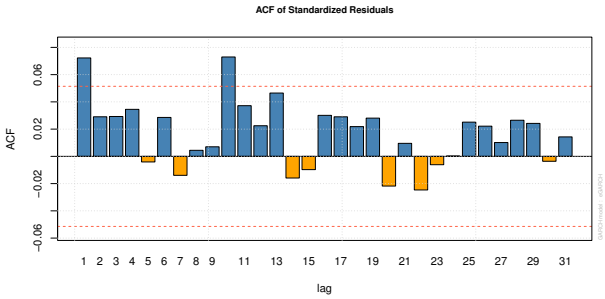


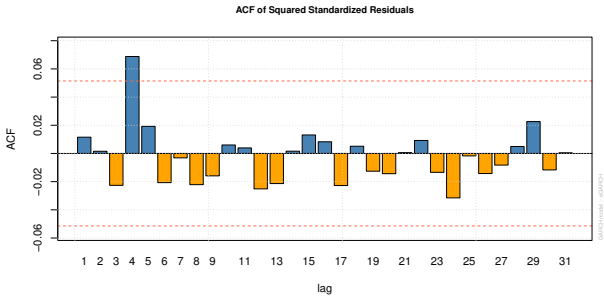
(a) *MA(2)-GARCH(1,1) std*



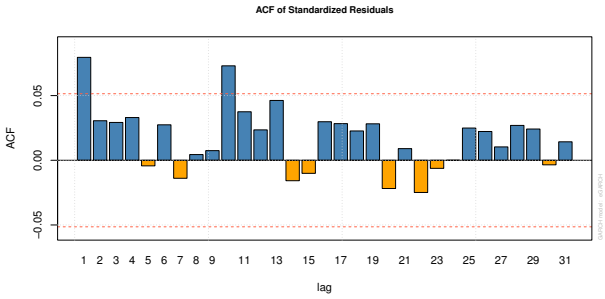


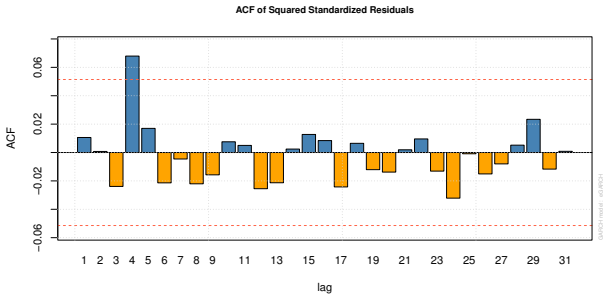
(b) *MA(2)-GARCH(1,1) nig*





(c) *MA(2)-EGARCH(1,1) std*





(d) *MA(2)-EGARCH(1,1) nig*

Figura 2.6: ACF dei residui standardizzati (a sinistra) e dei residui standardizzati al quadrato (a destra) dei migliori modelli stimati per binance.

Capitolo 3

Previsioni e Value-at-Risk

Per procedere alle previsioni e alle stime del VaR, il campione è stato diviso in due parti: *in-sample* dal 01/05/2017 al 31/10/2021 (1644 osservazioni, 1452 per binance che inizia dal 10/11/2017) e *out-of-sample* dal 01/11/2021 al 31/05/2022 (212 osservazioni). La frazione *in-sample* è stata utilizzata per la stima dei modelli visti nel secondo capitolo. Le previsioni sono state calcolate basandosi su modelli stimati in *rolling windows* di 1500 osservazioni (dato che la serie di binance è più breve perchè è una cripto più recente, la *rolling window* per binance è formata da 1452 osservazioni). Per determinare il miglior modello previsivo si sono fatte delle previsioni a 1, 5 e 10 passi in avanti. Come indicatori, si sono utilizzati l'errore quadratico medio (RMSE) e l'errore medio assoluto (MAE). Come *proxy* della volatilità si sono utilizzati i rendimenti logaritmici al quadrato *out-of-sample*. Per valutare le performance dei modelli GARCH si è utilizzato come *benchmark* il modello *RiskMetrics*.

3.1 Modello RiskMetrics

Il modello *RiskMetrics* (Longerstaey e Spencer 1996) è un modello a media mobile con pesi esponenziali, noto per la sua semplicità. Per le stime del VaR è molto efficiente, tanto da essere utilizzato da J.P. Morgan e altre banche. Questo modello può essere considerato un caso particolare del modello GARCH. L'equazione della volatilità condizionata è così

specificata:

$$\sigma^2_{t|t-1} = \lambda\sigma^2_{t-1|t-2} + (1-\lambda)r^2_{t-1}$$

Il parametro $\lambda \in (0,1)$ è chiamato *decay factor* o costante di lisciamento ed indica il grado di persistenza delle osservazioni passate. λ è fissato a 0.94 per dati giornalieri e r^2_t rappresenta il rendimento giornaliero al quadrato al tempo t della serie presa in considerazione.

3.2 Previsioni

Dai risultati in tabella 3.1 emerge come il miglior modello nelle previsioni su più giorni sia l'EGARCH per tutte e tre le criptovalute. Dalle previsioni un passo in avanti, tra bitcoin e binance, si evidenzia un'altra analogia, perchè, in entrambi i casi i migliori risultati in termini di MAE vengono forniti dal modello *RiskMetrics*. Osservando l'RMSE, il miglior modello pare essere invece il GARCH standard. È interessante notare che le previsioni ottenute con l'EGARCH per bitcoin siano molto più volatili rispetto alle previsioni delle altre due criptovalute ottenute con lo stesso modello (si vedano le figure A.1, A.2, A.3 in appendice A).

Tabella 3.1: Performance delle previsioni BTC. In grassetto il risultato migliore.

	BTC		ETH		BNB	
	mae	rmse	mae	rmse	mae	rmse
1 day ahead						
RiskM.	**0.001248**	0.002223	**0.001741**	**0.002985**	**0.001697**	0.003402
EGARCH	0.001749	0.002550	0.002015	0.003026	0.001956	0.003466
sGARCH	0.001281	**0.002190**	0.002018	0.003050	0.001813	**0.003395**
5 days ahead						
RiskM.	0.023157	0.032937	0.029966	0.039549	0.028976	0.039074
EGARCH	**0.022449**	**0.032361**	**0.029466**	**0.039155**	**0.028331**	**0.038536**
sGARCH	0.023010	0.032821	0.029617	0.039275	0.028670	0.038802
10 days ahead						
RiskM.	0.023118	0.032899	0.029941	0.039512	0.028983	0.039081
EGARCH	**0.022378**	**0.032334**	**0.029403**	**0.039087**	**0.028330**	**0.038522**
sGARCH	0.022950	0.032753	0.029562	0.039191	0.028614	0.038734

3.3 Value-at-Risk

Il *Value-at-Risk* è la misura di rischio più diffusa dopo la volatilità. A livello interpretativo, esso rappresenta la perdita che ci si aspetta ecceda solo con probabilità pari ad α, nel periodo considerato, quindi è il "migliore dei casi peggiori". Volendo dare una definizione a livello matematico, si ipotizzi che X sia la variabile casuale che descrive i guadagni e le perdite di un portafoglio di attività finanziarie. Partendo da questa ipotesi, il *Value–at–Risk*, del portafoglio VaR_α al livello di confidenza $\alpha \in (0, 1)$ dove $\alpha = (0.01, 0.05)$, è dato dal più piccolo valore v tale che la probabilità che una perdita ecceda v è minore o uguale ad α.

$$VaR_\alpha = inf\{v \in \mathbb{R} : \mathbb{P}(X + v \leq 0) \leq \alpha\} = -Q^+_\alpha(X),$$

dove $-Q^+_\alpha(X) = inf\{x \in \mathbb{R} : \mathbb{F}_X(x) > \alpha\}$

Per valutare la bontà delle stime VaR, si utilizzerà *'Actual over Expected exceedance ratio'* (A/E), ovvero il rapporto tra il numero di osservazioni che sforano il limite del VaR e quelle attese. Questo valore indica una buona copertura delle stime quanto più si avvicina ad 1. Inoltre, verranno considerati i test di Kupiec (1995) per la proporzione di sforamenti e il test di Christoffersen (1998) che verifica l'ipotesi nulla di indipendenza delle violazioni del VaR. I risultati sono riportati nelle tabelle 3.2, 3.3 e 3.4. Osservando le stime $VaR_{95\%}$ per bitcoin non si riscontrano risultati soddisfacenti: tutti e tre i modelli evidenziano l'allontanamento dall'ipotesi nulla dei due test e presentano un A/E elevato. Invece, osservando le stime $VaR_{99\%}$, il miglior modello risulta essere l'EGARCH. Tuttavia, questo modello sembra sovrastimare il rischio in diversi punti (figura 3.1). Per quanto riguarda ether (tabella 3.3), il miglior modello è l'EGARCH per le stime $VaR_{95\%}$ e il GARCH standard per le stime $VaR_{99\%}$. L'EGARCH è il modello migliore nelle stime $VaR_{95\%}$ per binance coin (tabella 3.4) ed è l'unico che non rifiuta l'ipotesi nulla nei due test. Per quanto riguarda il $VaR_{99\%}$, il GARCH e l'EGARCH mostrano risultati simili: il secondo è preferibile perchè presenta una violazione media e massima inferiori al primo.

Tabella 3.2: Backtesting del $VaR_{95\%}$ e del $VaR_{99\%}$ per bitcoin. Le colonne LR_{uc} e LR_{cc} riportano i *p-values* dell'*unconditional coverage* Kupiec test e il *conditional coverage* Christoffersen test.

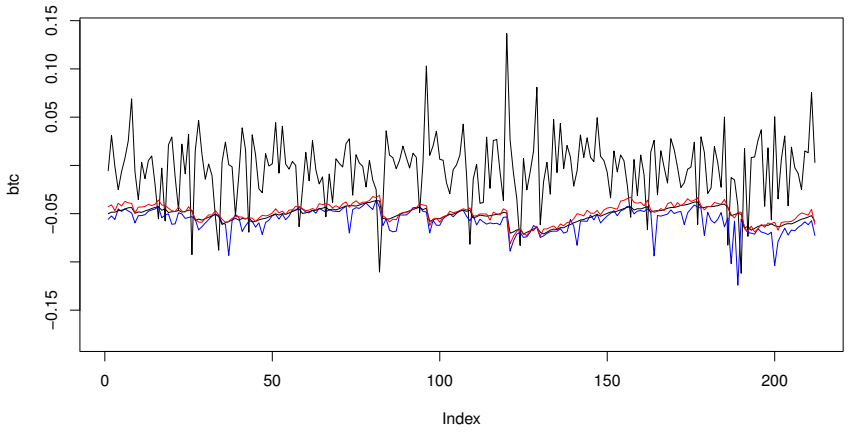
	Model	A/E	mean AD vio.	max AD vio.	Vio.	LR_{uc}	LR_{cc}
VaR95	RiskM	1.69811	0.02507	0.07380	18	0.03315	0.09129
	GARCH	1.79245	0.02659	0.07964	19	0.01666	0.04642
	EGARCH	1.79245	0.02012	0.06724	19	0.01666	0.04642
VaR99	RiskM	2.35849	0.01964	0.04800	5	0.09081	0.21197
	GARCH	1.88679	0.02054	0.05207	4	0.24776	0.47461
	EGARCH	0.94340	0.01327	0.02497	2	0.93336	0.97762

Tabella 3.3: Backtesting del $VaR_{95\%}$ e del $VaR_{99\%}$ per ether. Le colonne LR_{uc} e LR_{cc} riportano i *p-values* dell'unconditional coverage Kupiec test e il conditional coverage Christoffersen test.

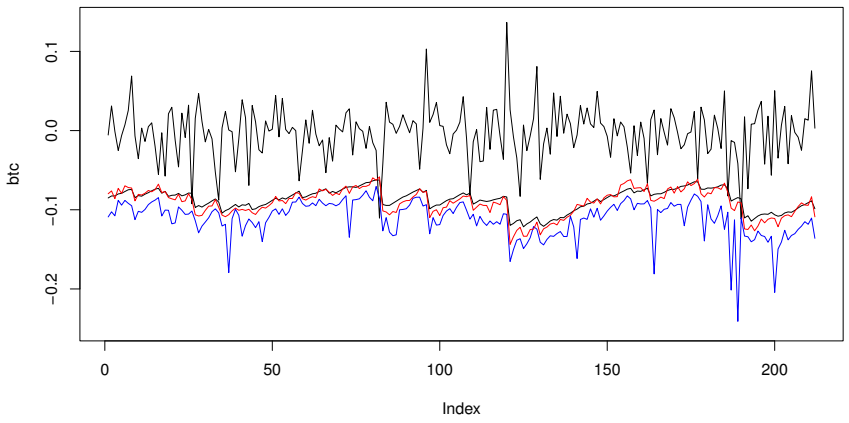
	Model	A/E	mean AD vio.	max AD vio.	Vio.	LR_{uc}	LR_{cc}
Var95	RiskM	1.6037	0.02769	0.10699	17	0.06265	0.03978
	GARCH	1.5094	0.02258	0.09904	16	0.11230	0.07615
	EGARCH	1.3208	0.02218	0.09663	14	0.30610	0.21883
Var99	RiskM	1.4151	0.02833	0.06876	3	0.56751	0.81326
	GARCH	0.9434	0.02983	0.05365	2	0.93336	0.97762
	EGARCH	0.4717	0.04866	0.04866	1	0.38866	0.68638

Tabella 3.4: Backtesting del $VaR_{95\%}$ e del $VaR_{99\%}$ per binance. Le colonne LR_{uc} e LR_{cc} riportano i *p-values* dell'unconditional coverage Kupiec test e il conditional coverage Christoffersen test.

	Model	A/E	mean AD vio.	max AD vio.	Vio.	LR_{uc}	LR_{cc}
VaR95	RiskM	1.6038	0.02749	0.12368	17	0.06265	0.03978
	GARCH	1.6981	0.02503	0.12330	18	0.03315	0.09129
	EGARCH	1.2264	0.02856	0.11435	13	0.46443	0.32571
VaR99	RiskM	1.4151	0.05385	0.08952	3	0.56751	0.81326
	GARCH	0.9434	0.05833	0.08053	2	0.93336	0.97762
	EGARCH	0.9434	0.04153	0.06397	2	0.93336	0.97762

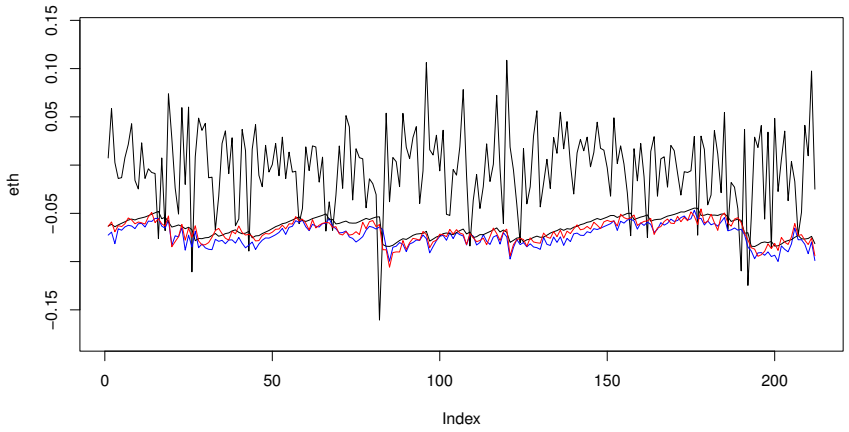


(a) *VaR 95% per bitcoin*

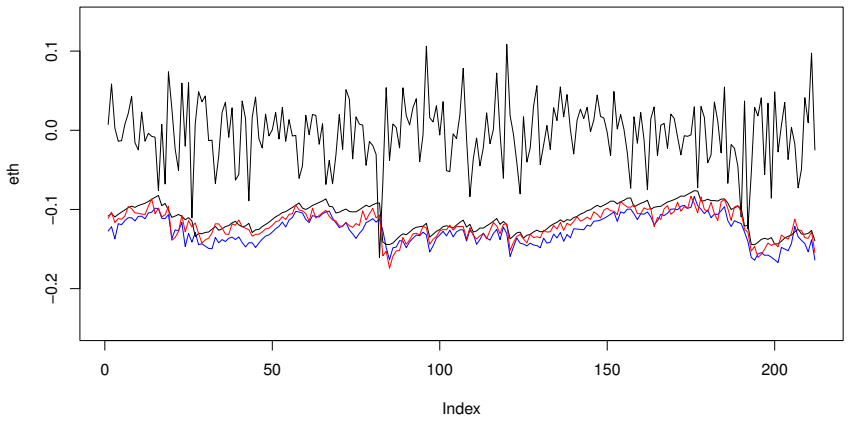


(b) *VaR 99% per bitcoin*

Figura 3.1: Stime del VaR per bitcoin. In **nero** il modello *RiskMetrics*, in **rosso** GARCH e in **blu** EGARCH, con distribuzione degli errori t di Student.

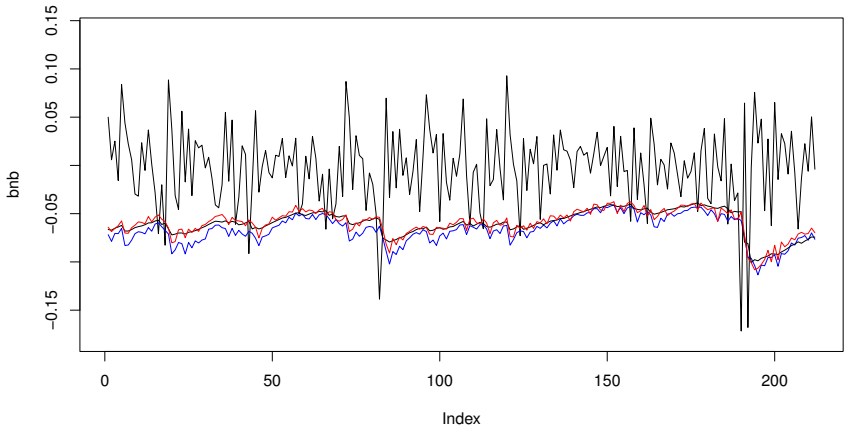


(a) *VaR 95% per ether*

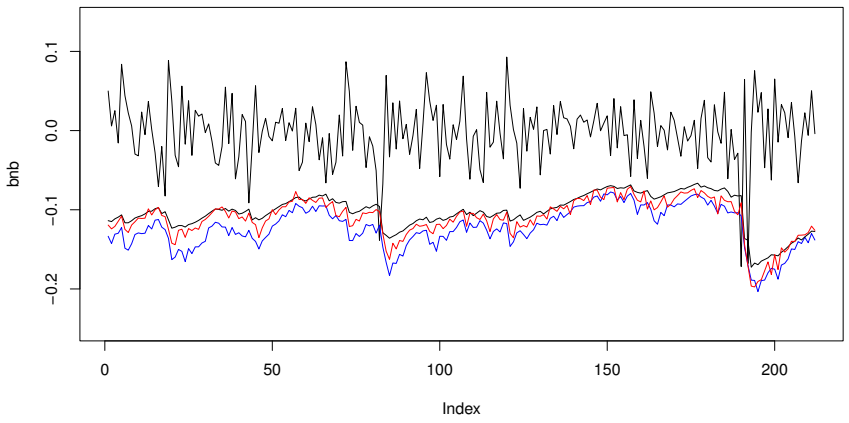


(b) *VaR 99% per ether*

Figura 3.2: Stime del VaR per ether. In **nero** il modello *RiskMetrics*, in **rosso** GARCH e in **blu** EGARCH, con distribuzione degli errori GED.



(a) *VaR 95% per Binance*



(b) *VaR 99% per Binance*

Figura 3.3: Stime del VaR per binance. In **nero** il modello *RiskMetrics*, in **rosso** GARCH e in **blu** EGARCH, con distribuzione degli errori t di Student.

Capitolo 4

Conclusioni

La volatilità delle criptovalute resta un problema attuale per gli investitori in questo periodo storico. In accordo con gli ultimi dati registrati, le quotazioni di tutte le principali criptovalute hanno subito un forte calo a causa della guerra in Ucraina e dell'aumento delle stime dell'inflazione, previste per quest'anno. Il trend ribassista è stato anche amplificato dalla crisi dello *stablecoin* TERRA-USD. In questi casi, per chi opera nei mercati finanziari, è fondamentale studiare la volatilità delle criptovalute allo scopo di gestire il rischio del proprio portafoglio. Questo elaborato analizza i migliori modelli della classe ARMA-GARCH per le principali criptovalute, considerando il loro periodo più importante dal 2017 ad oggi. Anche se chiamato spesso "oro digitale", nel periodo sopracitato, il bitcoin ha una debole correlazione con il celebre metallo prezioso. Dalle analisi effettuate, il bitcoin potrebbe rivelarsi un investimento interessante in termini di diversificazione di portafoglio, ma non è sicuramente un titolo *safe heaven*, dato il suo comportamento estremamente volatile.
Le analisi preliminari indicano la presenza di code pesanti nella distribuzione dei titoli considerati. Questo comportamento viene confermato dall'adattamento mostrato nei modelli del secondo capitolo. I migliori risultati si sono ottenuti con le distribuzioni t di Student, NIG e GED, note per modellare serie leptocurtiche. Durante la costruzione dei modelli, sono emersi dei risultati interessanti:

- generalmente l'EGARCH si è rivelato il modello più accurato a parità di distribuzione, fornendo i valori minori di AIC e BIC;

- binace e bitcoin nei modelli EGARCH mostrano un effetto asimmetrico che aumenta la volatilità dei loro rendimenti in presenza di notizie positive;
- la distribuzione NIG ha fornito un ottimo adattamento in tutti e tre i casi. Tuttavia, sono state preferite altre distribuzioni per lievi miglioramenti in termini di bontà di adattamento.

Per quanto concerne la previsione e il calcolo del *Value-at-Risk*, i modelli adattati sono stati in grado di fornire risultati migliori del *benchmark RiskMetrics*. Tenuto conto dei vari aspetti, per modellare la volatilità delle principali cripto, i migliori modelli sono il GARCH standard e l'EGARCH combinati ad un opportuno modello per la media della serie considerata.

Bibliografia

Bollerslev (1986). *Generalized Autoregressive Conditional Heteroskedasticity*. Journal of Econometrics 31, 307-327.

Brooks (2008). *Introductory Econometrics for Finance*. Cambridge University Press.

Caponera e Gola (2019). *Questioni di Economia e Finanza, Aspetti economici e regolamentari delle «cripto-attività»*. Banca d'Italia, 484.

Christoffersen (1998). *Evaluating Interval Forecasts*. International Economic Review, 39.4.

Chu et al. (2017). *GARCH modelling of cryptocurrencies*. Journal of Risk e Financial Management 10.4.

Ding et al. (1993). *A long memory property of stock market returns and a new model*. Journal of empirical finance, 1.1.

Gkillas e Longin (2019). *Is Bitcoin the new digital gold? Evidence from extreme price movements in financial markets*. Evidence From Extreme Price Movements in Financial Markets.

Gyamerah (2019). *Modelling the volatility of Bitcoin returns using GARCH models*. Quantitative Finance e Economics 3.4, 739-753.

Hardle et al. (2020). *Understanding Cryptocurrencies*. Journal of Financial Econometrics, 18.2, 181–208.

Kupiec (1995). *Techniques for Verifying the Accuracy of Risk Measurement Models*. The Journal of Derivatives, 3.2.

Kyriazis (2020). *Is Bitcoin similar to gold? An integrated overview of empirical findings*. Journal of Risk e Financial Management, 13.5.

Longerstaey e Spencer (1996). *Riskmetricstm—technical document*. Morgan Guaranty Trust Company of New York, 51.

López-Cabarcos et al. (2021). *Bitcoin volatility, stock market and investor sentiment. Are they connected?* Finance Research Letters, 38.

Nakamoto (2008). *Bitcoin: A peer-to-peer electronic cash system*. Decentralized Business Review.

Nelson (1991). *Conditional heteroskedasticity in asset returns: A new approach*. Econometrica: Journal of the Econometric Society, 59.2.

Schwert (1989). *Why Does Stock Market Volatility Change Over Time?* The Journal of Finance, 44.5.

Shen et al. (2021). *Bitcoin Return Volatility Forecasting: A Comparative Study between GARCH and RNN*. Journal of Risk e Financial Management, 14.7.

Zakoian (1994). *Threshold heteroskedastic models*. Journal of Economic Dynamics e control, 18.5.

Zia-Rehman et al. (2020). *Modeling cryptocurrencies volatility using GARCH models: a comparison based on Normal and Student's T-Error distribution*. Entrepreneurship e Sustainability Issues, 7.3.

Sitografia

Boffa (2022). *Dentro il mistero Satoshi Nakamoto: una webserie racconta chi è l'inventore dei Bitcoin*. URL: https://www.huffingtonpost.it/economia/2022/02/23/news/l_origine_dei_bitcoin-8815347/.

Che cos'è Binance Coin? (2022). URL: https://coinmarketcap.com/it/currencies/bnb/.

Che cos'è Ethereum? (2022). URL: https://coinmarketcap.com/it/currencies/ethereum/.

China declares all crypto-currency transactions illegal (2021). URL: https://www.bbc.com/news/technology-58678907.

Cina: la grande crescita silenziosa dello Yuan digitale (2022). URL: https://www.ilsole24ore.com/art/cina-grande-crescita-silenziosa-yuan-digitale-AEufHqDB.

Coinbase, la «Borsa» dei bitcoin vola al debutto poi rallenta (2021). URL: https://www.ilsole24ore.com/art/coinbase-quotazione-storica-nasdaq-mondo-criptovalute-sbarca-wall-street-AE185w.

Coinmetrics (2022). URL: https://coinmetrics.io/.

Corinto, Di (2022). *Che c'entra il Bitcoin con le rivolte in Kazakistan e gli scontri in piazza*. URL: https://www.repubblica.it/tecnologia/

2022/01/10/news/che_c_entra_il_bitcoin_con_le_rivolte_in_kazakistan_e_gli_scontri_in_piazza-333273222/.

Crypto war aid for Ukraine: Are donations in Bitcoin an innovation or just a sideshow? (2022). URL: https://www.euronews.com/next/2022/03/28/crypto-war-aid-for-ukraine-are-donations-in-bitcoin-an-innovation-or-just-a-sideshow.

La corsa al "Web3" (2022). URL: https://www.ilpost.it/2022/02/20/web3/.

L'Ucraina vuole puntare sulle criptovalute (2021). URL: https://www.ilpost.it/2021/11/28/ucraina-criptovalute/.

Marazzina (2022). *Stablecoin: cosa sono e perché se ne parla (e perché c'entra Facebook)*. URL: https://www.huffingtonpost.it/entry/stablecoin-cosa-sono-e-perche-se-ne-parla-e-perche-centra-facebook_it_61b31edfe4b0030da7d350ed/#:~:text=Gli\%20stablecoin\%20sono\%20crypto\%20asset,per\%20gli\%20scambi\%20di\%20Bitcoin..

Portale (2021). *Blockchain for business: in che modo le aziende stanno utilizzando la tecnologia Blockchain?* URL: https://blog.osservatori.net/it_it/blockchain-business-applicazioni-aziende?hsLang=it-it.

Rodeck e Schmidt (2022). *What Is Blockchain?* URL: https://www.forbes.com/advisor/investing/what-is-blockchain/.

Soldavini (2020). *PayPal apre alle criptovalute: il bitcoin sarà accettato per i pagamenti*. URL: https://www.ilsole24ore.com/art/paypal-apre-criptovalute-bitcoin-sara-accettato-i-pagamentii-ADxnOSx.

The R Project for Statistical Computing (2022). URL: https://cran.r-project.org/.

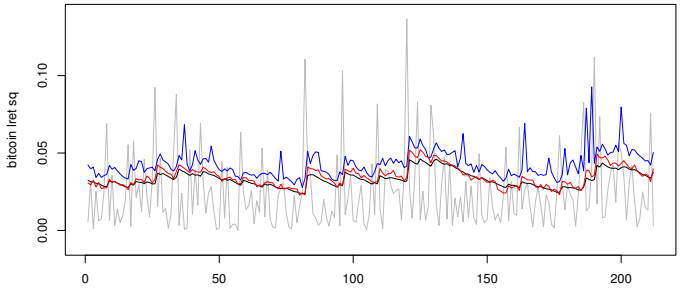
Un euro digitale (2022). URL: https://www.ecb.europa.eu/paym/digital_euro/html/index.it.html.

Winkie (2022). *Every bitcoin helps: why Ukraine is soliciting for cryptocurrency donations*. URL: https://www.theguardian.com/technology/2022/mar/03/bitcoin-donations-cryptocurrency-support-ukraine.

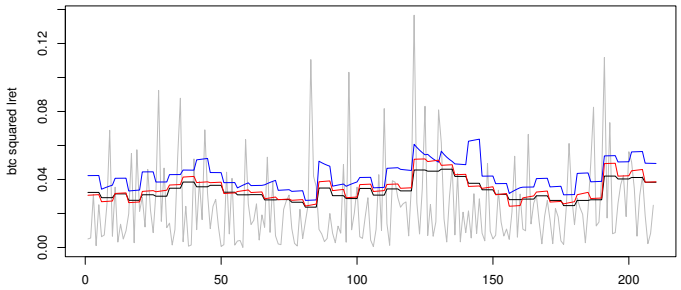
Yahoo Finance (2022). URL: https://finance.yahoo.com/.

Appendice A

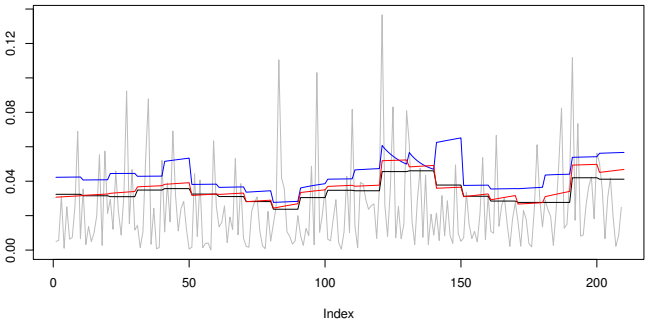
Grafici



(a) *previsioni 1 step ahead per bitcoin*

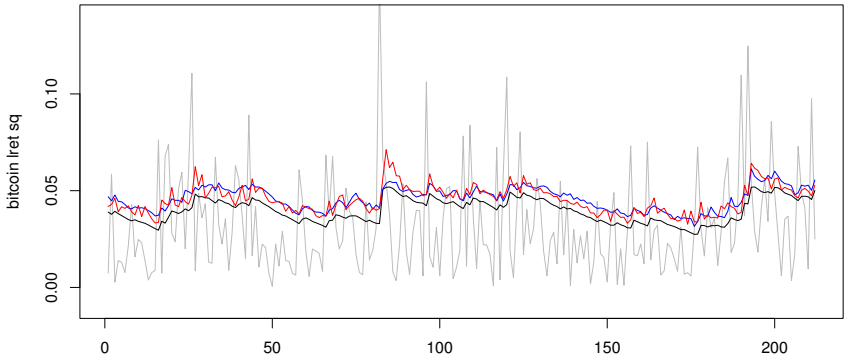


(b) *previsioni 5 step ahead per bitcoin*

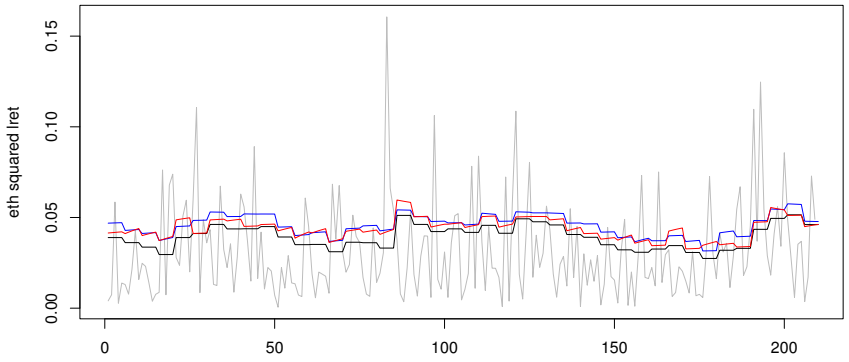


(c) *previsioni 10 step ahead per bitcoin*

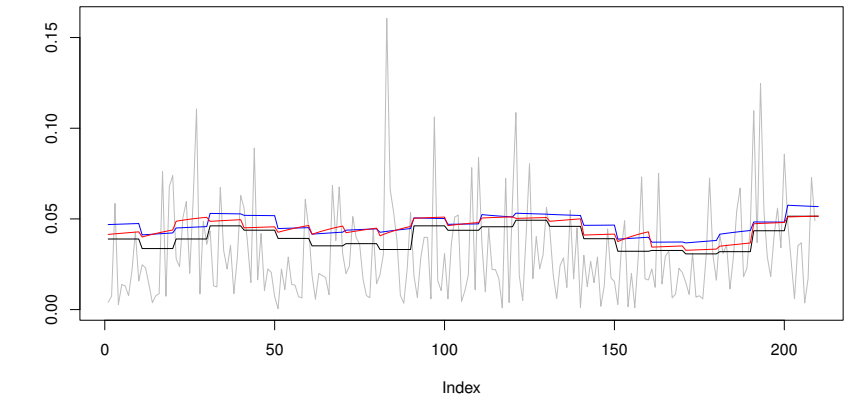
Figura A.1: Previsioni della volatilità per bitcoin. In **nero** il modello *RiskMetrics*, in **rosso** GARCH e in **blu** EGARCH, con distribuzione degli errori t di Student.



(a) *previsioni 1 step ahead per ether*

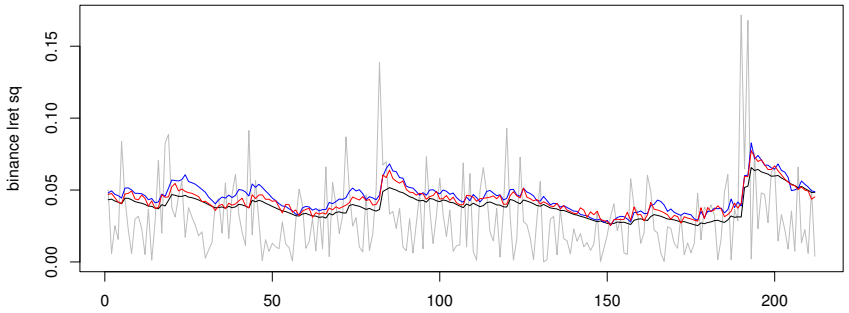


(b) *previsioni 5 step ahead per ether*

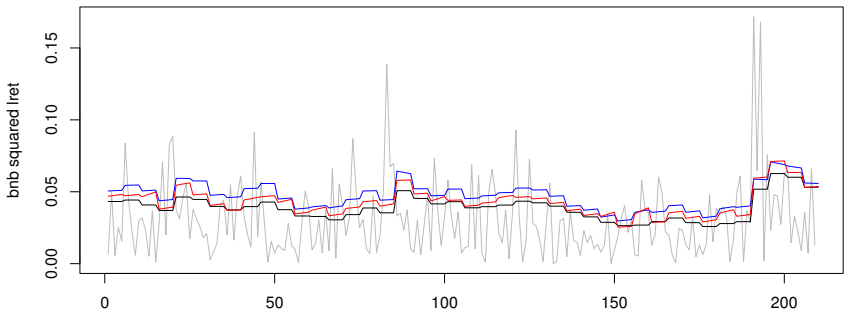


(c) *previsioni 10 step ahead per ether*

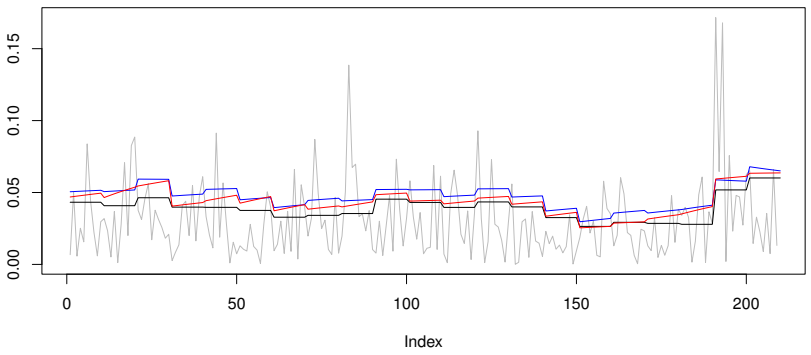
Figura A.2: Previsioni della volatilità per ether. In **nero** il modello *RiskMetrics*, in **rosso** GARCH e in **blu** EGARCH, con distribuzione degli errori GED.



(a) *previsioni 1 step ahead per binance*

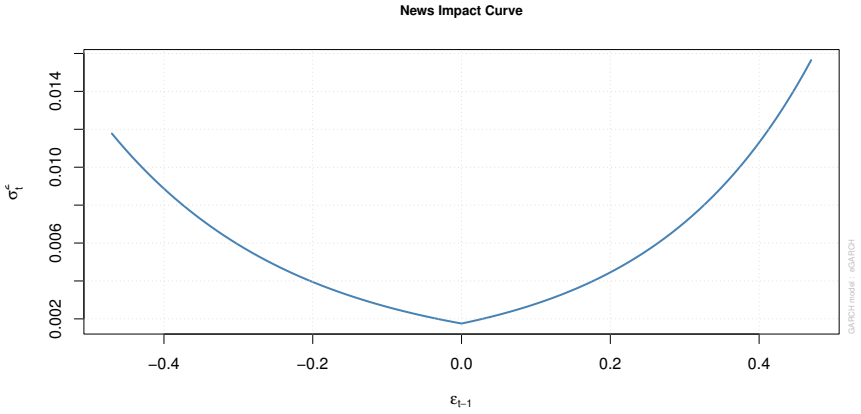


(b) *previsioni 5 step ahead per binance*

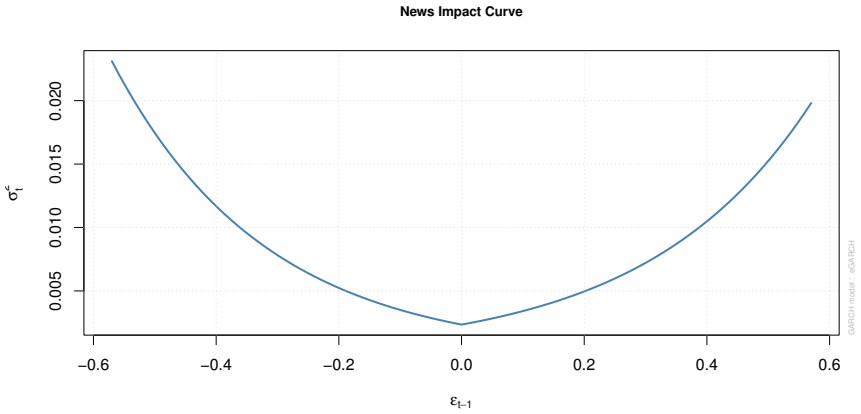


(c) *previsioni 10 step ahead per binance*

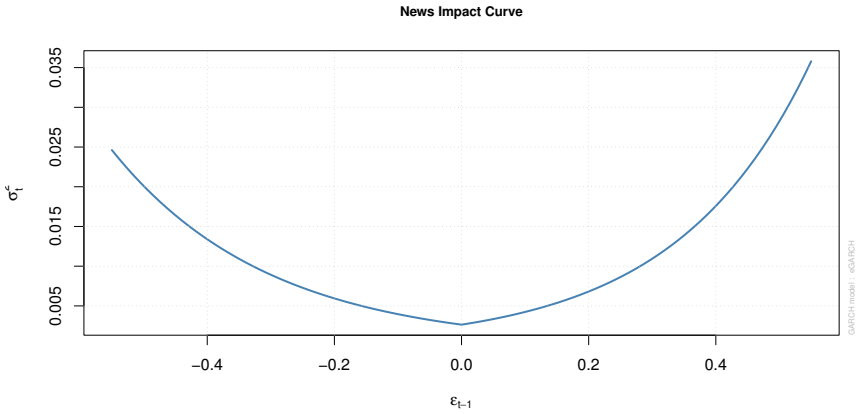
Figura A.3: Previsioni della volatilità per binance. In **nero** il modello *RiskMetrics*, in **rosso** GARCH e in **blu** EGARCH, con distribuzione degli errori t di Student.



(a) *bitcoin*



(b) *ether*



(c) *binance*

Figura A.4: *News Impact Curve* per i modelli EGARCH, con distribuzione GED per ether e t di Student per bitcoin e binance.

Appendice B

Codice R

```
library(tseries)
library(timeDate)
library(timeSeries)
library(zoo)
library(astsa)
library(urca)
library(fUnitRoots)
library(fBasics)
library(fImport)
library(stats)
library(fAssets)
library(forecast)
library(fPortfolio)
library(fRegression)
library(fTrading)
library(stabledist)
library(fGarch)
library(rugarch)
library(rmgarch)
library(xts)
library(rtsdata)
library(lmtest)
library(FinTS)
library(quantmod)
library(Metrics)
library(GAS)


#####################################################################
##############################BITCOIN
#####################################################################

dbtc<- read.csv("btc.csv", sep=",", header=T)
btc.y <- dbtc[,c(1,69)]
btc.y<-na.omit(btc.y)
btc<-as.timeSeries((btc.y))
```

```
#----grafico prezzi per tutto il periodo della serie
plot((seq.Date(as.Date("2010-07-18"), as.Date("2022-05-31"), by="day")
    ),btc[1:4336], type="l", xlab="time", ylab="BTC-USD")
#----grafico prezzi solo per il periodo considerato
plot( as.Date(btc.y$time[2480:4336]), btc.y$PriceUSD[2480:4336], type=
    "l", ylab="BTC-USD", xlab="tempo" ) #periodo analisi prezzi BTC

btctot.r <- na.omit(diff(log(btc.y[,2])))
btc.r <- btctot.r[2480:4335]

#----- grafico dei rendimenti per il periodo considerato
plot((seq.Date(as.Date("2017-05-02"), as.Date("2022-05-31"), by="day")
    ), btc.r,xlab="tempo", type="l") #rendimenti log BTC

#---- costruzione data-frame per Bitcoin
btc.p<-btc.y[2480:4336,]
n <- dim(btc.p)[1]
bitcoin <- data.frame(
  day = as.Date(btc.p$time)[-1],
  value = btc.p$PriceUSD[-1],
  lvalue = log(btc.p$PriceUSD)[-1],
  lret = log(btc.p$PriceUSD[2:n]) - log(btc.p$PriceUSD[1:(n-1)]),
  lret.sq = (log(btc.p$PriceUSD[2:n]) - log(btc.p$PriceUSD[1:(n-1)]))
      ^2,
  lret.abs = abs(log(btc.p$PriceUSD[2:n]) - log(btc.p$PriceUSD[1:(n-1)
      ]))
)

#-------- ACF E PACF RENDIMENTI BITCOIN
par(mfrow=c(1,2))
acf(btc.r, ylim=c(-0.1,0.1), main="")
pacf(btc.r, main="")

acf(btc.r^2, ylim=c(0,0.3), main="")
pacf(btc.r^2, main="")

acf(abs(btc.r), main="")
pacf(abs(btc.r), main="")
```

```
par(mfrow=c(1,1))
# ------------------statistiche base per tutte e 3 le cripto
round(basicStats(btc.r),5)
load("ethr.rda")
load("bnbr.rda")
sdes<-rbind(t(basicStats(bitcoin$lret)), t(basicStats(ether$lret)), t(
    basicStats(binance$lret)))
rownames(sdes)<- c("BTC", "ETH", "BNB")
xtable::xtable(sdes[,-c(2,5,6,9,10,11,12,13)], digits=5)

#------- Box-test e Arch Test
Box.test(btc.r^2, lag=8, type="Ljung–Box")
ArchTest(btc.r)

#----density plot
hist(btc.r, breaks=50, freq = F, ylim=c(0,17), xlim=c(-0.3,0.3), main=
    "")
lines(density(btc.r))
curve(dnorm(x,mean=mean(btc.r), sd=sd(btc.r)), col="red", add=T, xlim=
    c(-0.3,0.3))

#----QQ plot
qqnorm(btc.r)
qqline(btc.r)

#-----test di normalita' dei rendimenti
ksnormTest(btc.r)

#----modello per la media
auto.arima(btc.r)
m1<-arima(btc.r, order=c(1,0,0))
coeftest(m1)
AIC(m1)
acf(resid(m1)^2, lag=30, ylim=c(0,0.3))
pacf(resid(m1)^2, lag=30)
Box.test(resid(m1),lag=8 ,type="Ljung–Box")
Box.test(resid(m1)^2,lag=8 ,type="Ljung–Box")
```

```
#####################################################################
#####GRAFICO CORRELAZIONE BITCOIN, SP500, VIX
#####################################################################
sp500 <- (get.hist.quote("^GSPC", start="2017-05-01", end="2022-05-31"
    , quote="Close",retclass = "ts"))
gld <- (get.hist.quote("GLD", start="2017-05-01",end="2022-05-31",
    quote="Close", = "ts"))
vix <- (get.hist.quote("^VIX", start="2017-05-01", end="2022-05-31",
    quote="Close", retclass = "ts"))

rendi<-cbind( btc.r[-1],diff(log(as.numeric(vix))),diff(log(as.numeric
    (sp500))),(diff(log(as.numeric(gld)))))
names(rendi)<- c( "btc", "vix", "sp500", "gold", )
corre<-cor(rendi, use="pairwise.complete.obs")
colnames(corre)<- c("btc", "vix", "sp500", "gold")
rownames(corre)<- c("btc", "vix", "sp500", "gold")
corre
corrplot::corrplot(corre, method="square", addCoef.col = "black")

vix<-as.numeric(vix)
sp500 <-as.numeric(sp500)
dati1 <- (cbind(btc.r[-1], (diff(log(as.numeric((vix)))))))
dati2 <- (cbind(btc.r[-1], (diff(log(as.numeric((sp500)))))))
dati3 <- (cbind(btc.r[-1], (diff(log(as.numeric(gld))))))

RollCor<- (matrix(ncol=3, nrow=n-250))

for(i in (250+1):1856){
  RollCor[i-250,1]<- cor((dati1[(i-250):(i-1),1]), (dati1[(i-250):(i
      -1),2]), use="pairwise.complete.obs")
  RollCor[i-250,2]<- cor((dati2[(i-250):(i-1),1]), (dati2[(i-250):(i
      -1),2]), use="pairwise.complete.obs")
  RollCor[i-250,3]<- cor((dati3[(i-250):(i-1),1]), (dati3[(i-250):(i
      -1),2]), use="pairwise.complete.obs")
  }

cor(dati1, use="pairwise.complete.obs")
```

```
cor(dati2, use="pairwise.complete.obs")
cor(dati3, use="pairwise.complete.obs")

plot(RollCor[,1], type="l", ylim=c(-0.5,1), xlab="time", ylab="")
lines(RollCor[,2], type="l", col="violet")
lines(RollCor[,3], type="l", col="red")

legend("topleft",lty=1,col=c("black", "violet", "red"),
       legend=c("vix", "sp500", "gold"))

#####################################################################
##################BITCOIN MODELLI GARCH
#####################################################################
btc.r <- bitcoin$lret[1:1644]
nom<-c("norm", "snorm", "std", "sstd", "ged", "sged", "nig", "ghyp")
modelli <- matrix(nrow=4, ncol=8)
mbic <- matrix(nrow=4, ncol=8)
colnames(modelli)<-nom
colnames(mbic)<-nom
p=1
q=0
spec1<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit1<-ugarchfit(spec1, btc.r)

spec2<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit2<-ugarchfit(spec2, btc.r)

spec3<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="norm")
fit3<-ugarchfit(spec3, btc.r)

spec4<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
```

```
    distribution.model="norm")
fit4<-ugarchfit(spec4, btc.r)

modelli[1,1]<-infocriteria(fit1)[1]
modelli[2,1]<-infocriteria(fit2)[1]
modelli[3,1]<-infocriteria(fit3)[1]
modelli[4,1]<-infocriteria(fit4)[1]

mbic[1,1]<-infocriteria(fit1)[2]
mbic[2,1]<-infocriteria(fit2)[2]
mbic[3,1]<-infocriteria(fit3)[2]
mbic[4,1]<-infocriteria(fit4)[2]

spec1snorm<- ugarchspec(variance.model=list(model="sGARCH", garchOrder
    =c(1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit1snorm<-ugarchfit(spec1snorm, btc.r)

spec2snorm<- ugarchspec(variance.model=list(model="eGARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit2snorm<-ugarchfit(spec2snorm, btc.r)

spec3snorm<- ugarchspec(variance.model=list(model="fGARCH", garchOrder
    =c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T),
                    distribution.model="snorm")
fit3snorm<-ugarchfit(spec3snorm, btc.r)

spec4snorm<- ugarchspec(variance.model=list(model="apARCH", garchOrder
    =c(1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit4snorm<-ugarchfit(spec4snorm, btc.r)

modelli[1,2]<-infocriteria(fit1snorm)[1]
modelli[2,2]<-infocriteria(fit2snorm)[1]
modelli[3,2]<-infocriteria(fit3snorm)[1]
modelli[4,2]<-infocriteria(fit4snorm)[1]
```

```
mbic[1,2]<-infocriteria(fit1snorm)[2]
mbic[2,2]<-infocriteria(fit2snorm)[2]
mbic[3,2]<-infocriteria(fit3snorm)[2]
mbic[4,2]<-infocriteria(fit4snorm)[2]

spec1std<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit1std<-ugarchfit(spec1std, btc.r)

spec2std<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit2std<-ugarchfit(spec2std, btc.r, solver="hybrid")

spec3std<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="std")
fit3std<-ugarchfit(spec3std, btc.r)

spec4std<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit4std<-ugarchfit(spec4std, btc.r)
fit4std

modelli[1,3]<-infocriteria(fit1std)[1]
modelli[2,3]<-infocriteria(fit2std)[1]
modelli[3,3]<-infocriteria(fit3std)[1]
modelli[4,3]<-infocriteria(fit4std)[1]

mbic[1,3]<-infocriteria(fit1std)[2]
mbic[2,3]<-infocriteria(fit2std)[2]
mbic[3,3]<-infocriteria(fit3std)[2]
mbic[4,3]<-infocriteria(fit4std)[2]

spec1sstd<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
```

```
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit1sstd<-ugarchfit(spec1sstd, btc.r)

spec2sstd<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit2sstd<-ugarchfit(spec2sstd, btc.r)

spec3sstd<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T),  distribution.model="sstd")
fit3sstd<-ugarchfit(spec3sstd, btc.r)

spec4sstd<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit4sstd<-ugarchfit(spec4sstd, btc.r)

modelli[1,4]<-infocriteria(fit1sstd)[1]
modelli[2,4]<-infocriteria(fit2sstd)[1]
modelli[3,4]<-infocriteria(fit3sstd)[1]
modelli[4,4]<-infocriteria(fit4sstd)[1]

mbic[1,4]<-infocriteria(fit1sstd)[2]
mbic[2,4]<-infocriteria(fit2sstd)[2]
mbic[3,4]<-infocriteria(fit3sstd)[2]
mbic[4,4]<-infocriteria(fit4sstd)[2]

spec1ged<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit1ged<-ugarchfit(spec1ged, btc.r)

spec2ged<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit2ged<-ugarchfit(spec2ged, btc.r)
```

```
spec3ged<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel= "TGARCH"),  mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="ged")
fit3ged<-ugarchfit(spec3ged, btc.r, solver = "nloptr")

spec4ged<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit4ged<-ugarchfit(spec4ged, btc.r)

modelli[1,5]<-infocriteria(fit1ged)[1]
modelli[2,5]<-infocriteria(fit2ged)[1]
modelli[3,5]<-infocriteria(fit3ged)[1]
modelli[4,5]<-infocriteria(fit4ged)[1]

mbic[1,5]<-infocriteria(fit1ged)[2]
mbic[2,5]<-infocriteria(fit2ged)[2]
mbic[3,5]<-infocriteria(fit3ged)[2]
mbic[4,5]<-infocriteria(fit4ged)[2]

spec1sged<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit1sged<-ugarchfit(spec1sged, btc.r)

spec2sged<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit2sged<-ugarchfit(spec2sged, btc.r)

spec3sged<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="sged")
fit3sged<-ugarchfit(spec3sged, btc.r, solver="nloptr")

spec4sged<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
```

```
    distribution.model="sged")
fit4sged<-ugarchfit(spec4sged, btc.r)

modelli[1,6]<-infocriteria(fit1sged)[1]
modelli[2,6]<-infocriteria(fit2sged)[1]
modelli[3,6]<-infocriteria(fit3sged)[1]
modelli[4,6]<-infocriteria(fit4sged)[1]

mbic[1,6]<-infocriteria(fit1sged)[2]
mbic[2,6]<-infocriteria(fit2sged)[2]
mbic[3,6]<-infocriteria(fit3sged)[2]
mbic[4,6]<-infocriteria(fit4sged)[2]

spec1nig<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit1nig<-ugarchfit(spec1nig, btc.r)

spec2nig<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit2nig<-ugarchfit(spec2nig, btc.r)

spec3nig<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="nig")
fit3nig<-ugarchfit(spec3nig, btc.r)

spec4nig<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit4nig<-ugarchfit(spec4nig, btc.r)

modelli[1,7]<-infocriteria(fit1nig)[1]
modelli[2,7]<-infocriteria(fit2nig)[1]
modelli[3,7]<-infocriteria(fit3nig)[1]
modelli[4,7]<-infocriteria(fit4nig)[1]
```

```
mbic[1,7]<-infocriteria(fit1nig)[2]
mbic[2,7]<-infocriteria(fit2nig)[2]
mbic[3,7]<-infocriteria(fit3nig)[2]
mbic[4,7]<-infocriteria(fit4nig)[2]

spec1ghyp<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit1ghyp<-ugarchfit(spec1ghyp, btc.r)

spec2ghyp<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit2ghyp<-ugarchfit(spec2ghyp, btc.r)

spec3ghyp<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), .model="ghyp")
fit3ghyp<-ugarchfit(spec3ghyp, btc.r)

spec4ghyp<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit4ghyp<-ugarchfit(spec4ghyp, btc.r)

modelli[1,8]<-infocriteria(fit1ghyp)[1]
modelli[2,8]<-infocriteria(fit2ghyp)[1]
modelli[3,8]<-infocriteria(fit3ghyp)[1]
modelli[4,8]<-infocriteria(fit4ghyp)[1]

mbic[1,8]<-infocriteria(fit1ghyp)[2]
mbic[2,8]<-infocriteria(fit2ghyp)[2]
mbic[3,8]<-infocriteria(fit3ghyp)[2]
mbic[4,8]<-infocriteria(fit4ghyp)[2]
rownames(modelli)<-c("GARCH", "EGARCH", "TGARCH", "PGARCH")
rownames(mbic)<-c("GARCH", "EGARCH", "TGARCH", "PGARCH")
round(modelli,4)
library(xtable)
```

```
xtable::xtable(modelli, digits=c(0,rep(4, 8)))
xtable::xtable(mbic, digits=c(0,rep(4, 8)))

#####################################################################
##################PREVISIONE#########################################
#####################################################################
Tem=length(bitcoin$lret)
n=212 #1705 31-12-2021 #
N=Tem-n
previs=rep(NA,n)
rolling<-1500
pred1<-list()

#previsione 1 step ahead, RiskMetrics, BTC-USD
for(i in 1:n){
  mod.hw <- HoltWinters(bitcoin$lret[(N-rolling+(i-1)):(N+(i-1))]^2,
      alpha=(1.0-0.94), gamma=F, beta=F, start.periods = 1)
  pred1[["n1risk"]][i] <- predict(mod.hw,1)
  cat(i, "\n")
}

#previsione 1 step ahead, eGARCH STD, BTC-USD
for(i in 1:n){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=bitcoin$lret[(N-rolling+(i-1)):(N+(i
      -1))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 1)
  pred1[["n1eGARCH"]][i]=pred@forecast$sigmaFor
  cat(n-i, "\n")
}

#previsione 1 step ahead, sGARCH STD, BTC-USD
for(i in 1:n){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)), mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
```

```
  ug.fit1=ugarchfit(ug.spec,data=bitcoin$lret[(N-rolling+(i-1)):(N+(i
      -1))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 1)
  pred1[["n1sGARCH"]][i]=pred@forecast$sigmaFor
  cat(n-i, "\n")
}

observed <-  bitcoin$lret.sq[(N+1):(Tem)]

plot(sqrt(bitcoin$lret.sq[(N+1):Tem]),
      type="l", col="grey", ylab="bitcoin lret sq", ylim=c(-0.01,0.14)
          , xlab="")
lines(sqrt(pred1$n1risk), col="black")
lines((pred1$n1eGARCH), col="blue")
lines((pred1$n1sGARCH), col="red")

tabpred<- data.frame(matrix(nrow=9, ncol=2))
rownames(tabpred)<- c("risk1","egarch1","sgarch1",
                      "risk5","egarch5","sgarch5",
                      "risk10","egarch10","sgarch10")
colnames(tabpred)<- c("mae", "rmse")

tabpred[1,1]<-mae(observed, pred1$n1risk)
tabpred[1,2]<-rmse(observed, pred1$n1risk)
tabpred[2,1]<-mae(observed, pred1$n1eGARCH^2)
tabpred[2,2]<-rmse(observed, pred1$n1eGARCH^2)
tabpred[3,1]<-mae(observed, pred1$n1sGARCH^2)
tabpred[3,2]<-rmse(observed, pred1$n1sGARCH^2)
#####################################################################
##########PREVISIONI 5 STEP AHEAD
#####################################################################
for(i in 1:(n%/%5)){
  mod.hw <- HoltWinters(bitcoin$lret[(N-rolling+(5*(i-1))):(N+(5*(i-1)
      ))]^2, alpha=(1.0-0.94), gamma=F, beta=F, start.periods = 1)
  pred1[["n5risk"]] <- c(pred1[["n5risk"]],predict(mod.hw,5))
  cat(i, "\n")
}
```

```
for(i in 1:(n%/%5)){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=bitcoin$lret[(N-rolling+(5*(i-1))):(N
      +(5*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 5)
  pred1[["n5eGARCH"]]=c(pred1[["n5eGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%5, "\n")
}

for(i in 1:(n%/%5)){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=bitcoin$lret[(N-rolling+(5*(i-1))):(N
      +(5*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 5)
  pred1[["n5sGARCH"]]=c(pred1[["n5sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%5, "\n")
}

observed5 <-  bitcoin$lret.abs[(Tem-(n-1)):(Tem-(n%%5))]

plot(sqrt(bitcoin$lret.sq[N:(Tem-4)]), type="l", col="grey", ylab="btc
    squared lret",
    xlab="")
lines(sqrt(pred1$n5risk), col="black")
lines((pred1$n5eGARCH), col="blue")
lines((pred1$n5sGARCH), col="red")

tabpred[4,1]<-mae(observed5, pred1$n5risk)
tabpred[4,2]<-rmse(observed5, pred1$n5risk)
tabpred[5,1]<-mae(observed5, pred1$n5eGARCH^2)
tabpred[5,2]<-rmse(observed5, pred1$n5eGARCH^2)
tabpred[6,1]<-mae(observed5, pred1$n5sGARCH^2)
tabpred[6,2]<-rmse(observed5, pred1$n5sGARCH^2)
```

```
#####################################################################
#########PREVISIONI 10 STEP AHEAD
#####################################################################
for(i in 1:(n%/%10)){
  mod.hw <- HoltWinters(bitcoin$lret[(N-rolling+(10*(i-1))):(N+(10*(i
      -1)))]^2, alpha=(1.0-0.94), gamma=F, beta=F, start.periods = 1)
  pred1[["n10risk"]] <- c(pred1[["n10risk"]],predict(mod.hw,10))
  cat(i, "\n")
}

for(i in 1:(n%/%10)){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=bitcoin$lret[(N-rolling+(10*(i-1))):(
      N+(10*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 10)
  pred1[["n10eGARCH"]]=c(pred1[["n10eGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%10, "\n")
}

for(i in 1:(n%/%10)){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=bitcoin$lret[(N-rolling+(10*(i-1))):(
      N+(10*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 10)
  pred1[["n10sGARCH"]]=c(pred1[["n10sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%10, "\n")
}

observed10 <-  bitcoin$lret.abs[(Tem-(n-1)):(Tem-(n%%10))]

plot(sqrt(bitcoin$lret.sq[N:(Tem-4)]), type="l", col="grey", ylab="")
lines(sqrt(pred1$n10risk), col="black")
lines((pred1$n10eGARCH), col="blue")
lines((pred1$n10sGARCH), col="red")
```

```
tabpred[7,1]<-mae(observed10, pred1$n10risk)
tabpred[7,2]<-rmse(observed10, pred1$n10risk)
tabpred[8,1]<-mae(observed10, pred1$n10eGARCH^2)
tabpred[8,2]<-rmse(observed10, pred1$n10eGARCH^2)
tabpred[9,1]<-mae(observed10, pred1$n10sGARCH^2)
tabpred[9,2]<-rmse(observed10, pred1$n10sGARCH^2)

xtable(tabpred, digits=6)
#save(pred1, file="pred1btc.rda")

######################################################################
##########Value at Risk######
######################################################################
VaR <- list()
# ---Step 2: previsioni e stima del VaR col modello selezionato
alfa.95   <- qstd(0.05, nu = fit2std@fit$coef["shape"])
alfa.99   <- qstd(0.01, nu = fit2std@fit$coef["shape"])

alfa2.95   <- qstd(0.05, nu = fit1std@fit$coef["shape"])
alfa2.99   <- qstd(0.01, nu = fit1std@fit$coef["shape"])

rollwin <- 1500
1. ---stima del VaR con EGARCH(1,1)
for(i in 1:n){
  cat("mancano",n-i,"\n")
  y.rollwin <- bitcoin$lret[(N+i-1-1500):(N+i-1)]
  garch.rollwin <- ugarchfit(spec2std, data = y.rollwin,solver="nloptr
      ")
  previsione <- ugarchforecast(garch.rollwin, n.ahead = 1)
  VaR[["VaRegarch.95"]][i]<-previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa.95
  VaR[["VaRegarch.99"]][i]<-previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa.99
}
2. ---stima del VaR GARCH(1,1)
for(i in 1:n){
  cat("mancano",n-i,"\n")
```

```
  y.rollwin <- bitcoin$lret[(N+i-1-1500):(N+i-1)]
  garch.rollwin <- ugarchfit(spec1std, data = y.rollwin,solver="nloptr
      ")
  previsione <- ugarchforecast(garch.rollwin, n.ahead = 1)
  VaR[["VaRsgarch.95"]][i] <- previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa2.95
  VaR[["VaRsgarch.99"]][i] <- previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa2.99
}

# 3. ---stima del VaR con l'approccio RiskMetrics
lambda <- 0.94 # -> equivalente ad un modello IGARCH(1,1) con omega =
    0 e alpha = 1-0.94
nuest = (6/kurtosis(btc.r)[1])+4
alfa.95   <- qstd(0.05, nu = nuest)
alfa.99   <- qstd(0.01, nu = nuest)
for (i in 1:n) {
  y.rollwin <- bitcoin$lret[(N+i-1-1500):(N+i-1)]
  RiskMetrics <- HoltWinters(y.rollwin^2, alpha = (1-lambda), gamma =
      FALSE, beta = FALSE, start.periods = 1)
  vol.RiskMetrics <- predict(RiskMetrics, 1)
  VaR[["VaRrisk.95"]][i] <- sqrt(vol.RiskMetrics) * alfa.95
  VaR[["VaRrisk.99"]][i]<- sqrt(vol.RiskMetrics) * alfa.99
}

y <- bitcoin$lret[(Tem-(n-1)):Tem]

plot(y,type="l", ylim=c(-0.18,0.14), ylab="btc")
lines(VaR$VaRegarch.95, col="blue")
lines(VaR$VaRrisk.95, col="black")
lines(VaR$VaRsgarch.95, col="red")

plot(y,type="l", ylim=c(-0.25,0.14), ylab="btc")
lines(VaR$VaRegarch.99, col="blue")
lines(VaR$VaRrisk.99, col="black")
lines(VaR$VaRsgarch.99, col="red")

Var95 <- matrix(cbind(VaR$VaRrisk.95, VaR$VaRsgarch.95, VaR$VaRegarch
```

```
    .95), ncol=3)
resvar95 <- matrix(0, 3,6)
for(z in 1:3){
  backtest<-BacktestVaR(y, Var95[,z], alpha=0.05, Lags=4)
  resvar95[z, 1] <- backtest$AE
  resvar95[z, 2] <-backtest$AD[1]
  resvar95[z, 3] <-backtest$AD[2]
  resvar95[z, 4] <-sum(y<Var95[,z])
  resvar95[z, 5] <-backtest$LRuc[2]
  resvar95[z, 6] <-backtest$LRcc[2]
}

resvar95

Var99 <- matrix(cbind(VaR$VaRrisk.99, VaR$VaRsgarch.99, VaR$VaRegarch
    .99), ncol=3)
resvar99 <- matrix(0, 3,6)
for(z in 1:3){
  backtest<-BacktestVaR(y, Var99[,z], alpha=0.01, Lags=4)
  resvar99[z, 1] <- backtest$AE
  resvar99[z, 2] <-backtest$AD[1] #deviazione media in abs
  resvar99[z, 3] <-backtest$AD[2] #deviazione massima in abs
  resvar99[z, 4] <-sum(y<Var99[,z]) # numero di violazioni
  resvar99[z, 5] <-backtest$LRuc[2] # ipotesi nulla alfaobs=alfateo
  resvar99[z, 6] <-backtest$LRcc[2] # ipotesi di dipendenza delle
      deviazioni tipo catena di markov
}


#########################################################################
    ETHER
    ##########################################################################################

rm(list=ls())

data2<- read.csv("eth.csv", header = T, sep=",")
eth <- na.omit(data2[,c(1,72)])
#----------plot serie storica
```

```
plot((seq.Date(as.Date("2015-08-08"), as.Date("2022-05-31"), by="day")
    ), eth[1:2489,2], type="l", xlab="time", ylab="ETH-USD")

eth2 <- na.omit(data2[642:2498,c(1,72)])
eth <- na.omit(data2[642:2498,c(1,72)])

#-------plot serie storica periodo considerato
plot( as.Date(eth$time),eth$PriceUSD, type="l", xlab="tempo", ylab="
    ETH-USD")

#-------plot dei rendimenti
plot( as.Date(eth$time[-1]),diff(log(eth$PriceUSD)), type="l", xlab="
    tempo", ylab="eth.r")
head(eth)

# ------------costruzione del data-frame per ethereum
n <- dim(eth2)[1]
ether <- data.frame(
  day = as.Date(eth2$time)[-1],
  value = eth2$PriceUSD[-1],
  lvalue = log(eth2$PriceUSD)[-1],
  lret = log(eth2$PriceUSD[2:n]) - log(eth2$PriceUSD[1:(n-1)]),
  lret.sq = (log(eth2$PriceUSD[2:n]) - log(eth2$PriceUSD[1:(n-1)]))^2,
  lret.abs = abs(log(eth2$PriceUSD[2:n]) - log(eth2$PriceUSD[1:(n-1)])
      )
)
eth.r <- ether$lret

#------------statistiche base ethereum
round(basicStats(eth.r),5)
ksnormTest(eth.r)
#-----------density plot ethereum
hist(eth.r, freq=F, breaks=50, ylim=c(0,15), xlim=c(-0.3, 0.3), main="
    ")
lines(density(eth.r), ylim=c(0,15))
curve(dnorm(x, mean(eth.r), sd(eth.r)), add=T, col="red")
#--------------ACF E PACF per Ether
par(mfrow=c(1,2))
```

```
acf(eth.r, ylim=c(-0.1,0.2), main="")
pacf(eth.r, main="")

acf(eth.r^2, ylim=c(-0.1,0.2), main="")
pacf(eth.r^2, main="")

acf(abs(eth.r), main="")
pacf(abs(eth.r), main="")

Box.test(as.numeric(eth.r),lag=8, type="Ljung–Box")
Box.test(as.numeric(eth.r^2),lag=8, type="Ljung–Box")
ArchTest(eth.r)

#-------QQ-PLOT ether
qqnorm(eth.r)
qqline(eth.r)

eth.r<- ether$lret[1:1644]
acf(eth.r, ylim=c(-0.1,0.3))
pacf(eth.r)
Box.test(eth.r, lag=8, type="Ljung–Box")
auto.arima(eth.r)
m1e<-arima(eth.r, order=c(3,0,3))
m1e
coeftest(m1e)
Box.test(resid(m1e), lag=8, type="Ljung–Box")
acf(resid(m1e), ylim=c(-0.1,0.2))
pacf(resid(m1e))
Box.test(resid(m1e)^2, lag=8, type=("Ljung–Box"))
acf(resid(m1e)^2, ylim=c(-0.1,0.2))
pacf(resid(m1e)^2)

######################################################################
######################MODELLI GARCH
######################################################################
eth.r<- ether$lret[1:1644]
nom<-c("norm", "snorm", "std", "sstd", "ged", "sged", "nig", "ghyp")
modelli <- matrix( nrow=4, ncol=8)
```

```
mbic <- matrix(nrow=4, ncol=8)
colnames(modelli)<-nom
colnames(mbic)<-nom

p=3
q=3
spec1<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit1<-ugarchfit(spec1, eth.r)

spec2<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit2<-ugarchfit(spec2, eth.r)

spec3<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="norm")
fit3<-ugarchfit(spec3, eth.r)

spec4<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),mean.model=list(armaOrder=c(p,q), include.mean=T),
                   distribution.model="norm")
fit4<-ugarchfit(spec4, eth.r)

modelli[1,1]<-infocriteria(fit1)[1]
modelli[2,1]<-infocriteria(fit2)[1]
modelli[3,1]<-infocriteria(fit3)[1]
modelli[4,1]<-infocriteria(fit4)[1]

mbic[1,1]<-infocriteria(fit1)[2]
mbic[2,1]<-infocriteria(fit2)[2]
mbic[3,1]<-infocriteria(fit3)[2]
mbic[4,1]<-infocriteria(fit4)[2]

spec1snorm<- ugarchspec(variance.model=list(model="sGARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
```

```
    distribution.model="snorm")
fit1snorm<-ugarchfit(spec1snorm, eth.r)

spec2snorm<- ugarchspec(variance.model=list(model="eGARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit2snorm<-ugarchfit(spec2snorm, eth.r)

spec3snorm<- ugarchspec(variance.model=list(model="fGARCH", garchOrder
    =c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="snorm")
fit3snorm<-ugarchfit(spec3snorm, eth.r)

spec4snorm<- ugarchspec(variance.model=list(model="apARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit4snorm<-ugarchfit(spec4snorm, eth.r)

modelli[1,2]<-infocriteria(fit1snorm)[1]
modelli[2,2]<-infocriteria(fit2snorm)[1]
modelli[3,2]<-infocriteria(fit3snorm)[1]
modelli[4,2]<-infocriteria(fit4snorm)[1]

mbic[1,2]<-infocriteria(fit1snorm)[2]
mbic[2,2]<-infocriteria(fit2snorm)[2]
mbic[3,2]<-infocriteria(fit3snorm)[2]
mbic[4,2]<-infocriteria(fit4snorm)[2]

spec1std<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit1std<-ugarchfit(spec1std, eth.r)

spec2std<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit2std<-ugarchfit(spec2std, eth.r, solver="hybrid")
fit2std
```

```
spec3std<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel="TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="std")
fit3std<-ugarchfit(spec3std, eth.r)

spec4std<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit4std<-ugarchfit(spec4std, eth.r)

modelli[1,3]<-infocriteria(fit1std)[1]
modelli[2,3]<-infocriteria(fit2std)[1]
modelli[3,3]<-infocriteria(fit3std)[1]
modelli[4,3]<-infocriteria(fit4std)[1]

mbic[1,3]<-infocriteria(fit1std)[2]
mbic[2,3]<-infocriteria(fit2std)[2]
mbic[3,3]<-infocriteria(fit3std)[2]
mbic[4,3]<-infocriteria(fit4std)[2]

spec1sstd<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit1sstd<-ugarchfit(spec1sstd, eth.r)

spec2sstd<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit2sstd<-ugarchfit(spec2sstd, eth.r)

spec3sstd<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1), submodel="TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="sstd")
fit3sstd<-ugarchfit(spec3sstd, eth.r)

spec4sstd<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
```

```
    distribution.model="sstd")
fit4sstd<-ugarchfit(spec4sstd, eth.r)

modelli[1,4]<-infocriteria(fit1sstd)[1]
modelli[2,4]<-infocriteria(fit2sstd)[1]
modelli[3,4]<-infocriteria(fit3sstd)[1]
modelli[4,4]<-infocriteria(fit4sstd)[1]

mbic[1,4]<-infocriteria(fit1sstd)[2]
mbic[2,4]<-infocriteria(fit2sstd)[2]
mbic[3,4]<-infocriteria(fit3sstd)[2]
mbic[4,4]<-infocriteria(fit4sstd)[2]

spec1ged<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit1ged<-ugarchfit(spec1ged, eth.r)

spec2ged<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit2ged<-ugarchfit(spec2ged, eth.r)

spec3ged<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1),  submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="ged")
fit3ged<-ugarchfit(spec3ged, eth.r, solver = "nloptr")

spec4ged<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit4ged<-ugarchfit(spec4ged, eth.r)

modelli[1,5]<-infocriteria(fit1ged)[1]
modelli[2,5]<-infocriteria(fit2ged)[1]
modelli[3,5]<-infocriteria(fit3ged)[1]
modelli[4,5]<-infocriteria(fit4ged)[1]
```

```
mbic[1,5]<-infocriteria(fit1ged)[2]
mbic[2,5]<-infocriteria(fit2ged)[2]
mbic[3,5]<-infocriteria(fit3ged)[2]
mbic[4,5]<-infocriteria(fit4ged)[2]

spec1sged<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit1sged<-ugarchfit(spec1sged, eth.r)

spec2sged<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit2sged<-ugarchfit(spec2sged, eth.r)

spec3sged<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T),   distribution.model="sged")
fit3sged<-ugarchfit(spec3sged, eth.r, solver="nloptr")

spec4sged<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit4sged<-ugarchfit(spec4sged, eth.r)

modelli[1,6]<-infocriteria(fit1sged)[1]
modelli[2,6]<-infocriteria(fit2sged)[1]
modelli[3,6]<-infocriteria(fit3sged)[1]
modelli[4,6]<-infocriteria(fit4sged)[1]

mbic[1,6]<-infocriteria(fit1sged)[2]
mbic[2,6]<-infocriteria(fit2sged)[2]
mbic[3,6]<-infocriteria(fit3sged)[2]
mbic[4,6]<-infocriteria(fit4sged)[2]

spec1nig<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
```

```
fit1nig<-ugarchfit(spec1nig, eth.r)

spec2nig<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit2nig<-ugarchfit(spec2nig, eth.r)

spec3nig<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel="TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="nig")
fit3nig<-ugarchfit(spec3nig, eth.r)

spec4nig<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit4nig<-ugarchfit(spec4nig, eth.r)

modelli[1,7]<-infocriteria(fit1nig)[1]
modelli[2,7]<-infocriteria(fit2nig)[1]
modelli[3,7]<-infocriteria(fit3nig)[1]
modelli[4,7]<-infocriteria(fit4nig)[1]

mbic[1,7]<-infocriteria(fit1nig)[2]
mbic[2,7]<-infocriteria(fit2nig)[2]
mbic[3,7]<-infocriteria(fit3nig)[2]
mbic[4,7]<-infocriteria(fit4nig)[2]

spec1ghyp<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit1ghyp<-ugarchfit(spec1ghyp, eth.r)

spec2ghyp<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit2ghyp<-ugarchfit(spec2ghyp, eth.r)

spec3ghyp<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
```

```
    c(1,1), submodel= "TGARCH"),  mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="ghyp")
fit3ghyp<-ugarchfit(spec3ghyp, eth.r)

spec4ghyp<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit4ghyp<-ugarchfit(spec4ghyp, eth.r)

modelli[1,8]<-infocriteria(fit1ghyp)[1]
modelli[2,8]<-infocriteria(fit2ghyp)[1]
modelli[3,8]<-infocriteria(fit3ghyp)[1]
modelli[4,8]<-infocriteria(fit4ghyp)[1]

mbic[1,8]<-infocriteria(fit1ghyp)[2]
mbic[2,8]<-infocriteria(fit2ghyp)[2]
mbic[3,8]<-infocriteria(fit3ghyp)[2]
mbic[4,8]<-infocriteria(fit4ghyp)[2]

rownames(modelli)<-c("GARCH", "EGARCH", "TGARCH", "PGARCH")
rownames(mbic)<-c("GARCH", "EGARCH", "TGARCH", "PGARCH")
round(modelli,4)
#####################################################################
##################PREVISIONE#########################################
#####################################################################
Tem=length(ether$lret)
n=212
N=Tem-n
previs=rep(NA,n)

#####################################################################
##########PREVISIONI 1 STEP AHEAD
#####################################################################
rolling<-1500
pred1<-list()
#vol.hw=rep(NA,n)
#previsione 1 step ahead, RiskMetrics, BNB-USD
for(i in 1:n){
```

```
  mod.hw <- HoltWinters(ether$lret[(N-rolling+(i-1)):(N+(i-1))]^2,
      alpha=(1.0-0.94), gamma=F,beta=F, start.periods = 1)
  pred1[["n1risk"]][i] <- predict(mod.hw,1)
  cat(i, "\n")
}

#previsione 1 step ahead, eGARCH, BNB-USD
for(i in 1:n){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "ged")
  ug.fit1=ugarchfit(ug.spec,data=ether$lret[(N-rolling+(i-1)):(N+(i-1)
      )],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 1)
  pred1[["n1eGARCH"]][i]=pred@forecast$sigmaFor
  cat(n-i, "\n")
}
#previsione 1 step ahead, sGARCH, BNB-USD
for(i in 1:n){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=ether$lret[(N-rolling+(i-1)):(N+(i-1)
      )],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 1)
  pred1[["n1sGARCH"]][i]=pred@forecast$sigmaFor
  cat(n-i, "\n")
}

observed <-  ether$lret.sq[(N+1):(Tem)]

plot(sqrt(ether$lret.sq[(N+1):Tem]),
     type="l", col="grey", ylab="bitcoin lret sq", ylim=c(-0.01,0.14),
          xlab="")
lines(sqrt(pred1$n1risk), col="black")
lines(sqrt(pred1$n1eGARCH)^2, col="blue")
lines(sqrt(pred1$n1sGARCH)^2, col="red")
```

```
tabpred<- data.frame(matrix(nrow=9, ncol=2))
rownames(tabpred)<- c("risk1","egarch1","sgarch1",
                      "risk5","egarch5","sgarch5",
                      "risk10","egarch10","sgarch10")
colnames(tabpred)<- c("mae", "rmse")

tabpred[1,1]<-mae(observed, pred1$n1risk)
tabpred[1,2]<-rmse(observed, pred1$n1risk)
tabpred[2,1]<-mae(observed, pred1$n1eGARCH^2)
tabpred[2,2]<-rmse(observed, pred1$n1eGARCH^2)
tabpred[3,1]<-mae(observed, pred1$n1sGARCH^2)
tabpred[3,2]<-rmse(observed, pred1$n1sGARCH^2)
tabpred
#####################################################################
##########PREVISIONI 5 STEP AHEAD
#####################################################################
for(i in 1:(n%/%5)){
  mod.hw <- HoltWinters(ether$lret[(N-rolling+(5*(i-1))):(N+(5*(i-1)))
      ]^2, alpha=(1.0-0.94), gamma=F,beta=F, start.periods = 1)
  pred1[["n5risk"]] <- c(pred1[["n5risk"]],predict(mod.hw,5))
  cat(i, "\n")
}

for(i in 1:(n%/%5)){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "ged")
  ug.fit1=ugarchfit(ug.spec,data=ether$lret[(N-rolling+(5*(i-1))):(N
      +(5*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 5)
  pred1[["n5eGARCH"]]=c(pred1[["n5eGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%5, "\n")
}

for(i in 1:(n%/%5)){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "ged")
```

```
  ug.fit1=ugarchfit(ug.spec,data=ether$lret[(N-rolling+(5*(i-1))):(N
      +(5*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 5)
  pred1[["n5sGARCH"]]=c(pred1[["n5sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%5, "\n")
}

observed5 <-  ether$lret.abs[(Tem-(n-1)):(Tem-(n%%5))]

plot(sqrt(ether$lret.sq[N:(Tem-4)]), type="l", col="grey", ylab="eth
    squared lret",
     xlab="")
lines(sqrt(pred1$n5risk), col="black")
lines((pred1$n5eGARCH), col="blue")
lines((pred1$n5sGARCH), col="red")

tabpred[4,1]<-mae(observed5, pred1$n5risk)
tabpred[4,2]<-rmse(observed5, pred1$n5risk)
tabpred[5,1]<-mae(observed5, pred1$n5eGARCH^2)
tabpred[5,2]<-rmse(observed5, pred1$n5eGARCH^2)
tabpred[6,1]<-mae(observed5, pred1$n5sGARCH^2)
tabpred[6,2]<-rmse(observed5, pred1$n5sGARCH^2)



######################################################################
##########PREVISIONI 10 STEP AHEAD
######################################################################
for(i in 1:(n%/%10)){
  mod.hw <- HoltWinters(ether$lret[(N-rolling+(10*(i-1))):(N+(10*(i-1)
      ))]^2, alpha=(1.0-0.94), gamma=F, beta=F, start.periods = 1)
  pred1[["n10risk"]] <- c(pred1[["n10risk"]],predict(mod.hw,10))
  cat(i, "\n")
}

for(i in 1:(n%/%10)){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "ged")
```

```
  ug.fit1=ugarchfit(ug.spec,data=ether$lret[(N-rolling+(10*(i-1))):(N
      +(10*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 10)
  pred1[["n10sGARCH"]]=c(pred1[["n10sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%10, "\n")
}

for(i in 1:(n%/%10)){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "ged")
  ug.fit1=ugarchfit(ug.spec,data=ether$lret[(N-rolling+(10*(i-1))):(N
      +(10*(i-1)))], solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 10)
  pred1[["n10sGARCH"]]=c(pred1[["n10sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%10, "\n")
}

observed10 <-  ether$lret.abs[(Tem-(n-1)):(Tem-(n%%10))]

plot(sqrt(ether$lret.sq[N:(Tem-4)]), type="l", col="grey", ylab="")
lines(sqrt(pred1$n10risk), col="black")
lines((pred1$n10eGARCH), col="blue")
lines((pred1$n10sGARCH), col="red")

tabpred[7,1]<-mae(observed10, pred1$n10risk)
tabpred[7,2]<-rmse(observed10, pred1$n10risk)
tabpred[8,1]<-mae(observed10, pred1$n10eGARCH^2)
tabpred[8,2]<-rmse(observed10, pred1$n10eGARCH^2)
tabpred[9,1]<-mae(observed10, pred1$n10sGARCH^2)
tabpred[9,2]<-rmse(observed10, pred1$n10sGARCH^2)

#####################################################################
##########Value at Risk
    ###################################################################
VaR <- list()
# ---Step 2: previsioni e stima del VaR col modello selezionato
alfa.95   <- qged(0.05, nu = fit2ged@fit$coef["shape"])
```

```
alfa.99   <- qged(0.01, nu = fit2ged@fit$coef["shape"])

alfa2.95   <- qged(0.05, nu = fit1ged@fit$coef["shape"])
alfa2.99   <- qged(0.01, nu = fit1ged@fit$coef["shape"])

rollwin <- 1500
1. ---stima del VaR con EGARCH
for(i in 1:n){
  cat("mancano",n-i,"\n")
  y.rollwin <- ether$lret[(N+i-1-1500):(N+i-1)]
  garch.rollwin <- ugarchfit(spec2ged, data = y.rollwin,solver="nloptr
      ")
  previsione <- ugarchforecast(garch.rollwin, n.ahead = 1)
  VaR[["VaRegarch.95"]][i]<-previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa.95
  VaR[["VaRegarch.99"]][i]<-previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa.99
}
2. ---stima del VaR con GARCH
for(i in 1:n){
  cat("mancano",n-i,"\n")
  y.rollwin <- ether$lret[(N+i-1-1500):(N+i-1)]
  garch.rollwin <- ugarchfit(spec1ged, data = y.rollwin,solver="nloptr
      ")
  previsione <- ugarchforecast(garch.rollwin, n.ahead = 1)
  VaR[["VaRsgarch.95"]][i] <- previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa2.95
  VaR[["VaRsgarch.99"]][i] <- previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa2.99
}

# 3. ---stima del VaR con l'approccio RiskMetrics
lambda <- 0.94
barshape = (fit2ged@fit$coef["shape"]+fit1ged@fit$coef["shape"])/2
alfa3.95 <- qged(0.05, nu = barshape)
alfa3.99 <- qged(0.01, nu = barshape)
for (i in 1:n) {
  y.rollwin <- ether$lret[(N+i-1-1500):(N+i-1)]
```

```
  RiskMetrics <- HoltWinters(y.rollwin^2, alpha = (1-lambda), gamma =
      FALSE, beta = FALSE, start.periods = 1)
  vol.RiskMetrics <- predict(RiskMetrics, 1)
  VaR[["VaRrisk.95"]][i] <- sqrt(vol.RiskMetrics) * alfa3.95
  VaR[["VaRrisk.99"]][i]<- sqrt(vol.RiskMetrics) * alfa3.99
}

y <- ether$lret[(Tem-(n-1)):Tem]

plot(y,type="l", ylim=c(-0.18,0.14), ylab="eth")
lines(VaR$VaRegarch.95, col="blue")
lines(VaR$VaRrisk.95, col="black")
lines(VaR$VaRsgarch.95, col="red")

plot(y,type="l", ylim=c(-0.25,0.14), ylab="eth")
lines(VaR$VaRegarch.99, col="blue")
lines(VaR$VaRrisk.99, col="black")
lines(VaR$VaRsgarch.99, col="red")

Var95 <- matrix(cbind(VaR$VaRrisk.95, VaR$VaRsgarch.95, VaR$VaRegarch
    .95), ncol=3)
resvar95 <- matrix(0, 3,6)
for(z in 1:3){
  backtest<-BacktestVaR(y, Var95[,z], alpha=0.05, Lags=4)
  resvar95[z, 1] <- backtest$AE
  resvar95[z, 2] <-backtest$AD[1]
  resvar95[z, 3] <-backtest$AD[2]
  resvar95[z, 4] <-sum(y<Var95[,z])
  resvar95[z, 5] <-backtest$LRuc[2]
  resvar95[z, 6] <-backtest$LRcc[2]
}

resvar95

Var99 <- matrix(cbind(VaR$VaRrisk.99, VaR$VaRsgarch.99, VaR$VaRegarch
    .99), ncol=3)
resvar99 <- matrix(0, 3,6)
for(z in 1:3){
```

```
  backtest<-BacktestVaR(y, Var99[,z], alpha=0.01, Lags=4)
  resvar99[z, 1] <- backtest$AE
  resvar99[z, 2] <-backtest$AD[1] #deviazione media in abs
  resvar99[z, 3] <-backtest$AD[2] #deviazione massima in abs
  resvar99[z, 4] <-sum(y<Var99[,z]) # numero di violazioni
  resvar99[z, 5] <-backtest$LRuc[2] # ipotesi nulla alfaobs=alfateo
  resvar99[z, 6] <-backtest$LRcc[2] # ipotesi di dipendenza delle
      deviazioni tipo catena di markov
}
revar99

######################################################################
##########Binance Coin
######################################################################
rm(list=ls())

data3 <- read.csv("BNB-USD.csv", header=T)
head(data3)
tail(data3)
#---------------GARFICO SERIE STORICA
plot(seq(as.Date("2017-11-09"),as.Date("2022-05-31"), by="days"), bnb,
     ylab="BNB-USD", type="l", xlab="time")
head(bnb)

n <- dim(data3)[1]
binance <- data.frame(
  day = as.Date(data3$Date)[-1],
  value = data3$Close[-1],
  lvalue = log(data3$Close)[-1],
  lret = log(data3$Close[2:n]) - log(data3$Close[1:(n-1)]),
  lret.sq = (log(data3$Close[2:n]) - log(data3$Close[1:(n-1)]))^2,
  lret.abs = abs(log(data3$Close[2:n]) - log(data3$Close[1:(n-1)]))
)

bnb.r<- diff(log(bnb))
#--------------GRAFICO RENDIMENTI
plot(seq(as.Date("2017-11-10"),as.Date("2021-12-31"), by="days"), bnb.
    r[1:1513],type="l", xlab="time" )
```

```
round(basicStats(bnb.r),5)
#---------DENSITY PLOT
hist(bnb.r, freq=F, breaks=50, ylim=c(0,15), main="")
lines(density(bnb.r), ylim=c(0,15))
curve(dnorm(x, mean=mean(bnb.r), sd=sd(bnb.r)), col="red", add=T)
#-------------ACF E PACF
par(mfrow=c(1,2))
acf(bnb.r, ylim=c(-0.1,0.2),main="")
pacf(bnb.r, main="" )

acf(bnb.r^2, ylim=c(0,0.3),main="")
pacf(bnb.r^2, main="" )

acf(abs(bnb.r),main="")
pacf(abs(bnb.r), main="" )
#------------------Box.test, test di normalitÃă e arch test
ksnormTest(bnb.r)
Box.test(as.numeric(bnb.r),lag=8, type="Ljung–Box")
Box.test((bnb.r)^2,lag=8, type="Ljung–Box")
ArchTest((bnb.r))
par(mfrow=c(1,1))
#----------------------QQ-plot
qqnorm(bnb.r)
qqline(bnb.r)

#--------------------modelli per la media
bnb.r <- binance$lret[1:1452]
auto.arima(bnb.r)
m1e<-arima(bnb.r, order=c(0,0,2), include.mean = T)
AIC(m1e)
coeftest(m1e)
Box.test(resid(m1e),lag=8, type="Ljung–Box")
acf(resid(m1e), ylim=c(-0.1,0.2))
pacf(resid(m1e))
#####################################################################
############################MODELLI
#####################################################################
```

```
nom<-c("norm", "snorm", "std", "sstd", "ged", "sged", "nig", "ghyp")
modelli <- matrix( nrow=4, ncol=8)
colnames(modelli)<-nom
bnb.r <- binance$lret[1:1452]
mbic <- matrix(nrow=4, ncol=8)
colnames(mbic)<-nom
p=0
q=2
spec1<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit1<-ugarchfit(spec1, bnb.r)

spec2<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit2<-ugarchfit(spec2, bnb.r)

spec3<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1), submodel= "TGARCH"),mean.model=list(armaOrder=c(p,q),
    include.mean=T),  distribution.model="norm")
fit3<-ugarchfit(spec3, bnb.r)

spec4<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="norm")
fit4<-ugarchfit(spec4, bnb.r)

modelli[1,1]<-infocriteria(fit1)[1]
modelli[2,1]<-infocriteria(fit2)[1]
modelli[3,1]<-infocriteria(fit3)[1]
modelli[4,1]<-infocriteria(fit4)[1]

mbic[1,1]<-infocriteria(fit1)[2]
mbic[2,1]<-infocriteria(fit2)[2]
mbic[3,1]<-infocriteria(fit3)[2]
mbic[4,1]<-infocriteria(fit4)[2]
```

```
spec1snorm<- ugarchspec(variance.model=list(model="sGARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit1snorm<-ugarchfit(spec1snorm, bnb.r)

spec2snorm<- ugarchspec(variance.model=list(model="eGARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit2snorm<-ugarchfit(spec2snorm, bnb.r)

spec3snorm<- ugarchspec(variance.model=list(model="fGARCH", garchOrder
    =c(1,1), submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="snorm")
fit3snorm<-ugarchfit(spec3snorm, bnb.r)

spec4snorm<- ugarchspec(variance.model=list(model="apARCH", garchOrder
    =c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="snorm")
fit4snorm<-ugarchfit(spec4snorm, bnb.r)

modelli[1,2]<-infocriteria(fit1snorm)[1]
modelli[2,2]<-infocriteria(fit2snorm)[1]
modelli[3,2]<-infocriteria(fit3snorm)[1]
modelli[4,2]<-infocriteria(fit4snorm)[1]

mbic[1,2]<-infocriteria(fit1snorm)[2]
mbic[2,2]<-infocriteria(fit2snorm)[2]
mbic[3,2]<-infocriteria(fit3snorm)[2]
mbic[4,2]<-infocriteria(fit4snorm)[2]

spec1std<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit1std<-ugarchfit(spec1std, bnb.r)

spec2std<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
```

```
fit2std<-ugarchfit(spec2std, bnb.r)

spec3std<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1),  submodel= "TGARCH"),  mean.model=list(armaOrder=c(p,q),
    include.mean=T),  distribution.model="std")
fit3std<-ugarchfit(spec3std, bnb.r)

spec4std<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="std")
fit4std<-ugarchfit(spec4std, bnb.r)

modelli[1,3]<-infocriteria(fit1std)[1]
modelli[2,3]<-infocriteria(fit2std)[1]
modelli[3,3]<-infocriteria(fit3std)[1]
modelli[4,3]<-infocriteria(fit4std)[1]

mbic[1,3]<-infocriteria(fit1std)[2]
mbic[2,3]<-infocriteria(fit2std)[2]
mbic[3,3]<-infocriteria(fit3std)[2]
mbic[4,3]<-infocriteria(fit4std)[2]

spec1sstd<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)),   mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit1sstd<-ugarchfit(spec1sstd, bnb.r)

spec2sstd<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)),   mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit2sstd<-ugarchfit(spec2sstd, bnb.r)

spec3sstd<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1),  submodel= "TGARCH"),  mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="sstd")
fit3sstd<-ugarchfit(spec3sstd, bnb.r)

spec4sstd<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
```

```
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sstd")
fit4sstd<-ugarchfit(spec4sstd, bnb.r)

modelli[1,4]<-infocriteria(fit1sstd)[1]
modelli[2,4]<-infocriteria(fit2sstd)[1]
modelli[3,4]<-infocriteria(fit3sstd)[1]
modelli[4,4]<-infocriteria(fit4sstd)[1]

mbic[1,4]<-infocriteria(fit1sstd)[2]
mbic[2,4]<-infocriteria(fit2sstd)[2]
mbic[3,4]<-infocriteria(fit3sstd)[2]
mbic[4,4]<-infocriteria(fit4sstd)[2]

spec1ged<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit1ged<-ugarchfit(spec1ged, bnb.r)

spec2ged<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit2ged<-ugarchfit(spec2ged, bnb.r)

spec3ged<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1),  submodel= "TGARCH"), mean.model=list(armaOrder=c(p,q),
    include.mean=T),   distribution.model="ged")
fit3ged<-ugarchfit(spec3ged, bnb.r)

spec4ged<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ged")
fit4ged<-ugarchfit(spec4ged, bnb.r)

modelli[1,5]<-infocriteria(fit1ged)[1]
modelli[2,5]<-infocriteria(fit2ged)[1]
modelli[3,5]<-infocriteria(fit3ged)[1]
modelli[4,5]<-infocriteria(fit4ged)[1]
```

```
mbic[1,5]<-infocriteria(fit1ged)[2]
mbic[2,5]<-infocriteria(fit2ged)[2]
mbic[3,5]<-infocriteria(fit3ged)[2]
mbic[4,5]<-infocriteria(fit4ged)[2]

spec1sged<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit1sged<-ugarchfit(spec1sged, bnb.r)

spec2sged<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit2sged<-ugarchfit(spec2sged, bnb.r)

spec3sged<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1),  submodel= "TGARCH"),  mean.model=list(armaOrder=c(p,q),
    include.mean=T),  distribution.model="sged")
fit3sged<-ugarchfit(spec3sged, bnb.r)

spec4sged<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="sged")
fit4sged<-ugarchfit(spec4sged, bnb.r)

modelli[1,6]<-infocriteria(fit1sged)[1]
modelli[2,6]<-infocriteria(fit2sged)[1]
modelli[3,6]<-infocriteria(fit3sged)[1]
modelli[4,6]<-infocriteria(fit4sged)[1]

mbic[1,6]<-infocriteria(fit1sged)[2]
mbic[2,6]<-infocriteria(fit2sged)[2]
mbic[3,6]<-infocriteria(fit3sged)[2]
mbic[4,6]<-infocriteria(fit4sged)[2]

spec1nig<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
```

```
    distribution.model="nig")
fit1nig<-ugarchfit(spec1nig, bnb.r)

spec2nig<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit2nig<-ugarchfit(spec2nig, bnb.r)

spec3nig<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=c
    (1,1),  submodel= "TGARCH"),  mean.model=list(armaOrder=c(p,q),
    include.mean=T),
                      distribution.model="nig")
fit3nig<-ugarchfit(spec3nig, bnb.r)

spec4nig<- ugarchspec(variance.model=list(model="apARCH", garchOrder=c
    (1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="nig")
fit4nig<-ugarchfit(spec4nig, bnb.r)

modelli[1,7]<-infocriteria(fit1nig)[1]
modelli[2,7]<-infocriteria(fit2nig)[1]
modelli[3,7]<-infocriteria(fit3nig)[1]
modelli[4,7]<-infocriteria(fit4nig)[1]

mbic[1,7]<-infocriteria(fit1nig)[2]
mbic[2,7]<-infocriteria(fit2nig)[2]
mbic[3,7]<-infocriteria(fit3nig)[2]
mbic[4,7]<-infocriteria(fit4nig)[2]

spec1ghyp<- ugarchspec(variance.model=list(model="sGARCH", garchOrder=
    c(1,1)),   mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit1ghyp<-ugarchfit(spec1ghyp, bnb.r)

spec2ghyp<- ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)),  mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit2ghyp<-ugarchfit(spec2ghyp, bnb.r)
```

```
spec3ghyp<- ugarchspec(variance.model=list(model="fGARCH", garchOrder=
    c(1,1),  submodel= "TGARCH"),   mean.model=list(armaOrder=c(p,q),
    include.mean=T), distribution.model="ghyp")
fit3ghyp<-ugarchfit(spec3ghyp, bnb.r)

spec4ghyp<- ugarchspec(variance.model=list(model="apARCH", garchOrder=
    c(1,1)), mean.model=list(armaOrder=c(p,q), include.mean=T),
    distribution.model="ghyp")
fit4ghyp<-ugarchfit(spec4ghyp, bnb.r)

modelli[1,8]<-infocriteria(fit1ghyp)[1]
modelli[2,8]<-infocriteria(fit2ghyp)[1]
modelli[3,8]<-infocriteria(fit3ghyp)[1]
modelli[4,8]<-infocriteria(fit4ghyp)[1]

mbic[1,8]<-infocriteria(fit1ghyp)[2]
mbic[2,8]<-infocriteria(fit2ghyp)[2]
mbic[3,8]<-infocriteria(fit3ghyp)[2]
mbic[4,8]<-infocriteria(fit4ghyp)[2]
rownames(modelli)<-c("GARCH", "EGARCH", "TGARCH", "PGARCH")
rownames(mbic)<-c("GARCH", "EGARCH", "TGARCH", "PGARCH")
round(modelli,4)

######################################################################
##################PREVISIONE
######################################################################

Tem=length(binance$lret)
n=212
N=Tem-n
previs=rep(NA,n)

######################################################################
##########PREVISIONI 1 STEP AHEAD
######################################################################
rolling<-1452
pred1<-list()
```

```
#vol.hw=rep(NA,n)

for(i in 1:n){
  mod.hw <- HoltWinters(binance$lret[(N-rolling+(i-1)):(N+(i-1))]^2,
      alpha=(1.0-0.94), gamma=F, beta=F, start.periods = 1)
  pred1[["n1risk"]][i] <- predict(mod.hw,1)
  cat(i, "\n")
}

#previsione 1 step ahead, eGARCH STD, BTC-USD
for(i in 1:n){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "ged")
  ug.fit1=ugarchfit(ug.spec,data=binance$lret[(N-rolling+(i-1)):(N+(i
      -1))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 1)
  pred1[["n1eGARCH"]][i]=pred@forecast$sigmaFor
  cat(n-i, "\n")
}

for(i in 1:n){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=binance$lret[(N-rolling+(i-1)):(N+(i
      -1))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 1)
  pred1[["n1sGARCH"]][i]=pred@forecast$sigmaFor
  cat(n-i, "\n")
}

observed <-  binance$lret.sq[(N+1):(Tem)]

plot(sqrt(binance$lret.sq[(N+1):Tem]),
     type="l", col="grey", ylab="binance lret sq", xlab="")
lines(sqrt(pred1$n1risk), col="black")
lines(sqrt(pred1$n1eGARCH)^2, col="blue")
```

```
lines(sqrt(pred1$n1sGARCH)^2, col="red")
#legend("topleft",lty=1,col=c("black", "blue", "red"),
#legend=c("riskm", "EGARCH-std", "GARCH-std"))
dev.print(pdf,
          "C:/Users/Bruno/Desktop/Scuola+uni/21.tesi/immagini/imgpdf/
              pred1BNB.pdf")

tabpred<- data.frame(matrix(nrow=9, ncol=2))
rownames(tabpred)<- c("risk1","egarch1","sgarch1",
                      "risk5","egarch5","sgarch5",
                      "risk10","egarch10","sgarch10")
colnames(tabpred)<- c("mae", "rmse")

tabpred[1,1]<-mae(observed, pred1$n1risk)
tabpred[1,2]<-rmse(observed, pred1$n1risk)
tabpred[2,1]<-mae(observed, pred1$n1eGARCH^2)
tabpred[2,2]<-rmse(observed, pred1$n1eGARCH^2)
tabpred[3,1]<-mae(observed, pred1$n1sGARCH^2)
tabpred[3,2]<-rmse(observed, pred1$n1sGARCH^2)
tabpred
#####################################################################
##########PREVISIONI 5 STEP AHEAD
#####################################################################
for(i in 1:(n%/%5)){
  mod.hw <- HoltWinters(binance$lret[(N-rolling+(5*(i-1))):(N+(5*(i-1)
      ))]^2, alpha=(1.0-0.94), gamma=F,beta=F, start.periods = 1)
  pred1[["n5risk"]] <- c(pred1[["n5risk"]],predict(mod.hw,5))
  cat(i, "\n")
}

for(i in 1:(n%/%5)){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=binance$lret[(N-rolling+(5*(i-1))):(N
      +(5*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 5)
  pred1[["n5eGARCH"]]=c(pred1[["n5eGARCH"]], pred@forecast$sigmaFor)
```

```
  cat(i, "su", n%/%5, "\n")
}

for(i in 1:(n%/%5)){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=binance$lret[(N-rolling+(5*(i-1))):(N
      +(5*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 5)
  pred1[["n5sGARCH"]]=c(pred1[["n5sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%5, "\n")
}

observed5 <-  binance$lret.abs[(Tem-(n-1)):(Tem-(n%%5))]

plot(sqrt(binance$lret.sq[N:(Tem-4)]), type="l", col="grey", ylab="bnb
     squared lret",
     xlab="")
lines(sqrt(pred1$n5risk), col="black")
lines((pred1$n5eGARCH), col="blue")
lines((pred1$n5sGARCH), col="red")

tabpred[4,1]<-mae(observed5, pred1$n5risk)
tabpred[4,2]<-rmse(observed5, pred1$n5risk)
tabpred[5,1]<-mae(observed5, pred1$n5eGARCH^2)
tabpred[5,2]<-rmse(observed5, pred1$n5eGARCH^2)
tabpred[6,1]<-mae(observed5, pred1$n5sGARCH^2)
tabpred[6,2]<-rmse(observed5, pred1$n5sGARCH^2)

#####################################################################
##########PREVISIONI 10 STEP AHEAD
#####################################################################
for(i in 1:(n%/%10)){
  mod.hw <- HoltWinters(binance$lret[((10*(i-1))):(N+(10*(i-1)))]^2,
      alpha=(1.0-0.94), gamma=F, beta=F, start.periods = 1)
  pred1[["n10risk"]] <- c(pred1[["n10risk"]],predict(mod.hw,10))
  cat(i, "\n")
```

```
}

for(i in 1:(n%/%10)){
  ug.spec=ugarchspec(variance.model = list(model="eGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=binance$lret[(N-rolling+(10*(i-1))):(
      N+(10*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 10)
  pred1[["n10eGARCH"]]=c(pred1[["n10eGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%10, "\n")
}

for(i in 1:(n%/%10)){
  ug.spec=ugarchspec(variance.model = list(model="sGARCH",garchOrder=c
      (1,1)),mean.model=list(armaOrder=c(p,q),include.mean=T),
      distribution.model = "std")
  ug.fit1=ugarchfit(ug.spec,data=binance$lret[(N-rolling+(10*(i-1))):(
      N+(10*(i-1)))],solver="nloptr")
  pred=ugarchforecast(ug.fit1, n.ahead = 10)
  pred1[["n10sGARCH"]]=c(pred1[["n10sGARCH"]], pred@forecast$sigmaFor)
  cat(i, "su", n%/%10, "\n")
}

observed10 <-  binance$lret.abs[(Tem-(n-1)):(Tem-(n%%10))]

plot(sqrt(binance$lret.sq[N:(Tem-4)]), type="l", col="grey", ylab="")
lines(sqrt(pred1$n10risk), col="black")
lines((pred1$n10eGARCH), col="blue")
lines((pred1$n10sGARCH), col="red")

tabpred[7,1]<-mae(observed10, pred1$n10risk)
tabpred[7,2]<-rmse(observed10, pred1$n10risk)
tabpred[8,1]<-mae(observed10, pred1$n10eGARCH^2)
tabpred[8,2]<-rmse(observed10, pred1$n10eGARCH^2)
tabpred[9,1]<-mae(observed10, pred1$n10sGARCH^2)
tabpred[9,2]<-rmse(observed10, pred1$n10sGARCH^2)
```

```
xtable(tabpred, digits=6)

#####################################################################
#########Value at Risk
#####################################################################
VaR <- list()
# ---Step 2: previsioni e stima del VaR col modello selezionato
alfa.95   <- qstd(0.05, nu = fit2std@fit$coef["shape"])
alfa.99   <- qstd(0.01, nu = fit2std@fit$coef["shape"])
#nig beta=skewness par, alpha= shape
alfa2.95   <- qstd(0.05, nu = fit1std@fit$coef["shape"])
alfa2.99   <- qstd(0.01, nu = fit1std@fit$coef["shape"])

rollwin <- 1425
# 1. ---stima del VaR con EGARCH
for(i in 1:n){
  cat("mancano",n-i,"\n")
  y.rollwin <- binance$lret[(N+i-1-rollwin):(N+i-1)]
  garch.rollwin <- ugarchfit(spec2std, data = y.rollwin,solver="nloptr
      ")
  previsione <- ugarchforecast(garch.rollwin, n.ahead = 1)
  VaR[["VaRegarch.95"]][i]<-previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa.95
  VaR[["VaRegarch.99"]][i]<-previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa.99
}
# 2. ---stima del VaR con GARCH
for(i in 1:n){
  cat("mancano",n-i,"\n")
  y.rollwin <- binance$lret[(N+i-1-rollwin):(N+i-1)]
  garch.rollwin <- ugarchfit(spec1std, data = y.rollwin,solver="nloptr
      ")
  previsione <- ugarchforecast(garch.rollwin, n.ahead = 1)
  VaR[["VaRsgarch.95"]][i] <- previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa2.95
  VaR[["VaRsgarch.99"]][i] <- previsione@forecast$seriesFor+
      previsione@forecast$sigmaFor*alfa2.99
}
```

```
# 3. ---stima del VaR con l'approccio RiskMetrics
lambda <- 0.94 # -> equivalente ad un modello IGARCH(1,1) con omega =
    0 e alpha = 1-0.94
nuest = (6/kurtosis(bnb.r)[1])+4
alfa3.95   <- qstd(0.05, nu = nuest)
alfa3.99   <- qstd(0.01, nu = nuest)
for (i in 1:n) {
  y.rollwin <- binance$lret[(N+i-1-rollwin):(N+i-1)]
  RiskMetrics <- HoltWinters(y.rollwin^2, alpha = (1-lambda), gamma =
      FALSE, beta = FALSE, start.periods = 1)
  vol.RiskMetrics <- predict(RiskMetrics, 1)
  VaR[["VaRrisk.95"]][i] <- sqrt(vol.RiskMetrics) * alfa3.95
  VaR[["VaRrisk.99"]][i]<- sqrt(vol.RiskMetrics) * alfa3.99
}

#save(VaR, file="VaRbnb.rda")
y <- binance$lret[(Tem-(n-1)):Tem]

plot(y,type="l", ylim=c(-0.18,0.14), ylab="bnb")
lines(VaR$VaRegarch.95, col="blue")
lines(VaR$VaRrisk.95, col="black")
lines(VaR$VaRsgarch.95, col="red")



plot(y,type="l", ylim=c(-0.25,0.14), ylab="bnb")
lines(VaR$VaRegarch.99, col="blue")
lines(VaR$VaRrisk.99, col="black")
lines(VaR$VaRsgarch.99, col="red")

Var95 <- matrix(cbind(VaR$VaRrisk.95, VaR$VaRsgarch.95, VaR$VaRegarch
    .95), ncol=3)
resvar95 <- matrix(0, 3,6)
for(z in 1:3){
  backtest<-BacktestVaR(y, Var95[,z], alpha=0.05, Lags=4)
  resvar95[z, 1] <- backtest$AE
  resvar95[z, 2] <-backtest$AD[1]
  resvar95[z, 3] <-backtest$AD[2]
```

```
  resvar95[z, 4] <-sum(y<Var95[,z])
  resvar95[z, 5] <-backtest$LRuc[2]
  resvar95[z, 6] <-backtest$LRcc[2]
}

resvar95

Var99 <- matrix(cbind(VaR$VaRrisk.99, VaR$VaRsgarch.99, VaR$VaRegarch
    .99), ncol=3)
resvar99 <- matrix(0, 3,6)
for(z in 1:3){
  backtest<-BacktestVaR(y, Var99[,z], alpha=0.01, Lags=4)
  resvar99[z, 1] <- backtest$AE
  resvar99[z, 2] <-backtest$AD[1] #deviazione media in abs
  resvar99[z, 3] <-backtest$AD[2] #deviazione massima in abs
  resvar99[z, 4] <-sum(y<Var99[,z]) # numero di violazioni
  resvar99[z, 5] <-backtest$LRuc[2] # ipotesi nulla alfaobs=alfateo
  resvar99[z, 6] <-backtest$LRcc[2] # ipotesi di dipendenza delle
      deviazioni tipo catena di markov
}
resvar99
```