Università degli studi di Milano Bicocca

Decision Models

Final Project

Ottimizzazione di un portafoglio mediante la massimizzazione dell'indice di Sharpe

Authors:
Alessandro Riboni - 847160 - a.riboni2@campus.unimib.it
Federico Signoretta - 847343 - f.signoretta@campus.unimib.it
Diana Tenca - 789651- d.tenca@campus.unimib.it

July,15, 2019

Sommario

L'ottimizzazione di un portafoglio di titoli è uno dei problemi di maggior interesse nella matematica finanziaria. Facendo riferimento alla teoria del portafoglio di *Markowitz*, si definisce *portafoglio ottimo* la combinazione dei diversi titoli che massimizza l'indice di *Sharpe*, un indicatore dell'extra-rendimento rispetto ad un titolo *risk-free*.
In questo paper verranno presentate due metodologie risolutive per la ricerca del portafoglio ottimo. Nella prima verrà applicato iterativamente un algoritmo genetico con *crossing-over* aritmetico al fine di garantire il soddisfacimento dei vincoli di non negatività dei pesi. Nella seconda metodologia implementata, invece, i vincoli verranno inseriti all'interno della funzione obiettivo mediante l'utilizzo di un algoritmo di penalità. Dalle soluzioni quasi-ottimali ottenute con i due metodi sopracitati, verrà presentata una strategia per la ricerca locale al fine di raggiungere l'ottimo della funzione obiettivo.
In conclusione, verranno stimati i prezzi futuri dei titoli precedentemente scelti - attraverso un modello ARIMA - grazie ai quali sarà possibile ottenere il ritorno del portafoglio efficiente.

1 Introduzione

L'ottimizzazione di un portafoglio di titoli è uno dei problemi di maggior interesse nella matematica finanziaria. Seguendo la teoria di Markowitz, il miglior modo per ottenere il portafoglio ottimo è minimizzandone la volatilità massimizzandone il rendimento.

1.1 La teoria di Markowitz

Harry Markowitz è un economista statunitense noto per la sua teoria sull'ottimizzazione di un portafoglio finanziario meglio conosciuta come *teoria moderna del portafoglio*. Essa si fonda sull'idea che l'investitore razionale sia avverso al rischio e quindi portato ad investire in una combinazione di titoli tali da massimizzare il rendimento e minimizzare la volatilità.
Per semplicità la teoria di Markowitz verrà presentata nel caso di un portafoglio costituito da due soli titoli A e B con B più volatile di A. Si introducono alcune formule matematiche che descrivono il portafoglio. Il ritorno atteso è calcolato come la media pesata dei ritorni dei singoli titoli ossia:

$$\mathbb{E}(R_p) = \omega_A \mathbb{E}(R_A) + \omega_B \mathbb{E}(R_B) = \omega_A \mathbb{E}(R_A) + (1 - \omega_A)\mathbb{E}(R_B) \tag{1}$$

dove $\mathbb{E}(R_p)$ è il ritorno atteso del portafoglio e ω_i è il peso associato all'i-esimo titolo. La volatilità del portafoglio si calcola con la seguente formula:

$${\sigma_p}^2 = {\omega_A}^2 {\sigma_A}^2 + {\omega_B}^2 {\sigma_B}^2 + 2\omega_A \omega_B \sigma_A \sigma_B \rho_{AB} \tag{2}$$

dove σ_i è la deviazione standard di ogni titolo e ρ_{ij} è la correlazione tra gli stock.

La figura 1 rappresenta tutte le possibili combinazioni dei titoli in funzione del ritorno atteso e della deviazione standard, rispettivamente sull'asse delle ordinate e delle ascisse. Quest'insieme di combinazioni è anche noto come **frontiera efficiente** così chiamata perché è costituita dai portafogli che corrispondono al massimo rendimento atteso rispetto ad una data volatilità. Benché si possa pensare che il portafoglio MVP, ossia quello a associato alla minor varianza, sia il portafoglio ottimale in realtà non è così. Il portafoglio ottimale, anche detto portafoglio tangente, è quello che massimizza l'**indice di Sharpe**. L'indice di Sharpe è un indicatore dell'extra-rendimento rispetto al tasso risk-free ossia, è il rapporto tra il guadagno atteso del portafoglio

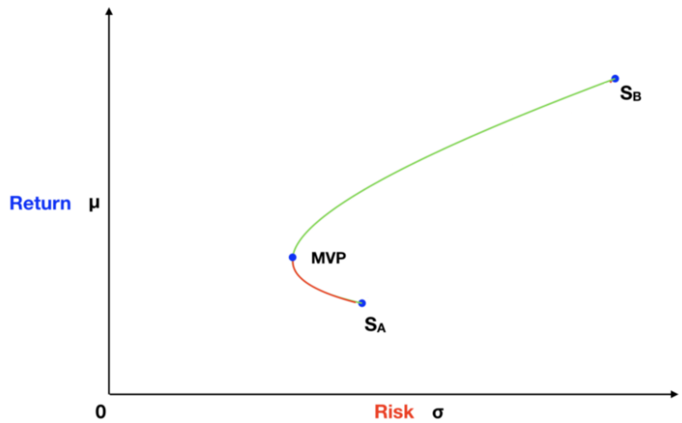

Figura 1: Portafogli efficienti per i titoli A e B

rispetto ad un titolo risk-free[1] e la volatilità. A livello matematico corrisponde:

$$S = \frac{R_P - R_{rf}}{\sigma_R} \tag{3}$$

dove R_P è il ritorno atteso del portafoglio, R_{rf} è il ritorno atteso del titolo risk-free e σ_P è la volatilità del portafoglio. In conclusione, l'indice di Sharpe è una misura che ottimizza il rendimento minimizzando la volatilità.

1.2 Ottimizzazione non-lineare vincolata

La ricerca di minimi e massimi costituisce una grande parte dell'analisi matematica e la complessità di questa ricerca dipende dalla funzione da ottimizzare e dalla presenza o meno di vincoli. Si parla di ottimizzazione non lineare vincolata quando la funzione è non lineare e sono presenti dei vincoli che riducono la regione ammissibile delle soluzioni.
Gli approcci risolutivi per problemi generici di ottimizzazione non-lineare possono essere classificati in tre categorie:

- adattamento al caso vincolato degli algoritmi per l'ottimizzazione svincolata
- tecniche basate sul rilassamento lagrangiano

[1] Si definisce titolo risk-free un titolo a varianza nulla che solitamente corrisponde ai buoni del tesoro di un paese economicamente stabile

- algoritmi di penalità

La metodologia risolutiva che verrà utilizzata per l'ottimizzazione e la semplificazione del problema sopra-descritto rientra negli **algoritmi di penalità**. Questi algoritmi sono basati sull'idea di rilassare alcuni vincoli del problema e di tenerne conto nella funzione obiettivo, definendo un problema ausiliario P_c:

$$min[P(x,c)] = f(x) + c\phi(x), \quad x \in \mathbb{R}^n \tag{4}$$

dove $c \geq 0$ è un parametro e $\phi : \mathbb{R}^n \to R$ è una funzione di penalità legata ai vincoli rilassati. Questa funzione deve soddisfare le seguenti condizioni:

- ϕ deve essere continua
- $\phi(\bar{x}) = 0 \ \forall \bar{x} \in S$
- $\phi(\bar{x}) > 0 \ \forall \bar{x} \notin S$

Una delle possibili scelte per la funzione $\phi(x)$ è:

$$\phi(x) = \sum_{i=1}^{m} [max(g_i(x), 0)]^2 + \sum_{j=1}^{p} [max(h_j(x), 0)]^2 \tag{5}$$

così da avere una penalizzazione proporzionale ai vincoli. Inoltre, questa scelta permette di avere una funzione differenziabile anche se con delle discontinuità nella derivata prima. Intuitivamente, per valori piccoli del parametro c la funzione obiettivo tiene poco conto del soddisfacimento dei vincoli, per cui è probabile che la soluzione ottima di (4) sia non ammissibile. Viceversa, per c grande, le soluzioni non ammissibili risultano fortemente penalizzate. Quindi, minimizzando $P(x,c)$ è probabile che si determini una soluzione ammissibile ma, non necessariamente ottima, dato che si utilizza una funzione obiettivo diversa da $f(x)$.

2 Datasets

Per effettuare le analisi di ottimizzazione del portafoglio sono stati utilizzati gli storici di 10 titoli attraverso la funzione *getSymbols* del pacchetto *quantmod*, attraverso il quale è stato possibile interfacciarci al portale *Yahoo Finance*. In particolare, è stato selezionato uno storico di *6* anni: dal *01-07-2013* al *01-07-2019*.
I titoli selezionati per l'analisi sono i seguenti: **AMZ** (Amazon), **AAPL** (Apple), **IBM** (IMB Corporation), **FB** (Facebook), **MSFT** (Microsoft), **JPM** (J.P. Morgan), **DIS** (Disney), **GOOG** (Google), **NFLX** (Netflix) e **TSLA** (Tesla).

3 L'approccio metodologico

L'analisi ha come scopo la ricerca del portafoglio ottimo, sotto i vincoli di non-negatività dei pesi, mediante la massimizzazione dell'indice di Sharpe. Le metodologie utilizzate per trovare i pesi ottimali dei titoli su cui investire sono entrambe basate su algoritmi genetici:

- con **arithmetic crossing-over** e funzione obiettivo ***S*** (*Sharpe Ratio*)
- con **two-point crossing-over** e funzione obiettivo ***P(x,c)*** (*Penality*)

Attraverso questi due approcci è stato possibile avvicinarsi al massimo della funzione adottata. Dopodiché, è stata implementata una funzione che permettesse di effettuare una ricerca locale della soluzione ed arrivare all'ottimo della funzione obiettivo. Trovato il *massimo* e i pesi associati ad esso, si è passati ad una previsione dei prezzi dei titoli attraverso un modello *ARIMA*, grazie al quale è stato possibile calcolare il rendimento del portafoglio futuro.
Di seguito verrà presentato l'algoritmo genetico adattato al caso particolare di ottimizzazione del portafoglio. In seguito, verranno trattate separatamente le due metodologie sopracitate.

3.1 Algoritmo Genetico

L'algoritmo genetico è una tecnica di ottimizzazione basata sui principi genetici e di selezione naturale. Utilizzato nei problemi di ottimizzazione nel *machine learning*, fornisce una strategia, in situazioni in cui la soluzione ottimale non è immediatamente calcolabile, per trovarne una buona approssimazione in modo iterativo.
L'algoritmo genera una popolazione casuale composta da individui (generalmente detti cromosomi) che, in questo caso, corrispondono alla sequenza dei pesi degli n titoli sui quali effettuare l'analisi.
I primi individui sono stati creati casualmente, rispettando i vincoli di non-negatività e totale investimento del capitale che matematicamente si traduce:

$$\begin{cases} \sum_i x_i \leq 1 \\ 0 \leq x_i \leq 1 \end{cases} \tag{6}$$

Partendo da una popolazione creata casualmente tale da soddisfare i vincoli (6), vengono selezionati i due individui migliori. Su di essi viene applicata la funzione *crossing-over* che seleziona gli l pesi ottimali dal primo genitore e i restanti n-l dal secondo che comporranno, mantenendo l'ordine in cui si presentano nel genitore,

l'individuo figlio. Nell'algoritmo è presente la funzione *swap-mutation* che permette di mantenere una componente stocastica nelle generazioni successive. Tale funzione crea casualmente una mutazione all'interno della popolazione con una probabilità $p \in [0, 1]$.
In seguito, si ricerca l'individuo migliore che, secondo il principio dell'*elitismo*, verrà preservato tra una generazione e l'altra al fine di mantenere le caratteristiche migliori. Il resto della popolazione viene sovrascritto da quella appena generata.

Pseudo-Code, part I

```
Population ← generateRandomPopulation(PopulationSize, NumberWeights)
fit ← evaluatePopulation(Population, objectiveFunction)
MaxFit ← max(fit)
MaxSol ← Population[which (fit == MaxFit)]
while(Counter ≤ Iteration):
.        Scores ← evaluatePopulation(Population, ObjectiveFunction)
.        Rs ← rouletteSelection(Scores, Population)
.        for(i in PopulationSize):
.              Child[i] ← generateOffspring(Rs_1, Rs_2)
.              Child[i] ← swapMutation(Child[i], Prob_mutation)
.        end for
.        if (Elitist):
.              Child[random(i)] ← MaxSol
.        end if
.        Population ← Child
```

Dopodiché, viene nuovamente valutata la popolazione e ricercato l'individuo migliore *TempSol*. Questo viene utilizzato nella funzione *MinWeigthstoMaxWeigths*: la sequenza migliore dei pesi appena trovata viene modificata sommando al peso maggiore quello minore e quest'ultimo viene posto a zero. Se questo nuovo individuo risulta migliore rispetto a *TempSol* ne prende il posto all'interno della popolazione, altrimenti, non verrà considerato nella prole futura. La funzione *MinWeigthstoMaxWeigths* viene eseguita solo se, l'individuo scelto ha meno di *DiversificationThreshold* zeri al suo interno. Questa soglia viene passata come input all'algoritmo e serve per diversificare il portafoglio, ovvero per evitare che l'investimento sia concentrato su pochi titoli.
Infine, viene confrontata la soluzione trovata con quella massima e, nel caso fosse migliore, viene sovra-scritta.

Questi passaggi vengono iterati *Iteration* volte, a meno che per un numero *run* di volte non venga trovato una soluzione migliore per la funzione obiettivo.
L'algoritmo restituisce la soluzione migliore trovata e i pesi associati ad essa.

Pseudo-Code, part II

```
.        fit ← evaluatePopulation(Population, objectiveFunction)
.        TempFit ← max(fit)
.        TempSol ← Population[which (fit == MaxFit)]
.        if (Number of Zeros in TempSol ≤ DiversificationThreshold):
.              zeroSol ← minWeightToMaxWeight(TempSol)
.              zeroFit ← ObjectiveFunction(zeroSol)
.              if (zeroFit ≥ TempFit):
.                    Population[which (fit == TempFit)] ← zeroSol
.                    TempFit ← zeroFit
.              end if
.        end if
.        if (TempFit ≥ MaxFit):
.              CounterRun ← 0
.              MaxFit ← TempFit
.              MaxSol ← Population[which (fit == MaxFit)]
.        end if
.        if(CountRun == Run):
.              return (MaxSol, MaxFit)
.        end if
.        Counter++, CounterRun++
end while
return (MaxSol, MaxFit)
```

3.1.1 Arithmetic crossing-over e funzione obiettivo *Sharpe*

La prima strategia risolutiva utilizza come funzione obiettivo l'indice di Sharpe ed i vincoli sono inseriti all'interno del codice. Il problema da gestire riguardava la violazione dei vincoli imposti durante il *crossing-over*.
La funzione **generateOffspring** permette di creare nuovi figli attraverso un *crossing-over* aritmetico: in questo modo il figlio non presenta nessun elemento identico ai genitori. Gli elementi della generazione successiva u_i sono ottenuti come combinazione

convessa dei genitori y_1 e y_2 ossia:

$$u_i = \alpha y_1 + (1-\alpha)y_2, \quad i = 1, \ldots, q \tag{7}$$

dove $\alpha \in [0,1]$ generato in modo casuale e q numero di iterazioni.
Questa funzione permette di ottenere figli che continuano a soddisfare le condizioni (6).

3.1.2 Two-point crossing-over e funzione obiettivo *Penality*

La seconda soluzione implementata utilizza un algoritmo di penalità, presentato in precedenza, per inserire i vincoli all'interno della funzione obiettivo. In questo caso la funzione è la seguente:

$$P(x_i, 100) = Sharpe(x_i) - 100\phi(x_i)$$

$$\text{dove} \quad \phi(x_i) = \left(\textstyle\sum_{i=1}^n x_i - 1\right)^2 + \textstyle\sum_{i=1}^n (max(0, x_i - 1))^2 + \textstyle\sum_{i=1}^n (max(0, -x_i))^2$$

Come anticipato durante la presentazione degli algoritmi di penalità, essi non assicurano che la soluzione ottimale appartenga alla regione ammissibile. Tuttavia, ponendo il parametro $c > 0$ sufficientemente grande, per esempio pari a 100, vengono penalizzate le soluzioni non accettabili facendo sì che non risultino ottimali.
Anche per questa seconda metodologia risolutiva si è partiti dall'algoritmo genetico seguendo step-by-step i passaggi precedentemente presentati. La differenza principale rispetto alla prima implementazione sta nella definizione della funzione *crossing-over*. Ogni membro della popolazione è costituito da una sequenza di n titoli: durante il *crossing-over*, $n/2$ pesi del primo genitore (con posizione di partenza casuale) vengono mantenuti anche nella generazione successiva. Le posizioni dei pesi rimanenti vengono allocate con i pesi del secondo genitore. Procedendo in questo modo è possibile ottenere anche delle soluzioni non ammissibili, ma per costruzione della funzione obiettivo, tali soluzioni saranno penalizzate, dato che si otterebbe $\displaystyle\sum_{i=1}^{n} x_i - 1 \neq 0$.

3.2 Metodo per la ricerca locale

Le soluzioni precedentemente descritte, consentono di arrivare a delle soluzioni *quasi-ottimali*. Per ottenere la sequenza di pesi che massimizza lo *Sharpe Ratio*, ossia quella che consente di individuare il *tangency portfolio*, è stato implementato un algoritmo

di ricerca locale che ha permesso di ottenere la soluzione ottima.
Di seguito verranno presentata la funzione per la ricerca locale (*Local Search*):

1. vengono presi due pesi x_i e x_j in modo casuale dal vettore della soluzione ottimale $\mathbf{x*} \in \mathbb{R}_+^{n-l}$, ottenuta con l'algoritmo genetico (n numero di titoli iniziali e l numero di titoli con peso uguale a zero);

2. dato $\varepsilon > 0$ sufficientemente piccolo, la nuova soluzione sarà ottenuta mediante i nuovi pesi $x_i^{(m)} = x_i^{(m-1)} - \varepsilon$ e $x_j^{(m)} = x_j^{(m-1)} + \varepsilon$, con $m = 1, \ldots, M$, dove M è il numero di iterazioni in cui la soluzione migliora;

3. se la soluzione ottenuta è migliore della precedente, allora l'algoritmo continuerà ad aggiungere e sottrarre ε agli stessi pesi x_i, x_j selezionati nel punto *1*;

4. se la soluzione trovata non è migliore di quella precedente, allora verranno ripetuti i punti *1-2-3*;

5. infine, viene fatto un controllo sulla norma del gradiente: se $\|\nabla f(\mathbf{x**})\| \leq tol$, con *tol* vicino a zero (in questo caso $tol = 10^{-9}$), allora la soluzione trovata è un massimo, dove x^{**} è la soluzione ottima finale.

Il processo appena descritto viene iterato per K volte ed il valore ε viene diminuito all'aumentare del numero di iterazioni (ε_{k_i} con $k_i \in K$).
In questa analisi sono stati utilizzati i seguenti parametri: $K = 500$, $\varepsilon_{200} = 10^{-7}$, $\varepsilon_{300} = 10^{-8}$, $\varepsilon_{400} = 10^{-9}$ e $\varepsilon_{450} = 10^{-10}$.

3.3 Forecasting con modello ARIMA

Trovati gli n pesi che massimizzano l'indice di *Sharpe*, si è deciso di effettuare una previsione dei prezzi futuri attraverso un modello *ARIMA* al fine di calcolare il valore del portafoglio tra h mesi. Dopo aver verificato che i titoli fossero distribuiti in modo normale e che fossero stazionari, è stato possibile applicare il modello.
Per compiere tale previsione sono stati considerati i prezzi di chiusura di ciascun titolo dal *01-07-2013* al *01-07-2019*. Attraverso il comando *arima* del pacchetto *stats*, è stato possibile *fittare* il modello, passato alla funzione *forecast* del pacchetto *forecast*. Successivamente, è stato possibile ottenere una previsione tra $h = 6$ mesi dei prezzi di ciascun titolo.

Dopo questa fase di *forecasting*, è stato possibile calcolare il ritorno atteso del portafoglio R nel seguente modo:

$$R = \sum_{i=1}^{n} w_i r_i, \quad r_i = \frac{P_i(t+h) - P_i(t)}{P_i(t)}$$

dove w_i è il peso e r_i è il ritorno atteso del *i-esimo* titolo; P_i rappresenta il prezzo del titolo, h è il periodo di previsione che si vuole considerare dall'istante inziale t.

4 Risultati e valutazioni

Nella seguente sezione verranno riportati sinteticamente i risultati ottenuti dai metodi precedentemente descritti. Questi sono stati ottenuti modificando i valori di input degli algoritmi implementati e conseguentemente confrontati con i risultati ottenuti attraverso i pacchetti di R: la funzione *GA* del pacchetto *GA*, la funzione *auglag* del pacchetto *nloptr* e la funzione *tangencyPortfolio* del pacchetto *fPortfolio*.

POP_SIZE	ITERATIONS	RUN	GA_PENALITY	GA_SHARPE	GA_PACCHETTO
10	40	20	0.1014965	0.1008867	-23.0230046
10	50	25	0.1014746	0.0995615	-0.9141324
10	100	50	0.1015827	0.1012661	0.0913186
15	200	100	0.1014288	0.1002206	0.0951617
20	200	100	0.1014351	0.1005842	0.096323
20	300	150	0.1013872	0.1010654	0.0990028
30	300	200	0.1013191	0.1002069	0.0982406
30	400	200	0.1015071	0.1012354	0.1005651
40	500	200	0.1014112	0.1007568	0.1001167
50	1000	300	0.1013883	0.1014249	0.0998902

Figura 2: Confronto tra i valori della soluzione obbiettivo ottenuti con i diversi algoritmi e diversi valori dei parametri in input

Nella figura 2 sono stai messi a confronto i valori dell'indice di Sharpe calcolati applicando i diversi algoritmi più volte e variando i valori in input: la grandezza della popolazione, il numero di iterazioni e il valore del parametro run.
In figura 3 sono presentati i pesi associati ad ogni titolo nella soluzione ottima trovata dagli algoritmi prima dell'applicazione della ricerca locale.
In figura 4 sono presentati i pesi associati ad ogni titolo nella soluzione ottima trovata dalla *Local Search* e confrontata con i risultati delle funzioni dei pacchetti *auglag* e *tangencyPortfolio*: queste ultime due funzioni hanno rappresentato per questa analisi

i risultati ottimimali da raggiungere.

TITOLI	GA_PENALITY	GA_SHARPE	GA_PACCHETTO
AMAZON	0.12336932	0.14996578	0.12458983
APPLE	0.12814785	0.14584855	0.12459110
IBM	0	0	0.00509414
FACEBOOK	0.18833473	0.21786239	0.19834450
MICROSOFT	0.15845753	0.17357608	0.20849016
JPM	0.05549513	0.07456871	0.05359265
DISNEY	0.12336932	0.09601650	0.09978044
GOOGLE	0	0	0.01413348
NETFLIX	0.14707953	0.14216199	0.14520577
TESLA	0	0	0.01114821

Figura 3: Confronto tra i pesi ottimali trovati con l'applicazione delle tre metodologie risolutive

TITOLI	GA + LS PENALITY	GA + LS SHARPE	TANGENCY PORTFOLIO	AUGLAG
AMAZON	0.1219172	0.1222449	0.1223329	0.1222449
APPLE	0.1376703	0.1380404	0.1382715	0.1380405
IBM	0	0	0	0
FACEBOOK	0.2002829	0.2008214	0.2007306	0.2008213
MICROSOFT	0.2011219	0.2016626	0.2014491	0.2016626
JPM	0.0426336	0.0427482	0.0429784	0.0427482
DISNEY	0.1314100	0.1317633	0.1315576	0.1317632
GOOGLE	0	0	0	0
NETFLIX	0.1622828	0.1627191	0.1626796	0.1627191
TESLA	0	0	0	0

Figura 4: Confronto tra i pesi ottimali trovati con l'applicazione della Local Search

Partendo dai pesi ottenuti, è stato calcolato il ritorno cumulato del portafoglio ottimale e confrontato graficamente in figura 5 con quelli dei singoli titoli, notando che il nostro portafoglio è il più stabile.

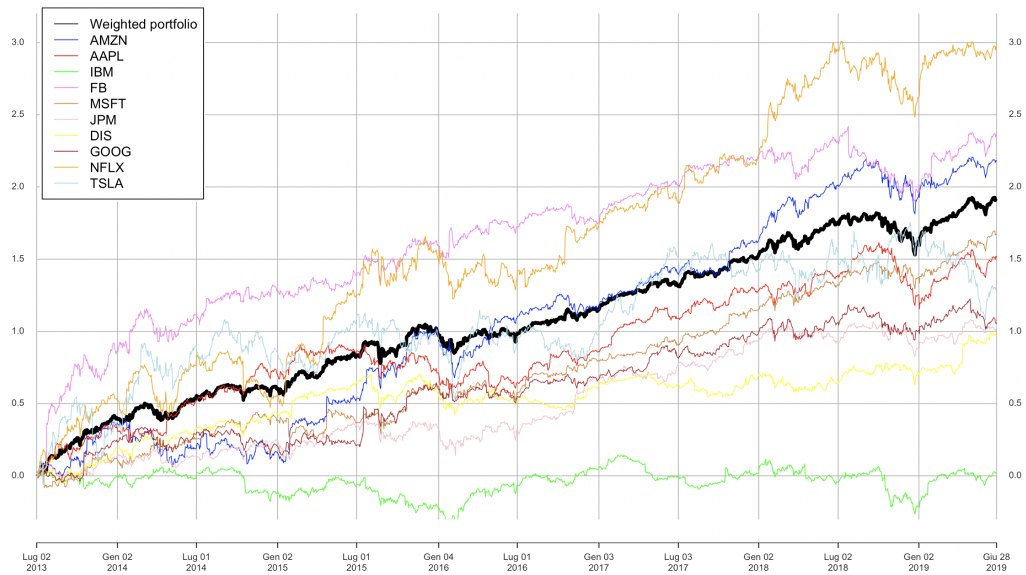

Figura 5: Confronto tra il ritorno atteso del portafoglio e quello dei singoli titoli

Calcolando i ritorni attesi futuri dei titoli, attraverso la previsione dei prezzi col modello ARIMA, è stato possibile calcolare il rendimento atteso del portafoglio a sei mesi.

AMZN	AAPL	IBM	FB	MSFT	JPM	DIS	GOOG	NFLX	TSLA
0.06229808	0	0	0.06839378	0.04959099	0.03344206	0.04538229	0.04993387	0.07647190	0

Figura 6: Ritorni a sei mesi dei titoli

Dati i pesi ottimali w_i^* ottenuti precedentemente è stato possibile calcolare il rendimento atteso del portafoglio:

$$R = \sum_{i=1}^{n} w_i^* r_i \simeq 0.0512 \tag{8}$$

Quindi, se si ha intenzione di investire un capitale $C = 1,000$ *Euro*, il guadagno stimato sarebbe $G = R \times C \simeq 54.43$ *Euro*.

5 Discussione

Come prima premessa si noti che gli algoritmi implementati presentano una componente casuale che, in questo caso, è stata gestita iterando più volte al fine di assicurare l'attendibilità dei dati. Osservando i valori della funzione obiettivo e del gradiente calcolato nei punti ottimi si vede che sia gli algoritmi implementati che quelli dei pacchetti (*fPortfolio* e *auglag*) permettono di raggiungere la soluzione ottima che corrisponde ad un valore dell'indice di Sharpe pari a 0.1016291.
Più dettagliatamente, osservando i valori del gradiente associati alla soluzione ottima (ottenuta dopo l'applicazione della funzione *Local Search* da noi implementata) è stato possibile constatare che, ad eccezione della funzione *tangencyPortfolio*, i gradienti sono dell'ordine $\sim 10^{-10}$.
Confrontando le tre versioni utilizzate per l'implementazione dell'algoritmo genetico, risulta evidente che il pacchetto di R non sia performante come gli altri due costruiti proprio partendo dal problema di ottimizzazione del portafoglio. Aumentando il numero di iterazioni e di individui della popolazione, la funzione preimpostata in R non riesce a raggiungere il valore massimo. L'algoritmo con funzione obiettivo *P(x,c)* massimizza il valore dell'indice di Sharpe anche con poche iterazioni e con una popolazione limitata.
Per quanto riguarda i risultati ottenuti con il modello ARIMA, si può affermare che i valori predetti confermano che la scelta dei pesi da attribuire ai titoli sia corretta avendo una previsione di guadagno dello 0.0512. Tuttavia, come ci si potrebbe aspettare, questi tipi di modelli predittivi non presentano un'elevata accuratezza.

6 Conclusioni

In questa analisi è stata fornita una metodologia di ottimizzazione di un portafoglio finanziario mediante algoritmi di ottimizzazione non-lineare con lo scopo di fornire una strategia di investimento e garantire il giusto *trade-off* tra rendimento atteso e volatilità.
Dopo aver stimato il portafoglio ottimale, è stato possibile effettuare una predizione sul suo rendimento atteso utilizzando una tecnica di *forecasting.* In particolare, investendo il capitale nei dieci titoli presi inizialmente in esame, in maniera proporzionale ai pesi trovati con i metodi sopra-descritti, è stato possibile stimare un ritorno atteso da qui a sei mesi del 5%.

Riferimenti bibliografici

[1] *Appunti del corso di Decision Models aa 2018-2019*, Vincenzina Messina & Michele Ciavotta

[2] *Appunti del corso di Business Intelligence per i servizi finanziari aa 2016-2017*, Antonio Candelieri

[3] *"Data Science: Theories, Models, Algorithms, and Analytics"*, Sanjiv Ranjan Das

[4] *https://medium.com/the-trading-scientist/portfolio-optimization-in-r-using-a-genetic-algorithm-8726ec985b6f*

[5] *"Optimization Method in Finance"*, Gerard Cornuejols and Reha Tutuncu

[6] *"Introduzione alla Programmazione Non Lineare"*, Michele Monaci,Dipartimento di Ingegneria dell'Informazione, Università di Padova

[7] *"Portfolio optimization through genetic algorithms in an artificial stock market"*, Andrea Chiamenti,Master's Degree in Quantitative Finance and Insurance, Università degli Studi di Torino