پردیس علوم
دانشکده‌ی ریاضی، آمار و علوم کامپیوتر

بهینه‌سازی پورتفولیو در بازار سهام ایران

نگارنده

محمد تیموری پابندی

استاد راهنما: دکتر سمانه افتخاری مهابادی

پایان‌نامه برای دریافت درجه کارشناسی
در رشته علوم کامپیوتر

مرداد ۹۹

چکیده

در بازار بورس و هنگام مدیریت حجم‌های بزرگ سرمایه، نهایتاً آن‌چه حرف آخر را می‌زند، مدیریت صحیح پورتفولیو و بهینه‌سازی ریسک آن است. در صورتی که پورتفولیویی کم‌ریسک و با بازده خوب داشته باشیم، می‌توانیم انتظار داشته باشیم که سرمایه‌مان در فراز و نشیب‌های موجود در بازار بورس، مسیر مناسبی را طی می‌کند و کار سرمایه‌گذاری را با اطمینان و نه نگرانی پیش می‌بریم. در این پروژه، ابتدا داده‌های مورد نیاز را از بازار سهام به صورت Realtime به دست می‌آوریم، سپس با استفاده از روش‌های مبتنی بر EigenDecomposition اقدام به تشکیل پورتفولیو با کم‌ترین ریسک می‌کنیم. نهایتاً با روش ابتکاری استفاده از SVD سرعت عملکرد روش مقاله‌ی اصلی را تا ۳ و نیم برابر افزایش می‌دهیم.

کلیدواژه‌ها: بورس، بهینه‌سازی پورتفولیو، تجزیه‌ی مقادیر منفرد

پیشگفتار

بهینه‌سازی پورتفولیو [1] ، روشی است که طی آن، سهام مختلف با بازده‌ها و ریسک‌های موجود، به نحوی ترکیب می‌شوند که پورتفولیوی حاصل ویژگی‌های مطلوبی را داشته باشد؛ برای مثال ممکن است پورتفولیو برای ریسک بهینه‌سازی شود و یا برای بازده. انتخاب اوزان مناسب برای سهام موجود، توسط روش‌های مبتنی بر داده‌کاوی قابل انجام است. در این پژوهش به بهینه‌سازی پورتفولیو برای کم‌ترین ریسک می‌پردازیم. در فصل اول، به بیان مفاهیم مقدماتی و ضرورت بهینه‌سازی پورتفولیو می‌پردازیم. در فصل دوم مراحل اصلی و شمای کلی بهینه‌سازی پورتفولیو را بررسی می‌کنیم و در فصل سوم، عملکرد خودمان را در این پژوهش به تفصیل به بررسی می‌گذاریم. در بخش عملی این پژوهش، ابتدا در خصوص جمع‌آوری داده‌ها به صورت Realtime صحبت می‌کنیم و جزئیات پیاده‌سازی آن را ذکر می‌کنیم. در ادامه به تمیزسازی داده‌ها و روش‌های پیشنهادی برای آن می‌پردازیم و در نهایت علاوه بر پیاده‌سازی روش‌های پیش‌نهادی مبتنی بر EigenDecomposition روی ماتریس کوواریانس، این روش را با استفاده از متد SVD پی می‌گیریم که در نتیجه‌ی آن، محاسبات مقاله‌ی اصلی تا ۳ و نیم برابر تسریع می‌شود.

[1] Portfolio Optimization

فهرست مطالب

فصل ۱

مقدمه

بهینه‌سازی پورتفولیو [۱] فرآیند انتخاب بهترین پورتفولیو(توزیع دارایی [۲])، از بین مجموعه‌ی تمام پورتفولیوهای قابل تشکیل، بر اساس یک هدف [۳] است. این هدف معمولاً بیشینه‌کردن فاکتورهایی چون امید بازگشت سرمایه [۴] و کمینه‌کردن هزینه‌هایی چون ریسک مالی [۵] است.

بر اساس استراتژی‌های مختلفی می‌توان عمل بهینه‌سازی پورتفولیو را انجام داد؛ برای مثال می‌توان از هر سهم با وزن مساوی در پورتفو قرار داد، می‌توان از هر سهم به نسبت ریسکش استفاده کرد، می‌توان سعی در ساخت یک پورتفولیو با کمینه‌ی ریسک کرد و یا بهینه‌سازی را بر اساس کمینه‌کردن ریسک و بیشینه‌کردن بازگشت سرمایه در عین حال انجام داد. در شکل ۱.۱ می‌توانیم تفاوت عملکرد این روش‌ها را بر روی یک پورتفولیوی فرضی ببینیم:

---

[۱] Portfolio Optimization

[۲] Assst Distribution

[۳] Objective

[۴] Expected Return

[۵] Financial Risk

شکل ۱.۱: عملکرد استراتژی‌های مختلف بهینه‌سازی پورتفولیو روی بازگشت سرمایه

فاکتورهای مختلفی چون ویژگی‌های شخصیتی سرمایه‌گذار(مثلاً میزان ریسک‌پذیری وی) می‌توانند مسیر بهینه‌سازی پورتفولیو را تغییر دهند. هم‌چنین از عناصر دیگری که روی نتیجه‌ی کار تأثیرگذار هستند، می‌توان به میزان هم‌بستگی میان سهم‌های پورتفولیو و شاخص بازار مالی نیز اشاره کرد.

در صورتی که به درستی، حضور در بازار سرمایه را یک امر بلندمدت بدانیم، رویکرد صحیح حضور در این بازار، نه ورود و خروج‌های پی‌درپی به سهام مختلف، بلکه نگهداری و مدیریت یک پورتفولیو به شکل مناسب است. این نگهداری بایستی با استفاده از ابزارهای مالی و ریاضی انجام شود و در این مطالعه، سعی خواهیم کرد نشان دهیم که استفاده از ابزار صحیح، چگونه می‌تواند به سود حداکثری از این بازار منجر شود.

۱.۱ مفاهیم

۱.۱.۱ پورتفولیو

همان‌طور که پیش‌تر به طور ضمنی ارائه شد، مفهوم «پورتفولیو» [۶] را به عنوان یک مجموعه از سهام بازار بورس، به همراه اوزان مشخص آن‌ها در نظر می‌گیریم. برای مثال یک پورتفولیو می‌تواند مانند جدول ۱.۱ تعریف شود.

سهم	وزن
غپینو	0.2
کبافق	0.4
زشریف	0.15
شبندر	0.25

جدول ۱.۱: یک پورتفولیوی نمونه با چهار سهم

آن‌گونه که مشخص است، وزن هر سهم(w_i) کوچک‌تر یا مساوی با ۱ است و هم‌چنین حاصل $\sum_i w_i$ برابر با ۱ خواهد بود.

به صورتی رسمی‌تر، اگر یک مجموعه‌ی S از سهم‌ها داشته باشیم و اوزان مورد انتخابمان را به صورت بردار $w = [w_i]_{i \in S}$ نشان دهیم، دوتایی (S, w) توصیف یک پورتفولیو را به ما می‌دهد.

۲.۱.۱ بازگشت [۷]

بازگشت، در واقع شاخصی از بهره‌وری یک سهم ارائه میکند. اگر فرض کنیم من ۱۰۰دلار از سهمی را قبلاً خریده‌ام و الآن با افزایش قیمت، ارزش سهامم به ۱۵۰دلار رسیده است، این یعنی سهام من ٪۵۰ بازگشت داشته است. بازگشت یک سهم i را در زمان t، می‌توانیم با $r_{i,t}$ نشان دهیم

[۶] Portfolio
[۷] Return

که از رابطه‌ی زیر قابل محاسبه است:

$$r_{i,t} = \frac{p_{i,t} - p_{i,t-1}}{p_{i,t-1}}$$

که $p_{i,t}$ همان قیمت سهام i در زمان t است. مشخص است که با این تعریف، بازگشت در روز اول تعریف نمی‌شود.

۳.۱.۱ امید بازگشت [۸]

امید بازگشت، برابر با حاصل‌ضرب میزان بازگشت سناریوهای مختلف در احتمال رخ‌داد آن‌هاست. برای مثال می‌توانیم یک سرمایه‌گذاری خطرپذیر [۹] را در نظر بگیریم که در آن صرفاً ٪۱ احتمال دارد که استارت‌آپ مورد تأسیس، به سودآوری برسد. ولی در صورتی که به سودآوری برسد، حجم بازارش به او این اجازه را خواهد داد که میزان ۱ میلیون‌دلار در سال سود داشته باشد. هم‌چنین برای سادگی در محاسبات، فرض می‌کنیم که در حالت‌های دیگر مطلقاً درآمدی کسب نمی‌کنیم و میزان سرمایه‌گذاری‌مان هم ۱۰۰۰ دلار است. در این سناریو امید بازگشت سرمایه‌مان برابر است با:

$$R = 0.01 \times \$1000000 + 0.99 \times -\$1000 = \$10000 - \$990 = \$9010$$

یعنی امید بازگشت برابر با ۹۰۱۰ دلار خواهد بود. از جهت تعریف، امید بازگشت شباهت زیادی به «امید ریاضی» در آمار دارد. امید بازگشت دیدگاه خوبی از حاصل‌ضرب احتمالات در بازگشت‌ها به ما می‌دهد، لکن صرفاً بر اساس آن تصمیم گرفتن نوعی سادگی است؛ چرا که در همین مثال بالا، فقط ٪۱ سرمایه‌گذاران خطرپذیر در این حوزه می‌توانند به موفقیت برسند و الباقی کاملاً از رده خارج می‌شوند، اما با این حال انتظار ۹۰۱۰ دلار بازگشت را در محاسباتمان می‌بینیم که عدد

[۸] Expected Return

[۹] Venture Capital

خوبی است. برای پرکردن این خلأ در پیش‌گویی، از یک شاخص دیگر به نام «ریسک مالی» استفاده می‌کنیم.

۴.۱.۱ ریسک مالی [۱۰]

ریسک مالی احتمال از دست‌دادن پول در یک سرمایه‌گذاری و یا کسب‌وکار است. یکی از شاخص‌هایی که برای اندازه‌گیری ریسک استفاده می‌شود، «واریانس»(یا انحراف معیار) است.

بیایید برای مثال در سرمایه‌گذاری خطرپذیری که پیش‌تر تعریف کردیم، میزان ریسک را با استفاده از شاخص واریانس اندازه‌گیری کنیم.

$$\sigma^2 = \sum x^2 p - \mu^2 = (1000000^2 \times 0.01 + (-1000)^2 \times 0.99) - 9010^2 = 9919809900$$

ویژگی‌ای که واریانس را برای اندازه‌گیری ریسک مناسب می‌کند، این است که هر چقدر همه‌ی نقاط مختلف در فضای حالت، از نظر بازگشت سرمایه نزدیک به یک‌دیگر باشند، میزان ریسک نیز پایین خواهد بود؛ ولی هر قدر پیش‌آمدهای مختلف با یک‌دیگر فاصله داشته باشند، ریسک نیز عدد بالایی را خروجی خواهد داد. در این‌جا مقدار واریانس آن‌قدر زیاد است که با یک نگاه ما را متوجه میزان خطری که متوجه این نوع سرمایه‌گذاری است، می‌کند. پس در مثال یادشده، با وجود این‌که هر شرکت‌کننده‌ای به صورت میانگین حدود ۹۰۰۰دلار بازگشت سرمایه خواهد داشت، ولی از آن‌جا که ٪۹۹ سرمایه‌گذاران، ضرر خواهند کرد، پس سرمایه‌گذاری پرریسکی را داریم که با شاخص واریانس قابل اندازه‌گیری است.

[۱۰] Financial Risk

۵.۱.۱ نسبت شارپ [۱۱]

نسبت شارپ که توسط William F. Sharpe توسعه داده شده است، می‌تواند به سرمایه‌گذاران کمک کند تا دیدی از امید بازگشت یک سرمایه‌گذاری در مقایسه با ریسک آن پیدا کنند. در واقع نسبت شارپ به ما می‌گوید که به ازای هر واحد ریسکی که به جان بخریم، چه مقدار امید بازگشتمان به صورت متوسط افزایش می‌یابد. نسبت شارپ از طریق فرمول زیر محاسبه می‌شود:

$$SharpeRatio = \frac{R_p - R_f}{\sigma_p}$$

که در آن، R_p بازگشت پورتفولیو، R_f، بازگشت سرمایه‌ی بدون ریسک(مثلاً اگر نرخ سود بانکی ٪۲۰ است، این مقدار ٪۲۰ خواهد بود) و σ_p انحراف استاندارد بازگشت پورتفولیو است. در صورتی که نرخ سود بانکی را ۰ در نظر بگیریم، می‌توان از مقدار R_f صرف‌نظر کرد. با توجه به مطالب ذکر شده، هر نوع سرمایه‌گذاری که مقدار نسبت شارپ آن بیش‌تر باشد، به طور کلی یک سرمایه‌گذاری جذاب‌تر است.

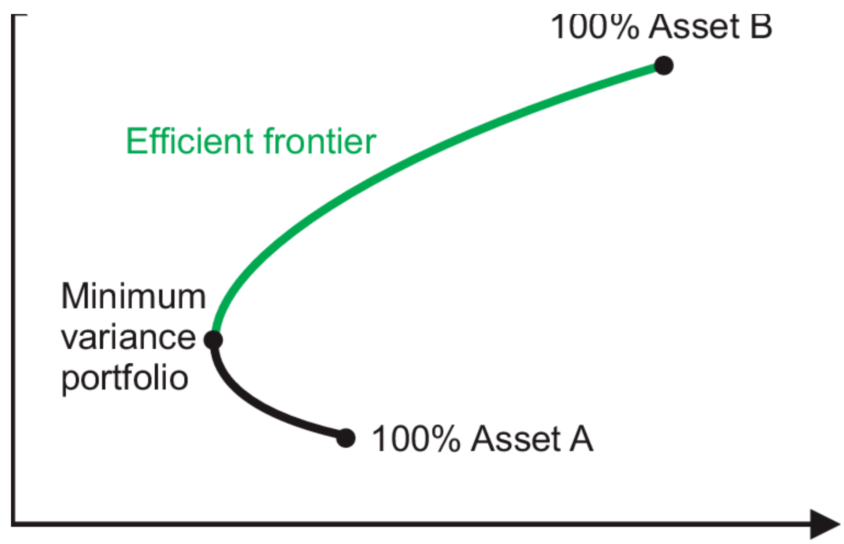
۶.۱.۱ هم‌بستگی [۱۲]

هم‌بستگی مد نظر ما در این مطالعه، مشابه هم‌بستگی در مفهوم کلی آماری است. هم‌بستگی میان دو سهم را، هم‌بستگی پیرسن بین مقادیر Return موجود در دو سهم در یک سری زمانی در نظر می‌گیریم.

[۱۱] Sharpe Ratio
[۱۲] Correlation

۷.۱.۱ Efficient Frontier

در واقع Efficient Frontier مجموعه‌ی تمام پورتفولیوهایی است که دارای بیش‌ترین امید بازگشت، به ازای مقدار مشخصی از ریسک هستند. هر پورتفولیویی که زیر Efficient Frontier بیفتد، یک پورتفولیوی غیربهینه خواهد بود؛ چرا که با قبول همان میزان از ریسک، می‌توانیم یک نقطه روی Efficient Frontier انتخاب کنیم که دارای بازدهی بهتر باشد. یک نمونه از Frontier Efficient در شکل ۲.۱ آمده است.



شکل ۲.۱: Efficient Frontier for a Two Assets Example

۲.۱ ضرورت بهینه‌سازی پورتفولیو

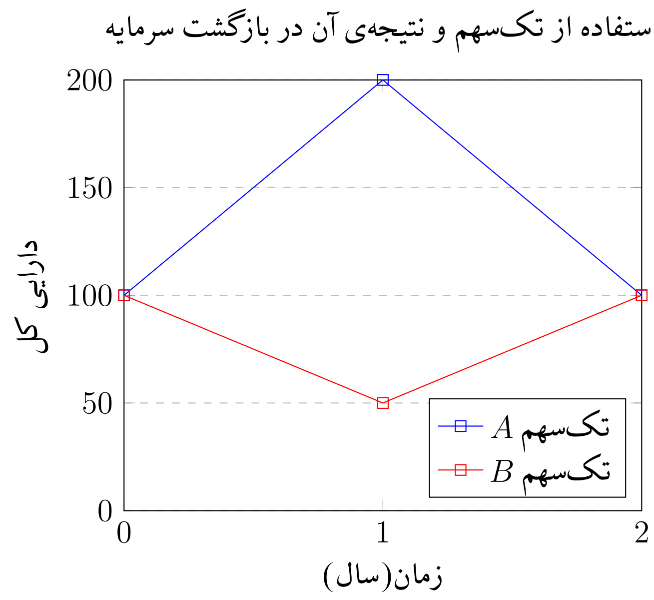
سؤالی که ممکن است پیش بیاید، این است که «اصلاً چرا بایستی یک پورتفولیو بچینیم»؟ و «آیا بهتر نیست یک سهم بخریم و منتظر رشد آن باشیم»؟ پاسخ به این سؤالات را با استفاده از یک مثال خواهیم داد. تصور کنید دو نوع سهم A و B در یک بازار مالی موجودند. سهم A در سال اول

دوبرابر و در سال دوم نصف می‌شود، و سهم B در سال اول نصف و در سال دوم دوبرابر می‌شود. حال می‌خواهیم ببینیم آیا چیدن سبد در این‌جا معقول است یا خرید تک‌سهم؟

۱.۲.۱ استراتژی تک‌سهم

تصور کنید در ابتدای کار، تمام ۱۰۰ دلار دارایی‌مان را A بخریم. این سهم در سال اول دوبرابر می‌شود؛ پس در انتهای سال اول ۲۰۰ دلار خواهیم داشت. و از آن‌جا که در سال دوم نصف می‌شود، در انتهای سال دوم دوباره ۱۰۰ دلار دارایی A در دستمان خواهد بود. پس با توضیحات ذکر شده، میزان بازگشت سرمایه در این روش ۰٪ می‌شود.

حال تصور کنید در ابتدای کار تمام ۱۰۰ دلارمان را B می‌خریدیم. از آن‌جا که در سال اول این سهم نصف می‌شود، در انتهای سال اول ۵۰ دلار دارایی کلمان خواهد بود و با توجه به دوبرابرشدن این سهام در سال دوم، دارایی ما دوباره به عدد اولیه‌ی ۱۰۰ دلار برمی‌گردد که نشان‌گر این است که در این روش نیز بازگشت سرمایه‌مان ۰٪ خواهد بود.



شکل ۳.۱: بررسی پورتفولیوی تک‌سهم

در نتیجه در استراتژی تک‌سهم، با این توصیفات ارائه‌شده هر کدام از سهام را که در روز اول بخریم، در نهایت بازگشتی برایمان در پی نخواهد داشت. این مسئله در شکل ۳.۱ مشهود است.

۲.۲.۱ استراتژی چینش پورتفولیو

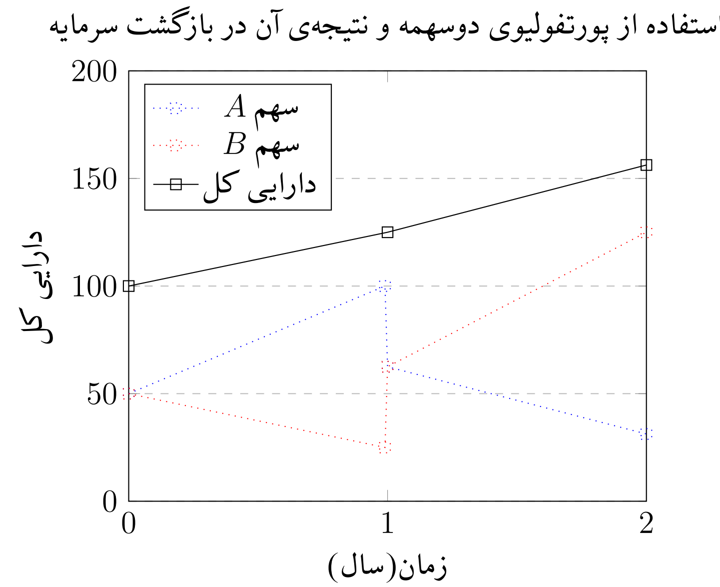
حال می‌خواهیم با توصیفات ذکر شده در بالا، یک پورتفولیوی دوسهمه از A و B بسازیم؛ بدین‌شکل که وزن هر کدام برابر با 0.5 باشد. در این استراتژی، ۱۰۰دلار روز اولمان را بین هر کدام از سهام به صورت مساوی تقسیم می‌کنیم؛ یعنی ۵۰دلار از A و ۵۰دلار هم از B می‌خریم. با گذشت سال اول، با دوبرابرشدن A و نصف‌شدن B، دارایی ما به مجموع ۱۲۵دلار می‌رسد؛ چرا که داریم:

$$50 \times 2 + 50 \times 0.5 = 100 + 25 = 125$$

هنگامی که سال اول گذشت، دوباره اوزان را به نصف می‌رسانیم(یعنی کمی A می‌فروشیم و به جای آن B می‌خریم تا از هر کدام 62.5 دلار داشته باشیم). در نتیجه پس از گذشت سال دوم، با دوبرابرشدن A و نصف‌شدن B، مقدار 156.25 دلار دارایی خواهیم داشت؛ چون

$$62.5 \times 2 + 62.5 \times 0.5 = 125 + 31.25 = 156.25$$

و این نشانگر آن است که در شرایط ذکرشده، پورتفولیوی دوسهمه با وزن‌های مساوی از سهام A و B، عملکرد بهتری نسبت به انتخاب تک‌سهم دارد. این استراتژی در شکل ۴.۱ بررسی شده است.



شکل ۴.۱: بررسی پورتفولیوی دوسهمه

همان‌طور که مشاهده می‌شود، استفاده از یک پورتفولیو به جای خرید تک‌سهم، بازگشت نهایی سرمایه را در این مثال بسیار بالاتر می‌برد. به نوعی می‌توان Diversification یا ایجاد تنوع در سبد دارایی را The Free Lunch of Investing نامید.

فصل ۲

مراحل تشکیل پورتفولیوی بهینه

۱.۲ عوامل مؤثر بر تشکیل پورتفولیوی بهینه

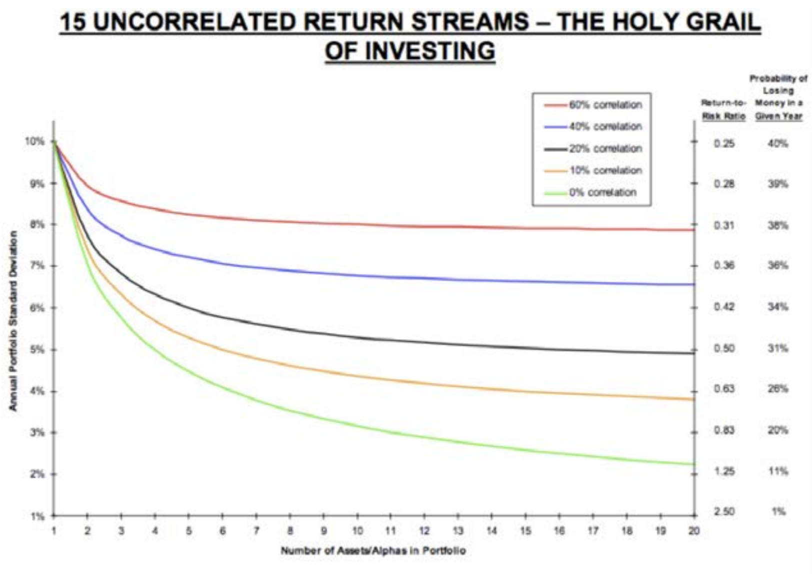
در تشکیل پورتفولیوی بهینه، لازم است که در عین بیشینه‌کردن امید بازگشت، ریسک مالی را کمینه کنیم. همان‌طور که پیش‌تر ذکر کردیم، عوامل مختلفی در تشکیل پورتفولیوی بهینه اثرگذارند که در این‌جا به بررسی آن‌ها می‌پردازیم.

۱.۱.۲ ویژگی‌های شخصیتی سرمایه‌گذار

تاب‌آوری ریسک یک سرمایه‌گذار، می‌تواند عامل مهمی در انتخاب یک پورتفولیو داشته باشد و شاید بتوان آن‌را مهم‌ترین عامل موجود برای انتخاب سهام مختلف دانست.

۲.۱.۲ هم‌بستگی

هم‌بستگی نقش بسیار پررنگی در تشکیل پورتفولیوی بهینه بازی می‌کند. ری دالیو [۱] در کتاب خودش، «اصول اساسی» [۲] از روزی صحبت می‌کند که یک متخصص آمار، با محاسباتی به وی نشان داد که «تعدد سهام» و «هم‌بستگی»، چگونه می‌توانند دست در دست هم‌دیگر، یک پورتفوی امن و با بازگشت خوب بسازند. ری دالیو این نمودار را «جام مقدس سرمایه‌گذاری» [۳] نامیده است. این نمودار در شکل ۱.۲ نمایش داده شده است.



شکل ۱.۲: جام مقدس سرمایه‌گذاری

همان‌طور که قابل مشاهده است، یک پورتفولیوی ۴ سهمه‌ی متشکل از سهام بدون هم‌بستگی، بسیار بهتر از یک پورتفولیو با ۲۰ سهم که ۶۰٪ با یک‌دیگر هم‌بستگی دارند عمل می‌کند. یعنی عمل Diversification بایستی به صورت عاقلانه انجام شود و صرفاً استفاده از تعداد زیادی سهام کافی

[۱] Ray Dalio
[۲] Principles
[۳] The Holy Grail of Investing

نیست و باید سهم‌های انتخاب‌شده، هم‌بستگی کمی با هم‌دیگر داشته باشند.

۳.۱.۲ وضعیت بازار

گاهی اوقات و با توجه به وضعیت بازار، ممکن است استراتژی‌های مختلفی بهترین پورتفولیو را بسازند؛ برای مثال در شرایط خاصی ممکن است ریسک و امید بازگشت، هم‌راستا عمل کنند ولی در شرایط دیگری امکان دارد که این دو در خلاف یک‌دیگر باشند. در نتیجه این که بدانیم وضعیت بازار در چه موقعیتی است، ما را در بهتر چیدن پورتفولیوی بهینه یاری می‌کند. در این مطالعه از این عامل صرف‌نظر کرده‌ایم.

۲.۲ مراحل بهینه‌سازی پورتفولیو

۱.۲.۲ به دست آوردن اطلاعات زنده

بایستی ابتدا اطلاعات سهام مختلفمان را در قالب مناسب به دست آوریم. برای این امر، از سایت «نهایت‌نگر» و web scrapping روی آن استفاده کرده‌ایم. جزئیات بیش‌تر را در فصل بعدی آورده‌ایم.

۲.۲.۲ محاسبه‌ی ریسک و اوزان مرتبط با ریسک کمینه

در صورتی که به ماتریس کوواریانس Σ دسترسی داشته باشیم، می‌توانیم ریسک(واریانس) یک پورتفولیو را از طریق اوزان به شکل زیر به دست آوریم:

$$risk = w^T \Sigma w$$

در واقع این فرمول بسیار مهمی است که در انواع مختلفی از مسائل بهینه‌سازی ظاهر می‌شود، اما در خصوصش مسئله‌ای وجود دارد که شایان ذکر است؛ اگر بخواهیم ریسک کمینه را هنگامی که وزن‌ها از اندازه‌ی یک در L^2 هستند به دست آوریم، می‌توانیم ببینیم که وزن‌های مذکور برابر با eigenvector های معادل با کوچک‌ترین eigenvalue می‌شوند و ریسک هم برابر با کوچک‌ترین eigenvalue خواهد بود. این ارتباط اساسی، یکی از مهم‌ترین کاربردهای eigen-vector ها می‌باشد.

۳.۲.۲ ارتباط بین eigenvector ها و پورتفولیوی بهینه

به صورت کاربردی دیده شده است که بزرگ‌ترین eigenvalue و eigenvector های متناظر آن، به شدت با بازار مالی هم‌بستگی دارند. لذا معمولاً تصمیم به کنار گذاشتن این eigenvector و استفاده از اوزان موجود در بردارهای بعدی گرفته می‌شود.

فصل ۳

بخش عملی پژوهش

۱.۳ جمع‌آوری داده

با توجه به اینکه می‌خواهیم تحقیقات را روی داده‌های مرتبط به بازار سهام ایران انجام دهیم، بایستی ابتدا داده‌های مورد نیازمان را از یکی از سایت‌های ارائه‌دهنده‌ی داده‌ها دریافت کنیم. یکی از سایت‌هایی که ابزار تحلیل تکنیکال ارائه می‌کند، سایت «نهایت‌نگر» است. در این وب‌گاه، مرورگر کاربر با ارسال یک درخواست REST اطلاعات سهام را دریافت می‌کند. با تحلیل ترافیک عبوری از مرورگرمان، متوجه می‌شویم که مرورگر اطلاعات هر سهم را از آدرس زیر دریافت می‌کند:

```
https://www.nahayatnegar.com/tv/chart/history?symbol=IRO3PZGZ00013&
    resolution=D&from=1562690329&to=1594312789
```

چند نکته در خصوص این URL مورد توجه هستند که به توضیح آنها می‌پردازیم.

۳.۱.۱ مقدار Symbol

مقدار symbol که در URL فوق قرار دارد، در واقع نشانگر(id) سهام مورد نظر ماست. هر سهم مقدار symbol خاص خودش را داراست و maping بین نام سهام و symbol را مرورگر از طریق API زیر به دست می‌آورد:

```
https://www.nahayatnegar.com/tv/chart/search?limit=30&query=%D8%BA%
    D9%BE&type=&exchange=&symbolsClientToken=
    a25e55e7926e74f6b7cc1acb902d8654&adjustmentType=3
```

که در آن query همان چیزی است که کاربر در کادر جستجو تایپ کرده. یعنی ابتدا با یک API Call به آدرس بالا، مقدار symbol را به دست می‌آورد و با ارسال آن symbol در آدرس اولیه، history مقادیر سهم را دریافت می‌کند.

۳.۱.۲ مقدار Resolution

در واقع این مقدار D به معنای Day است و یعنی بازه‌های زمانی که داده‌ها برایش دریافت می‌شوند، هر یک روز است. با توجه به اینکه ما هم می‌خواهیم Return سهم‌ها را در بازه‌های روزانه مورد بررسی قرار دهیم، این مقدار را به حال خود رها می‌کنیم.

۳.۱.۳ مقادیر From و To

این دو مقدار، در واقع Timestamp فیلتر شروع و پایان هستند و مشخص می‌کنند که از چه بازه‌ای تا چه بازه‌ای داده‌ها را درخواست می‌کنیم. این Timestamp به صورت Unix Timestamp ارسال می‌شود. این دو مقدار را نیز تغییر نمی‌دهیم.

۴.۱.۳ جمع‌بندی

به نظر می‌رسد ابتدا بایستی مقادیر symbol را برای هر سهمی که می‌خواهیم آن را مورد بررسی قرار دهیم بیابیم، سپس با ارسال API Call به آدرس اولیه، قیمت آن سهام را در بازه‌ی دلخواه به دست آوریم.

۲.۳ پیاده‌سازی

پیاده‌سازی این عمل در فایل `data_gathering_utils.py` انجام شده است. برای سهولت در ارسال و دریافت درخواستهای HTTP از کتابخانه‌ی Requests در پایتون استفاده شده که بایستی به عنوان Dependecy پروژه نصب گردد. تمام Dependency های پروژه در فایل `requirements.txt` ذخیره شده‌اند و با استفاده از دستور زیر قابل نصب هستند:

```
pip install -r requirements.txt
```

مستندات پیاده‌سازی

در فایل `data_gathering_utils.py` ، دو تابع قرار گرفته؛ یکی `get_stock_symbol(name)` که با استفاده از نام سهام، مقدار symbol را برای آن به دست می‌آورد و دیگری `get_stock_price_history(symbol)` که با استفاده از مقدار symbol سهام، تاریخچه‌ی قیمت سهام را به دست می‌دهد.

تست پیاده‌سازی

```
>>> data_gathering_utils.get_stock_symbol('GHAPINO')
'IRO1MINO00013'
```

```
>>> data_gathering_utils.get_stock_price_history('IRO1MIN000013')
dict(timestamps=[669859200, 670896000, ...], closing_prices=[450,
    450 ,451, ...])
```

تست‌ها نشان می‌دهند که توابع پیاده‌سازی شده به درستی عمل می‌کنند.

۳.۳ پاک‌سازی داده‌ها

۱.۳.۳ داده‌های ناموجود

یکی از مهم‌ترین بخش‌های برخورد با داده‌های بازار سهام، این است که بتوانیم با داده‌های missing رفتار صحیحی انجام دهیم. به طور کلی دو نوع missing value در بازار ایران داریم:

- نبود داده به علت بسته بودن کل بازار: مثلاً در تعطیلات عید نوروز
- نبود داده به علت بسته‌بودن سهم به خصوص: مثلاً بسته‌بودن سهم به خاطر برگزاری مجمع عمومی سالانه

نبود داده به علت بسته بودن کل بازار

در این مواقع، داده‌های آن روز را به طور کل در محاسباتمان نمی‌آوریم. این پیشنهادی است که بسیاری از محققان [1] در این خصوص ارائه می‌کنند.

[1] برای مثال Prof. Mazin A. M. Al Janabi در Dealing with the Problem of Missing Data: A Short Note.

نبود داده به علت بسته‌بودن سهم به خصوص

در این صورت، یکی از رویکردهای بسیار خوب، استفاده از geometric average بین مقادیر قبلی و بعدی برای تخمین‌زدن مقادیر مابین است که در بعضی تحقیقات [۲] به آن اشاره شده. پیشنهاد دیگر، استفاده از میانگین مقادیر قبلی و بعدی است که در این صورت، نتیجه سبب می‌شود که volatility سهم‌ها کمتر از آنچه واقعاً هستند تخمین زده شود. ما برای سهولت و همین‌طور از آنجا که volatility سهم برایمان موضوعیت ندارد، از روش تکرار مقدار قبلی استفاده خواهیم کرد. قطعا برای بهبودهای عمده در خروجی کار، بایستی روشی بهتر اتخاذ شود.

۲.۳.۳ قالب‌ریزی [۳] مقادیر به انواع صحیح

دیگر مسئله‌ی مهم در پاک‌سازی داده‌ها، آن است که بتوانیم ستون‌ها را در قالب‌های صحیحی وارد کنیم. ابتدای امر، مقادیر زمانی به صورت Unix Timestamp قرار دارند که توسط انسان به سختی قابل درکند. این مقادیر را به مقادیر رشته‌ای قابل فهم و با تاریخ جلالی تبدیل خواهیم کرد. همچنین خروجی تابع `get_stock_price_history` یک tuple است که بایستی به مقادیر مورد نیاز در کتابخانه‌ی قدرتمند `pandas` تبدیل شود که کار با داده به سهولت قابل انجام باشد.

۳.۳.۳ پیاده‌سازی

پیاده‌سازی ابزارهای مرتبط با پاک‌سازی داده‌ها در فایل `data_cleansing_utils.py` انجام گردیده است. همان‌گونه که ذکر شد، برای سهولت کار با داده‌های موجود، از کتابخانه‌ی `pandas` در پایتون استفاده شده است.

[۲] همان

[۳] Cast

برای این منظور، تابع خصوصی `timestamp_to_jalali` پیاده‌سازی شده است. این تابع یک Unix Timestamp دریافت کرده و به عنوان خروجی، تاریخ جلالی را ارائه می‌کند.

```
>>> str(data_cleansing_utils._timestamp_to_jalali_string(669859200)
    )
'1370-01-05'
```

با استفاده از متد `apply` روی Pandas DataFrame می‌توانیم یک تابع را روی تمام مقادیر یک ستون اجرا کنیم. این عمل در تابع `stock_data_to_pandas_df` پیاده‌سازی شده است. نهایتاً این تابع، خروجی تابع `get_stock_price_history` را دریافت کرده و یک Pandas DataFrame مانند شکل ۱.۳ خروجی می‌دهد.

	closing_prices
1398-11-30	4333
1398-12-03	4490
1398-12-04	4590
1398-12-05	4543
1398-12-06	4638
1398-12-07	4449
1398-12-10	4285
1398-12-11	4548
1398-12-12	4745
1398-12-13	4982
1398-12-14	5231
1398-12-17	5492
1398-12-19	5297
1398-12-20	5622
1398-12-21	5841
1398-12-24	5828
1398-12-25	5714
1398-12-26	6084

شکل ۱.۳ : قیمت‌های پایانی به صورت صحیحی از تابع به دست می‌آیند

که فرمتی قابل درک توسط انسان است. اما به وضوح نقاطی ناموجود در این داده‌ها داریم؛ برای

مثال بازه‌ی ۱۴ام تا ۱۷ام اسفندماه سال ۱۳۹۸ (که فعلاً نمی‌دانیم به دلیل تعطیلی بازار بوده یا بسته بودن سهم).

داده‌های ناموجود

در خصوص داده‌های ناموجود نوع اول (یعنی زمانی که بازار بسته است) ، نیازی به انجام کاری نیست، چرا که آن‌ها از قبل از بین مقادیر حذف شده‌اند. اما برای داده های نوع دوم (یعنی زمانی که سهم بسته بوده است) ، بایستی در روزهای مذکور، مقادیر بازه‌ی وسط را به وسیله‌ی مقدار قبلی پر کنیم. یعنی در صورتی که برایمان روشن شود که ۱۴ام تا ۱۷ام اسفند ۱۳۹۸ ، سهم بسته بوده است، مقدار ۵۲۳۱ را برای ۱۵ام و ۱۶ام قرار می‌دهیم.
برای پیاده‌سازی این مسئله، لازم است که سهم‌ها را به نحوی در یک Pandas DataFrame ترکیب کنیم تا بتوانیم روزهایی راکه یک سهام به خصوص بسته است را از روی بازبودن سهم‌های دیگر تشخیص دهیم. برای این منظور، تابع `combine_stocks_df` را ارائه می‌کنیم. عملکرد این تابع به این صورت است که دو Pandas DataFrame به عنوان ورودی گرفته و بر اساس کلید تاریخ، مقادیر را در کنار هم قرار می‌دهد و لذا می‌توانیم تشخیص دهیم که در چه روزهایی، کدام یک از سهم‌ها بسته و دیگری باز بوده‌اند.

	closing_prices_IRO1MINO00013	closing_prices_IRO1PKOD00013
1399-03-20	12103.00000	7245.00000
1399-03-21	12270.00000	7372.00000
1399-03-24	12682.00000	7730.00000
1399-03-25	13290.00000	8110.00000
1399-03-26	13470.00000	8460.00000
1399-03-27	nan	8840.00000
1399-03-31	nan	9280.00000
1399-04-01	nan	9740.00000
1399-04-02	12660.00000	9750.00000
1399-04-03	12030.00000	nan
1399-04-04	11720.00000	nan
1399-04-08	11880.00000	11190.00000
1399-04-09	11760.00000	11680.00000
1399-04-10	12140.00000	11700.00000
1399-04-14	11320.00000	12730.00000
1399-04-15	11980.00000	13330.00000
1399-04-16	12150.00000	13990.00000
1399-04-17	11670.00000	14430.00000

شکل ۲.۳: قیمت‌های پایانی سهام مختلف به درستی با یک‌دیگر ترکیب می‌شوند

این تابع به شکلی پیاده‌سازی شده که قابلیت ترکیب تعداد نامحدودی DataFrame را داشته باشد. به سادگی با ارائه‌ی DataFrame ها، می‌توانیم ترکیب آن‌ها را به دست آوریم و همان‌طور که در شکل ۲.۳ نیز واضح است، زمان‌هایی را که یک سهم به خصوص در بازار بسته بوده را تشخیص دهیم.

حال وقت آن است که مقادیر ناموجودی را که حاصل بسته‌بودن یک سهم به خصوص در زمان مشخص هستند پر کنیم. برای این کار تابع `fill_missing_values` را پیاده‌سازی میکنیم. این تابع یک DataFrame را دریافت می‌کند و به جای مقادیر NaN آن، مقدار قبلی را قرار می‌دهد.

	closing_prices_IRO1MINO00013	closing_prices_IRO1PKOD00013
1399-03-20	12103.00000	7245.00000
1399-03-21	12270.00000	7372.00000
1399-03-24	12682.00000	7730.00000
1399-03-25	13290.00000	8110.00000
1399-03-26	13470.00000	8460.00000
1399-03-27	13470.00000	8840.00000
1399-03-31	13470.00000	9280.00000
1399-04-01	13470.00000	9740.00000
1399-04-02	12660.00000	9750.00000
1399-04-03	12030.00000	9750.00000
1399-04-04	11720.00000	9750.00000
1399-04-08	11880.00000	11190.00000
1399-04-09	11760.00000	11680.00000
1399-04-10	12140.00000	11700.00000
1399-04-14	11320.00000	12730.00000
1399-04-15	11980.00000	13330.00000
1399-04-16	12150.00000	13990.00000
1399-04-17	11670.00000	14430.00000

شکل ۳.۳: قیمت‌های ناموجود به درستی پر می‌شوند

همان‌طور که در شکل ۳.۳ قابل مشاهده است، مقادیر روزهای ۲۷ و ۳۱ خرداد و ۱ تیر در سهم غپینو، توسط مقدار روز ۲۶ خرداد جایگزین شده‌اند که نشان از عملکرد صحیح این تابع دارد.

۴.۳ محاسبه‌ی وزن‌های پورتفولیو

حال که داده‌های مورد نیازمان را در یک فرمت مناسب قرار داده‌ایم، می‌توانیم تمرکز خودمان را بر روی پیاده‌سازی روش‌های جبر خطی برای یافتن بهترین اوزان پورتفولیو بگذاریم و دغدغه‌ی جمع‌آوری و یا استفاده‌ی مناسب از داده‌ها را نداشته باشیم.

۱.۴.۳ تئوری

با توجه به مطالبی که پیش‌تر ارائه شد، می‌دانیم که Eigenvalue ها و Eigenvector های ماتریس Covariance بازگشت‌های روزانه، معانی مشخصی دارند؛ بدین شکل که

- مقادیر Eigenvector در واقع اوزان مشخصی را برای یک پورتفولیو پیشنهاد می‌کنند.
- مقادیر Eigenvalue، میزان ریسک پورتفولیوی مشخص‌شده با Eigenvector را به دست می‌دهند.

لذا تلاش ما بر این خواهد بود که ابتدا مقادیر بازگشت روزانه را در DataFrame به دست آوریم(آنرا A می‌نامیم)، سپس با یافتن Eigenvector ها و Eigenvalue های ماتریس Covarince که همان A^TA است، پورتفولیومان را بهینه‌سازی کنیم.

بررسی روش‌های مختلف یافتن این مقادیر

راه مستقیم و البته نه خیلی هوشمندانه، شکل‌دادن ماتریس Covariance و سپس یافتن Eigen- value Eigenvector های آن است. ما این کار را برای شروع انجام خواهیم داد. اما مشخصاً می‌توان این امر را بهبود داد.

بهترین روش بهبود در این خصوص، به نظر من توجه به تناظر عمل مورد نظر با عملیات SVD روی ماتریس A است؛ چراکه ما ابتدا ماتریس A^TA را تشکیل می‌دهیم و سپس اقدام به محاسبه‌ی Eigenvalue Eigenvector های آن میکنیم و این امر در واقع تعریف محاسبه‌ی U در SVD روی ماتریس A است. پس در ادامه‌ی کار با استفاده از روش‌های مورد توصیه در کتب مرجع جبر خطی عددی، اقدام به بهینه‌سازی SVD مذکور خواهیم کرد و در خصوص سرعت روش‌های مختلف، Benchmark ارائه خواهیم داد.

۲.۴.۳ پیاده‌سازی

محاسبه‌ی ماتریس A

برای محاسبه‌ی ماتریس A که حاوی بازگشت‌های روزانه‌ی سهم‌هاست، تابع `get_returns_df` را پیاده‌سازی می‌کنیم. این تابع در واقع یک DataFrame که حاوی مقادیر قیمتی برای هر سهم است را دریافت کرده و میزان تغییرات روزانه‌ی هر سهم را با قیمت آن جایگزین می‌کند.

محاسبه‌ی ماتریس Covariance

تابع `get_covariance_df` مقدار `returns_df` را ورودی گرفته و ماتریس Covariance را خروجی می‌دهد.

محاسبه‌ی Eigenvalue هایEigenvector ماتریس Covariance

برای محاسبه‌ی این بخش، از کتابخانه‌ی numpy استفاده شده است. تابع `get_eigens` به ترتیب Eigenvalue ها و Eigenvector ها را باز می‌گرداند.

محاسبه‌ی وزن‌ها توسط روش ساده

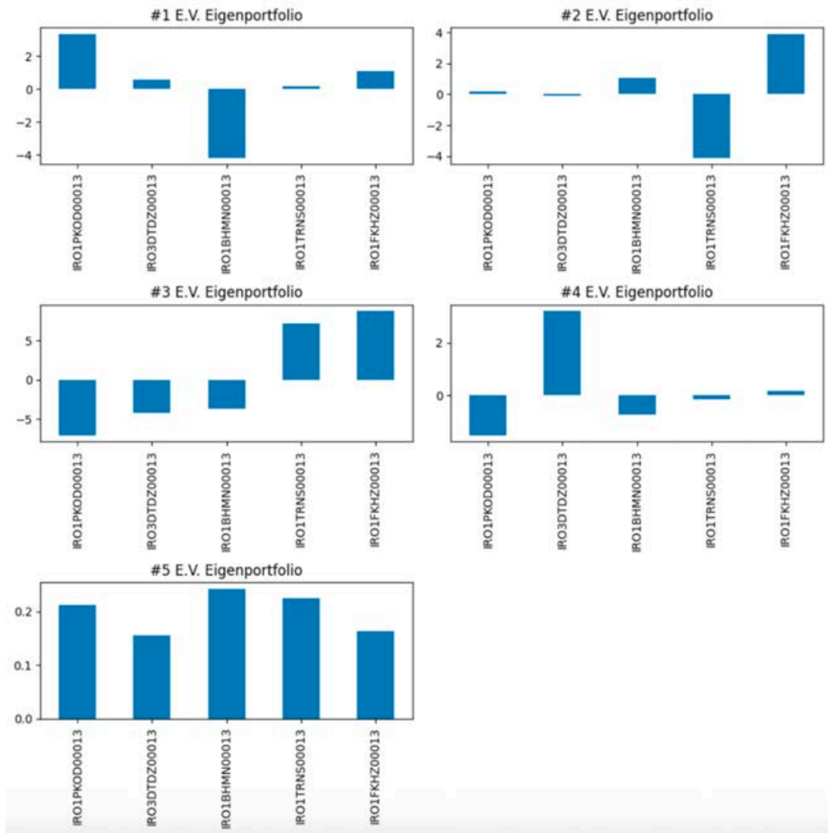
با قطعه‌کد زیر، می‌توانیم یک بار وزن‌های مناسب را توسط این روش برای پورتفولیوی پنج سهمه‌ی «خپارس»، «دتولید»، «خبهمن»، «بترانس» و «فخوز» به دست آوریم:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import data_gathering_utils
import data_cleansing_utils
```

```
from data_manipulation_utils import get_eigens, get_covariance_df,
    get_returns_df
SAHAM = ['KHEPARS', 'DETOLID', 'KHEBAHMAN', 'BETERANS', 'FAKHOOZ']
symbols = []
for sahm in SAHAM:
  symbols.append( data_gathering_utils.get_stock_symbol(sahm))
data = []
for symbol in symbols:
  data.append( data_gathering_utils.get_stock_price_history(symbol))
df = []
for d in data:
  df.append( data_cleansing_utils.stock_data_to_pandas_df(d))

df = data_cleansing_utils.combine_stocks_df(*df)
df = data_cleansing_utils.fill_missing_values(df)

returns_df = get_returns_df(df).tail(100)
covariance_matrix = get_covariance_df(returns_df)
D, S = get_eigens(covariance_matrix)

tickers = returns_df.columns.copy()

f = plt.figure(figsize=(10, 10))
```

```python
for i in range(S.shape[1]):
  ax = plt.subplot(3, 2, i + 1)

  eigenportfolio_i = S[:, i] / np.sum(S[:, i]) # Normalize to sum to 1
  eigenportfolio = pd.DataFrame(data=eigenportfolio_i, columns=['Investment Weight'], index=tickers)
  eigenportfolio.plot(kind='bar', ax=ax, legend=False)
  plt.title("#{} E.V. Eigenportfolio".format(i + 1))
```

این قطعه‌کد که در فایل `test_simple_way.py` قرار داده شده است، ابتدا ماتریس Covariance را محاسبه می‌کند، سپس با محاسبه‌ی Eigenvalue Eigenvector های آن، پورتفولیوی معادل با Eigenvector ها را ساخته و اوزان را به صورت نمودار ستونی نشان می‌دهد. خروجی این روش در شکل ۴.۳ قابل مشاهده است.

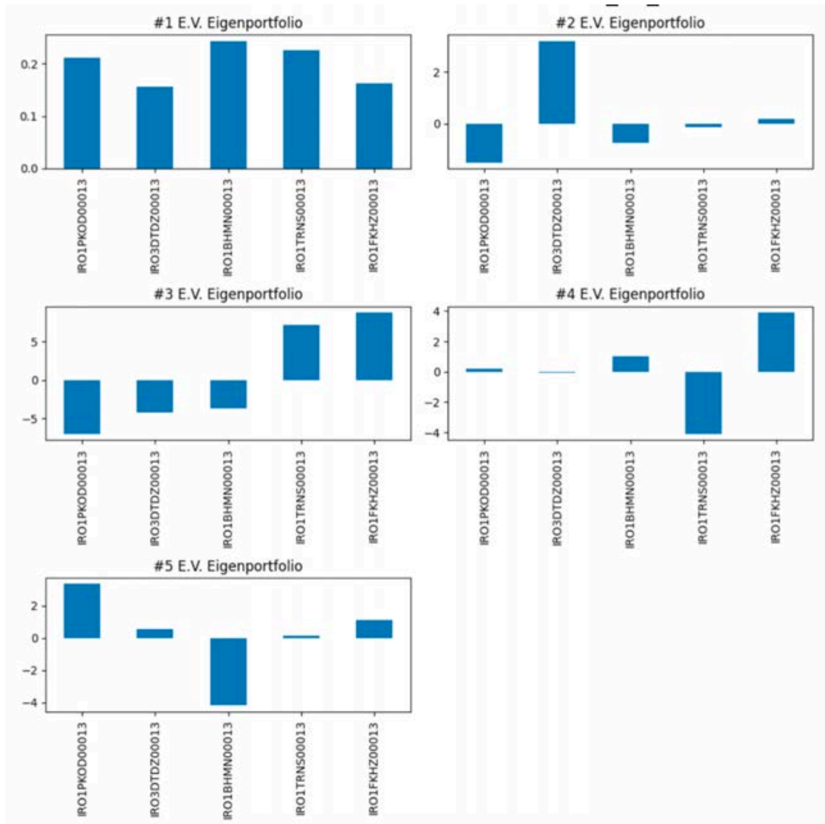


شکل ۴.۳: اوزان به دست آمده با استفاده از روش ساده.

اوزان منفی به معنی short کردن هستند که در بازار ایران هنوز در دسترس نیست.

محاسبه‌ی وزن‌ها توسط روش SVD

می‌دانیم که ستون‌های V در SVD همان Eigenvector های A^TA هستند؛ لذا ابتدا یک‌بار SVD را روی A اجرا می‌کنیم تا به V دست پیدا بکنیم و با استفاده از ستون‌های V می‌توانیم پورتفولیوی خود را شکل دهیم. پیاده‌سازی این روش در فایل `test_svd_way.py` آمده است. خروجی این روش در شکل ۵.۳ قابل دیدن است.



شکل ۵.۳: اوزان به دست‌آمده از روش تجزیه‌ی مقادیر منفرد

این روش در مقاله‌ی اصلی موجود نبود و با توجه به ویژگی‌های SVD و ماتریس Covariance و همین‌طور مطالعه در خصوص PCA به دست آمد. توجه به خروجی‌های دو روش، نشان می‌دهد که هر دوی آنها به نتیجه‌ی مشابهی دست پیدا می‌کنند.

۳.۴.۳ مقایسه‌ی سرعت دو روش محاسبه‌ی اوزان

برای بررسی سرعت دو روش، از benchmarking مبتنی بر تکرار استفاده شده است؛ بدین صورت که ابتدا ۱۰۰ بار محاسبه‌ی اوزان از روش SVD انجام شده است، سپس ۱۰۰ بار به روش Simple انجام شده و نهایتاً میانگین زمان لازم برای یک محاسبه از طریق میانگین‌گیری روی تمام زمان‌های

موجود به دست آمده است. همان‌طور که پیش‌بینی می‌شد، داده‌های به دست آمده نشان‌گر آن هستند که استفاده از روش محاسبه‌ی مبتنی بر SVD حدود ۳ و نیم برابر سریع‌تر از تشکیل ماتریس Covariance و محاسبه‌ی Eigenvalue Eigenvector های آن به طول می‌انجامد:

```
Mean time of SVD: 000136.0
Mean time of Simple: 000506.0
```

پیادهسازی این Benchmark در فایل `benchmark.py` آمده است و قابل تکرار و مقایسه می‌باشد.

۵.۳ نتیجه‌گیری

همان‌طور که ذکر شد، محاسبه‌ی Eigenvector های ماتریس Covariance بازگشت‌های روزانه‌ی سهام، می‌تواند به وزن‌های بهینه برای پورتفولیو منجر شود. در این پروژه، موفق شدیم با استفاده از روش SVD سرعت رویکردی که در مقاله‌ی اصلی وجود داشت را تا ۳ و نیم برابر افزایش دهیم؛ چرا که برای یافتن این Eigenvector ها نیازی به تشکیل ماتریس Covariance و یا محاسبه‌ی Eigenvector ها نیست.
ماتریس Covariance عبارت است از:

$$C = A^T A$$

از آنجا که ماتریس Covariance یک ماتریس متقارن است، می‌توان آن را به صورت زیر قطری نمود:

$$C = VLV^T$$

که V ماتریس Eigenvector هاست و L هم ماتریس قطری Eigenvalue ها می‌باشد. در واقع Eigenvalue های ماتریس Covariance همان Principal Axes در PCA هستند. حال اگر

SVD را روی ماتریس اولیه (A) انجام دهیم، خواهیم داشت:

$$A = U\Sigma V^T$$

که در آن V معادل همان Principal Axes خواهد شد؛ چرا که داریم:

$$C = V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T$$

پس به جای استفاده از روش مستقیم تشکیل ماتریس Covariance و محاسبه‌ی Eigenvector ها، می‌توانیم به سادگی از SVD استفاده کنیم و به نتیجه برسیم. همچنین با Benchmark موجود در انتهای پروژه، نشان دادیم که این تغییر می‌تواند ۳ و نیم برابر سرعت کارمان را افزایش دهد که هنگام کار با ماتریس‌های بسیار بزرگ(برای پورتفولیوهای دارای تعداد زیاد سهام)، بسیار عالی خواهد بود.

واژه‌نامه فارسی به انگلیسی

مجموعه داده	Dataset
پورتفولیو	Portfolio
بازگشت	Return
امید بازگشت	Expected Return
توزیع دارایی	Asset Distribution
ریسک مالی	Financial Risk
نسبت شارپ	Sharpe Ratio
هم‌بستگی	Correlation
مقدار ویژه	Eigenvalue
بردار ویژه	Eigenvector
داده‌ی ناموجود	Missing Value
قالب‌ریزی	Cast
وزن	Weight

مراجع

[1] Ruey S. Tsay, “Analysis of Financial Time Series, 3rd Edition”, WILEY, 2010.

[2] MIT OCW 18.S096 ”Topics in Mathematics with Applications in Finance”, Fall 2013.

[3] Scott Rome, ”Eigen-vesting I. Linear Algebra Can Help You Choose Your Stock Portfolio”.

[4] ”Dealing with the Problem of Missing Data: A Short Note”, Prof. Mazin A. M. Al Janabi, PhD, SNI II.

[5] Ray Dalio, ”Principles”, Simon and Schuster, 2017.

[6] Sonam Srivastava, ”Portfolio Optimization Methods”.

[7] Alex Putkov, ”Portfolio optimization in Modern Portfolio Theory”.

[8] ”Portfolio Optimization”, Wall Street Mojo.

Abstract

When working with huge amounts of assets, what counts more than any other factor is the correct way of portfolio management and risk optimization. If we optimize our portfolio for the least amount of risk possible, we can make sure that no matter how the market is, our capital isn't at risk. In this project, we first gather our needed data using a real-time approach, then we use EigenDecomposition-based methods to form a portfolio that has the least amount of risk. Lastly, we use SVD to make the referenced article's approach much faster(more than 3.5 times).
Keywords: Stock Exchange, Portfolio Optimization, Singular Value Decomposition

College of Science

School of Mathematics, Statistics, and Computer Science

Portfolio Optimization in Tehran Stock Exchange

Mohammad Teimori Pabandi

Supervisor: Dr. Samaneh Eftekhari Mahabadi

A thesis submitted to Graduate Studies Office

in partial fulfillment of the requirements for the degree of

B.Sc.in

Computer Science

August , 2020