Mean-Variance Portfolio Management with Alpha Quantitative Investing Strategy

Binbin Chen
Department of Computer Science
2019533081
chenbb@shanghaitech.edu.cn

Yuling Yuan
Department of Computer Science
2019533107
yuanyl@shanghaitech.edu.cn

Abstract

In this project, we build mean-variance portfolio management with the alpha quantitative investing strategy in the SSEC financial market dataset from 2010 to 2021. We first select the proper dataset, do feature engineering, and adjust the dataset to company-parallel form for covariance computation. Then we have a series of data cleaning methods to deal with the potential issue of data. Next, in alpha strategy, we first derive vital signals and themes for use, and then build an alpha model through a multi-factor framework via OLS and WLS regression in 2 stages: combine signals to themes, and combine themes to a multi-factor model. We begin this strategy in 2015, and iteratively operate it with a monthly expanding window. Similarly, after we construct a portfolio with the top-k stock selection strategy and analyze the covariance of these stocks, we optimize our portfolio monthly again to have weights redistribution. When the loop ends, we compare the monthly returns of our portfolio to that of the equal-weight portfolio and get better median results.

1 Introduction

Financial trading has been a widely researched topic with various proposed methods over the last few decades, and how to make profitable decisions in trading is significant and challenging for investors. While according to the efficient market hypothesis, the cornerstone of modern financial theory, share prices reflect all information and consistent alpha generation is impossible, in reality, however, most markets do display some level of inefficiencies, suggesting that it could be possible to outperform the overall market through expert stock selection and quantitative investing strategy.

With the non-linear and dynamic nature and uncertainty of markets, the weak form efficiency stimulates more and more investors to have active investments, including fundamental analysis, technical analysis, and algorithm trading. Recently, the alpha investing strategy and mean-variance portfolio optimization method has attracted various attention in the financial market, and it's a common belief that this investment process, designed to achieve specified target return and risk levels, has an excellent ability to identify valid information, extract relevant patterns and outperform the market.

2 Data Preparation

2.1 Dataset Selection

After an initial review of the given Chinese financial market datasets in CNT, SZI, and SSEC, we found that a large percentage of CNT stocks have NAN values between 2010 and 2019, especially in the early years, due to the fact that most GEM stocks are emerging companies that are not yet listed

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

in that period. These NAN values make less training data available. Also, the price data of the CNT market is highly volatile and potentially depends on other important factors rather than the financial signals we will use.

In contrast, only a few of the 500 companies in the SZI and SSEC datasets have invalid price data during the decade. As the most popular financial markets in China, the data seems more robust and solid. While SZI and SSEC markets share similar advantages regarding data feasibility, the SSEC dataset provides a more compact stock basket for selection. Thus, we finally decided to use SSEC market data for the subsequent alpha construction, stock selection, and portfolio management.

2.2 Feature Engineering

Return First, we use the price data in the SSEC market to calculate returns as target variable using the following formula1, to be more specific, we acquire 1-month return, 3-month return, 6-month return, 9-month return, and 12-month return. Then the current month's returns after these months (retf1,retf3,retf6,retf9,retf12) are convenient for the current month's financial data to make an Alpha strategy. Owing to the absence of corresponding price data, some returns in 2021 retain the default value of -1 for the portion of the returns that cannot be calculated.

$$\frac{CLOSE_{t+1,3,6,9,12} - CLOSE_t}{CLOSE_t} \tag{1}$$

annualROE When viewing ROE we find that only at March, June, September and December of each year there is ROE data, and ROE at the rest of the months are all NAN, which we believe is due to the fact that ROE data comes from the quarterly and annual reports of enterprises. However, such a high percentage of missing values will lead to the ROE column being directly deleted in the following process of data cleaning. Therefore, we annualize and adjust the data in the ROE column by adding a new column of annualROE. The calculation method is as follows: for 3, 6, 9 and 12 months, annualROE is ROE*4, ROE*2, ROE/3*4 and ROE respectively; for other months, annualROE is the same as the annualROE of the stock 1 or 2 months ago. Therefore, it's safe and available for us to use this feature as a valid signal in the alpha strategy afterwards.

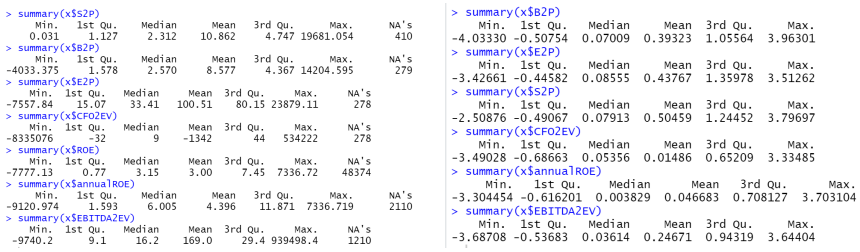
Dataset Adjustment To better deal with the covariance between different companies in the SSEC market which can help optimize our financial portfolio, we build new datasets concerning companies' covariance with different returns via transforming the original dataset to the new form. Difference from the returns mentioned above, these returns should not include future information, so we use the current month's returns before these months (retf1,retf3,retf6,retf9,retf12). Then we split stocks at each return into a single file (like figure1) for the convenience of covariance calculation, also the stocks at each alpha into a single file that we will cover later. In simple terms, when the alpha strategy uses retf9, the corresponding return to calculate the covariance should be retf9. At this point, the portfolio data is obtained and the covariance information between the returns of the companies in the portfolio over a certain period of time can be calculated.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	yyyymm	600000.SH	600004.SH	600006.SH	600007.SH	600008.SH	600009.SH	600010.SH	600011.SH	600012.SH	600015.SH	600016.SH	600017.SH
2	201010	-0.2619776	-0.038282	0.0652921	0.0523416	-0.0302115	-0.2398922	0.1191067	-0.125	-0.0622463	0.20241411	-0.2448133	-0.386236
3	201011	-0.3881389	-0.2410866	-0.1243863	0.07582516	-0.161529	-0.3646813	-0.0337349	-0.2113821	-0.2820197	0.00455789	-0.308322	-0.4424899
4	201012	-0.4561018	-0.2249561	-0.2305246	-0.1443124	-0.132732	-0.3313546	-0.1040189	-0.2131148	-0.192623	-0.149766	-0.3472042	-0.4680574
5	201101	-0.3822244	-0.164557	-0.1546053	-0.0093071	-0.0639777	-0.1246883	-0.0153061	-0.1967213	-0.1104101	-0.1234277	-0.2834758	-0.4312321
6	201102	-0.3144552	-0.0577342	0.06163022	0.16135881	0.12274959	0.08825729	0.41297935	-0.0988655	0.11052632	0.08254269	-0.2186544	-0.2951907
7	201103	0.00147059	-0.0260476	0.14606742	0.04008016	0.24680073	0.14045416	1.50167224	-0.1263823	0.14634146	0.12859712	-0.0760331	-0.2228464
8	201104	-0.046729	-0.1553589	-0.0212766	-0.1692982	0.09933775	0.0685241	1.03529412	-0.0484375	0.02857143	0.0170593	0.10017889	-0.3227273
9	201105	-0.0092924	-0.1793313	-0.2024648	-0.25	0.00829187	0.00754148	1.01176471	-0.0738916	-0.0104712	0.05726496	0.12430427	-0.3587786
10	201106	-0.2407407	-0.173822	-0.2021858	-0.216566	-0.0675453	0.024	1.53030303	-0.1400651	-0.011236	0.00648148	0.12352941	0.00240964
11	201107	-0.3508287	-0.2436893	-0.2983871	-0.1815009	-0.1199377	-0.0723404	0.68736142	-0.2282609	-0.2582973	-0.2046332	-0.003663	-0.1075515
12	201108	-0.2679275	-0.2013423	-0.2654206	-0.2222222	-0.2161765	0.08388158	0.64339152	-0.2216495	-0.1903945	-0.0435572	0.18145957	-0.1334951
13	201109	-0.3107345	-0.19161	-0.25	-0.1001984	-0.2704309	-0.047619	0.46701847	-0.2690972	-0.2724196	-0.0724771	0.09960159	-0.2132353
14	201110	-0.2759434	-0.1503497	-0.2568093	0.05532359	-0.1738484	-0.0897436	0.32901554	-0.1150278	-0.2003546	-0.0044843	0.21471173	-0.1788413
15	201111	-0.3312352	-0.2034682	-0.3539326	-0.0987203	-0.1982507	-0.1092784	-0.0772443	-0.1294964	-0.3191153	-0.0893953	0.13502935	-0.2894118
16	201112	-0.376652	-0.2802326	-0.3843137	-0.1213873	-0.2375367	-0.0973451	-0.4491979	-0.0325497	-0.314239	-0.1051793	0.05366726	-0.3493976
17	201201	-0.3543417	-0.2479627	-0.3675889	-0.0337909	-0.2138554	-0.119098	-0.2962428	-0.0968801	-0.2973856	-0.048722	0.04390244	-0.3668904
18	201202	-0.3116883	-0.1641975	-0.1479029	0.10816777	-0.0921053	-0.0179641	-0.2309942	-0.0301418	-0.1922399	-0.0646726	0.08745875	-0.2761905
19	201203	-0.0924797	-0.1457541	-0.2465753	0.12444934	-0.130742	0.009375	-0.2874251	-0.0340909	-0.1515152	-0.0128795	0.09424084	-0.3125
20	201204	0.00106383	-0.0937099	-0.2	0.16950959	-0.040708	-0.0091743	-0.2076216	0.06639839	-0.0758755	0.08834951	0.22242647	-0.2205128
21	201205	-0.0559742	-0.0070028	-0.1043257	0.14818763	0.03752345	-0.0091047	-0.0394537	0.20309051	-0.0423729	-0.0645161	0.06343907	-0.1344538
22	201206	-0.0480094	-0.0448808	-0.1322314	0.14663727	-0.0040733	0.07966102	-0.1223022	0.53206651	-0.0139535	-0.0633037	0.08514493	-0.1401869
23	201207	-0.1682953	-0.1399177	-0.2408377	-0.0187933	-0.1978417	-0.0187793	-0.0253411	0.4129979	-0.1308204	-0.2072072	-0.0163666	-0.1779141

Figure 1: retf9

2.3 Data Cleaning

The unprocessed raw data set has many problems, such as some of the data is missing and some of the data is clearly far beyond the normal data range2(a)2(b). In order to make the regression model more accurate and reduce the impact of these outliers, we need to clean the raw data.3



(a) Data before cleaning (b) Data after cleaning

Figure 2: Dataset Exploration

```
for(j in 1:signals.num)
{
  x_train = dataMiss(x_train,signals.name[j],0.3)
  x_train = dataTruncate(x_train,signals.name[j])
  x_train = dataStandardize(x_train,signals.name[j])
  x_train = dataWinsorize(x_train,signals.name[j]) # This is done after the standarization
  x_train = dataNeutralize(x_train,signals.name[j], 'sectorName')
  x_train = dataStandardize(x_train,signals.name[j])
  x_train = dataReplaceMissing(x_train,signals.name[j],miss.fill = "mean")
  x_train = dataDistribution(x_train,signals.name[j],1)
}
```

Figure 3: Code for Data Cleaning Process

Miss value detection If there are too many missing values in a feature, it is considered that there is too little information available from the feature. Moreover, missing values are not permitted in the regression process. That means all missing values must be replaced. But if there is little data available, replacement becomes tricky. We directly delete features with a missing value ratio greater than 0.3, such as "ROE". Meanwhile, We can also replace missing values with the feature's mean.

Outlier Treatment If we use raw values full of outliers and errors for investment the results will be totally meaningless. Once a signal passes the missing values threshold, we will clean up the data by removing errors and outliers and then standardize the scores and winsorize the outliers. First, for the truncation, we set values of more than 5 standard deviations (sigmas) to NA. Then values are subtracted by the feature's mean and divided by the feature's standard deviation, which can help to compare factors apples-to-apples. Next, similar to truncation in the sense of outlier treatment but different in value adjustment, here we keep outliers and shrink values of more than 3 sigmas to 3 sigmas, while the truncation simply removes outliers.

Neutralization Industries have special impacts on signal values, which implies that without de-meaning to remove industry effects, stock selection will be highly concentrated and skewed in some industries resulting in more of an industry selection than a stock selection. To tackle this problem, values are subtracted by the feature's mean to neutralize the industry effects. Then Re-standardization is employed to make sure factors have the same mean and standard deviation because industry demeaning may change the distribution of features' values.

Distribution Distribution Matters depend on many considerations. For example, for an investment universe of small-cap companies or emerging markets, feature's values are usually more wild than for large-cap companies in developed markets. The former may require ranking, while the latter can be normalized.

3 Alpha Strategy

3.1 Signal Exploration

After the process of data preparation, there are 6 signals as independent variables for the regression of further alpha strategy: B2P, E2P, S2P, CFO2EV, annualROE, and EBITDA2EV. They mainly belong to 2 themes distinctively that can impact the future stock returns: Profitability and Value.

Profitability A company needs to make profits, which are important for the company's survival and growth. Furthermore, a company is more investable if it makes sustainable profits, not as just a one-time phenomenon. So when we investigate signals like annualROE and EBITDA2EV, we want the corresponding company to be more profitable which indicating the greater its ability to create value for investors.

The signal ROE (Return on Equity) measures the value created for shareholders, where the return is measured by net income from the income statement as shown in formula2. Notice that the numerator is called the trailing twelve-month (TTM) (the information is published quarterly) value when the t indicates one quarter. Therefore, to better tackle this signal on a monthly basis, we transform it to annual value thus creating the signal annualROE.

$$ROE = \frac{NI}{Equity} = \frac{\sum_{t=1}^{4} NI_t}{AverageShareholder'Equity} \tag{2}$$

As a popular evaluation signal, EBITDA2EV is earnings before interest, taxes, depreciation, and amortization to the enterprise value ratio comparing the value of a company debt included-to the company's cash earnings less noncash expenses as in formular3. EV calculates a company's total value or assessed worth, while EBITDA measures a company's overall financial performance and profitability. The inversed ratio EV2EBITDA is most commonly used to compare companies in the same industry or sector. like the P/E ratio (price-to-earnings), the higher the EBITDA2EV, the cheaper the valuation for a company. There are benefits to using the P/E ratio along with the EBITDA2EV. For example, many investors look for companies that have both low valuations using P/E and EBITDA2EV and solid dividend growth.

$$EBITDA2EV = \frac{EBITDA}{EV} \tag{3}$$

Value Value signals measure the difference between market and accounting valuations of a public company, and investors usually utilize value signals from three different angles: net income, cash flow, and balance sheet. So when we investigate signals like B2P, E2P, S2P, and CFO2EV, we want to find out whether a company is overvalued or undervalued by the stock market.

B2P (Book value to market capitalization) is the value signal from the balance sheet, measured by the accounting value of the company's common shares divided by the market value of those common shares as in formula4, where the book value is the total shareholders' equity and the market capitalization equals to the multiple of stock price and total shares outstanding.

$$B2P = \frac{BookValue}{MarketCapitalization} \tag{4}$$

E2P is the value signal from the income statement, reflecting the earnings support per share for the stock price: the price comes from earnings and how the market evaluates per dollar of earnings as in formula5, where the EPS (Earnings per share) is calculated as a company's profit divided by the outstanding shares of its common stock with resulting number serves as an indicator of a company's profitability.

$$E2P = \frac{NI}{Marketcapitalization} = \frac{EPS}{Price} \tag{5}$$

The S2P (sales per share to price) ratio is a valuation ratio that compares a company's revenues to its stock price as in formula6, where the SPS states sales per share and the MVS states market value per

share. It is an indicator of the value that financial markets have placed on each dollar of a company's sales or revenues. Like all ratios, the P/S ratio is most relevant when used to compare companies in the same sector. A high ratio may indicate the stock is undervalued, while a ratio that is significantly below the average may suggest overvaluation.

$$S2P = \frac{SPS}{MVS} \tag{6}$$

CFO2EV (cash flow to enterprise value) is an alternative value signal of measuring cash flow generating ability as in formula7, comparing the ability to generate cash flow with the total valuation of the company. The higher the CFO2EV ratio, the faster a company can pay back the cost of its acquisition or generate cash to reinvest in its business.

$$CFO2EV = \frac{CFO}{EV} \tag{7}$$

Correlation Before moving on to the construction of a multi-factor model, we need to analyze the correlation between signals and returns with univariate analysis and also the self-correlation of these signals with bivariate analysis. Since a 0.4 correlation for OLS is seriously high enough to cause collinearity, we can find in figure4 that it's a potential possibility for S2P and E2P to have collinearity issues. We also investigate the efficacy of each signal with forwarding stock returns of 1, 3, 6, 9, and 12 months as in figure5. It's an interesting phenomenon that all these signals have a long-term effect, that is, the longer the investment horizon, the higher the forecasting power as measured by correlation, which means they are suitable for medium- to long-term investing strategies.

	B2P	E2P	S2P	CFO2EV	annualROE	EBITDA2EV
B2P	1.00000000	0.30352034	0.55385655	0.09054925	0.106041385	0.45708170
E2P	0.22397961	1.00000000	0.27069210	0.03785084	-0.217706926	0.65169921
S2P	0.50141895	0.19368430	1.00000000	0.05179170	0.001634019	0.47925975
CFO2EV	0.05565814	0.03272613	0.02763944	1.00000000	0.098710871	0.04699757
annualROE	0.08090507	-0.11894279	0.01305626	0.07753953	1.000000000	-0.13879162
EBITDA2EV	0.35764852	0.59374043	0.38968353	0.03078143	-0.020222533	1.00000000

Figure 4: Self Correlation of Signals

	retf1	retf3	retf6	retf9	retf12
B2P	-0.035184774	-0.047909216	-0.049592401	-0.060471322	-0.067460134
E2P	0.002917819	-0.002179522	-0.001123461	-0.003183001	-0.004510204
S2P	-0.030406481	-0.043318919	-0.051671121	-0.063002290	-0.069949645
CFO2EV	-0.006260176	-0.011298922	-0.018862805	-0.026993617	-0.037345915
annualROE	0.015291602	0.005067693	-0.022071432	-0.045066966	-0.056594943
EBITDA2EV	-0.014480129	-0.026533802	-0.031347414	-0.040247931	-0.045488806

Figure 5: Pearson Correlation between Signals and Returns

3.2 Alpha Construction

3.2.1 Combine signals into a theme

```
fit <- lm(retf1~B2P+E2P+S2P+CFO2EV,data=x_train[!is.na(x_train$retf1),])
summary(fit)
#install.packages("lmtest")
library(lmtest)
print(bptest(fit))
fit_wls<- lm(retf1~B2P+E2P+S2P+CFO2EV,data=x_train[!is.na(x_train$retf1),],weights=1/abs(fit$residuals))
dataweight(fit_wls)
weight1[1,1:4]<-dataweight(fit_wls)
x_train$VALUE = weight1[1,1]*x_train$B2P + weight1[1,2]*x_train$E2P + weight1[1,3]*x_train$S2P + weight1[1,4]*x_train$CFO2EV
```

Figure 6: Code for Obtaining Theme VALUE

We construct ALPHA model based on retf1 data. Since there are four signals(B2P,E2P,S2P and CFO2EV) which related to VALUE, we combine those four signals to a theme VALUE. First, we do regression between retf1 and B2P+E2P+S2P+CFO2EV. Then we do bptest to this fit. It turned out that the p-value of Breusch-Pagan test is small, which means the issue of heteroscedasticity affect a lot.

So we re-estimate the model by taking consideration of variation of errors with OLS, which means weighted least squares (WLS). At the end, we save the weight from fit's t-value with $W_k = \frac{t_k}{\sum_{i=1}^{4} t_i}$ and obtained 4 weights6. We do similar operation to retf1 and annualROE+EBITDA2EV. Then we combine them to a theme PROF with 2 weights.

3.2.2 Combine themes into ALPHA

```
fit <- lm(PROF~VALUE,data=x_train)
summary(fit)
x_train$PROF.resid = fit$residuals
fit <- lm(retf1 ~ VALUE + PROF.resid, data = x_train[!is.na(x_train$retf1),])
print(bptest(fit))
fit_wls<-lm(retf1 ~ VALUE + PROF.resid, data = x_train[!is.na(x_train$retf1),],weights=1/abs(fit$residuals))
weight1[1,7:8]<-dataweight(fit_wls)
x_train$ALPHA1 = weight1[1,7]*x_train$VALUE + weight1[1,8]*x_train$PROF
```

Figure 7: Code for Remove Correlation between Themes

After achieving VALUE and PROF, we do standardization to these two themes. Then we do regression between PROF and VALUE to see whether there is correlation between themes. The R square seems not very high, but we still employ a residual approach, which means using PROF.residuals as a "cleaned" factor. So We do regression between retf1 and VALUE+PROF.resid and achieve 2 weights7. Now we can obtain ALPHA1 from two themes. At last, we do regression between retf1 and ALPHA1, since there may exist difference in orders of magnitude. We achieve the last 2 weights from the fit's coefficients, with w1*ALPHA1+w2=retf1.

3.2.3 Computing ALPHA and do forecast

We do ALPHA computing and forecasting on retf1. We separate the data set to train and test set. If training set is too small, it is hard for ALPHA to achieve a great performance. So we decide that at the first loop, training set start from the beginning(201001) to 201412. We do data cleaning and ordering on training set and then use them to obtain the first relevant parameters of the ALPHA strategy. Testing set also start from beginning, while ending 1 month after training set(201501). After doing data cleaning and ordering on testing set, we use ALPHA strategy's weights to forecast ALPHA of 201502 using 201501's features. For the second loop, training set ends at 201501, while testing set ends at 201502. We do this loop until testing set ends at 202111.

4 Portfolio Management

4.1 Top-K Stock Selection Strategy

Rather than just picking stocks from the SSEC basket randomly, we consider stock selection as an important procedure of portfolio construction. When combining all impacts of alpha, real return, and the predictive ability of alpha, we make a comprehensive top-k stock selection strategy.

```
racor <- alphacorrelation(retf9s, alpha9s, cr9s)
alpha_mean = apply(alpha9s,2,mean)
a9mean <- array(dim = c(length(cr9s),1),dimnames = list(cr9s))
for(j in 1:length(cr9s))
{
  a9mean[j,1] = alpha_mean[j]
}
ret_mean = apply(retf9s,2,mean)
r9mean <- array(dim = c(length(cr9s),1),dimnames = list(cr9s))
for(j in 1:length(cr9s))
{
  r9mean[j,1] = ret_mean[j]
}

racor <- scale(racor)
a9mean <- scale(a9mean)
r9mean <- scale(r9mean)
score = 0.4*racor[,1]+0.3*a9mean+0.3*r9mean
selection <- cbind(racor[,1],a9mean,r9mean,score)
selection = selection[order(selection[,4],decreasing = TRUE),]
write.csv(selection,'selection.csv')
```

Figure 8: Sectional Code for Top-K Stock Selection

Since it's hard to calculate covariance in the period of portfolio management with NAN price value, we first drop off the stocks with NAN value in the price data from the financial basket. Then we separate the dataset as data before 2015 and data after 2015 as before. In the data before 2015, we just have an overall calculation on the training of the alpha without a rolling window, which means we use the dataset from 201001 to 201412 to build alpha and use the alpha to construct a portfolio. In the data after 2015, we can iterate the portfolio optimization each month based on the stocks chosen before, and have portfolio management with a rolling window.

Considering the outstanding predictive ability of alpha9 (forecast return after 9 months), we utilize factors including the mean alpha9 value, the mean retf9 value, and the accuracy of fitting of alpha9 of each stock to build weighted scores for top-k stock selection as shown in figure8: We first calculate the former 2 factors separately on the single file of alpha9 and retf9 and compute the correlation between alpha9 and retf9 to assess the fitting performance of alpha9. Then to combine these factors together, we do scaling on them, the same thing as standardization. Next, we assign 30%, 30%, and 40% weights on the 3 factors in sequence to acquire the evaluation scores for final stock selection.

In this way, when we set K as 10, we can get the following stocks in figure9. Since the stock selection strategy is more robust and solid than random pick, and it's an executable method based on financial intuition, we can further choose other K values for portfolio construction. But in order to prevent a high turnover rate, we don't apply this strategy iteratively, and thus it's a trade-off between a useful stock selection strategy and implementation cost.

	V1	V2	V3	V4
X600381.SH	-1.218278222	-0.333041853	9.789882067	2.3497408
X600446.SH	2.741784717	-0.240631672	3.490150760	2.0715696
X600155.SH	1.642044416	1.626908031	2.010620774	1.7480764
X600228.SH	1.679216254	1.138273557	1.921804550	1.5897099
X600330.SH	2.285820495	0.499174767	1.667509710	1.5643335
X600192.SH	3.015273139	0.816781898	0.068578975	1.4717175
X600566.SH	2.111391959	-0.595982753	2.645967717	1.4595523
X600200.SH	2.367371408	-0.684153712	1.985012504	1.3372062
X600539.SH	1.961869630	1.363841340	0.327673166	1.2922022
X600398.SH	1.741838574	-0.048766457	1.973064809	1.2740249

Figure 9: Top-10 Stocks for Portfolio

4.2 Portfolio Optimization

After constructing portfolio, we need to find a best method to making allocations between stocks in the portfolio. According to modern portfolio theory, the aim of allocate fund is to achieve more return conditioned on the same risk. Combining two stocks with positive and negative covariance can greatly reduce portfolio's volatility and risk. So we compute the covariance of 10 stocks' retf1 from

```
#优化
yyyymm=x_test$yyyymm[split+500]
Forecast_return<-array(dim=c(1,10))
stock=list("600381.SH","600446.SH", "600155.SH", "600228.SH", "600330.SH", "600192.SH", "600566.SH", "600200.SH", "600539.SH", "600398.SH")
for(j in 1:10)
{
  #yyyymm
  #stock[j]
  #x_test[x_test$yyyymm==yyyymm & x_test$Codes==stock[j],27]
  Forecast_return[1,j] = x_test[x_test$yyyymm==yyyymm & x_test$Codes==stock[j],27]
}
Forecast_return
print(Forecast_return)
alpha_target=-0.5

portfolio_weight=dataOptimize1(Covariance, Forecast_return, alpha_target)$solution
print(dataOptimize1(Covariance, Forecast_return, alpha_target))
```

Figure 10: Sectional Code for Optimization Process

beginning date(201001) to testing set's ending date. Together with the forecast ALPHA we computed in the loops, and set α to -0.5, which is the expected returns for securities, we can do optimization by formula8:

$$\min_{w^*} \left(W^T \Omega W\right) \quad s.t. \sum w_i = 1, w_i \geq 0, W^T \alpha \geq \alpha_0 \tag{8}$$

Where Ω is the variance defining risk levels, and W is a vector of portfolio weights. The w_i we obtain from solution is the best proportion to allocate funds according to our ALPHA strategy as shown in figure10.

5 Performance Evaluation

After the considerable iterative portfolio optimization, we now evaluate the performance of our portfolio by comparing the actual monthly returns of our portfolio with the actual monthly returns of the equal-weight portfolio11, for we want to figure out whether the monthly redistribution of stock weights can factually optimize the performance of our portfolio. Since our portfolio construct in 2015, and we assign stock weights monthly until 2021 according to the mean-variance portfolio optimization strategy, the following figure12 shows the comparing performance in this period, where the blue line indicates our portfolio and the yellow line indicates the equal-weight portfolio.

```
##与equal weight比较
Real_return<-array(dim=c(1,10))
yyyymm=x$yyyymm[split+501]
Real_return= retf1_select[retf1_select$yyyymm==yyyymm,2:11]
Real_return
sum(Real_return)
equal_weight_return=sum(Real_return)/10
our_weight_return=0
for(j in 1:10)
{
  our_weight_return=our_weight_return+portfolio_weight[j]*Real_return[j]
}
our_weight_return=as.numeric(our_weight_return)
print(equal_weight_return)
print(our_weight_return)
equalAndOurWeightReturnArray[i,1]=equal_weight_return
equalAndOurWeightReturnArray[i,2]=our_weight_return
if (equal_weight_return<our_weight_return)
{
  print("nice forcast!!")
  nicecount=nicecount+1
}
```

Figure 11: Sectional Code for Comparison Process

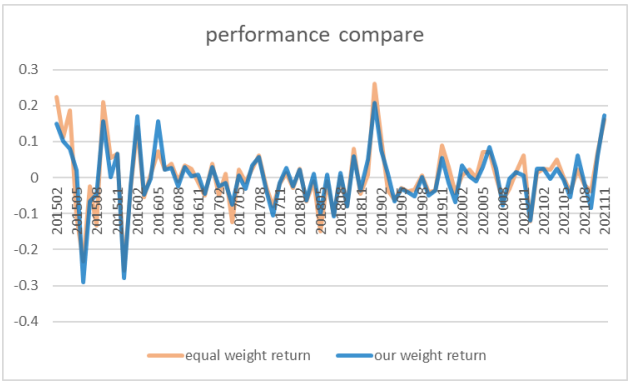


Figure 12: Portfolio Performance Evaluation with equal-weight

The figure illustrates that on average our portfolio has similar performance to the equal-weight portfolio, and the exceeding rate is 44.58% which means our portfolio's performance has exceeded the equal-weight portfolio 37 times in a total of 83 months. Also, we have the following description table1 of their returns sharing the parallel analysis. Since our portfolio has a smaller min value than the equal-weight portfolio, we have to say that our portfolio might not handle financial crises well according to the max drawdown. Then, we are surprised to find our median result is better than that of equal-weight, and the mean-variance portfolio management strategy can contribute to this thanks to the minimized risk conditioned on a target return. Also, we have a smaller max value and mean value, which are all results of the mean-variance portfolio.

In further work, we can fine-tune the parameters of the mean-variance portfolio management strategy to get better performance, like the selection number of stocks, alpha target, heteroscedasticity and stock redistribution frequency. Also, since retf1 used in this work shows the least relevance with alpha1, we can use retf9 or retf12 in portfolio management later for better performance of the mean-variance portfolio.

Table 1: Result Analysis of Portfolios

Retf1	Our Portfolio	Equal-weight Portfolio
Min	-0.2909	-0.2573
1st Qu.	-0.0382	-0.0375
Median	0.0040	0.0035
Mean	0.0000	0.0038
3rd Qu.	0.0289	0.0388
Max	0.2078	0.2600