«Бізнес аналітика»

Автор: Євген Пенцак

Домашнє завдання №12 (від 8.12.2020)

Виконав: Трохимович Микола

Заняття 19-20. Моделі оптимального управління портфелем акцій та розв'язування задач квадратичного програмування. Знаходження вартості під ризиком (VaR) інвестиційного портфеля.

Завдання 1 (20 балів). Розглянемо два цінні папери FB та TRIP з файлу ADJ_PRICES.xls. Знайдіть значення їх середніх доходностей та стандартних відхилень. Знайдіть залежність очікуваної доходності та стандартного відхилення портфеля цих акцій від вагових коефіцієнтів w_1 та $1-w_1$, відповідно. Зобразіть графічно у вигляді кривої описові характеристики портфеля в системі координат σ (вісь x) та μ (вісь y).

Розв`язок:

1. Знайдемо значення їх середніх доходностей та стандартних відхилень:

```
df = pd.read_excel('ADJ_PRICES_12.xlsx', sheet_name='DAY')

# Зчитуємо щоденну дохідність акцій відповідних компаній
rev_trip_fb = df[['RTRIP','RFB']]

# середніх доходностей
print('Середні доходності TRIP: ', round(rev_trip_fb['RTRIP'].mean()*100,2), '%')
print('Середні доходності FB: ', round(rev_trip_fb['RFB'].mean()*100,2), '%')

print()
# середніх відхилення
print('Стандартне відхилен TRIP: ', round(rev_trip_fb['RTRIP'].std()*100,2), '%')
print('Стандартне відхилен FB: ', round(rev_trip_fb['RFB'].std()*100,2), '%')
```

```
Середні доходності TRIP:  -0.01 %
Середні доходності FB:  0.14 %

Стандартне відхилен TRIP:  2.67 %
Стандартне відхилен FB:  2.01 %
```

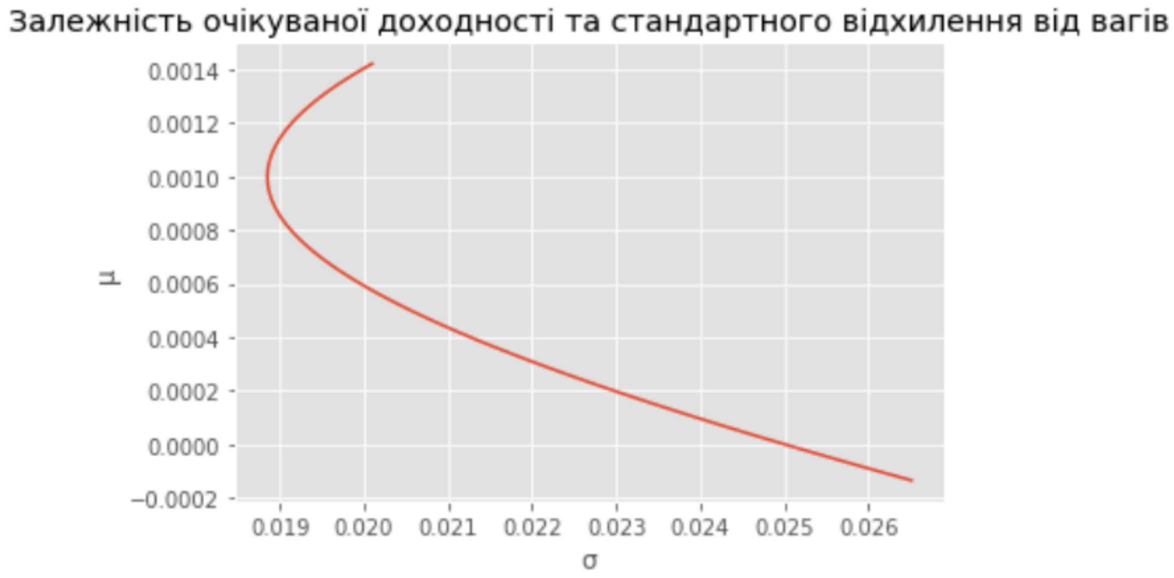
---

2. Зобразимо графічно залежність очікуваної доходності та стандартного відхилення портфеля цих акцій від вагів у вигляді кривої описові характеристики портфеля в системі координат σ (вісь x) та μ (вісь y).

Для обчислення стандартного відхилення портфеля використаємо дану формулу

$$\sigma_{portfolio} = \sqrt{w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + 2w_1w_2Cov_{1,2}}$$

```
import matplotlib.pyplot as plt
plt.style.use('ggplot')

ws = np.arange(0, 1, 0.01)

means = []
stds = []

cov_daily = rev_trip_fb.cov()

for w in ws:
    wec = np.array([w, 1-w])

    means.append(np.dot(wec, rev_trip_fb[:-1].T).mean())
    stds.append(np.sqrt(np.dot(wec.T, np.dot(cov_daily, wec))))

plt.plot(stds, means)
plt.xlabel('σ')
plt.ylabel('μ')
plt.title('Залежність очікуваної доходності та стандартного відхилення від вагів')
plt.show()
```



Завдання 2 (40 балів).

Розглянемо файл stock_data.xlsx з даними тижневими доходностями акцій компаній

AAPL GOOGL MSFT AMZN YHOO NFLX

та індексу S&P500.

2.1 **(20 балів)** Знайдіть копульну (Клейтона) залежність і відкалібруйте відповідну копулу, що характеризує взаємну поведінку акцій MSFT та AMZN. Використайте коефіцієнт Кендала та функцію copulaparam для знаходження відповідного параметра сім'ї Клейтона. Для параметричного оцінювання граничних розподілів використайте гама розподіл.

Згенеруйте двовимірну випадкову величину згідно до параметрів копули і параметрів оцінених граничних (гама) розподілів MSFT та AMZN.

Зчитаємо дані про дохідності:

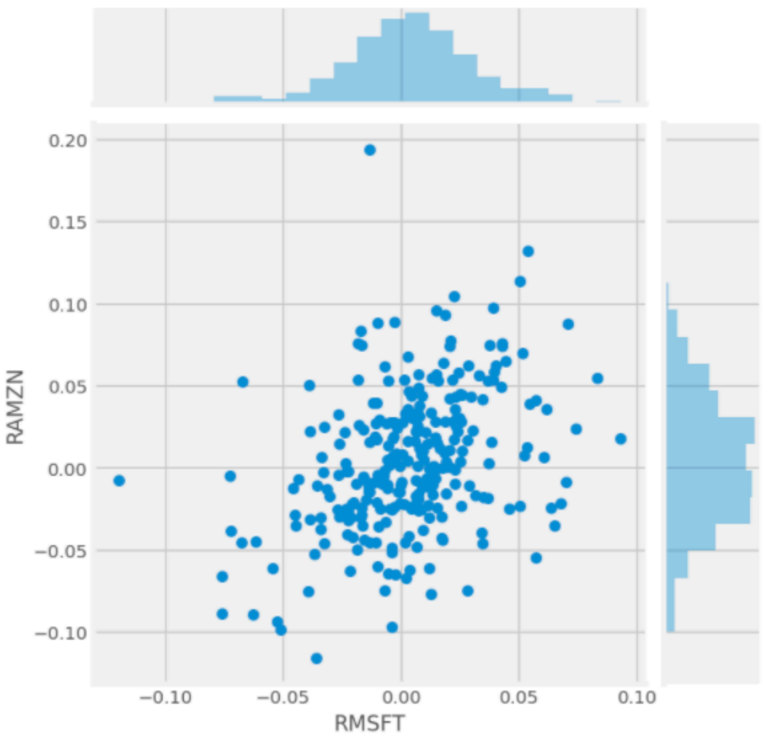
```
In [114]: table = pd.read_excel('stock_data_12.xlsx', sheet_name='page1')[['RAAPL', 'RGOOGL','RMSFT','RAMZN','RYHOO','RNFLX','RS
          table.head()
```

Out[114]:

	RAAPL	RGOOGL	RMSFT	RAMZN	RYHOO	RNFLX	RS&P500
0	0.030980	0.027951	0.028512	0.062200	0.002516	0.045089	0.031954
1	0.052111	0.037994	0.038971	0.053645	-0.000627	0.062120	0.022613
2	-0.022054	-0.003632	0.004254	-0.024893	-0.034526	0.010787	-0.001920
3	0.003327	0.017177	-0.013477	0.016042	-0.024707	-0.035351	0.000101
4	-0.036478	0.009064	0.025761	0.044330	0.012667	-0.028003	0.013275

Наші дані про тижневі дохідності акцій MSFT та AMZN розподілені так:

```
In [116]: import seaborn as sns
          sns.jointplot(data = table, x='RMSFT', y='RAMZN')

Out[116]: <seaborn.axisgrid.JointGrid at 0x7fb635c75fd0>
```



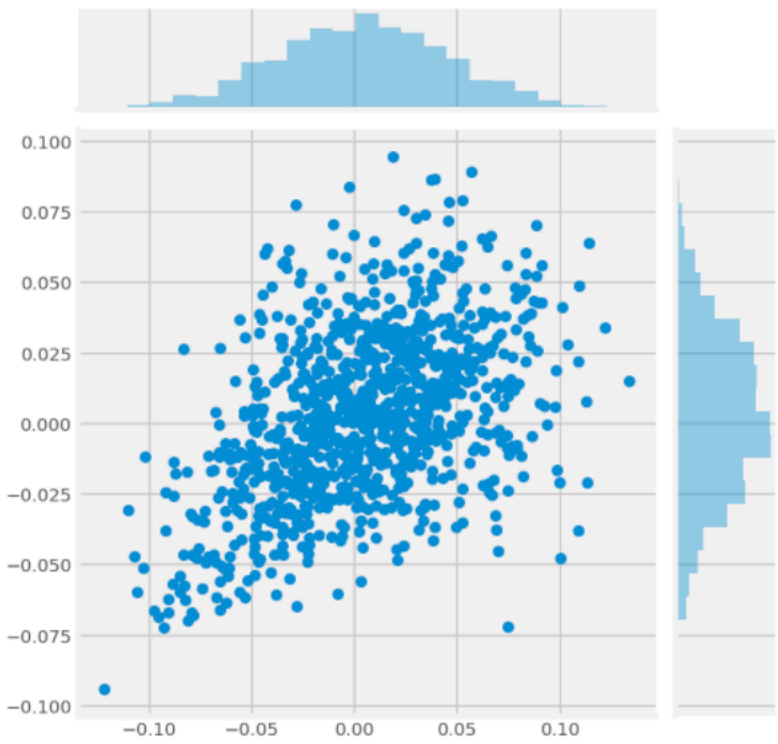
Далі генерувати дані будемо за допомогою Python та статистичного пакету scipy. Для копул використаємо пакет copulae:

```python
from scipy import stats
from copulae.archimedean import ClaytonCopula

# коефіцієнт Kendall's tau
k = stats.kendalltau(table['RMSFT'], table['RAMZN'])[0]
print('Koeff Kendall original data: ', k)

# Параметри для гама розподілів, що відповідають заданим вибіркам дохідностей
p_a = stats.gamma.fit(table['RAMZN'])
p_m = stats.gamma.fit(table['RMSFT'])

# Використовуємо копулу для знаходження параметра сім'ї Клейтона
copula = ClaytonCopula(dim = 2)
copula = copula.fit(table[['RMSFT', 'RAMZN']], method = 'itau')
random_copula_values = copula.random(1000)

# Зенеруйте двовимірну випадкову величину згідно до параметрів копули

generated_m = stats.gamma.ppf(random_copula_values[:,0], *p_m)
generated_a = stats.gamma.ppf(random_copula_values[:,1], *p_a)

# коефіцієнт Kendall's tau
k = stats.kendalltau(generated_m, generated_a)[0]
print('Koeff Kendall generated data: ', k)
```

```
Koeff Kendall original data:  0.27049767750497 68
Koeff Kendall generated data:  0.2538978978978979
```

Графік для згенерованих даних:

```
]: <seaborn.axisgrid.JointGrid at 0x7fb6391e2da0>
```



2.2 **(20 балів)** Знайдіть копульну (Гумбеля) залежність і відкалібруйте відповідну копулу, що характеризує взаємну поведінку акцій MSFT та AMZN. Використайте коефіцієнт Кендала та функцію copulaparam для знаходження відповідного параметра сім'ї Клейтона. Для параметричного оцінювання використайте гама розподіл.

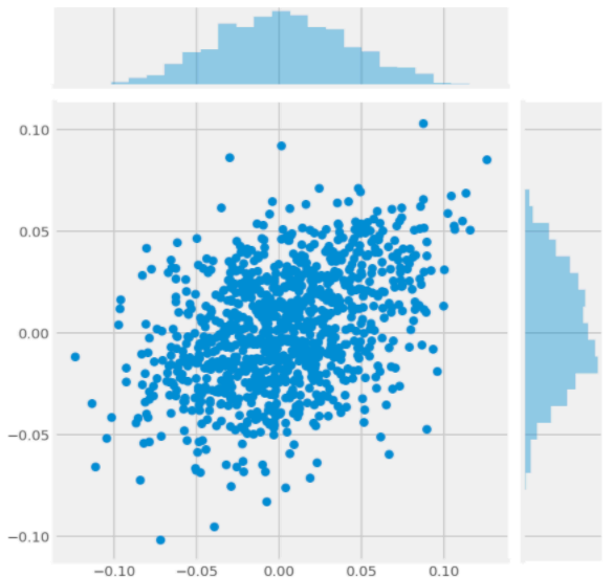
Згенеруйте двовимірну випадкову величину згідно до параметрів копули і параметрів оцінених граничних (гама) розподілів MSFT та AMZN.

```
from scipy import stats
from copulae.archimedean import GumbelCopula

# коефіцієнт Kendall's tau
k = stats.kendalltau(table['RMSFT'], table['RAMZN'])[0]
print('Koeff Kendall original data: ', k)

# Параметри для гама розподілів, що відповідають заданим вибіркам дохідностей
p_a = stats.gamma.fit(table['RAMZN'])
p_m = stats.gamma.fit(table['RMSFT'])

# Використовуємо копулу для знаходження параметра сім'ї Клейтона
copula = GumbelCopula(dim = 2)
copula = copula.fit(table[['RMSFT', 'RAMZN']], method = 'itau')
random_copula_values = copula.random(1000)

# Зенеруйте двовимірну випадкову величину згідно до параметрів копули

generated_m = stats.gamma.ppf(random_copula_values[:,0], *p_m)
generated_a = stats.gamma.ppf(random_copula_values[:,1], *p_a)

# коефіцієнт Kendall's tau
k = stats.kendalltau(generated_m, generated_a)[0]
print('Koeff Kendall generated data: ', k)
```

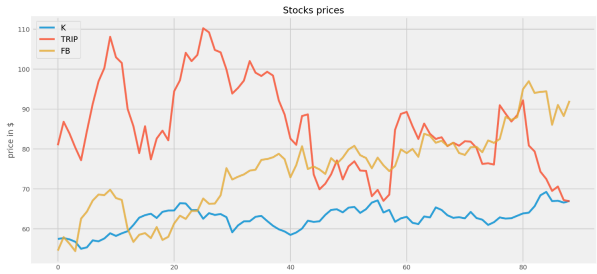
```
Koeff Kendall original data:  0.2704976775049768
Koeff Kendall generated data:  0.2638518518518519
```

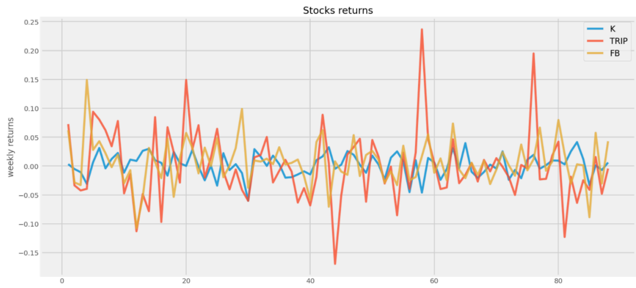
```
sns.jointplot(generated_a, generated_m)
```

```
<seaborn.axisgrid.JointGrid at 0x7fb63ab517f0>
```



Завдання 3 (20 балів). Оптимізуйте портфель з акцій K, FB, TRIP, використовуючи аналітичну модель теорії Марковиця і тижневі дані з файлу ADJ_PRICES.xlsx.





Як можемо бачити актив TRIP має стрибки стрибки як в позитивну так і в негативну сторону, що вказує на його високу волатильність.

За початкового розділення, тобто якщо кожному активу буде рівна частка 1/3 маємо таку

```
In [88]: # Сума вагів має бути 1 тому всі ініціалізуємо однаковими вагами
         weights = np.array([1/3,1/3,1/3])
         weights

Out[88]: array([0.33333333, 0.33333333, 0.33333333])

In [89]: # корегуємо, так як у нас тижневі дані
         cov_matrix_annual = returns.cov() * 52
         cov_matrix_annual

Out[89]:
```

	K	TRIP	FB
K	0.020277	0.005514	-0.000148
TRIP	0.005514	0.197669	0.042851
FB	-0.000148	0.042851	0.080701

```
In [90]: # Expected portfolio variance= WT * (Covariance Matrix) * W
         port_variance = np.dot(weights.T, np.dot(cov_matrix_annual, weights))
         port_variance

Out[90]: 0.043897914185429585

In [91]: # Expected portfolio volatility= SQRT (WT * (Covariance Matrix) * W)
         port_volatility = np.sqrt(port_variance)
         port_volatility

Out[91]: 0.2095182908135459

In [92]: portfolioSimpleAnnualReturn = np.sum(returns.mean()*weights) * 52
         portfolioSimpleAnnualReturn

Out[92]: 0.14435200366736117

In [93]: percent_var = str(round(port_variance, 4) * 100) + '%'
         percent_vols = str(round(port_volatility, 4) * 100) + '%'
         percent_ret = str(round(portfolioSimpleAnnualReturn*100, 3))+'%'
         print("Expected annual return : "+ percent_ret)
         print('Annual volatility/standard deviation/risk : '+percent_vols)
         print('Annual variance : '+percent_var)

         Expected annual return : 14.435%
         Annual volatility/standard deviation/risk : 20.95%
         Annual variance : 4.390000000000001%
```

картину:

Далі знаходимо оптимальний портфель: Для цього використаємо готовий програмний пакет в Python, що дозволяє проводити такі обчислення.

In [99]:
```python
# Using https://randerson112358.medium.com/python-for-finance-portfolio-optimization-66882498847
# !pip install PyPortfolioOpt

from pypfopt.efficient_frontier import EfficientFrontier
from pypfopt import risk_models
from pypfopt import expected_returns

mu = returns.mean() * 52
S = cov_matrix_annual

ef = EfficientFrontier(mu, S)
weights = ef.max_sharpe() #Maximize the Sharpe ratio, and get the raw weights
cleaned_weights = ef.clean_weights()
print(cleaned_weights) #Note the weights may have some rounding error, meaning they may not add up exactly to 1 but sh
ef.portfolio_performance(verbose=True)
```

```
OrderedDict([('K', 0.49445), ('TRIP', 0.0), ('FB', 0.50555)])
Expected annual return: 22.6%
Annual volatility: 16.0%
Sharpe Ratio: 1.29
```

Out[99]:
```
(0.22634195432648238, 0.15971511946779973, 1.2919375135807547)
```

Отже ми бачимо, що ми опитмізуємо портфоліо маючи 49.45% K та 50.55% FB. Також бачимо, що очікуваний річний прибуток при такій конфігурації зростає на 22.6% при цьому ризики становлять 16%. Це портфоліо має Sharpe ratio 1.29, що є добре. До речі у цьому прикладі Виколистав саме оптимізацію по Sharpe ratio, так як вона дає як на мене кращий результат ніж кватратична. У випадку квадратичної оптимізації ми все вкладаємо в один актив, при цьому так, маємо більший очікуваний прибуток, проте і ризики значно вищі, та і Sharpe ratio є очікувано менше.

```python
# Using https://randerson112358.medium.com/python-for-finance-portfolio-optimization-66882498847
# !pip install PyPortfolioOpt

from pypfopt.efficient_frontier import EfficientFrontier
from pypfopt import risk_models
from pypfopt import expected_returns

mu = returns.mean() * 52
S = cov_matrix_annual

ef = EfficientFrontier(mu, S)
weights = ef.max_quadratic_utility() # Maximises the quadratic utility, given some risk aversion.
cleaned_weights = ef.clean_weights()
print(cleaned_weights) #Note the weights may have some rounding error, meaning they may not add up exactly to 1 but sh
ef.portfolio_performance(verbose=True)
```

```
OrderedDict([('K', 0.0), ('TRIP', 0.0), ('FB', 1.0)])
Expected annual return: 34.9%
Annual volatility: 28.4%
Sharpe Ratio: 1.16

(0.34942692541881026, 0.2840791719620477, 1.1596306872607363)
```

Завдання 4 (20 балів). Оптимізуйте портфель акцій K, FB, TRIP, використовуючи чисельну модель оптимізації Марковиця і тижневі дані з файлу ADJ_PRICES.xlsx. У моделі використайте обмеження: додатні вагові коефіцієнти і у кожну акцію не можна інвестувати більше 40% вартості портфеля і не менше 25%. Візьміть коефіцієнт несприйняття ризику інвестором a=7.

Аналогічно з попереднім завданням розвяжемо цю задачу додавши необхідні параметри в модель:

```
# Using https://randerson112358.medium.com/python-for-finance-portfolio-optimization-66882498847
# !pip install PyPortfolioOpt

from pypfopt.efficient_frontier import EfficientFrontier
from pypfopt import risk_models
from pypfopt import expected_returns

mu = returns.mean() * 52
S = cov_matrix_annual

ef = EfficientFrontier(mu, S, weight_bounds=(0.25,0.4))
weights = ef.max_quadratic_utility(risk_aversion = 7) # Maximises the quadratic utility, given some risk aversion.
cleaned_weights = ef.clean_weights()
print(cleaned_weights) #Note the weights may have some rounding error, meaning they may not add up exactly to 1 but sh
ef.portfolio_performance(verbose=True)
```

```
OrderedDict([('K', 0.35), ('TRIP', 0.25), ('FB', 0.4)])
Expected annual return: 17.1%
Annual volatility: 19.3%
Sharpe Ratio: 0.78

(0.17072746162664987, 0.19298741218733942, 0.7810222434628722)
```

Як бачимо, з заданими обмеженнями маємо очікувано гірший результат. Волатильність підвищилася при цьому очікуваний дохід зменшився і став навіть менший від волатильності. Тобто при такій конфігурації навіть є деяка ймовірність піти в збиток. Такий портфель є не дуже хорошим.