

Prova Final

September 26, 2022

1 Prova Final

1.1 Portfólio com Ações Do Mercado De Criptoativos

```
[35]: import copy
      from datetime import date

      import yfinance
      import numpy as np
      import pandas as pd
      import plotly.express as px
      import matplotlib.pyplot as plt

      from pypfopt import plotting
      from pypfopt import risk_models
      from pypfopt import expected_returns
      from pypfopt.risk_models import CovarianceShrinkage
      from pypfopt.efficient_frontier import EfficientFrontier
```

Utilizou-se a base de dados do yahoo finance para buscar criptoativos, isto é, ativos do mercado de criptomoedas. O Yahoo Finance, no entanto, não conta com dados pareados dos ativos em reais, assim, foi utilizada a moeda fiduciária *Dólar Americano*. Como índice para a bolsa, por se tratar de um mercado não regulado e altamente volátil, como indicador do mercado foi utilizado o ativo *USDC* que é uma *stablecoin*.

Stablecoins, também chamadas de moedas estáveis, são criptomoedas pareadas em algum ativo estável ou cesta de ativos, de modo a controlar a volatilidade. Neste caso o ativo *USDC* tem seu valor pareado ao *Dólar Americano*.

O período analisado diz respeito ao início de 2019 até os dias atuais. A data de início escolhida tem ligação com a criação do ativo *USDC* que começou a ser negociado no final de 2018.

```
[2]: acoes = ["BTC-USD", "ADA-USD", "LTC-USD", "ETH-USD", "USDC-USD"]
      data_inicio = "2019-01-01"
      data_fim = date.today().strftime("%Y-%m-%d")

      acoes_df = yfinance.download(acoes, data_inicio, data_fim)['Close']
```

```
[*****100%*****] 5 of 5 completed
```

```
[3]: acoes_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1364 entries, 2019-01-01 to 2022-09-25
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  -
 0   ADA-USD     1364 non-null   float64
 1   BTC-USD     1364 non-null   float64
 2   ETH-USD     1364 non-null   float64
 3   LTC-USD     1364 non-null   float64
 4   USDC-USD    1364 non-null   float64
dtypes: float64(5)
memory usage: 63.9 KB
```

1.1.1 Preço Individual dos Ativos

```
[4]: preco_individual = px.line(acoes_df, title="Preço Individual dos Ativos")
preco_individual.show()
```

Como visto acima, o Bitcoin parece dominar a escala do gráfico, pois o preço absoluto da ação é muito alto. Os gráficos de todas as outras ações são achatados. Um gráfico como este não é muito útil para comparar o desempenho relativo das ações.

1.2 Otimização do Portfólio

1.2.1 Otimizando o Índice de Sharpe

A fronteira eficiente é o conjunto de carteiras ótimas que oferecem o maior retorno esperado para um nível definido de risco (volatilidade) ou o menor risco (volatilidade) para um determinado nível de retorno esperado. É representado por uma linha no gráfico Retorno vs Volatilidade. A carteira do índice max Sharpe encontra-se na fronteira eficiente.

Para representar tudo visualmente, o código abaixo gera 10000 portfólios de nossas ações com pesos aleatórios e plota seus retornos e volatilidade. A fronteira eficiente e a carteira de razão máxima de Sharpe também são plotadas no gráfico.

```
[55]: # calculate the mean and variance
# mu = expected_returns.ema_historical_return(acoes_df, frequency=365)
mu = expected_returns.mean_historical_return(acoes_df, compounding=True,
↪frequency=1363)
sigma = risk_models.exp_cov(acoes_df, frequency=1363)
```

```
[56]: sigma
```

```
[56]:          ADA-USD  BTC-USD  ETH-USD  LTC-USD  USDC-USD
ADA-USD  3.032158  1.740664  2.420416  2.254734 -0.000063
BTC-USD  1.740664  1.659040  1.993108  1.666970  0.000013
ETH-USD  2.420416  1.993108  3.263691  2.424511 -0.000539
```

```
LTC-USD    2.254734  1.666970  2.424511  2.694087  0.000677
USDC-USD   -0.000063  0.000013 -0.000539  0.000677  0.000123
```

```
[57]: # get the efficient frontier
      ef = EfficientFrontier(mu, sigma)
```

```
[8]: sharpe_weights = ef.max_sharpe()
      ef.clean_weights()
```

```
[8]: OrderedDict([('ADA-USD', 0.85667),
                  ('BTC-USD', 0.0),
                  ('ETH-USD', 0.14333),
                  ('LTC-USD', 0.0),
                  ('USDC-USD', 0.0)])
```

```
[9]: ef.portfolio_performance(verbose=True)
```

```
Expected annual return: 930.1%
Annual volatility: 169.9%
Sharpe Ratio: 5.46
```

```
[9]: (9.301140742493702, 1.6990252540964919, 5.462626715003851)
```

O portfólio ideal que maximiza o Sharpe Ratio é investir em ADA (85%) e em ETH (15%).

Agora, desejamos uma carteira com os ativos de menor volatilidade.

```
[58]: pesos = ef.min_volatility()
```

```
[59]: pesos
```

```
[59]: OrderedDict([('ADA-USD', 0.0),
                  ('BTC-USD', 0.0),
                  ('ETH-USD', 0.0002025751397655),
                  ('LTC-USD', 0.0),
                  ('USDC-USD', 0.9997974248602344)])
```

Para um portfólio com a menor volatilidade devemos investir 99.99% em USDC e 0.001% em ETH.

```
[60]: ef.portfolio_performance(verbose=True)
```

```
Expected annual return: -1.1%
Annual volatility: 1.1%
Sharpe Ratio: -2.85
```

```
[60]: (-0.01149612756940949, 0.011068364126435453, -2.845599151746805)
```

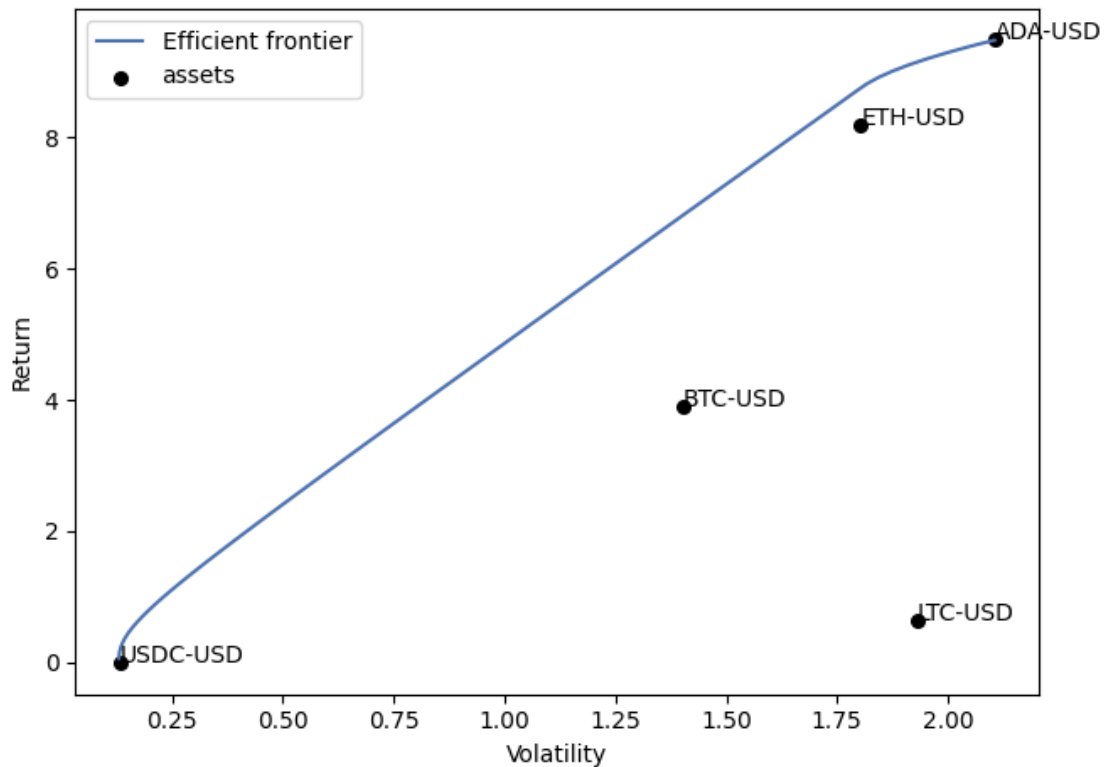
1.2.2 Plot da Fronteira Eficiente

```
[52]: mu_frontier = expected_returns.mean_historical_return(acoes_df,
    ↪compounding=True, frequency=1363)
sigma_frontier = risk_models.sample_cov(acoes_df, frequency=1363)

[53]: ef_frontier = EfficientFrontier(mu_frontier, sigma_frontier)

[54]: fig, ax = plt.subplots()
plotting.plot_efficient_frontier(ef_frontier, ax=ax, show_assets=True)

for i, asset in enumerate(ef_frontier.tickers):
    ax.annotate(asset, ((np.diag(ef_frontier.cov_matrix) ** (1/2))[i],
    ↪ef_frontier.expected_returns[i]))
```



Com o gráfico da fronteira eficiente é possível inferir que o ativo *ADA-USD* apresenta o maior retorno dentro do período medido em contrapartida o ativo *LTC-USD* apresenta quase o mesmo risco, no entanto, com um retorno muito inferior.

O ativo *USDC-USD* apresenta a menor volatilidade e também o menor retorno no período observado. Os resultados são esperados uma vez que o ativo é uma *stablecoin* e tem seu valor lastreado no dólar americano.

1.3 Com Portfólios Aleatórios

```
[41]: ef_frontier = EfficientFrontier(mu_frontier, sigma_frontier,
    ↪weight_bounds=(None, None))
ef_frontier.add_constraint(lambda w: w[0] >= 0.2)
ef_frontier.add_constraint(lambda w: w[2] == 0.15)
ef_frontier.add_constraint(lambda w: w[3] + w[4] <= 0.10)

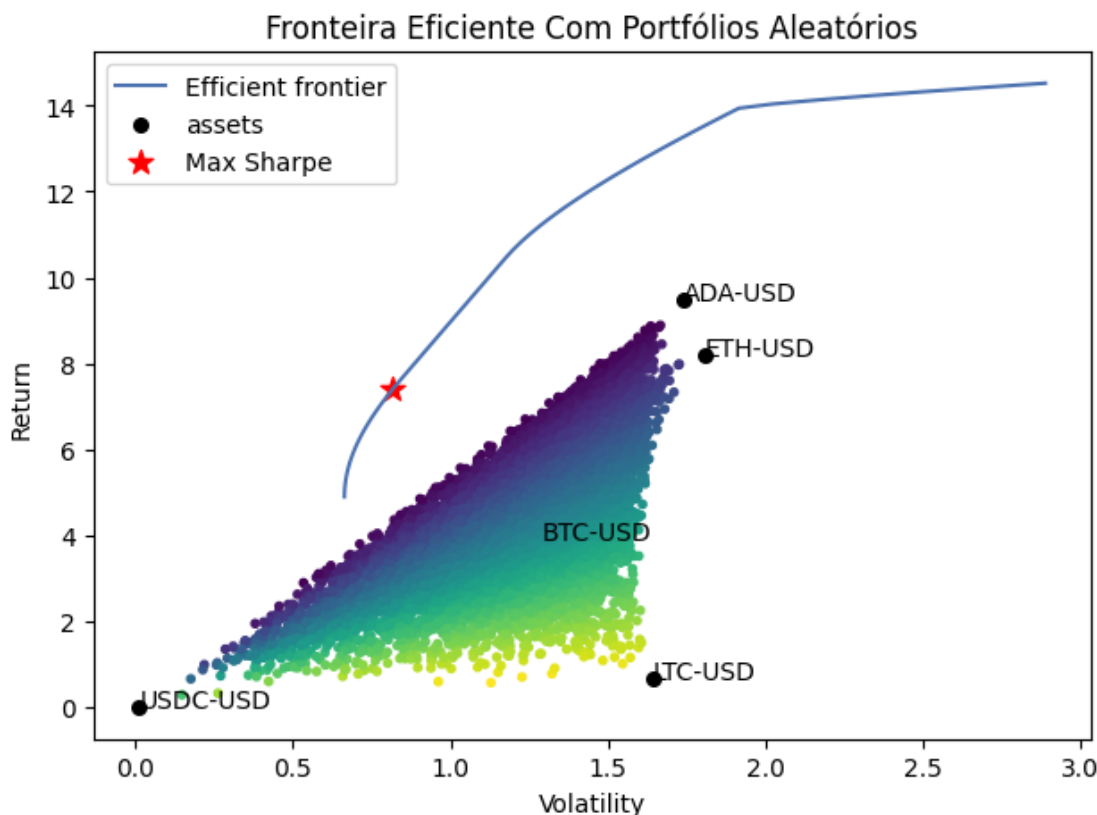
[42]: fig, ax = plt.subplots()
ef_max_sharpe = copy.deepcopy(ef_frontier)
plotting.plot_efficient_frontier(ef_frontier, ax=ax, show_assets=True)

for i, asset in enumerate(ef_frontier.tickers):
    ax.annotate(asset, ((np.diag(ef_frontier.cov_matrix) ** (1/2))[i],
    ↪ef_frontier.expected_returns[i]))

# Find the tangency portfolio
ef_max_sharpe.max_sharpe()
ret_tangent, std_tangent, _ = ef_max_sharpe.portfolio_performance()
ax.scatter(std_tangent, ret_tangent, marker="*", s=100, c="r", label="Max_
    ↪Sharpe")

# Generate random portfolios
n_samples = 10_000
w = np.random.dirichlet(np.ones(ef.n_assets), n_samples)
rets = w.dot(ef.expected_returns)
stds = np.sqrt(np.diag(w @ ef.cov_matrix @ w.T))
sharpes = rets / stds
ax.scatter(stds, rets, marker=".", c=sharpes, cmap="viridis_r")

# Output
ax.set_title("Fronteira Eficiente Com Portfólios Aleatórios")
ax.legend()
plt.tight_layout()
plt.show()
```



1.4 Análise Descritiva dos Dados

1.4.1 Retornos Diários

O retorno diário de uma ação é o ganho fracionário (ou perda) em um determinado dia em relação ao dia anterior, é dado por $(\text{preço de fechamento do dia atual} - \text{preço de fechamento do dia anterior}) / (\text{preço de fechamento do dia anterior})$. Por ser um valor relativo, proporciona uma comparação mais justa entre os retornos das ações, independentemente dos preços absolutos das ações. O método `pct_change()` pode ser usado para obter os retornos diários de forma eficiente.

```
[13]: retornos = acoes_df.pct_change()
      retornos.head()
```

```
[13]:
```

	ADA-USD	BTC-USD	ETH-USD	LTC-USD	USDC-USD
Date					
2019-01-01	NaN	NaN	NaN	NaN	NaN
2019-01-02	0.063718	0.025989	0.101039	0.045458	0.004808
2019-01-03	-0.056918	-0.027050	-0.038135	-0.042083	-0.004514
2019-01-04	0.026475	0.005467	0.036523	0.011786	-0.005344
2019-01-05	0.020291	-0.003246	0.006836	0.078160	0.002827

Agora, traçando os retornos diários das ações ADA-USD, BTC-USD, ETH-USD, USDC-USD,

durante todo o período medido. Notavelmente, as flutuações são muito maiores durante um período de alta volatilidade (ou seja, durante o crash do Covid em março de 2020).

```
[21]: fig = px.line(retornos[["BTC-USD", "ADA-USD", "LTC-USD", "ETH-USD",  
    ↪ "USDC-USD"]], title='Retornos Diários')  
fig.show()
```

1.4.2 Média dos Retornos Diários

```
[15]: retornos.mean()
```

```
[15]: ADA-USD      0.003344  
BTC-USD      0.001901  
ETH-USD      0.002850  
LTC-USD      0.001754  
USDC-USD     -0.000003  
dtype: float64
```

1.4.3 Desvio Padrão dos Retornos Diários

```
[16]: retornos.std()
```

```
[16]: ADA-USD      0.057045  
BTC-USD      0.038006  
ETH-USD      0.048861  
LTC-USD      0.052284  
USDC-USD      0.003586  
dtype: float64
```

1.4.4 Matriz de Covariância

```
[17]: retornos.cov()
```

```
[17]:          ADA-USD  BTC-USD  ETH-USD  LTC-USD  USDC-USD  
ADA-USD  0.003254  0.001447  0.002050  0.002148 -0.000012  
BTC-USD  0.001447  0.001444  0.001514  0.001564 -0.000009  
ETH-USD  0.002050  0.001514  0.002387  0.002094 -0.000012  
LTC-USD  0.002148  0.001564  0.002094  0.002734 -0.000008  
USDC-USD -0.000012 -0.000009 -0.000012 -0.000008  0.000013
```

1.4.5 Mapa de Calor a Partir da Matriz de Correlação dos Ativos

A matriz de correlação nos dá os coeficientes de correlação entre cada par de ações. Os coeficientes de correlação são indicadores da força da relação linear entre duas variáveis diferentes. É um valor de 0 a 1, com 1 indicando a relação mais forte. Se for um valor negativo, então as duas variáveis estão inversamente relacionadas.

```
[18]: def most_correlated(dataframe):
        """
        Returns a dataframe that contains the most correlated features

        dataframe: dataframe that gives the column names and their correlation value
        """
        corr_values = dataframe.abs().unstack()
        sorted_values = pd.DataFrame(corr_values.sort_values(kind="quicksort"),
        ↪ index= None)
        sorted_values = sorted_values[(sorted_values[0] > 0.6) & (sorted_values[0]
        ↪ < 1)]

        return sorted_values.drop_duplicates()
```

```
[19]: corr_df = acoes_df.corr().round(2) # round to 2 decimal places
fig_corr = px.imshow(corr_df, title = 'Correlação Entre Criptoativos')
fig_corr.show()
```

```
[20]: most_correlated(corr_df)
```

```
[20]:
          0
LTC-USD ETH-USD  0.75
ADA-USD LTC-USD  0.79
BTC-USD LTC-USD  0.86
ADA-USD BTC-USD  0.89
ETH-USD ADA-USD  0.91
BTC-USD ETH-USD  0.92
```

1.5 ANOVA e Testes de Hipóteses

```
[ ]:
```

```
[ ]:
```