Prova Final

September 26, 2022

1 Prova Final

1.1 Portfólio com Ações Do Mercado De Criptoativos

[88]:
```
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

import scipy.stats as stats
import statsmodels.api as sm
```

Utilizou-se a base de dados do yahoo finance bara 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 *Dolár 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 *Dolár Americano*.

O período analisado diz respeito ao ínicio 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.

[12]:
```
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
```

[41]:
```
acoes_df.head()
```

[41]:

Date	ADA-USD	BTC-USD	ETH-USD	LTC-USD	USDC-USD
2019-01-01	0.042547	3843.520020	140.819412	31.979931	1.013301
2019-01-02	0.045258	3943.409424	155.047684	33.433681	1.018173
2019-01-03	0.042682	3836.741211	149.135010	32.026699	1.013577
2019-01-04	0.043812	3857.717529	154.581940	32.404167	1.008160
2019-01-05	0.044701	3845.194580	155.638596	34.936867	1.011010

[15]:
```
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

[16]:
```
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.

[17]:
```
# 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)
```

[18]:
```
sigma
```

[18]:

	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

[19]:
```
# get the efficient frontier
ef = EfficientFrontier(mu, sigma)
```

[20]:
```
sharpe_weights = ef.max_sharpe()
ef.clean_weights()
```

[20]:
```
OrderedDict([('ADA-USD', 0.85667),
             ('BTC-USD', 0.0),
             ('ETH-USD', 0.14333),
             ('LTC-USD', 0.0),
             ('USDC-USD', 0.0)])
```

[21]:
```
ef.portfolio_performance(verbose=True)
```

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

[21]:
```
(9.301140742493702, 1.6990252540964919, 5.462626715003851)
```

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

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

[23]:
```
mu_min_v = expected_returns.mean_historical_return(acoes_df, compounding=True,␣
 ↪frequency=1363)
sigma_min_v = risk_models.exp_cov(acoes_df, frequency=1363)
ef_min_v = EfficientFrontier(mu, sigma)
```

[24]:
```
pesos = ef_min_v.min_volatility()
```

[25]:
```
pesos
```

[25]:
```
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.

[26]:
```
ef_min_v.portfolio_performance(verbose=True)
```

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

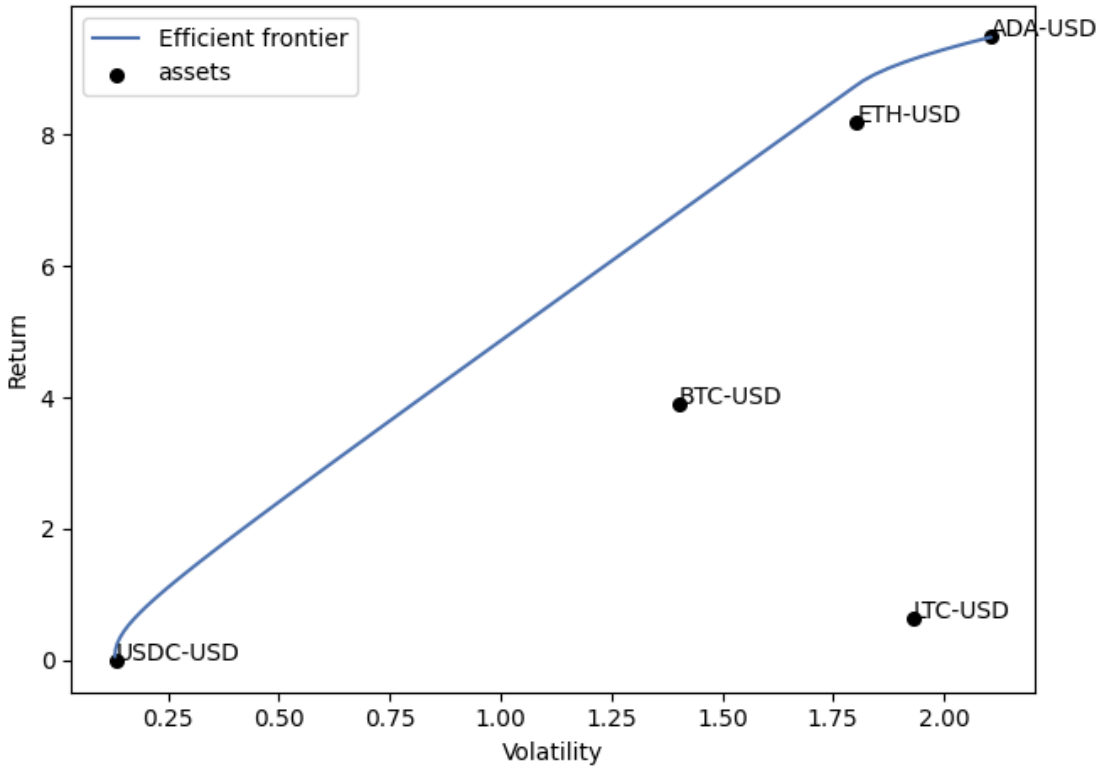
[26]:
```
(-0.011496127569409 49, 0.011068364126435453, -2.845599151746805)
```

1.2.3 Plot da Fronteira Eficiente

[27]:
```
mu_frontier = expected_returns.mean_historical_return(acoes_df,␣
 ↪compounding=True, frequency=1363)
sigma_frontier = risk_models.sample_cov(acoes_df, frequency=1363)
```

[28]:
```
ef_frontier = EfficientFrontier(mu_frontier, sigma_frontier)
```

[29]:
```
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.

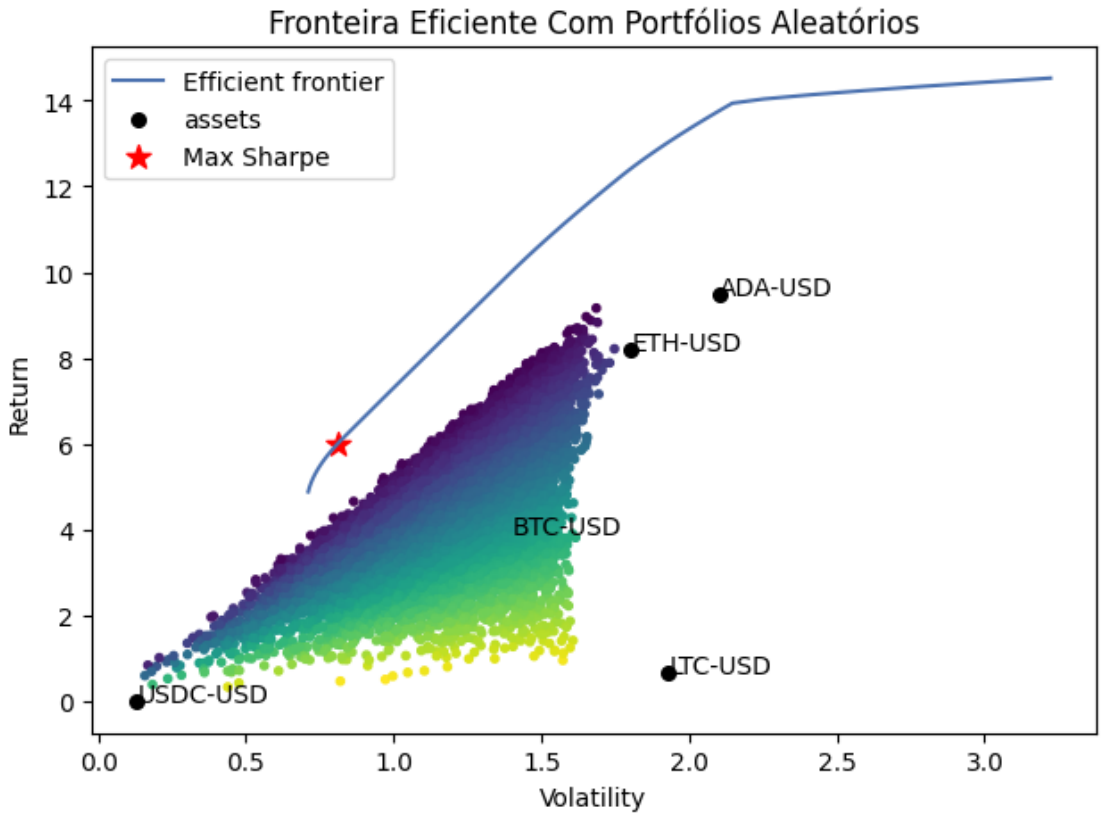
O ativo *USDC-USD* apresenta a menor volatilidade e também o menor retorno no período observado. Os resultados são esperados uma vez que o ativo é uma *stablecoin* e tem seu valor lastreado no dólar americano.

1.3 Com Portfólios Aleatórios

[30]:
```
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)
```

[31]:
```
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 (preço de fechamento do dia atual - preço de fechamento do dia anterior) / (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.

[32]:
```
retornos = acoes_df.pct_change()
retornos.head()
```

[32]:

Date	ADA-USD	BTC-USD	ETH-USD	LTC-USD	USDC-USD
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).

[33]:
```
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

[34]:
```
retornos.mean()
```

[34]:
```
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

[35]:
```
retornos.std()
```

[35]:
```
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

[36]:
```
retornos.cov()
```

[36]:
```
            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.

[37]:
```
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()
```

[38]:
```
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()
```

[39]:
```
most_correlated(corr_df)
```

[39]:
```
                     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

Para o teste de ANOVA é necessário que algumas suposições se provem verdadeiras:

- Normalidade: Cada amostra é de uma população normalmente distribuída.
- Indepedência: As amostras são independentes.
- Homocedasticidade: Os desvios padrão da população dos grupos são todos iguais.

Serão coletados dados dos ativos ao longo de um ano.

[77]:
```
acoes = ["BTC-USD", "ADA-USD", "LTC-USD", "ETH-USD", "USDC-USD"]
data_inicio = "2019-01-01"
data_fim = "2019-12-31"

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

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

[78]:
```
cripto_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 365 entries, 2019-01-01 to 2019-12-31
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   ADA-USD   365 non-null    float64
 1   BTC-USD   365 non-null    float64
 2   ETH-USD   365 non-null    float64
 3   LTC-USD   365 non-null    float64
 4   USDC-USD  365 non-null    float64
dtypes: float64(5)
memory usage: 17.1 KB
```

1.5.1 Teste de Shapiro-Wilk - Normalidade

O teste de Shapiro-Wilk é um teste de normalidade. É usado para determinar se uma amostra vem ou não de uma distribuição normal.

Para realizar um teste Shapiro-Wilk em Python podemos usar a função scipy.stats.shapiro()

[79]:
```
shapiro_df = pd.DataFrame(columns=[k for k in cripto_df.keys()],␣
 ↪index=["estatistica", "p_valor"])

for coluna in shapiro_df:
    shapiro_df[coluna] = stats.shapiro(cripto_df[coluna])
```

```
shapiro_df
```

[79]:

	ADA-USD	BTC-USD	ETH-USD	LTC-USD	\
estatistica	8.611770e-01	9.289624e-01	9.320979e-01	9.371803e-01	
p_valor	1.611809e-17	3.673350e-12	7.715647e-12	2.700180e-11	

	USDC-USD
estatistica	9.196935e-01
p_valor	4.621984e-13

1.5.2 Teste de Levene - Homocedasticidade

[83]:
```
levene_df = pd.DataFrame(columns=[k for k in cripto_df.keys() if k !=␣
 ↪"USDC-USD"], index=["estatistica_F", "p_valor"])

for coluna in levene_df:
    levene_df[coluna] = stats.levene(acoes_df["USDC-USD"], cripto_df[coluna])

levene_df
```

[83]:

	ADA-USD	BTC-USD	ETH-USD	LTC-USD
estatistica_F	6.681113e+02	3177.897543	1.806899e+03	1.900131e+03
p_valor	8.203753e-125	0.000000	8.187182e-271	1.388372e-280

Como o valor de p é menor que 0.05 nos testes de Shapiro-Wilk e Levene, rejeitamos a hipótese nula. Temos evidências suficientes para dizer que os dados da amostra não vêm de uma distribuição normal.

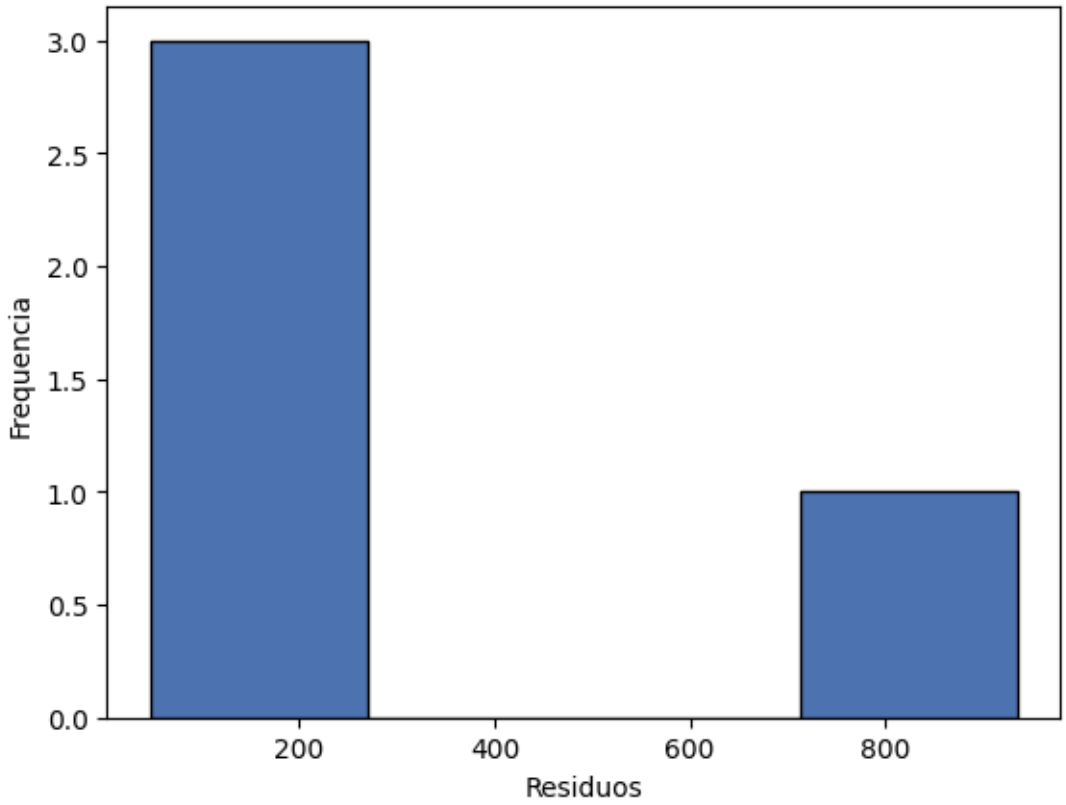
1.5.3 ANOVA

[99]:
```
anova_df = pd.DataFrame(columns=[k for k in cripto_df.keys() if k !=␣
 ↪"USDC-USD"], index=["f_valor", "p_valor"])

for coluna in anova_df:
    anova_df[coluna] = stats.f_oneway(cripto_df["USDC-USD"], cripto_df[coluna])

anova_df
```

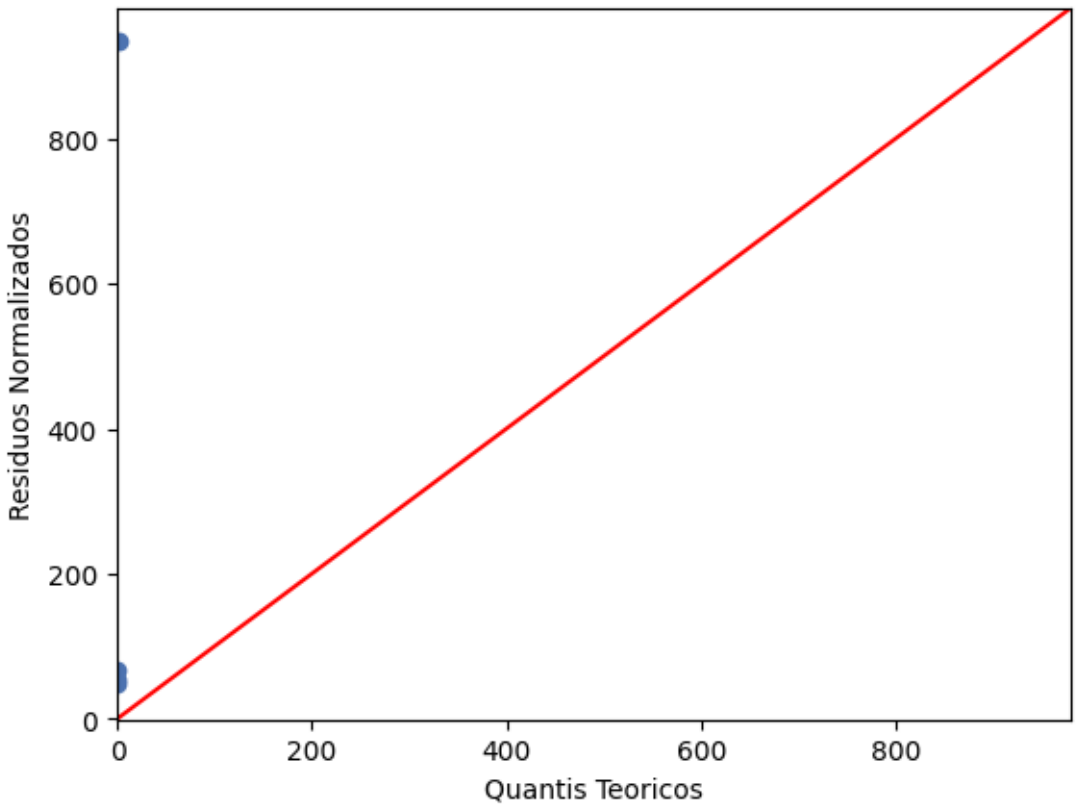
[99]:

	ADA-USD	BTC-USD	ETH-USD	LTC-USD
f_valor	873502.484407	2.866299e+03	4.698495e+03	2.271495e+03
p_valor	0.000000	1.235932e-254	8.967291e-320	5.025830e-226

[96]:
```
# histograma
plt.hist(np.sqrt(anova_df.iloc[0]), bins="auto", histtype="bar", ec="k")
plt.xlabel("Residuos")
plt.ylabel("Frequencia")
plt.show()
```



[97]:
```
sm.qqplot(np.sqrt(anova_df.iloc[0]), line="45")
plt.xlabel("Quantis Teoricos")
plt.ylabel("Residuos Normalizados")
plt.show()
```


Residuos Normalizados
0
200
400
600
800
Quantis Teoricos
0
200
400
600
800

[ ]: