**UNIVERSIDADE FEDERAL DE SANTA CATARINA**
**DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA**

Arthur Hortmann Erpen
Lucas João Martins

# Criptomoedas: Uma Visão Geral Sobre Bitcoin E Análise Dos Principais Altcoins

Relatório final

Florianópolis
2017

## Bitcoin

O comércio na internet é baseado em instituições financeiras, que processam pagamentos eletrônicos e garantem a confiança entre vendedores e compradores. Nesse sistema, não são possíveis transações completamente não-reversíveis. Também aceita-se que uma porcentagem das transações sejam fraudes inevitáveis.

Bitcoin é uma criptomoeda criada para possibilitar que duas partes realizem uma transação diretamente, sem necessidade de uma terceira parte que garanta a confiança. Foi criada por uma entidade desconhecida sob o nome Satoshi Nakamoto e lançada em 2009 como software open source.

O sistema funciona ponta-a-ponta e as transações acontecem entre usuários diretamente, sem intermédio de alguma instituição, sendo irreversíveis e assim garantindo proteção contra fraudes. As transações são verificadas pelos nodos da rede e registradas em uma estrutura chamada de blockchain, ou cadeia de blocos.

As moedas são criadas como recompensa em um processo chamado de mineração e podem ser trocadas por outras moedas, produtos ou serviços, podendo também servir como investimento financeiro.

## Cadeia de Blocos

A cadeia de blocos é uma estrutura similar à um banco de dados distribuído e guarda todas as transações realizadas com a moeda. Ela é mantida pela rede de nodos conectados ao software do bitcoin. Cada nodo da rede armazena sua própria cópia da cadeia de blocos, validando as transações que recebe e propagando as adições à cadeia para outros nodos.

A cada 10 minutos, um bloco é criado com as novas transações que foram validadas pela rede e adicionado à cadeia de blocos. Esse bloco é propagado pela rede entre os nodos. A cadeia contém todo o histórico de transações com a moeda desde sua criação.

## Transações

Transações são definidas em uma linguagem de script. O usuário designa cada endereço e a quantidade de bitcoin a ser enviada pro endereço. A origem do dinheiro numa transação sempre deve referir a uma saída prévia da blockchain não gasta, isso previne o problema do gasto duplo. Usuários podem enviar bitcoins a multiplos destinatários em uma mesma transação.

## Geração de Moedas

As moedas são criadas a partir de um processo chamado mineração, que utiliza poder computacional. É a mineração que mantém a cadeia de blocos

consistente, completa e inalterável por repetidamente verificar e coletar novas transações transmitidas em um novo bloco. Cada bloco contém um hash do bloco anterior, utilizando o algoritmo SHA-256. É daí que vem o encadeamento dos blocos.

Para um novo bloco ser aceito pelo resto da rede, este deve conter uma prova de trabalho. O trabalho consiste em encontrar um número que combinado com o conteúdo do bloco e aplicada a função de hash o resultado é numericamente menor do que o alvo de dificuldade da rede.

Esse alvo de dificuldade é adaptado baseado no próprio poder total de mineração da rede, de modo a manter o tempo médio entre novos blocos em 10 minutos.

O nodo minerador que consegue encontrar o novo bloco é recompensado com novos bitcoins gerados. Todos os bitcoins em circulação foram criados dessa maneira.

**Comercialização de Bitcoins**

A comercialização de bitcoins acontece de maneira a similar à operações de câmbio. A compra e venda são realizadas por meio de plataformas onlines, conhecidas como exchanges ou casas de câmbio, ou também diretamente entre pessoas. Uma das principais características do mercado de bitcoins é sua volatilidade.

**Carteira Digital**

As carteiras digitais guardam as credenciais de posse de bitcoins. Sao necessárias para realizar transações com a moeda, permitindo que usuários gastem suas propriedades.

As carteiras digitais fazem uso de criptografia de chave pública, utilizando um par de chaves geradas, uma pública e outra privada. Na cadeia de blocos, os bitcoins são registrados endereços de bitcoin. Uma chave privada é utilizada para criação de um endereço.

Para gastar bitcoins, é necessário assinar digitalmente a transação com a chave privada relacionada ao endereço que os bitcoins pertencem. A rede verifica a assinatura usando a chave pública relacionada.

**Comercialização de produtos com Bitcoin**

Cada vez mais bitcoin é aceito como meio de pagamento para a comercialização de bens e serviços. Há alguns anos, grandes empresas como Microsoft, Dell e PayPal aceitam bitcoin como meio de pagamento.

Uma das grandes vantagens do bitcoin está na economia por não pagar taxas usuais de bancos e operadoras de cartão.

**Crítica ao Bitcoin**

As vantagens que tornam o bitcoin atrativo estão na anonimidade, facilidade de transferência, ausência de supervisão governamental e acessibilidade em qualquer país.

Se esses fatores são vantagens para o lado dos consumidores, por outro lado representam grandes desvantagens para governos e instituições bancárias. Sem controle da economia, um governo perderia a possibilidade de criação de moeda, perdendo a capacidade de guiar sua economia. Não seria possível acompanhar, controlar e taxar valores monetários transferidos para fora do país.

Mesmo assim, governos poderiam fazer uso da tecnologia da cadeia de blocos. Outras criptomoedas vem inserindo novas funcionalidades tecnológicas em seus sistemas. A moeda Ethereum, por exemplo, inseriu o conceito de contratos inteligentes, onde as regras são estabelecidas através da programação e mantidas pelo sistema.

Não é possível afirmar se o bitcoin será a moeda do futuro, mas com certeza o impacto tecnológico trazido pela moeda irá mudar drasticamente nossos sistemas financeiros.

**Introdução ao projeto desenvolvido**

Após ter sido apresentado uma visão geral sobre o Bitcoin, o objetivo desse projeto é analisar as principais altcoins, buscando verificar se há correlação entre fatores como preço, volume de transação e capitalização de mercado. A principal justificativa para a realização do projeto é o impacto que as criptomoedas e as tecnologias agregadas a elas impactam na sociedade.

As informações sobre as altcoins e também sobre o Bitcoin foram coletadas do site coinmarketcap.com e correspondem aos valores do último ano. As seguintes características foram escolhidas e apresentadas de cada altcoin:

- Preço.
- Capitalização de mercado (market cap);
- Volume de transação diário;
- País de criação;
    - Quando a própria altcoin não afirma ser de nenhum país, utilizou-se como país de criação o país de nascimento do criador.
- Ano de criação.
- Razão de criação;

Esses tipos de características foram escolhidas para que fosse possível realizar tanto uma análise quantitativa quanto qualitativa das altcoins. Vale frisar que para a realização da análise qualitativa se utilizou medidas estatísticas e frameworks especializados nessa área.

**Altcoins**

O seu significado está no próprio nome: “alt” + “coin”, ou seja, criptomoedas alternativas. Surgiram após o Bitcoin e são consideradas bifurcações do mesmo, sendo que algumas foram até mesmo criadas a partir de alguma modificação no código fonte do Bitcoin, enquanto que outras foram criadas do zero.

Atualmente o coinmarketcap.com lista 1292 coins, sendo que 1291 são altcoins, e, muitas delas são podem ser consideradas como moedas “golpes” por determinadas pessoas. Sendo que para estar na lista desse site basta, de maneira resumida, ter somente um registro de transação financeira recente com a moeda.

Por fim, apesar dessa imensidão de alternativas, perceba que o Bitcoin ainda é o líder no segmento de criptomoedas. Por outro lado, os altcoins são necessários, pois alguns deles possuem o intuito de adicionar novas funcionalidades que a rede do Bitcoin pode não possuir.

As altcoins escolhidas para realização do projeto foram:

- Litecoin;
- Dogecoin;
- Monacoin;
- NEO;
- NEM;
- Steem;
- Monero;
- Bytecoin.

Essas altcoins podem ser categorizadas conforme o seu algoritmo de hash da seguinte maneira:

- Scrypt
    - Litecoin;
    - Dogecoin;
    - Monacoin;
- Família SHA
    - NEO;
    - NEM;
    - Steem;
- CryptoNight

- Monero
- Bytecoin.

Isso atende nosso requisito de possuir oito altcoins de no mínimo três categorias diferentes, sendo que há no mínimo duas altcoins para cada categoria escolhida. Buscou-se escolher as altcoins de cada categoria que possuíam o maior valor de capitalização de mercado listado no coinmarketcap.com.

Pode-se dizer que o scrypt é um algoritmo que foi criado com o principal objetivo de dificultar a mineração por hardwares especializados em minerar Bitcoin. Portanto, a razão para ser escolhido para o projeto é essa rivalidade com o Bitcoin, e também a grande quantidade de importantes altcoins que utilizam esse tipo de algoritmo.

A “família SHA” corresponde a qualquer algoritmo que seja um padrão do algoritmo de hash SHA. Foi escolhido para o projeto por causa que o Bitcoin utiliza o algoritmo SHA-256, então queríamos trabalhar com altcoins que tivessem algoritmos da mesma categoria.

O CryptoNight é um algoritmo que dificulta a mineração por hardwares especializados, incentivando a mineração em CPUs comuns. Originalmente desenvolvido pela comunidade do CryptoNote, e, esse foi o motivo de ser um algoritmo escolhido para o projeto, pois o CryptoNote é um protocolo que aumenta a privacidade nas transações de criptomoedas.

Agora vamos apresentar cada altcoin com mais detalhe, onde mostramos as características coletadas. Importante lembrar que aqui os valores quantitativos são valores de um único dia, e, que na análise foram utilizados dados históricos de um ano.

**Litecoin (LTC)**

- Preço: $63,61
- Capitalização de mercado: $3,4 bi
- Volume de transação (24h): $170,1 mi
- País de criação: EUA
- Ano de criação: 2011
- Razão de criação: Utiliza scrypt ao invés de SHA-256, um algoritmo de hash CPU-friendly, dificultando que mineradoras ASIC controlem a rede e tornando a mineração mais democrática. Possui um máximo de 84 milhões de coins, quatro vezes mais que bitcoin. A rede processa um bloco confirmando transações a cada 2,5 minutos, quatro vezes mais rápido que bitcoin.

**Dogecoin (DOGE)**

- Preço: $0,0012

- Capitalização de mercado: $134,3 mi
- Volume de transação (24h): $2,6 mi
- País de criação: EUA
- Ano de criação: 2013
- Razão de criação: Nasceu como uma brincadeira porém rapidamente ganhou valor por ser utilizada na internet como gorjeta de usuários para criadores de conteúdo em redes sociais. Requer apenas 1 minuto para confirmação de transação e não possui um limite no número máximo de moedas que podem ser mineradas. O número total de dogecoins em circulação já ultrapassa 100 bilhões.

**Monacoin (MONA)**

- Preço: $2,88
- Capitalização de mercado: $159,6 mi
- Volume de transação (24h): $2,2 mi
- País de criação: Japão
- Ano de criação: 2014
- Razão de criação: É mais rápida que bitcoin e litecoin, transações são confirmadas a cada 1,5 minutos pela rede. Foi motivada principalmente pelo interesse de japoneses em terem sua própria criptomoeda, hoje diversas lojas do país oferecem produtos e serviços por monacoins. Inicialmente utilizou o algoritmo scrypt porém passou a utilizar o Lyra2Rev2, como força de proteção à ASICs.

**NEO (NEO)**

- Preço: $29,86
- Capitalização de mercado: $1,9 bi
- Volume de transação (24h): $38,1 mi
- País de criação: China
- Ano de criação: 2014
- Razão de criação: Originalmente chamava-se Antshares e foi o primeiro projeto de blockchain chinês. Faz uso de contratos inteligentes (smart contracts), em que regras similares às de um contrato comum são escritas através da programação e garantidas pelo sistema. Coopera com agências autorizadas da China.

**NEM (XEM)**

- Preço: $0,19
- Capitalização de mercado: $1,7 bi

- Volume de transação (24h): $5,9 mi
- País de criação: Singapura
- Ano de criação: 2014
- Razão de criação: Utiliza um algoritmo descentralizado de prova de importância (proof-of-importance) ao invés do algoritmo de prova de trabalho (proof-of-work) comumente utilizado por outras moedas. Cada usuário possui uma pontuação de importância na rede, que determina a chance de se obter recompensas ao criar novos blocos na blockchain.

**Steem (STEEM)**

- Preço: $0,95
- Capitalização de mercado: $235,7 mi
- Volume de transação (24h): $1,1 mi
- País de criação: EUA
- Ano de criação: 2016
- Razão de criação: É uma criptomoeda criada para recompensar usuários da rede Steemit, um site similar ao Reddit que roda sobre uma blockchain. As moedas são distribuídas para criadores de conteúdos diariamente, por votação da comunidade. Podem ser convertidas facilmente para Bitcoin ou Ethereum.

**Monero (XMR)**

- Preço: $122,20
- Capitalização de mercado: $1,9 bi
- Volume de transação (24h): $55,2 mi
- País de criação: Desconhecido
- Ano de criação: 2014
- Razão de criação: Diferente de criptomoedas que derivam do bitcoin, é baseada no protocolo CryptoNote e utiliza o algoritmo CryptoNight. Tem como objetivo prover um nível superior de privacidade. Cada unidade da moeda é indistinguível. É muitas vezes utilizada para quebrar o link entre transações por usuários de bitcoins.

**Bytecoin (BCN)**

- Preço: $0.001152
- Capitalização de mercado: $211 mi
- Volume de transação (24h): $2,7 mi
- País de criação: Desconhecido
- Ano de criação: 2012

- Razão de criação: Também baseado no protocolo CryptoNote com o uso do algoritmo CryptoNight. Por isso, possui foco na privacidade e anonimato das transações. Portanto, sua blockchain é considerada mais resistente a análise. Além disso, foi projetada para ser minerada com mais facilidade em um computador pessoal do que em um hardware especializado.

## Estatísticas utilizadas

Realizamos a descrição dos dados coletados para a realização da análise através de uma medida de valor central, uma medida de dispersão e uma medida de correlação. Optamos por esses três tipos de medidas porque eles são capazes de fornecerem uma boa imagem de como os dados estão dispostos e também por serem suficientes para responder os objetivos desse projeto.

### Medida de valor central

A medida de valor central escolhida foi a média aritmética, por ser a mais popular, direta e simples de se trabalhar. Ela indica o centro de um conjunto de valores, ou seja, resume esse conjunto em termos da posição central, ou do valor mais típico, porém não informa nada sobre o aspecto da distribuição dos dados. Ela é influenciada por valores discrepantes.

### Medida de dispersão

A medida de dispersão escolhida foi o desvio padrão, porque se trata de algo em termos absolutos, portanto possui a mesma unidade de medida dos dados em análise, e, é simples e direto de se trabalhar. De maneira simplória, trata-se da raiz quadrada positiva da variância, que é outra medida de dispersão.

Um baixo desvio padrão indica que os valores estão menos dispersos, ou seja, estão mais próximos da média. Logo, o contrário é que quanto maior um desvio padrão, então há mais variação nos valores. Por estarmos trabalhando com uma amostra e não com toda a população utilizamos a correção de Bessel. Por fim, vale citar que o desvio padrão também é influenciado por valores discrepantes.

### Medida de correlação

A medida de correlação escolhida foi o coeficiente de correlação de Pearson, pois é uma das mais difundidas no meio acadêmico. Como qualquer coeficiente de correlação, ele mede o grau pelo qual duas variáveis tendem a mudar juntar e também descreve a força e a direção dessa relação.

O coeficiente de Pearson pode assumir valores entre -1 e 1. Onde um valor de coeficiente igual a 1 indica uma correlação perfeita positiva entre as duas

variáveis. Isto é, se uma aumenta, a outra sempre aumenta. Por outro lado, um valor de coeficiente igual a -1 indica uma correlação perfeita negativa entre as duas variáveis. Portanto, se uma aumenta, a outra sempre diminui. Por fim, um valor de coeficiente igual a 0 indica que não há dependência linear entre as duas variáveis.

## Tecnologias utilizadas

### Python

Linguagem de programação de alto nível para programação de propósito geral extremamente difundida. Criada por Guido van Rossum e lançada em 1991. Foi a linguagem escolhida para o desenvolvimento da parte prática, já que os membros da equipe possuem conhecimento nela e por os trabalhos anteriores da disciplina também terem sido desenvolvidos nessa linguagem. Geralmente em um ambiente unix o Python já vem instalado por default.

### lxml

Biblioteca open source que serve para processar XML e HTML em Python. Foi a primeira biblioteca XML para Python que demonstrou possuir um alto desempenho e suporte nativo para o Xpath. Na parte prática a biblioteca foi utilizada para trabalhar com o HTML das páginas que possuíam os dados históricos das altcoins.

Optamos por utilizar essa biblioteca devido ao grande número de material sobre ela disponível na internet, além do fato dela ser simples de trabalhar. Para obter, em um ambiente unix com o gerenciador de pacotes de Python pip instalado, basta no terminal digitar *pip install lxml* e no arquivo .py fazer o import necessário. No nosso projeto utilizamos o *from lxml import html*.

### Xpath

Sigla para XML Path Language, trata-se de uma linguagem de busca para selecionar nodos de um arquivo XML. Foi desenvolvido pela W3C (World Wide Web Consortium) em 1999. Na parte prática essa linguagem foi utilizada para coletar as informações necessárias das páginas HTML trabalhadas com o lxml.

Nossa escolha por essa linguagem foi devido ao suporte nativo que a biblioteca lxml fornece para ela, e, também por causa que os membros da equipe já tinham tido um breve contato com ela anteriormente. Para usar o Xpath não precisa de nada específico instalado, basta possuir uma biblioteca que trabalhe com a linguagem.

### Pandas

Biblioteca open source que fornece ferramentas de análise de dados fáceis de usar e com alto desempenho para o Python. Teve seu desenvolvimento iniciado em 2008 por Wes McKinney, um estatístico americano. Na parte prática utilizamos essa biblioteca para calcular todas as estatísticas que foram apresentadas.

Optamos por utilizar essa biblioteca devido a sua simplicidade e a vasta documentação que possui. Para obter, em um ambiente unix com o gerenciador de pacotes de Python pip instalado, basta no terminal digitar *pip install pandas* e no arquivo .py fazer o import necessário, como por exemplo: *import pandas as pd*.

**Numpy**

Biblioteca open source considerada fundamental para computação científica em Python. Foi desenvolvida inicialmente com o Numeric em 1995 por Travis Oliphant, um cientista de dados americano. Na parte prática utilizamos essa biblioteca para criar um tipo de estrutura de dados específico.

Nossa escolha por essa biblioteca deve-se ao fato de que ela fornece uma variedade de diferentes estruturas e funções matemáticas para trabalhar com essas estruturas de maneira fácil e bem documentada. Para obter, em um ambiente unix com o gerenciador de pacotes de Python pip instalado, basta no terminal digitar *pip install numpy* e no arquivo .py fazer o import necessário, como por exemplo: *import numpy*.

**Git**

Sistema de controle de versões distribuído open source projetado para trabalhar com velocidade e eficiência em projetos de qualquer tamanho. Teve seu desenvolvimento iniciado por Linus Torvalds, principal desenvolvedor do kernel Linux, em 2005. Junto com o github.com foi utilizado na parte prática para realizar o versionamento do código desenvolvido.

Optamos por utilizar o Git por ser o líder no segmento e também por causa da familiaridade que a dupla já possui com a ferramenta. Para obter, em um ambiente unix com o gerenciador de pacotes apt instalado, basta no terminal digitar *apt install git*.

**Matplotlib**

Biblioteca open source para Python que gera gráficos 2D em diversos formatos diferentes. Desenvolvido inicialmente por John D. Hunter (in memoriam) em 2003. Na parte prática utilizamos essa biblioteca para criar os gráficos de todas as estatísticas calculadas.

Nossa escolha por essa biblioteca deve-se ao fato de que ela faz coisas fáceis continuarem sendo fáceis ao mesmo tempo que transforma coisas difíceis em coisas possíveis. Para obter, em um ambiente unix com o gerenciador de pacotes de Python pip instalado, basta no terminal digitar *pip install matplotlib* e no arquivo .py fazer o import necessário. No nosso projeto utilizamos o *import matplotlib.ticker* e *import matplotlib.pyplot as plt*.

**Requests**

Biblioteca open source para interagir com o protocolo HTTP em Python. Teve seu desenvolvimento iniciado em 2011 por Kenneth Reitz. Segundo o próprio site, em tradução literal, requests é a única biblioteca HTTP não geneticamente modificada que é segura para consumo humano.

Na parte prática utilizamos ela para pegar as páginas da web que possuíam o conteúdo do nosso interesse. Optamos por utilizar ela devido a sua simplicidade e facilidade de uso. Para obter, em um ambiente unix com o gerenciador de pacotes de Python pip instalado, basta no terminal digitar *pip install requests* e no arquivo .py fazer o import necessário, como por exemplo: *import requests*.

## Estrutura do código

Optamos por desenvolver o código para a parte prática em um único arquivo com o paradigma de programação estruturada, já que acreditamos que ele seria eficiente e suficiente para resolver o nosso problema de maneira mais direta. O programa desenvolvido pode ser dividido na seguinte estrutura:

1. Declaração de estruturas para armazenamento das informações em memória;
2. Coleta de dados da web, limpeza desses dados e armazenamento nas estruturas declaradas previamente;
3. Cálculo das estatísticas para geração de gráficos da mesma.

Posteriormente será detalhado um pouco de cada uma dessas partes com o auxílio de imagens do código.

## Código desenvolvido

Todos os arquivos desenvolvidos podem ser encontrados no seguinte link https://github.com/lucasjoao/computer_security/tree/master/tf_altcoins e o arquivo principal da parte prática, o *atlcoins.py*, segue na íntrega aqui:

```
# -*- coding: utf-8 -*-
from lxml import html
import requests
import pandas as pd
import matplotlib.ticker
import matplotlib.pyplot as plt
import numpy
import sys

##############################################################################
# programa somente funciona com python 2, por isso possui o seguinte check:
if sys.version_info[0] >= 3:
    raise Exception("Executar com Python 2.")
##############################################################################

##############################################################################
# um dict para cada coin que possuira todos os dados coletados
LTC = {'marketcap': [], 'price': [], 'volume': []}
MONA = {'marketcap': [], 'price': [], 'volume': []}
DOGE = {'marketcap': [], 'price': [], 'volume': []}
NEO = {'marketcap': [], 'price': [], 'volume': []}
XEM = {'marketcap': [], 'price': [], 'volume': []}
STEEM = {'marketcap': [], 'price': [], 'volume': []}
XMR = {'marketcap': [], 'price': [], 'volume': []}
BCN = {'marketcap': [], 'price': [], 'volume': []}
BTC = {'marketcap': [], 'price': [], 'volume': []}

# dict que associa os dicts declarados previamente com os seus respectivos ids
# utilizados no site https://coinmarketcap.com/
ALTCOINS = {'id-litecoin': LTC,
            'id-monacoin': MONA,
            'id-dogecoin': DOGE,
            'id-neo': NEO,
            'id-nem': XEM,
            'id-steem': STEEM,
            'id-monero': XMR,
            'id-bytecoin-bcn': BCN,
            'id-bitcoin': BTC}
##############################################################################

##############################################################################
# inicio do parser que acessa os dados historicos em https://coinmarketcap.com/
# e pega os valores desejados para que alimente os dicts ja declarados

# todas as urls que possuem dados historicos do ultimo ano
urls = []
urls.append('https://coinmarketcap.com/historical/20171112/')
urls.append('https://coinmarketcap.com/historical/20171105/')
urls.append('https://coinmarketcap.com/historical/20171029/')
urls.append('https://coinmarketcap.com/historical/20171022/')
urls.append('https://coinmarketcap.com/historical/20171015/')
urls.append('https://coinmarketcap.com/historical/20171008/')
urls.append('https://coinmarketcap.com/historical/20171001/')
urls.append('https://coinmarketcap.com/historical/20170924/')
urls.append('https://coinmarketcap.com/historical/20170917/')
urls.append('https://coinmarketcap.com/historical/20170910/')
urls.append('https://coinmarketcap.com/historical/20170903/')
```

```
urls.append('https://coinmarketcap.com/historical/20170827/')
urls.append('https://coinmarketcap.com/historical/20170820/')
urls.append('https://coinmarketcap.com/historical/20170813/')
urls.append('https://coinmarketcap.com/historical/20170806/')
urls.append('https://coinmarketcap.com/historical/20170730/')
urls.append('https://coinmarketcap.com/historical/20170723/')
urls.append('https://coinmarketcap.com/historical/20170716/')
urls.append('https://coinmarketcap.com/historical/20170709/')
urls.append('https://coinmarketcap.com/historical/20170702/')
urls.append('https://coinmarketcap.com/historical/20170625/')
urls.append('https://coinmarketcap.com/historical/20170618/')
urls.append('https://coinmarketcap.com/historical/20170611/')
urls.append('https://coinmarketcap.com/historical/20170604/')
urls.append('https://coinmarketcap.com/historical/20170528/')
urls.append('https://coinmarketcap.com/historical/20170521/')
urls.append('https://coinmarketcap.com/historical/20170514/')
urls.append('https://coinmarketcap.com/historical/20170507/')
urls.append('https://coinmarketcap.com/historical/20170430/')
urls.append('https://coinmarketcap.com/historical/20170423/')
urls.append('https://coinmarketcap.com/historical/20170416/')
urls.append('https://coinmarketcap.com/historical/20170409/')
urls.append('https://coinmarketcap.com/historical/20170402/')
urls.append('https://coinmarketcap.com/historical/20170326/')
urls.append('https://coinmarketcap.com/historical/20170319/')
urls.append('https://coinmarketcap.com/historical/20170312/')
urls.append('https://coinmarketcap.com/historical/20170305/')
urls.append('https://coinmarketcap.com/historical/20170226/')
urls.append('https://coinmarketcap.com/historical/20170219/')
urls.append('https://coinmarketcap.com/historical/20170212/')
urls.append('https://coinmarketcap.com/historical/20170205/')
urls.append('https://coinmarketcap.com/historical/20170129/')
urls.append('https://coinmarketcap.com/historical/20170122/')
urls.append('https://coinmarketcap.com/historical/20170115/')
urls.append('https://coinmarketcap.com/historical/20170108/')
urls.append('https://coinmarketcap.com/historical/20170101/')
urls.append('https://coinmarketcap.com/historical/20161225/')
urls.append('https://coinmarketcap.com/historical/20161218/')
urls.append('https://coinmarketcap.com/historical/20161211/')
urls.append('https://coinmarketcap.com/historical/20161204/')
urls.append('https://coinmarketcap.com/historical/20161127/')
urls.append('https://coinmarketcap.com/historical/20161120/')
urls.append('https://coinmarketcap.com/historical/20161113/')

for url in urls:
    page = requests.get(url)
    tree = html.fromstring(page.content)

    for idcoin, coin in ALTCOINS.iteritems():
        # xpath e utilizado para acessar o conteudo html
        tr = '//tr[@id="{}"]'.format(idcoin)

        td_class = '//td[@class="no-wrap market-cap text-right"]/text()'
        cap = tree.xpath(''.join((tr, td_class)))
        clean_cap = float(cap[0].strip().replace('$', '').replace(',', ''))

        a_price = '//a[@class="price"]/text()'
        price = tree.xpath(''.join((tr, a_price)))
        clean_price = float(price[0].replace('$', ''))
```

```
        a_volume = '//a[@class="volume"]/@data-usd'
        volume = tree.xpath(''.join((tr, a_volume)))
        clean_volume = round(float(volume[0].replace('$', '').replace(',', '')),
                             2)

        # valores que alimentam as listas dentro dos dicts sao ja sao limpos no
        # momento da extracao
        coin['marketcap'].append(clean_cap)
        coin['price'].append(clean_price)
        coin['volume'].append(clean_volume)
################################################################################

################################################################################
# dicts auxiliares para serem utilizados nos calculos estatisticos

MARKETCAP = {'LTC': LTC['marketcap'],
             'MONA': MONA['marketcap'],
             'DOGE': DOGE['marketcap'],
             'NEO': NEO['marketcap'],
             'XEM': XEM['marketcap'],
             'STEEM': STEEM['marketcap'],
             'XMR': XMR['marketcap'],
             'BCN': BCN['marketcap'],
             'BTC': BTC['marketcap']}

PRICE = {'LTC': LTC['price'], 'MONA': MONA['price'], 'DOGE': DOGE['price'],
         'NEO': NEO['price'], 'XEM': XEM['price'], 'STEEM': STEEM['price'],
         'XMR': XMR['price'], 'BCN': BCN['price'], 'BTC': BTC['price']}

VOLUME = {'LTC': LTC['volume'], 'MONA': MONA['volume'], 'DOGE': DOGE['volume'],
          'NEO': NEO['volume'], 'XEM': XEM['volume'], 'STEEM': STEEM['volume'],
          'XMR': XMR['volume'], 'BCN': BCN['volume'], 'BTC': BTC['volume']}
################################################################################

################################################################################
# calculo e plotagem dos coeficientes de correlacao por pearson para market
# cap, volume e preco

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=MARKETCAP).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 9, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(MARKETCAP.keys())
ax.set_yticklabels(MARKETCAP.keys())
plt.title(u'Correlação por Pearson do market cap no último ano')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=PRICE).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 9, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
```

```
ax.set_xticklabels(PRICE.keys())
ax.set_yticklabels(PRICE.keys())
plt.title(u'Correlação por Pearson do preço no último ano')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=VOLUME).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 9, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(VOLUME.keys())
ax.set_yticklabels(VOLUME.keys())
plt.title(u'Correlação por Pearson do volume de transação no último ano')
plt.show()
################################################################################

################################################################################
# valores do bitcoin so foram utilizados em uma comparacao geral, agora nao
# sao mais necessarios
del MARKETCAP['BTC']
del PRICE['BTC']
del VOLUME['BTC']
################################################################################

################################################################################
# calculo e plotagem das medias do market cap, preço e volume

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.plot(pd.DataFrame(data=MARKETCAP).mean())
y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
ax.yaxis.set_major_formatter(y_formatter)
ax.grid(True)
plt.title(u'Média em dólares do market cap no último ano')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.plot(pd.DataFrame(data=PRICE).mean())
y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
ax.yaxis.set_major_formatter(y_formatter)
ax.grid(True)
plt.title(u'Média em dólares do preço no último ano')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.plot(pd.DataFrame(data=VOLUME).mean())
y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
ax.yaxis.set_major_formatter(y_formatter)
ax.grid(True)
plt.title(u'Média em dólares do volume de transação no último ano')
plt.show()
################################################################################

################################################################################
```

```
# calculo e plotagem dos desvios padroes do market cap, preço e volume

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.plot(pd.DataFrame(data=MARKETCAP).std())
y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
ax.yaxis.set_major_formatter(y_formatter)
ax.grid(True)
plt.title(u'Desvio padrão do market cap no último ano')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.plot(pd.DataFrame(data=PRICE).std())
y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
ax.yaxis.set_major_formatter(y_formatter)
ax.grid(True)
plt.title(u'Desvio padrão do preço no último ano')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.plot(pd.DataFrame(data=VOLUME).std())
y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
ax.yaxis.set_major_formatter(y_formatter)
ax.grid(True)
plt.title(u'Desvio padrão do volume de transação no último ano')
plt.show()
##############################################################################

##############################################################################
# calculo e plotagem dos coeficientes de correlacao por pearson entre market
# cap, volume e preco de cada altcoin analisada

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=LTC).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(LTC.keys())
ax.set_yticklabels(LTC.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do LTC')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=MONA).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(MONA.keys())
ax.set_yticklabels(MONA.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do MONA')
plt.show()
```

```
fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=DOGE).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(DOGE.keys())
ax.set_yticklabels(DOGE.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do DOGE')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=NEO).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(NEO.keys())
ax.set_yticklabels(NEO.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do NEO')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=XEM).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(XEM.keys())
ax.set_yticklabels(XEM.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do XEM')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=STEEM).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(STEEM.keys())
ax.set_yticklabels(STEEM.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do STEEM')
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=XMR).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(XMR.keys())
ax.set_yticklabels(XMR.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do XMR')
```

```
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(pd.DataFrame(data=BCN).corr(), vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = numpy.arange(0, 3, 1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(BCN.keys())
ax.set_yticklabels(BCN.keys())
plt.title(u'Correlação por Pearson do market cap, preço e volume do BCN')
plt.show()
################################################################################
```

## Explicação de alguns pontos do código

Como foi falado anteriormente, aqui será detalhado uma parte do código para cada estrutura do programa.

### Parte um: estruturas

```
16  ################################################################################
17  # um dict para cada coin que possuira todos os dados coletados
18  LTC = {'marketcap': [], 'price': [], 'volume': []}
19  MONA = {'marketcap': [], 'price': [], 'volume': []}
20  DOGE = {'marketcap': [], 'price': [], 'volume': []}
21  NEO = {'marketcap': [], 'price': [], 'volume': []}
22  XEM = {'marketcap': [], 'price': [], 'volume': []}
23  STEEM = {'marketcap': [], 'price': [], 'volume': []}
24  XMR = {'marketcap': [], 'price': [], 'volume': []}
25  BCN = {'marketcap': [], 'price': [], 'volume': []}
26  BTC = {'marketcap': [], 'price': [], 'volume': []}
27
28  # dict que associa os dicts declarados previamente com os seus respectivos ids
29  # utilizados no site https://coinmarketcap.com/
30  ALTCOINS = {'id-litecoin': LTC,
31              'id-monacoin': MONA,
32              'id-dogecoin': DOGE,
33              'id-neo': NEO,
34              'id-nem': XEM,
35              'id-steem': STEEM,
36              'id-monero': XMR,
37              'id-bytecoin-bcn': BCN,
38              'id-bitcoin': BTC}
39  ################################################################################
```

Por simplicidade e praticidade todas as informações coletadas foram armazenadas em memória. Para armazenas essas informações de forma eficiente, utilizamos associações de dois tipos de estruturas que o Python fornece: os dicionários e as listas. No fim geramos:

- Uma lista com as urls que possuiam as informações históricas das coins;
- Um dicionário para cada coin que possuia três listas, uma para cada tipo de informação que seria coletado;

- Um dicionário que associa um id, utilizado nas urls com as informações históricas, com o dicionário detalhado no item anterior;
- Três dicionários auxiliares que associam a sigla da coin com a lista da informação específica que foi declarada dentro do dicionário da coin. São três dicionários já que há três tipos de informações diferentes.

**Parte dois: informações**

```
for url in urls:
    page = requests.get(url)
    tree = html.fromstring(page.content)

    for idcoin, coin in ALTCOINS.iteritems():
        # xpath e utilizado para acessar o conteudo html
        tr = '//tr[@id="{}"]'.format(idcoin)

        td_class = '//td[@class="no-wrap market-cap text-right"]/text()'
        cap = tree.xpath(''.join((tr, td_class)))
        clean_cap = float(cap[0].strip().replace('$', '').replace(',', ''))

        a_price = '//a[@class="price"]/text()'
        price = tree.xpath(''.join((tr, a_price)))
        clean_price = float(price[0].replace('$', ''))

        a_volume = '//a[@class="volume"]/@data-usd'
        volume = tree.xpath(''.join((tr, a_volume)))
        clean_volume = round(float(volume[0].replace('$', '').replace(',', '')),
                             2)

        # valores que alimentam as listas dentro dos dicts sao ja sao limpos no
        # momento da extracao
        coin['marketcap'].append(clean_cap)
        coin['price'].append(clean_price)
        coin['volume'].append(clean_volume)
```

As informações necessárias para calcular as estatísticas estavam distribuídas em cerca de 50 links diferentes no domínio www.coinmarketcap.com, já que a nossa análise foi feita sobre dados históricos do último ano das coins. Para realizar o processo de coleta e armazenamento dessas informações o código desenvolvido realiza de maneira genérica o seguinte:

1. Para cada URL pega o seu HTML e o transforma em String;
2. Então para cada coin escolhida se pega as informações desejadas com o auxílio do Xpath;
3. Após a coleta da informação, limpa-se ela para deixar no formato necessário para o cálculo das estatísticas;
4. Por fim, armazena a informação limpa nas estruturas declaradas na primeira parte do programa.

**Parte três: estatísticas e gráficos**

```
234  fig = plt.figure()
235  ax = fig.add_subplot(111)
236  cax = ax.plot(pd.DataFrame(data=MARKETCAP).std())
237  y_formatter = matplotlib.ticker.FormatStrFormatter('$%1.2f')
238  ax.yaxis.set_major_formatter(y_formatter)
239  ax.grid(True)
240  plt.title(u'Desvio padrão do market cap no último ano')
241  plt.show()
```

É possível calcular as estatísticas em somente uma linha com as funções fornecidas pelo pandas. Na imagem isso pode ser visto na linha 236, onde a estrutura com os dados de entrada é passada como argumento para o *DataFrame* e logo em seguida a estatística já é calculada com o *std()*. Portanto, todas as estatísticas foram calculadas através dos pandas, conforme a seguinte relação:

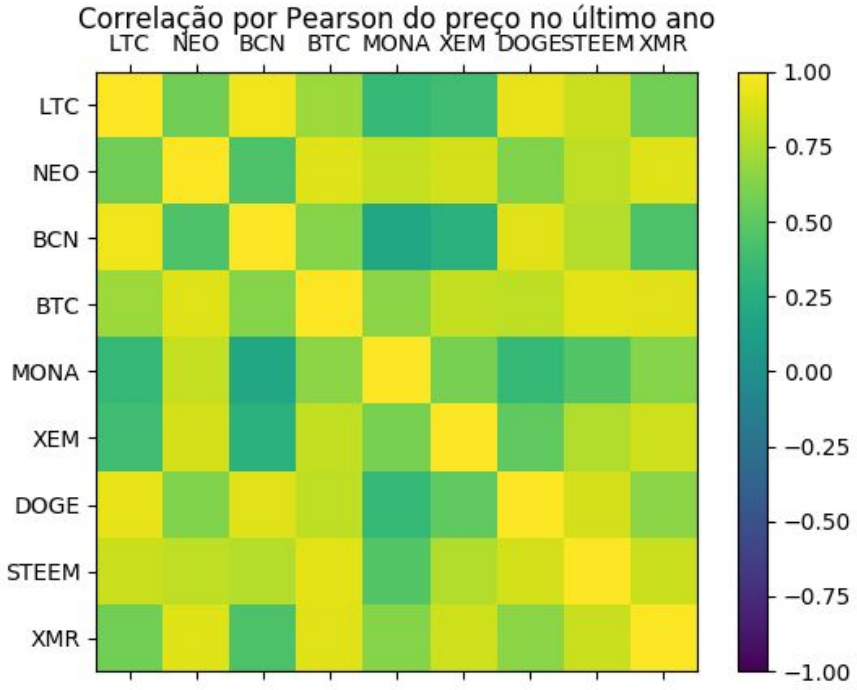
- Correlação de dados através do coeficiente de correlação de Pearson através da função *corr()* que já utiliza Pearson como coeficiente default;
- Desvio padrão através da função *std()* que utiliza a correção de Bessel por default;
- Média aritmética através da função *mean()*.

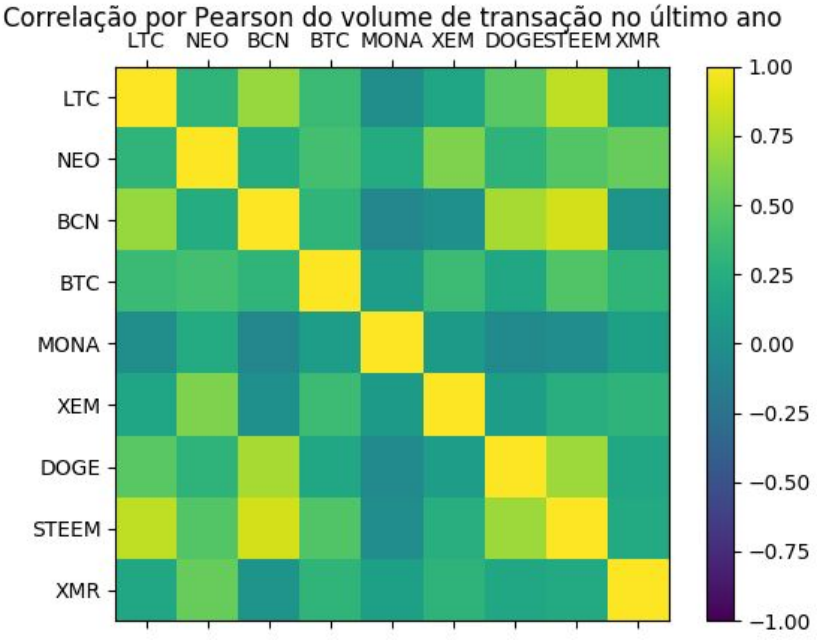
Cada gráfico é apresentado para o usuário após um bloco de configuração do gráfico que possui um cálculo de estatística. Isso é feito na linha 241 com o *plt.show()*. As linhas anteriores servem para criação e configuração do gráfico. Note que optamos por realizar o cálculo da estatística ao mesmo tempo que o resultado já é enviado para o gráfico em uma das linhas da configuração (nesse caso a 236).

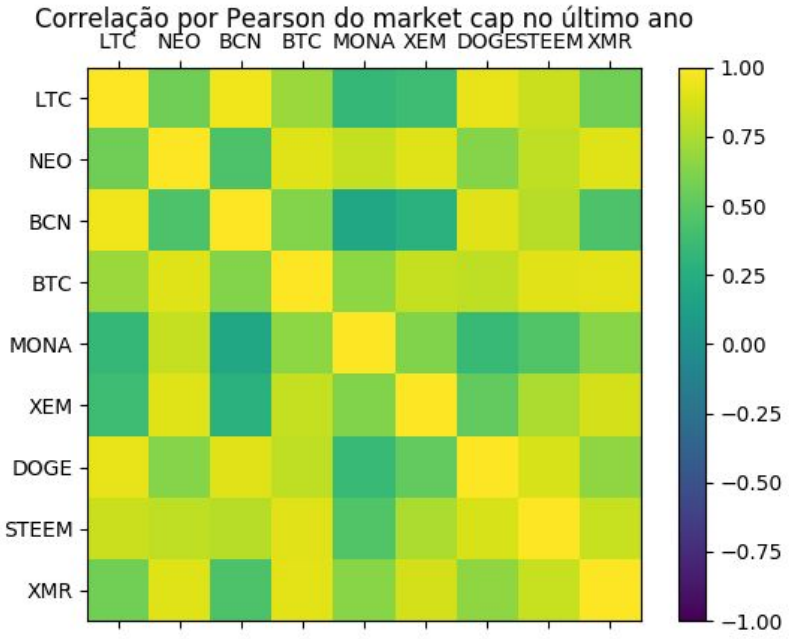
## Resultados

Os resultados da nossa análise serão apresentados em seguida. A apresentação será separada em categorias, onde para cada categoria os gráficos serão mostrados e uma conclusão será dita.

### Correlação por Pearson entre todas as coins


Correlação por Pearson do preço no último ano
LTC
NEO
BCN
BTC
MONA
XEM
DOGE
STEEM
XMR
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Correlação por Pearson do volume de transação no último ano
LTC
NEO
BCN
BTC
MONA
XEM
DOGE
STEEM
XMR
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

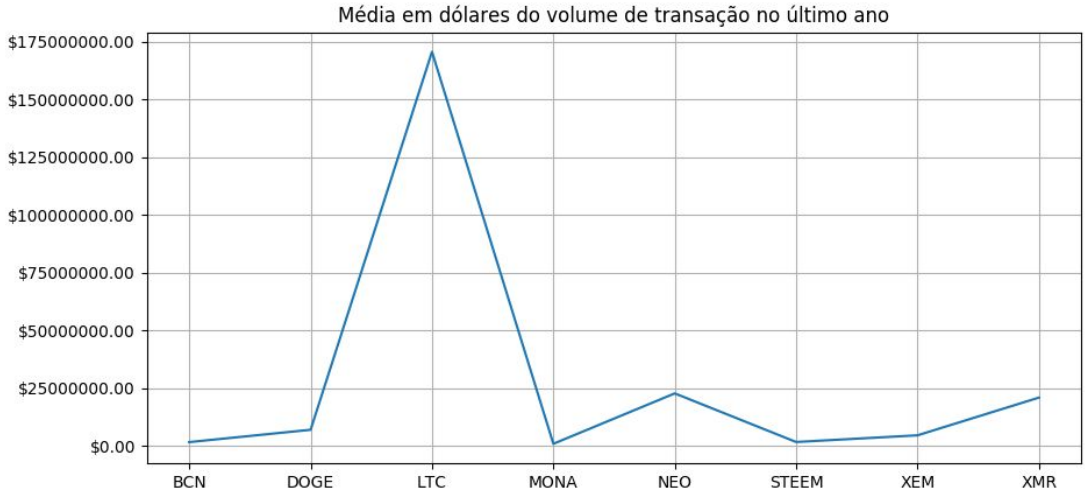


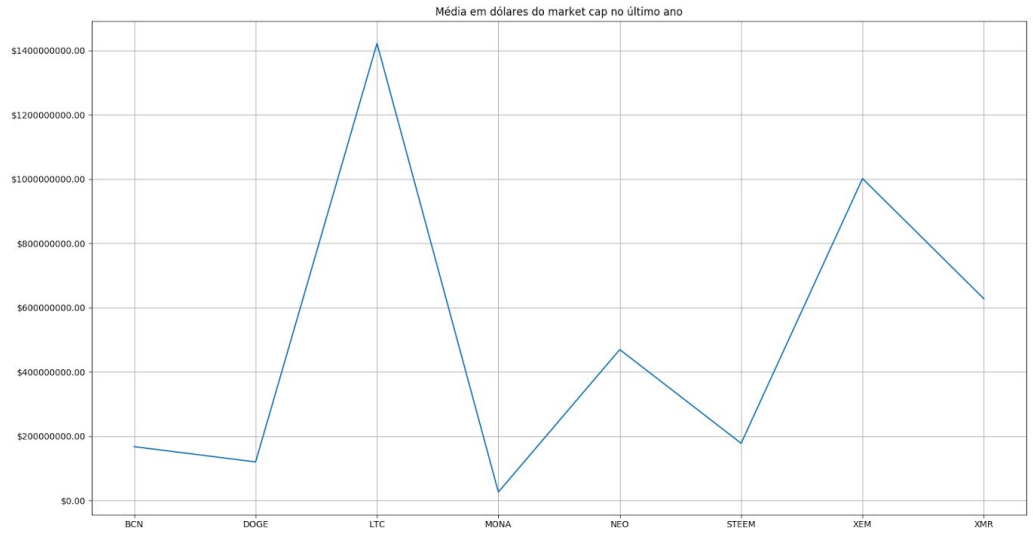

Aqui utilizamos o coeficiente de correlação de Pearson para verificar se há correlação entre os valores de capitalização de mercado, volume diário e preço entre as coins escolhidas, ou seja, um gráfico para cada tipo de informação. Essa foi a única análise que incluímos o Bitcoin (BTC), pois tínhamos interesse em verificar se ele era fortemente correlacionado com alguma altcoin.

Como podemos visualizar, não há uma forte correlação que acontece em todos os gráficos entre o Bitcoin e qualquer outra altcoin. Além disso, não pode-se afirmar que coins com características em comuns, que poderiam ser colocadas em uma mesma categoria, possuem sempre forte correlação. Por fim, o gráfico de preço e capitalização de mercado possuem uma cor no geral mais clara, ou seja, há um grau maior de correlação positiva. Enquanto que o gráfico de volume possui uma cor no geral mais escura, ou seja, há um grau maior de correlação negativa. Note que os blocos da diagonal principal correspondem ao grau de correlação entre dados da mesma coin (iguais) e por isso eles possuem o valor máximo de correlação.

**Medida de valor central**


Média em dólares do preço no último ano
$40.00
$30.00
$20.00
$10.00
$0.00
BCN
DOGE
LTC
MONA
NEO
STEEM
XEM
XMR

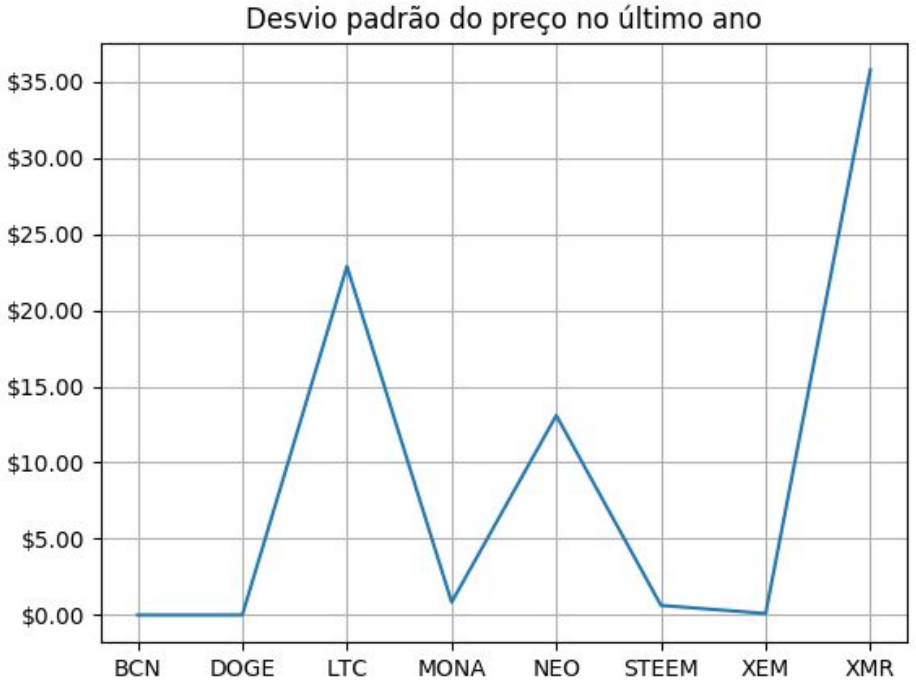

Média em dólares do volume de transação no último ano
$175000000.00
$150000000.00
$125000000.00
$100000000.00
$75000000.00
$50000000.00
$25000000.00
$0.00
BCN
DOGE
LTC
MONA
NEO
STEEM
XEM
XMR

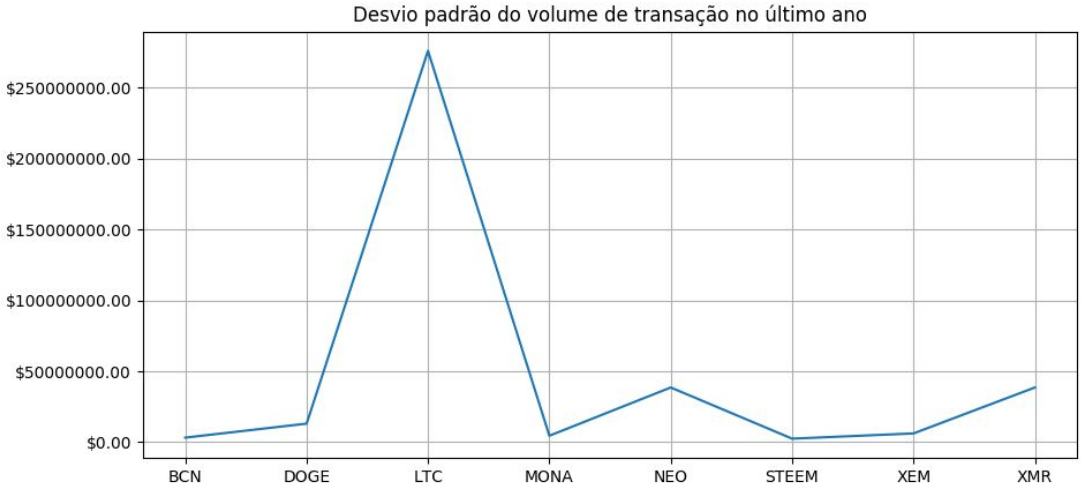


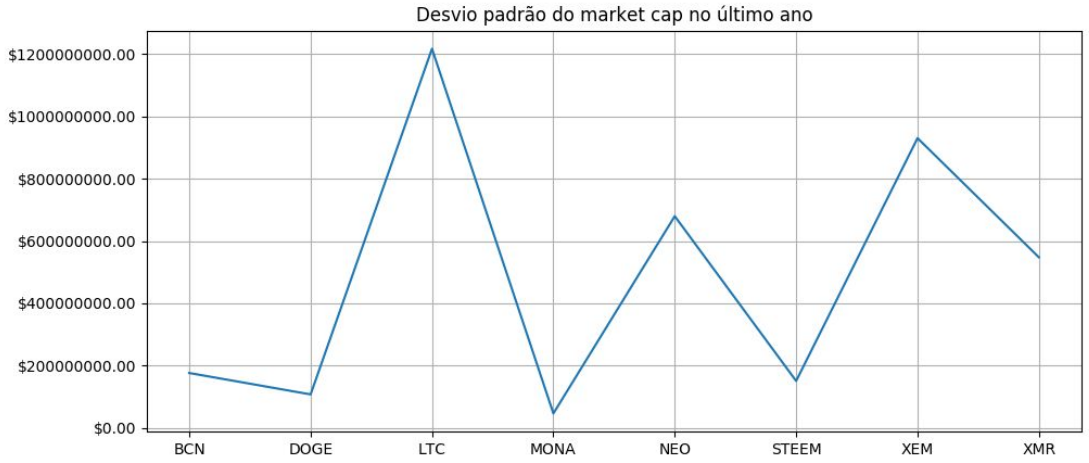

A medida de valor central utilizada foi a média. Aqui a análise focou somente nas altcoins e o Bitcoin não está presente em nenhum gráfico. Novamente há um gráfico para comparar capitalização de mercado, um para volume diário e um para preço que possui o valor para todas as altcoins.

Como já foi dito anteriormente, não pode-se afirmar que coins com características em comuns, que poderiam ser colocadas em uma mesma categoria, irão possuir valores centrais similares. Também é interessante notar pela média de preço das altcoins no último ano que elas não acompanham o boom do preço que o Bitcoin vive recentemente, apesar de possuírem média de valores de capitalização de mercado e de volume diário em escalas expressivas.

**Medida de dispersão**


Desvio padrão do preço no último ano
$35.00
$30.00
$25.00
$20.00
$15.00
$10.00
$5.00
$0.00
BCN
DOGE
LTC
MONA
NEO
STEEM
XEM
XMR

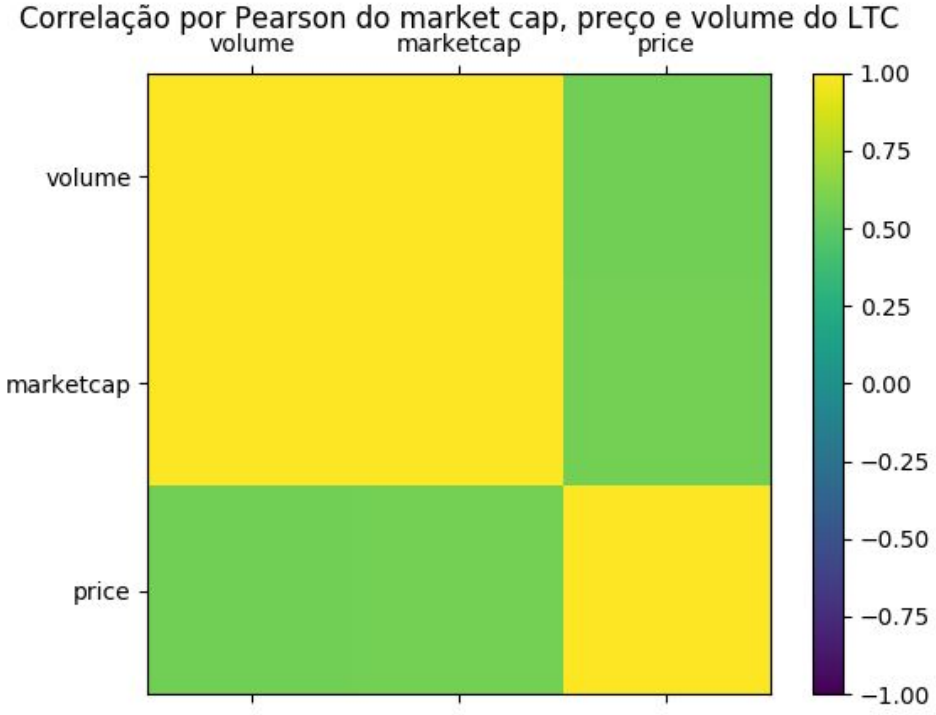

Desvio padrão do volume de transação no último ano
$250000000.00
$200000000.00
$150000000.00
$100000000.00
$50000000.00
$0.00
BCN
DOGE
LTC
MONA
NEO
STEEM
XEM
XMR

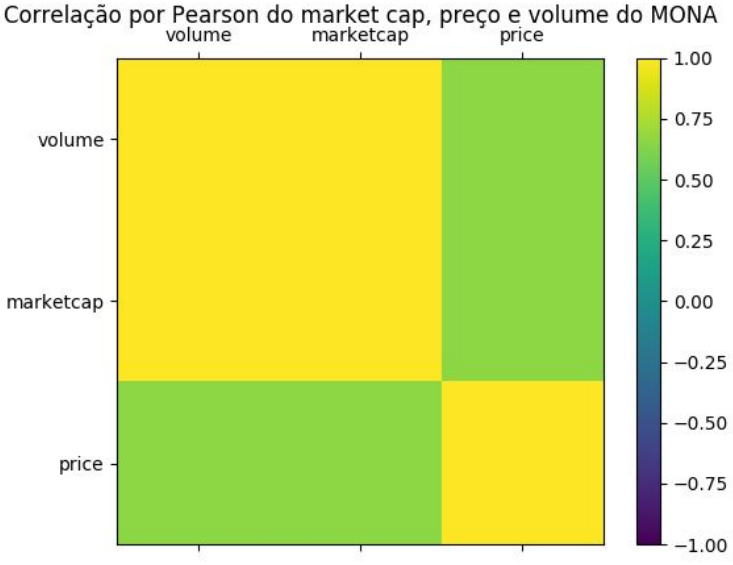


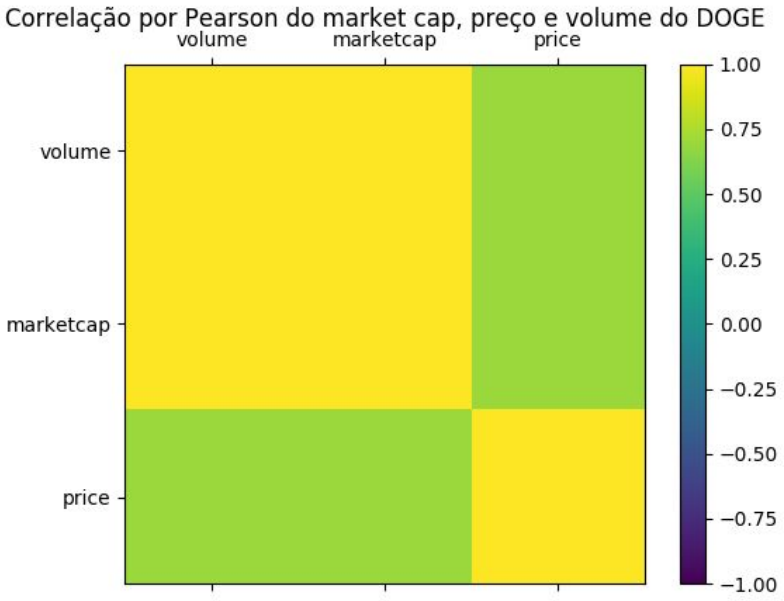

A medida de dispersão utilizada foi o desvio padrão. Novamente, a análise aqui focou somente nas altcoins e o Bitcoin não está presente em nenhum gráfico. Além disso, também há um gráfico para comparar capitalização de mercado, um para volume diário e um para preço, sendo que todos os gráficos possuem o valor para todas as altcoins.

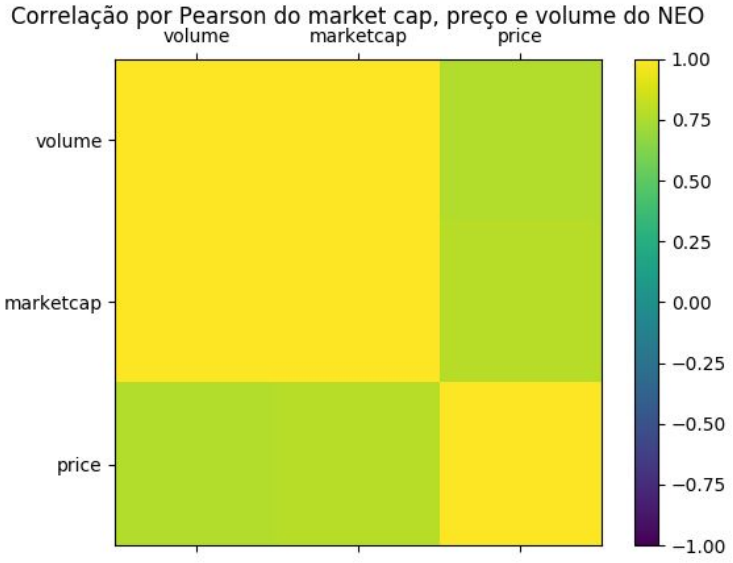
Mais uma vez é possível visualizar que não se pode afirmar que coins com características em comuns, que poderiam ser colocadas em uma mesma categoria, irão possuir medidas de dispersão similares. Por outro lado, percebe-se que os gráficos de média e de dispersão de uma mesma característica possuem no geral formas parecidas (por exemplo, o gráfico de média do preço de todas as altcoins possui um formato similar ao gráfico de desvio padrão do preço de todas as altcoins), e, isso provavelmente deve acontecer por causa que ambas as medidas são influenciadas por valores discrepantes.

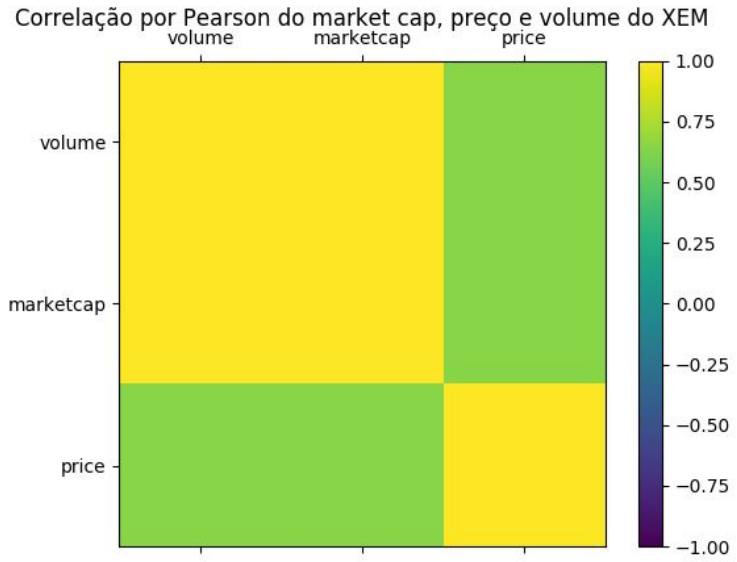
**Correlação por Pearson entre si**

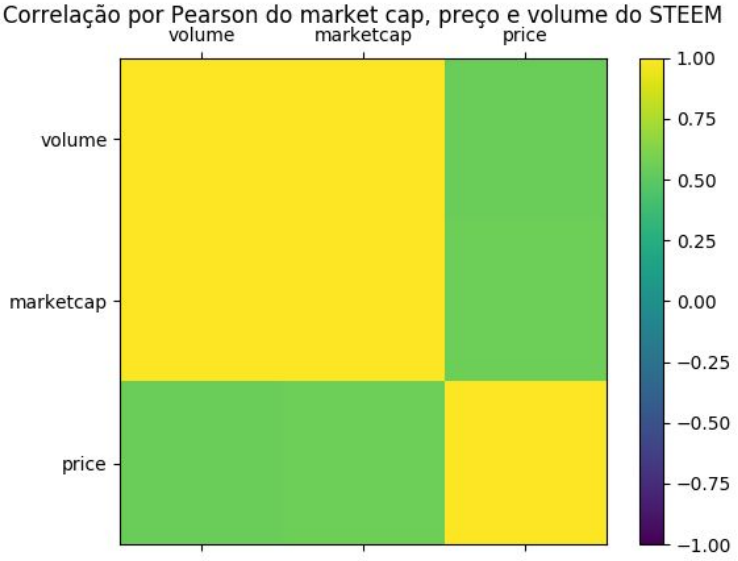

Correlação por Pearson do market cap, preço e volume do LTC
volume
marketcap
price
volume
marketcap
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

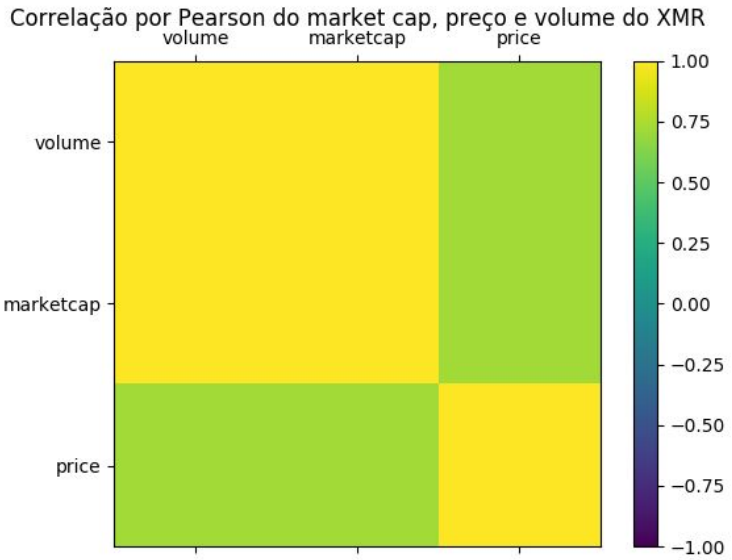

Correlação por Pearson do market cap, preço e volume do MONA
volume
marketcap
price
volume
marketcap
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

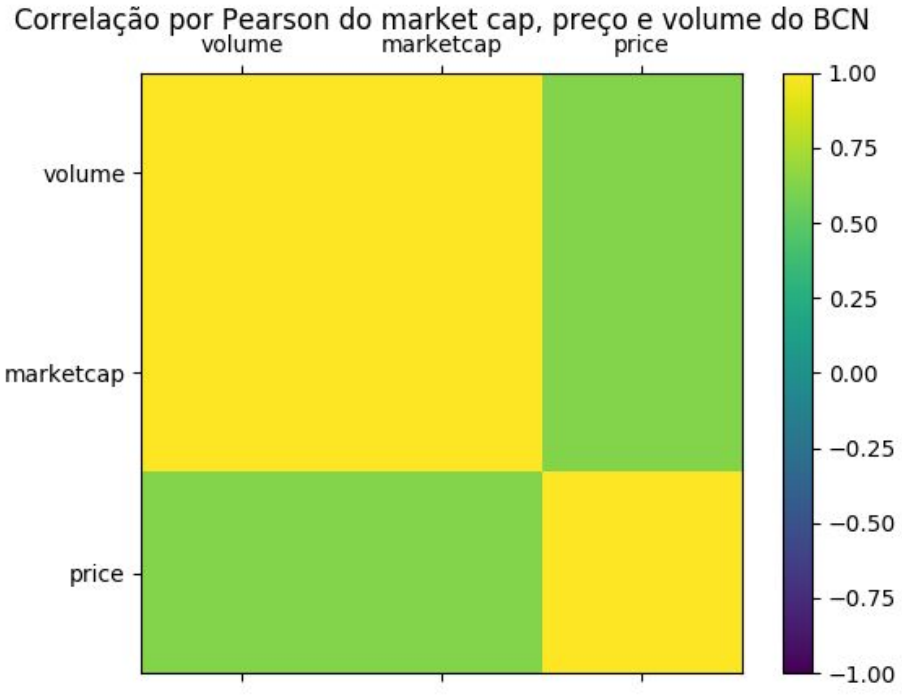

Correlação por Pearson do market cap, preço e volume do DOGE
volume
marketcap
price
volume
marketcap
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Correlação por Pearson do market cap, preço e volume do NEO
volume
marketcap
price
volume
marketcap
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Correlação por Pearson do market cap, preço e volume do XEM
volume
marketcap
price
volume
marketcap
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Correlação por Pearson do market cap, preço e volume do STEEM
volume
marketcap
price
volume
marketcap
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00




Aqui novamente utilizamos o coeficiente de correlação de Pearson, porém com outro objetivo. Decidimos verificar se há correlação entre os dados de capitalização de mercado (market cap), de volume diário e de preço do último ano de uma coin.

O resultado dessa análise foi interessante, pois descobrimos que todas as coins escolhidas possuem o mesmo padrão de correlação entre si, e, por isso, todos os gráficos possuem uma coloração semelhante. Isso acontece porque volume e capitalização de mercado são muito relacionados entre si, e, esses dois possuem um grau mediano de correlação com o preço. Aqui novamente, note que os blocos

da diagonal principal correspondem ao grau de correlação entre dados da mesma característica (iguais) e por isso eles possuem o valor máximo de correlação.

## Conclusão

A principal conclusão que tiramos após a análise sobre os coins é de que apesar de diferentes coins possuírem certas semelhanças, que até poderiam servir para uma categorização, como região de surgimento, ano ou algoritmo de hash, não é possível generalizar e prever comportamentos similares em valores financeiros. Mesmo com todas as altcoins tendo o mesmo padrão de correlação “interno”, ainda assim cada uma possui suas regras e seu próprio mercado.

Aliás, isso também vale para o relacionamento entre altcoins e Bitcoins. Por fim, vemos que o Bitcoin e toda tecnologia agregada a ele é uma das principais novidades do século XXI, mas ainda há pontos polêmicos em torno dessa novidade, e, o grande público leigo ainda não entende muito bem como que isso tudo funciona.

Esse projeto possui as seguintes possibilidades de trabalhos futuros:

- Aumentar a amostra de altcoins da análise;
- Buscar algum método para analisar de forma mais precisa os dados de características que foram coletados em forma de texto;
- Deixar de armazenar as informações coletadas da web em memória e passar a guardar elas em um banco de dados.

Os arquivos desenvolvidos para esse projeto podem ser encontrados no repositório do github que possui o seguinte link https://github.com/lucasjoao/computer_security/tree/master/tf_altcoins, enquanto que o vídeo com a apresentação do projeto pode ser visto no link https://www.youtube.com/watch?v=ox65ienNeiY e os slides utilizados no vídeo podem ser acessado através do https://drive.google.com/open?id=1MqvP9iSMrEuWUPLd-jj74d5U_oCg6_dTJLGHuwixjQY.

## Referências


- https://www.youtube.com/watch?v=x23C1sQg6wQ
- https://en.wikipedia.org/wiki/Virtual_currency#Centralized_versus_decentralized_virtual_currencies
- https://www.youtube.com/watch?v=PQQ0NpwqMlg
- https://www.youtube.com/watch?v=h87O2xfUSv8
- http://explosm.net/comics/3479/
- https://pt.wikipedia.org/wiki/Altcoins

- https://en.wikipedia.org/wiki/Bitcoin
- https://en.wikipedia.org/wiki/Cryptocurrency
- https://en.wikipedia.org/wiki/Blockchain
- https://en.wikipedia.org/wiki/List_of_cryptocurrencies
- https://coinmarketcap.com/
- https://blockexplorer.com/
- http://alt19.com/
- https://cryptocoincharts.info/tools/api
- https://github.com/bitcoin-abe/bitcoin-abe
- https://bitcoin.org/bitcoin.pdf
- https://freedom-to-tinker.com/2015/01/30/nine-awesome-bitcoin-projects-at-princeton/
- http://bitcoin-class.org/pages/project-ideas.html
- http://lxml.de/
- https://www.ibm.com/developerworks/library/x-hiperfparse/index.html
- https://en.wikipedia.org/wiki/XPath
- https://en.wikipedia.org/wiki/Pandas_(software)
- https://en.wikipedia.org/wiki/Wes_McKinney
- http://pandas.pydata.org/
- https://en.wikipedia.org/wiki/John_D._Hunter
- https://en.wikipedia.org/wiki/Matplotlib
- https://matplotlib.org/
- http://docs.python-requests.org/en/master/
- https://en.wikipedia.org/wiki/Requests_(software)
- https://en.wikipedia.org/wiki/Git
- https://git-scm.com/
- https://en.wikipedia.org/wiki/Linus_Torvalds
- https://en.wikipedia.org/wiki/NumPy
- https://en.wikipedia.org/wiki/Travis_Oliphant
- http://www.numpy.org/
- https://en.wikipedia.org/wiki/Python_(programming_language)
- https://en.wikipedia.org/wiki/Structured_programming
- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.mean.html
- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.std.html
- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.corr.html
- https://en.wikipedia.org/wiki/Bessel%27s_correction
- https://github.com/lucasjoao/lecture_notes/blob/master/05/prob.md

- https://support.minitab.com/pt-br/minitab/18/help-and-how-to/statistics/basic-statistics/supporting-topics/correlation-and-covariance/a-comparison-of-the-pearson-and-spearman-correlation-methods/
- https://en.wikipedia.org/wiki/Pearson_correlation_coefficient
- http://cryptocoin.cc/
- https://en.bitcoin.it/wiki/CryptoNote
- https://en.bitcoin.it/wiki/CryptoNight
- https://monero.stackexchange.com/questions/1386/is-it-better-to-mine-with-a-cpu-or-gpu
- https://en.wikipedia.org/wiki/Secure_Hash_Algorithms
- https://www.cryptocompare.com/coins/guides/what-is-scrypt/
- https://en.wikipedia.org/wiki/Scrypt
- https://www.coindesk.com/litecoin-founder-charles-lee-on-the-origins-and-potential-of-the-worlds-second-largest-cryptocurrency/
- https://en.wikipedia.org/wiki/Dogecoin
- https://www.investopedia.com/articles/investing/022016/bitcoin-vs-litecoin-vs-dogecoin-comparing-virtual-currencies.asp
- https://www.youtube.com/watch?v=DDkGGf5n9Uo&feature=youtu.be
- https://www.coindesk.com/why-japan-fell-love-monacoin-cat-meme-cryptocurrency/
- https://bitcoiner.today/en/monacoin-the-crypto-coin-to-mine-with-lyra2rev2-algorithm/
- https://www.btcsoul.com/noticias/antshares-sera-renomeado-para-neos/
- https://pt.wikipedia.org/wiki/Contrato_inteligente
- https://blockgeeks.com/guides/smart-contracts/
- https://en.wikipedia.org/wiki/NEM_(cryptocurrency)
- https://themerkle.com/what-is-proof-of-importance/
- https://cointelegraph.com/news/proof-of-importance-nem-is-going-to-add-reputations-to-the-blockchain
- https://en.wikipedia.org/wiki/Steemit
- https://steem.io/
- https://en.wikipedia.org/wiki/Monero_(cryptocurrency)
- https://en.wikipedia.org/wiki/Bytecoin_(cryptocurrency)