UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

JOÃO GABRIEL SILVA FERNANDES

PREVISÃO GENERALIZADA À CURTO PRAZO DE CRIPTOMOEDAS COM REDES NEURAIS PROFUNDAS

Goiânia
2021

UNIVERSIDADE FEDERAL DE GOIÁS

INSTITUTO DE INFORMÁTICA

AUTORIZAÇÃO PARA PUBLICAÇÃO DE TRABALHO DE CONCLUSÃO DE CURSO EM FORMATO ELETRÔNICO

Na qualidade de titular dos direitos de autor, **AUTORIZO** o Instituto de Informática da Universidade Federal de Goiás – UFG a reproduzir, inclusive em outro formato ou mídia e através de armazenamento permanente ou temporário, bem como a publicar na rede mundial de computadores (*Internet*) e na biblioteca virtual da UFG, entendendo-se os termos "reproduzir" e "publicar" conforme definições dos incisos VI e I, respectivamente, do artigo 5º da Lei nº 9610/98 de 10/02/1998, a obra abaixo especificada, sem que me seja devido pagamento a título de direitos autorais, desde que a reprodução e/ou publicação tenham a finalidade exclusiva de uso por quem a consulta, e a título de divulgação da produção acadêmica gerada pela Universidade, a partir desta data.

Título: PREVISÃO GENERALIZADA À CURTO PRAZO DE CRIPTOMOEDAS COM REDES NEURAIS PROFUNDAS

Autor(a): João Gabriel Silva Fernandes

Goiânia, 14 de Junho de 2021.

João Gabriel Silva Fernandes – Autor

Wellington Santos Martins – Orientador

Evandro Carrijo Taquary – Co-Orientador

JOÃO GABRIEL SILVA FERNANDES

PREVISÃO GENERALIZADA À CURTO PRAZO DE CRIPTOMOEDAS COM REDES NEURAIS PROFUNDAS

Trabalho de Conclusão apresentado à Coordenação do Curso de Computação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Bacharel em Computação.

Área de concentração: Área de Concentração.

Orientador: Prof. Wellington Santos Martins

Co-Orientador: Prof. Evandro Carrijo Taquary

Goiânia
2021

JOÃO GABRIEL SILVA FERNANDES

PREVISÃO GENERALIZADA À CURTO PRAZO DE CRIPTOMOEDAS COM REDES NEURAIS PROFUNDAS

Trabalho de Conclusão apresentado à Coordenação do Curso de Computação do Instituto de Informática da Universidade Federal de Goiás como requisito parcial para obtenção do título de Bacharel em Computação, aprovada em 14 de Junho de 2021, pela Banca Examinadora constituída pelos professores:

Prof. Wellington Santos Martins
Instituto de Informática – UFG
Presidente da Banca

Prof. Evandro Carrijo Taquary
Instituto de Informática – UFG

Prof. Eduardo Simões de Albuquerque
Instituto de Informática – UFG

Resumo

S. Fernandes, João Gabriel. **PREVISÃO GENERALIZADA À CURTO PRAZO DE CRIPTOMOEDAS COM REDES NEURAIS PROFUNDAS**. Goiânia, 2021. 34p. Relatório de Graduação. Instituto de Informática, Universidade Federal de Goiás.

Nos últimos anos o mercado de criptomoedas vem ganhando espaço no radar de investidores de alto risco, e consequentemente o interesse em prever movimentos deste mercado aumentou, problema o qual ainda se mostra desafiador. Esse trabalho faz o uso de redes neurais recorrentes **LSTM**, num modelo de regressão, para predição da variação diária de preço de Bitcoin e Altcoins. No lugar da previsão do preço, o alvo escolhido é sua variação (%). Validação cruzada K-Fold é realizada para uma avaliação mais confiável do modelo. Uma análise dos resultados é feita para a adição de dados sociais como volume de tweets, google trends e flags de eventos, ao final comparando com outros trabalhos. Indicadores técnicos do preço, volume de tweets, google-trends e calendário de eventos são avaliados como features adicionais. Duas criptomoedas estáveis(BTC, ETH) e duas *Altcoins* (TRX, EOS) são usadas. Os resultados obtidos mostram um **MSE** de $3,9 \cdot 10^{-4}$ e acurácia de *aumento ou queda no preço* de 77,8% para Bitcoin, e performance promissora para treino e teste cruzado entre moedas.

Palavras–chave

LSTM, Criptomoedas, Bitcoin, Modelo Generalizado, Tweet, Google Trends, Coinmarketcal

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1

Introdução

O mercado de criptomoedas já capitaliza um total de 7,5 trilhões de Reais, causando aumento no interesse de investidores, que, junto do aumento no acesso a poder computacional [Hwang 2018], vêm tornando mais comum o uso de DeepLearning em aplicações de investimentos no mercado.

Apesar da previsão de mercado de ações e criptomoedas ser uma área já bem explorada, a maioria esmagadora dos trabalhos foca em treinar/ajustar seus modelos para um único mercado alvo (ação na bolsa de valores ou criptomoeda). A especialização tem seus benefícios de extrair o máximo de performance possível, porém também carregam seus pontos negativos como a incapacidade de aplicar um modelo treinado à outros alvos (economizando tempo e recursos). Por exemplo, assim como um carro autônomo treinado para dirigir no Japão não funcionaria no Brasil sem ajustes, uma Rede Neural treinada para prever Bitcoin não teria eficácia para prever outra criptomoeda pouco conhecida.

Modelos de LSTM, comumente aplicados para problemas de NLP (*Natural Language Processing*), se mostram uma ótima escolha para previsão de criptomoedas como apontado em [Tandon et al. 2019] e [Felizardo et al. 2019]. Ela é capaz de manter informação, que julgar importante, entre passos (isto é, capacidade de memória), tornando-se uma ótima escolha para o problema de previsão de série temporal, neste trabalho.

Também foi estudada a correlação de tweets e google trends com preço do Bitcoin por [Abraham, Jethin; Higdon, Daniel; Nelson, John; e Ibarra, Juan 2018], concluindo que índices de interesse (volume de tweets e google trends) estão altamente correlacionados ao preço do Bitcoin, porém a analise de sentimentos de tweets não é um indicador confiável quando os preços estão caindo.

Sendo assim, neste trabalho vamos atacar o problema de **DomainShift** e **ConceptDrift** utilizando a **variação (%) do preço** no lugar do preço, com objetivo de representar informação do comportamento do preço de forma universal, e através disso construir e analisar um **modelo genérico** capaz prever a variação (%) do preço de criptomoedas, atingindo performance comparáveis em criptomoedas nunca vistas pelo modelo antes.

Para isso vamos primeiro, definir e explicar conceitos utilizados ao longo do do-

cumento, e métricas de erro/correlação e também de acurácia binária mesmo se tratando de um modelo de regressão.
Então converter os dados de preço das quatro criptomoedas *Bitcoin(BTC), Ethereum(ETH), Tron(TRX), EOS(EOS)* para **variação** (em porcentagem) **de preço** e mostrar que o uso da tal no lugar do preço "crú"é capaz de contornar o problema de *Conceptdrift* e torna o modelo capaz de ser treinado por uma criptomoeda e prever outras (tendo elas características similares ou não).

Também será analisado o uso de features adicionais. Features estatísticas, derivadas de **indicadores técnicos** bem conhecidos no mercado de ações. E features sociais, onde ferramentas foram construídas para extrair dados de websites, e assim gerar séries temporais do **volume de tweets**, **google trends** e **calendário de eventos de criptomoedas** (calendários dirigidos pela comunidade).

Ao final, conclui-se que resultados promissores foram obtidos para a generalização da previsão de criptomoedas, e que dados sociais guardam potencial. Sendo que ambos mostram a capacidade de maior exploração e avaliação das estratégias abordadas, propondo as tais como possíveis trabalhos futuros.

CAPÍTULO 2

Fundamentação Teórica

Com o aumento no poder computacional, o uso de DeepLearning vem se tornando mais acessível em aplicações de investimentos no mercado de ações. Modelos de RNN (*Recurrent Neural Network*), comumente aplicados para problemas de NLP (*Natural Language Processing*), também se mostraram uma ótima escolha para predição de séries temporais. Muitos modelos utilizam histórico de preços e indicadores técnicos [Weng, Ahmed e Megahed 2017] como fonte de *features*, e alguns estudos mostram a relação de dados sociais/obtidos da internet com o preço [Abraham, Jethin; Higdon, Daniel; Nelson, John; e Ibarra, Juan 2018], [Weng, Ahmed e Megahed 2017]

2.1 Rede Neural Recorrente e LSTM

Redes neurais tradicionais não possuem persistência de dados, isto é, informações não são mantidas entre etapas durante a execução. Isto afeta diretamente a natureza de séries temporais, por exemplo: dado um histórico de preços do Bitcoin, a ocorrência de uma alta á 7 dias atrás pode resultar em uma resistência próxima deste preço hoje.

Para isso redes com recorrência (RNN) foram criadas, permitindo persistência de informação. A imagem 2.1 mostra uma célula $\boldsymbol{A}$ de uma RNN ao longo do tempo, onde dada uma sequência X como entrada tal que x_t é o elemento de índice t em X, uma célula RNN recebe a entrada x_t, gera a saída h_t e passa informação para ela mesma através do loop.

Entretanto uma RNN tem dificuldades em manter informação à longo prazo, para isso a LSTM (*Long Short Term Memory*), uma subclasse da RNN, [Hochreiter e Schmidhuber 1997] foi criada, onde por projeto é capaz de "lembrar"de informações por longos períodos de tempo. Elas se tornaram muito populares para trabalhar com dados sequenciais, como séries temporais.

Uma LSTM possui um vetor de memória c_t chamado *cell state*, e utiliza os portões *input gate* i_t, *forget gate* f_t e *output gate* o_t para explicitamente controlar o valor de c_t. Esses portões permitem que a rede aprenda o que salvar, o que descartar, o que

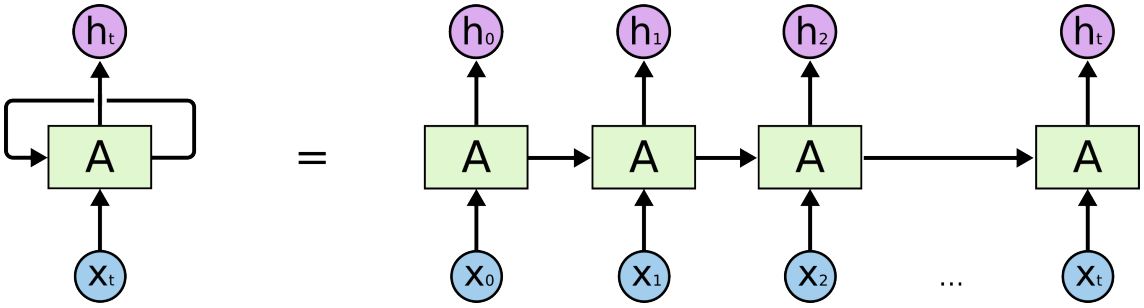

Figura 2.1: *Uma mesma célula RNN ao longo do tempo*
imagem retirada de https://colah.github.io

lembrar e o que dar como saída. A arquitetura de uma LSTM pode ser vista mais à fundo em [Olah 2015] e [Ming et al. 2017].

2.2 Validação cruzada K-Fold

Validação cruzada é um método estatístico usado para estimar a capacidade/performance geral de um modelo. É um método popular pela simplicidade e por resultar em estimativas de performance menos enviesadas que outros métodos.
Para isso ele treina e valida todo o dataset através da rotação na divisão do conjunto entre treino e teste.
O método possui um único parâmetro K, onde K é o número de grupos (folds) de tamanho igual que o conjunto de dados será divido. Após a divisão, cada possível combinação de $k-1$ *folds* para treino e 1 *fold* para teste é iterada (veja a figura 2.2), treinando uma nova rede (isto é, com seus pesos "limpos") e salvando os resultados da validação.

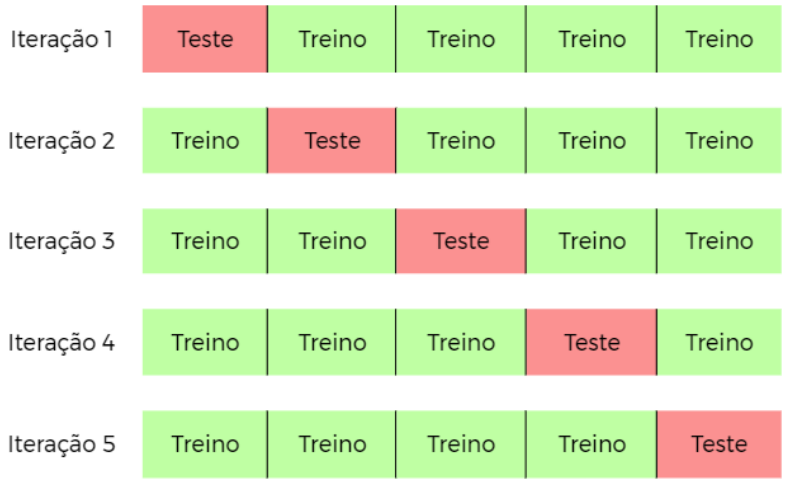

Figura 2.2: *Processo do K-Fold (k=5)*
cada iteração divide os dados em k-1 folds para treino, e 1 fold para teste

Após treinar e validar/testar todas iterações, a média da performance é calculada,

e neste trabalho em adicional, também é calculado o desvio padrão para medir o quão consistente é a performance do modelo com os conjuntos de dados.

2.3 Indicadores técnicos

Neste trabalho vamos usar dois dos três indicadores técnicos usados em [Weng, Ahmed e Megahed 2017] (pois dentre *oscilador estocástico*, *%R de Willians* e *RSI*, os dois primeiros diferem em apenas como são escalados), devido a similaridade da natureza especulativa entre o mercado de ações e mercado de criptomoedas.

2.3.1 Oscilador Estocástico (%K)

$$\%K = \left(\frac{C - Low_t}{High_t - Low_t} \right) \times 100$$

- C = o preço de fechamento
- Low_t = o menor preço nos últimos t dias
- $High_t$ = o maior preço nos últimos t dias
- $\%K$ = valor atual do indicador

Definido pela fórmula acima, é um indicador de *momentum* que compara um preço específico com as máximas e mínimas recentes, retornando um valor entre 0 e 100. Ele é usado para indicar sinais de ***overbought*** (sobre-comprado) e ***oversold*** (sobre-vendido). A prática para o valor padrão de t é $t = 14\ dias$.
Normalmente valores acima de 80 são considerados como ***overbought*** e abaixo de 20 como ***oversold***.

2.3.2 Índice de força relativa (RSI)

O RSI também é um indicador de *momentum* que retorna valores entre 0 e 100, mas diferente do Oscilador Estocástico, dá sinais de ***overbought*** e ***oversold*** medindo a velocidade do movimento de preços, ou em outras palavras, medindo a magnitude de mudanças recentes no preço.

Ele é definido pela seguinte fórmula:

$$RSI = 100 - \left(\frac{100}{1 + \frac{Average\ gain}{Average\ loss}} \right)$$

onde *Average gain* e *Average loss* são expressos em valores positivos definidos por:

$$Average\ gain = \frac{((media\ dos\ ganhos\ nos\ ultimos\ t\ dias) \times (t-1) + ganho\ atual)}{t}$$

$$Average\ loss = \frac{((media\ das\ perdas\ nos\ ultimos\ t\ dias) \times (t-1) + perda\ atual)}{t}$$

A prática para o valor padrão de t também é $t = 14\ dias$.
A pratica comum entre investidores é de considerar valores acima de 70 como ***overbought*** e abaixo de 20 como ***oversold***.

Em geral o *Oscilador Estocástico* é mais util para mercados agitados que sobem e descem numa certa frequência mas sem uma tendência para uma direção, e *RSI* para mercados com tendência.
Desta forma, selecionando estes dois indicadores, conseguimos ter dados adequados ambos para comportamentos cíclicos e de tendência.

CAPÍTULO 3

Proposta e Metodologia

A proposta deste trabalho é analisar a viabilidade de um modelo genérico de Redes Neurais de Regressão para predição de criptomoedas, isto é, um modelo que após ser treinado seja aplicado para diversas outras criptomoedas sem re-treina-lo. E também analisar o uso de features extraídas de redes sociais e do calendário de eventos dirigido pela comunidade.

3.1 Modelo da Rede Neural

A rede construída é composta por 5 camadas, como mostrado em 3.1, sendo a primeira e última camadas de entrada e saída respectivamente. A primeira camada interna (2ª camada da rede) é uma LSTM com 750 unidades, a segunda camada interna 175 unidades Densas e a terceira 75 unidades Densas. Todas as camadas internas possuem *dropout* de 0.1 (10%).

Os dados de treino para uma LSTM é uma matriz de 3 dimensões [$Amostras \times TamanhoJanela \times Features$], onde $TamanhoJanela$ é a quantidade de dias anteriores que uma célula LSTM olhará. Uma Amostra (que corresponde a uma entrada na rede) possui o histórico de $TamanhoJanela$ dias dos valores de cada feature.
Nossa implementação adotou $TamanhoJanela = 30$, por apresentar melhores resultados dentre os valores testados de $5, 15, 30, 60$.

A configuração acima foi a que apresentou melhores resultados, dado a seguinte busca de arquitetura:

- **camadas internas**: 1 LSTM + N camadas Densas, onde a partir de 2 camadas Densas não houve melhora na performance.
- **unidades na camada**: 250+ unidades para LSTM, onde não apresentou ganho que compensasse o aumento no tempo de execução a partir de 750 unidades.
  E 50 a 250 unidades para a primeira camada Densa, 175 apresentando o melhor resultado; as camadas Densas seguintes gradualmente possuem menos unidade, formando um "funil"até a última camada (1 unidade).

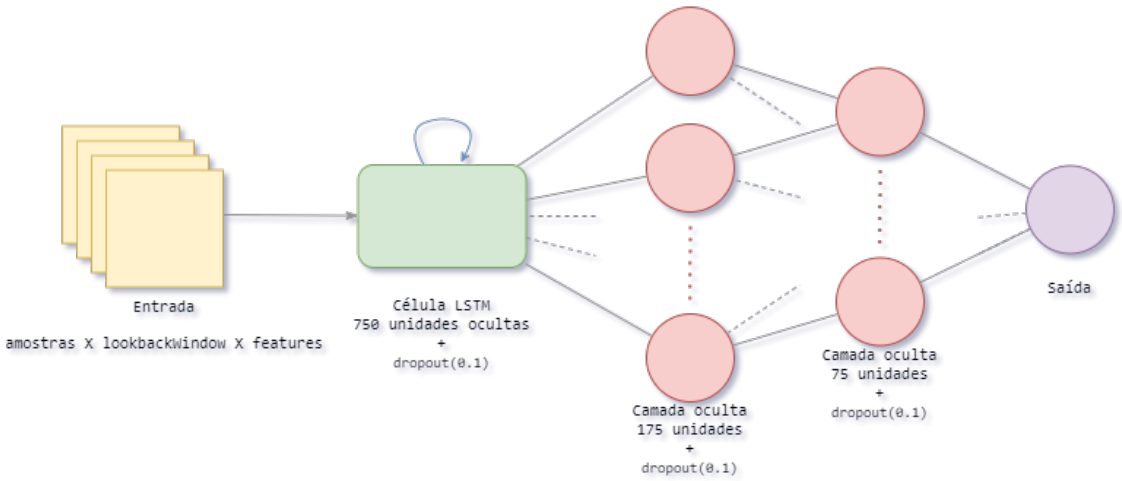

Figura 3.1: *Modelo da Rede Neural*

O modelo foi implementado em `python` utilizando a biblioteca `Keras`, e utiliza como função *loss* (função de erro) o método *MeanSquaredError*(Erro Quadrático Médio).

3.2 Obtenção dos dados e features

Foram usados dois tipos de dados:

1. **estatísticos/técnicos**: preço, Oscilador Estocástico, Índice de força relativa (RSI)
2. **dados sociais**: volume de tweets, google trends, eventos de criptomoedas;

Os dados estatísticos são obtidos de um dataset do Kaggle [Rajkumar 2021], composto por arquivos .csv de diversas criptomoedas, onde cada arquivo possui o histórico de granularidade diária do preço de abertura, fechamento, alta, baixa e volume. Então é calculado o **preço** ($= abertura - fechamento$), e os **indicadores técnicos** usando a biblioteca `pandas-ta` do python.

`O código feito para extração dos dados sociais está disponível no repositório deste trabalho`

Já os dados sociais são extraídos de múltiplas fontes (acompanhe o diagrama 3.2):

- **Volume de tweets**: retirado dos gráficos encontrados no bitinfocharts.com, usando um webScrapper implementado em python. O webScapper faz uma requisição para URL da pagina com o gráfico desejado, faz um parse do HTML obtido, e então

utiliza expressões *Regex* para extrair a `data` e `valor` de cada ponto do histórico. Em seguida tudo é compilado num `DataFrame pandas` e retornado como uma série temporal.

- **Google trends**: também retirado dos gráficos encontrados no bitinfocharts.com usando o webScapper contruído, e em caso de não existir dados para a criptomoeda desejada, é utilizada a API do googleTrends com o método de reconstrução por sobreposição proposto em [TSENG 2019]:
  - A API do google trends tem um limite de tamanho do intervalo de 9 meses para granularidade diária, 5 anos para semanal e acima de 5 anos o dados obtido é mensal.
  - A API do google trends retorna valores normalizados para aquele período específico, entre 0 e 100. Tornando impossível comparar valores de intervalos diferentes.
  - Como solução, o método de sobreposição faz várias requisições de até 9 meses que se interseccionam, e outra(s) de >9 meses para escalar corretamente os valores.
- **Calendário de eventos**: o website coinmarketcal.com provê calendários dirigidos pela comunidade com eventos de criptomoedas. Foi implementado um webScrapper em python com a biblioteca *Selenium* (Ferramenta para automação de testes de interface de apliacções web), para extrair uma lista de eventos:
  - Cada evento é apresentado como um "*Card*"no website, do qual possui 4 campos: `[data de divulgação, data prevista do evento, quantidade de votos totais, % de votos que foram positivos]`;
  - o WebScapper força o carregamento de todos os cards interagindo com a pagina, extrai os 4 campos citados, e monta uma lista com todos os eventos de um criptomoeda;
  - Então para cada item da lista, os campos são renomeados para `[data de divulgação, data prevista para ocorrer, votos, confiabilidade]`;
  - Ao final, um `DataFrame pandas` do histórico de eventos da criptomoeda é gerado e retornado;

  O DataFrame final é então convertido em múltiplas séries temporais:

  1. Duas séries temporais para o anúncio de um evento, uma para sua pontuação e outra para sua confiabilidade.
  2. Várias séries temporais informando que em *n* dias ele irá acontecer.

  Dado que um evento possui os atributos `[data de divulgação, data prevista para ocorrer, votos, confiabilidade]`, um total de 2(N+1) séries temporais serão criadas, onde $N = quantos\ dias\ informar\ com\ antecedencia$.

por exemplo, dado os seguintes eventos:

	Anunciado dia	Acontece dia	Votos	Confiabilidade
Evento A	0	6	250	0.85
Evento B	5	8	300	0.99

numa série temporal de 11 dias, vamos informar a rede com **3 dias de antecedência** ($N = 3$):

```
# o índice representa o dia, e o valor os votos|confiabilidade do evento
# dia n°              0    1    2    3    4    5    6    7    8  9 10
anuncio_de_evento = [250, 0,   0,   0  , 0  , 300, 0  , 0  , 0, 0, 0]
anunc_evento_confi= [.85, 0,   0,   0  , 0  , .99, 0  , 0  , 0, 0, 0]

evento_falta_3_dia =[0  , 0,   0,   250, 0  , 300, 0  , 0  , 0, 0, 0]
evento_falta_2_dia =[0  , 0,   0,   0  , 250, 0,   300, 0  , 0, 0, 0]
evento_falta_1_dia =[0  , 0,   0,   0  , 0  , 250, 0  , 300, 0, 0, 0]

evento_3_dia_confi =[0  , 0,   0,   .85, 0  , .99, 0  , 0  , 0, 0, 0]
evento_2_dia_confi =[0  , 0,   0,   0  , .85, 0,   .99, 0  , 0, 0, 0]
evento_1_dia_confi =[0  , 0,   0,   0  , 0  , .85, 0  , .99, 0, 0, 0]
```

em caso de intersecção entre eventos na mesma série temporal, é feita a soma da pontuação e mesclagem proporcional da confiabilidade:

$$votes = v_1 + v_2$$

$$merged\ confidence = \frac{(v_1 * c_1) + (v_2 * c_2)}{v_1 + v_2}$$

onde, dados dois eventos X e Y que se interseccionam, v_1 e c_1 são o *votos* e *confiabilidade* de X, e v_2 e c_2 são o *votos* e *confiabilidade* de Y, respectivamente.

Após extrair e mesclar todos os dados sociais num único dataset, eles são unidos com o dataset dos dados estatísticos/técnicos e calculado mais três features: `variação do volume de tweets`, `variação do google trends` e `variação do preço`, onde variação é a mudança em porcentagem entre o valor do *timestep* anterior e atual:

$$variacao[i] = \left(\frac{valor[i] - valor[i-1])}{valor[i-1]} \right)$$

Ao final, temos a seguinte lista de features/séries temporais disponíveis:

1. Preço
2. Variação do preço (%)
3. Oscilador Estocástico
4. Índice de força relativa (RSI)

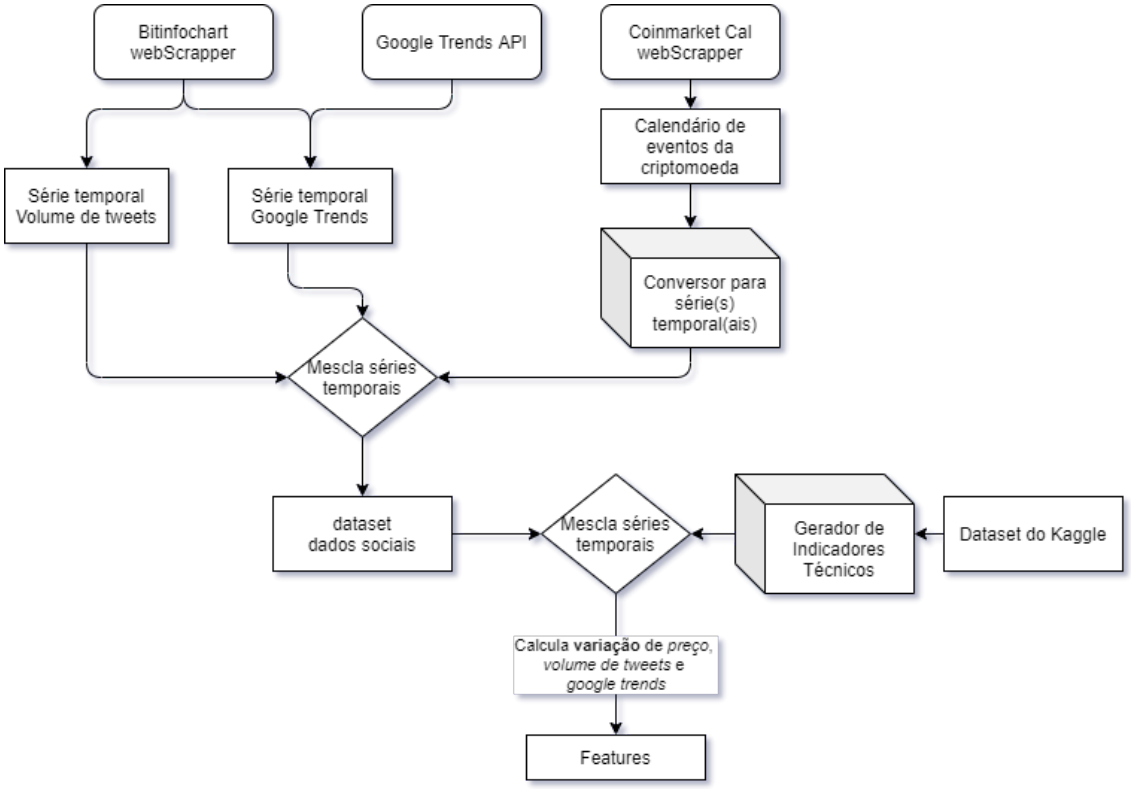

Figura 3.2: *Fluxograma da obtenção de features*

5. Volume de tweets
6. Variação do volume de tweets (%)
7. Google trends
8. Variação do google trends (%)
9. Anúncio de evento - votos
10. Anúncio de evento - confiabilidade
11. 1 dia para evento acontecer - votos
12. 1 dia para evento acontecer - conf.

...

N. *n* dias para evento acontecer - conf.

Com exceção das features de variação (%), todas são normalizadas no intervalo [-1, 1].

3.3 Metodologia

O trabalho será composto por duas partes:

1. Análise e validação do modelo genérico para previsão de criptomoedas.
2. Análise e validação do uso de features de indicadores técnicos e sociais.

Para a **análise e validação do modelo genérico**, quatro criptomoedas serão testadas: **Bitcoin**(BTC), **Ethereum**(ETH), **Tron**(TRX) e **EOS**(EOS);
As duas primeiras são moedas conhecidas de menor volatilidade, enquanto as duas últimas são ***Altcoins*** ("moedas alternativas") relativamente novas (lançadas no mercado em 2017 e 2018 respectivamente) que possuem maior volatilidade quando comparadas com Bitcoin e Ethereum. O objetivo é testar como um modelo se comporta na predição de outras

moedas com comportamentos de mercado similares e não-similares, isto é, outras moedas das quais não foram usadas em seu treino.

Para testar a performance base de cada criptomoeda, será usado o método de K-Fold (K=5) com **early stopping** (técnica que seleciona e salva a melhor versão do modelo de um fold), treinando e testando com dados da **mesma criptomoeda**.
Então vamos testar a performance para cada combinação (A,B) tal que $A,B \in \{BTC,ETH,TRX,EOS\}$ e $A \neq B$, treinando o modelo com todos os dados da criptomoeda A e testando com todos os dados da criptomoeda B, isto é, treinar e prever com **diferente criptomoedas**.

Já para a **análise e validação do uso de indicadores técnicos e de dados sociais**, será usado apenas o **Bitcoin** (BTC) como base, por ser a moeda de menor volatilidade. E então verificar o aumento e/ou perca de performance ao adicionar uma feature por vez.

3.3.1 Métricas

Por ser um modelo de regressão, a métrica *MeanSquaredError*(Error quadrático médio) foi adotada, mas também optamos por adicionar duas outras métricas de acurácia,

- **MeanSquaredError** (MSE): uma das métricas mais comuns para comparar estimadores. Ela indica o quão próximos estão os valores da predição em relação dos valores esperados, onde quanto mais próximo MSE for de 0, mais próximos os valores. Ela é definida por:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(|y_i - \tilde{y_i}|)^2$$

  Onde y_i é o valor esperado no ponto i, e $\tilde{y_i}$ o valor previsto no ponto i.
- **Movement Accuracy** (MovAcc): taxa de acerto da direção do movimento.
  Para predição do preço "crú"isso é equivalente à taxa de acerto de se o preço subiu ou desceu. Para predição de variação (%) do preço, é equivalente a taxa de acerto de se a % foi maior ou menor que a anterior.

$$up(prev,now) = \begin{cases} 1 & \text{se } prev < now \\ 0 & \text{se } prev \geq now \end{cases}$$

$$sameDirection(past,true,pred) = \begin{cases} 1 & \text{se } up(past,true) = up(past,pred) \\ 0 & \text{se } up(past,true) \neq up(past,pred) \end{cases}$$

$$MovAcc(A,B) = \frac{\sum_{i=1}^{|A|} sameDirection(A_{i-1},A_i,B_i)}{|A|}$$

Onde A e B são os valores alvo e de predição do modelo, respectivamente.

- **Price Up Down Accuracy** (UpDownAcc): taxa de acerto de aumento ou diminuição no preço, **métrica aplicável apenas para predição de variação (%) do preço**:

$$aboveZero(x) = \begin{cases} 1 & \text{se } x > 0 \\ 0 & \text{se } x \leq 0 \end{cases}$$

$$sameSide(x,y) = \begin{cases} 1 & \text{se } aboveZero(x) = aboveZero(y) \\ 0 & \text{se } aboveZero(x) \neq aboveZero(y) \end{cases}$$

$$UpDownAcc(A,B) = \frac{\sum_{i=1}^{|A|} sameSide(A_i, B_i)}{|A|}$$

Onde A e B são os valores alvo e de predição do modelo, respectivamente.

Mesmo sendo um modelo de regressão, as duas métricas adicionais são importantes para medir também a viabilidade da aplicação deste modelo.
Pois saber se o preço irá subir ou descer é de extrema utilidade (mesmo sem saber o quanto). Também saber se a variação (%) do preço do preço vai estar acima ou abaixo do dia anterior, pode servir como indicador de alguma tendência da variação (%) de preço, isto é, alertar que provavelmente a (%) irá mover em determinada direção.

CAPÍTULO 4

Testes e Resultados

Todas execuções foram executadas em GPU (GTX 1060), usando a biblioteca `Keras` em `python`, usando com `epochs=600` e `batch_size=128`. `batch_size=128` se mostrou o melhor valor a GPU utilizada (abaixo disso parte da GPU ficava inutilizada, causando maior tempo de execução, e acima não havia ganho). Já para os `epochs`, durante os treinos foi observado pico de performance do modelo entre os epochs 200 e 400 500, então foi escolhido `epochs=600` para garantir que a melhor versão do modelo não seria perdida pelo *earlystoping*.

4.1 Testes Preliminares

O conteúdo a seguir tem como objetivo de confirmar e/ou dar suporte à seguinte hipótese:

O uso de variação é capaz de representar melhor e de forma **generalizada/universal** a informação do comportamento do preço numa série temporal, e elimina o fator da mudança brusca do *domínio de valores* entre o inicio e fim da série temporal (problema conhecido como **concept drift** ou **data drift** pela área de aprendizado de máquinas).

4.1.1 Prevendo o Preço

Primeiro criamos um modelo base, prevendo o **preço**, assim como nos trabalhos relacionados [Weng, Ahmed e Megahed 2017] [Caux, Bernardini e Viterbo 2020] [Abraham, Jethin; Higdon, Daniel; Nelson, John; e Ibarra, Juan 2018] [Tandon et al. 2019] [Jaquart, Dann e Weinhardt 2021] , do dia seguinte.

Usando apenas o **preço** como feature, com K-Fold (k=5), obtivemos a performance mostrada em 4.1.

Adicionando a feature **variação (%) do preço** junto do **preço**, com K-Fold (k=5), a performance melhorou tanto na média como no desvio padrão (como apontado em 4.2).

	MeanSquaredError	MovementDiretionAccuracy
Média	1,19 e-4	56,93 %
Desv. Padrão	4,49 e-5	2,50 e-2

Tabela 4.1: *perfomance para previsao do bitcoin. Features: [preço]; alvo: preço*

	MeanSquaredError	MovementDiretionAccuracy
Média	9,25 e-5 ,	66,30 %
Desv. Padrão	2,63 e-5	1,57 e-2

Tabela 4.2: *perfomance para previsao do bitcoin. Features: [preço, variação]; alvo: preço*

`Uma imagem de predição (sem K-Fold) do preço do bitcoin pode ser encontrada no apêndice A`

4.1.2 Prevendo a Varição (%) do preço

Após confirmar melhora na performance (acreditando que o uso de variação é capaz de representar melhor a informação do comportamento do preço), optamos por prever a mesma, ao invés do preço "crú".
Então, usando apenas **variação (%) do preço** como feature, com K-Fold (k=5), obtivemos a performance apresentada em 4.3.

	MeanSquaredError	MovementDiretion Acc.	PriceUpDown Acc.
Média	4,00 e-4	73,20 %	75,39 %
Desv. Padrão	4.44 e-5	2,13 e-2	1,14 e-2

Tabela 4.3: *perfomance para previsao do bitcoin. Features: [variação]; alvo: variação*

Pode-se notar que mudar o problema para prever a **variação (%) do preço** trás uma performance superior [1].
Lembrando que a métrica *PriceUpDownAccuracy* para predição de variação(%) equivale à métrica *MovementDiretionAccuracy* para predição de preço, e vice-versa.

`Um gráfico mostrando o preço junto da variação do preço pode ser visto em 4.1`

[1] Note que o aumento do *MSE* é esperado, e não se traduz em pior performance, pois o *range*(amplitude) de prováveis/possíveis valores num ponto para predição de preço é muito menor do que para predição de variação (%) do preço. Então o *MSE* é naturalmente maior para previsão de variação (%), mesmo quando comparado à um modelo de mesma performance que prevê o preço (ao invés da variação).

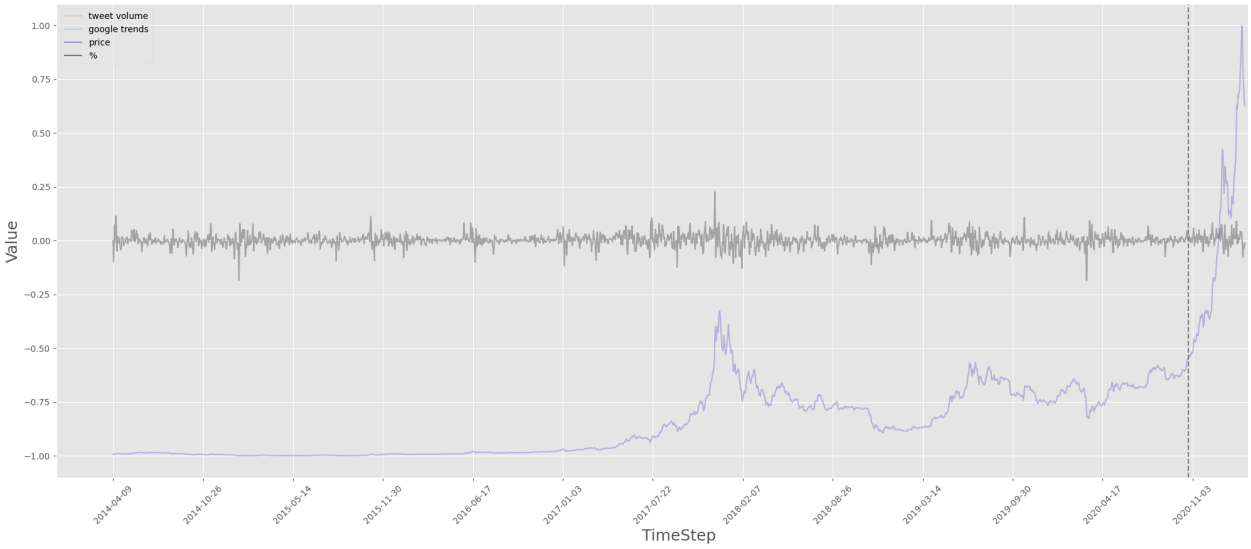

Figura 4.1: *Gráfico comparando preço e variação de preço (Bitcoin)*

4.2 Resultados do Modelo Genérico

Para o modelo genérico, vamos fazer uma validação cruzada entre datasets distintos, isto é, a rede é treinada com **todo o dataset** de uma **criptomoeda base**, e então validada com **todo o dataset** de uma **criptomoeda alvo**. O objetivo é testar a performance do modelo com uma criptomoeda nunca vista antes, tendo ela características (ex.: volatilidade, capitalização de mercado, comportamento,proposta da moeda, etc.) similares ou assimilares.
E assim descobrir se o uso da variação (%) de preço torna o modelo capaz de **DomainShift** (capacidade do modelo de funcionar em outro domínio, isto é, capacidade de funcionar no domínio de outras criptomoedas).

Quatro criptomoedas foram testadas ao total, duas mais estabelecidas no mercado e menos voláteis (*BTC* e *ETH*), e duas *AltCoins* mais voláteis(*TRX* e *EOS*), quando comparadas às anteriores. Os resultados podem ser vistos na tabela 4.7

As diagonais em cinza na tabela 4.7 são o resultado do treino e validação de uma mesma moeda com K-Fold (k=5). As células restantes são os resultados obtidos ao treinar com uma moeda base (linhas) e validar com outra moeda alvo (colunas).

A tabela 4.8 mostra o desvio padrão [2], da execução com K-Fold(k=5), para cada métrica.

[2]Observou-se que o desvio padrão varia numa pequena quantidade entre execuções (provavelmente devido ao não determinismo das funções randômicas utilizadas pelo keras), então foi feita a média de 3 repetições.

	BTC	ETH	TRX	EOS
BTC	3,97e-4	8,65e-4	8,84e-4	9,07e-4
ETH	4,00e-4	8,62e-4	8,93e-4	9,04e-4
TRX	3,98e-4	8,67e-4	8,86e-4	9,01e-4
EOS	4,02e-4	8,68e-4	8,89e-4	9,05e-4

Tabela 4.4: *Mean Squared Error (média)*

	BTC	ETH	TRX	EOS
BTC	73,42%	73,87%	73,41%	72,69%
ETH	72,37%	75,44%	73,23%	72,87%
TRX	72,05%	73,87%	75,42%	72,96%
EOS	72,25%	73,76%	73,23%	74,88%

Tabela 4.5: *Movement Accuracy (média)*

	BTC	ETH	TRX	EOS
BTC	75,55%	76,34%	76,90%	72,06%
ETH	74,58%	77,69%	76,90%	72,15%
TRX	74,34%	76,62%	78,42%	72,42%
EOS	74,26%	76,85%	76,54%	74,44%

Tabela 4.6: *Price Up Down Accuracy (média)*

Tabela 4.7: *Média das métricas do modelo genérico*

eixo y: moeda usada para treinar; eixo x: moeda testada

		BTC	ETH	TRX	EOS
Desvio Padrão	MeanSqrdErr	4.695 e-5	1,266 e-4	1,289 e-4	1,624 e-4
	Mov. Acc.	2,331 e-2	1,649 e-2	2,911 e-2	2,448 e-4
	UpDown Acc.	1,447 e-2	1,755 e-2	2,344 e-2	2,262 e-4

Tabela 4.8: *Desvio padrão (K-Fold)*

4.3 Resultados do uso de Features Adicionais

4.3.1 Features sociais

Como apontado por [Abraham, Jethin; Higdon, Daniel; Nelson, John; e Ibarra, Juan 2018], e mostrado nas figuras 4.2 e 4.3, o preço do Bitcoin e indicadores de interesse (google trends e volume de tweets) estão fortemente relacionados.

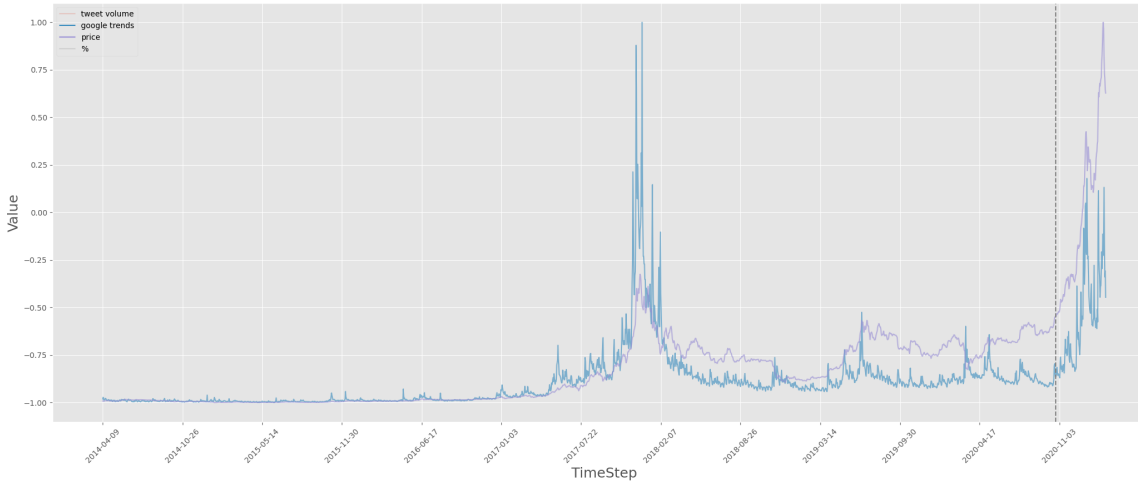

Figura 4.2: *Séries temporais (Bitcoin) do preço e google trends comparadas*

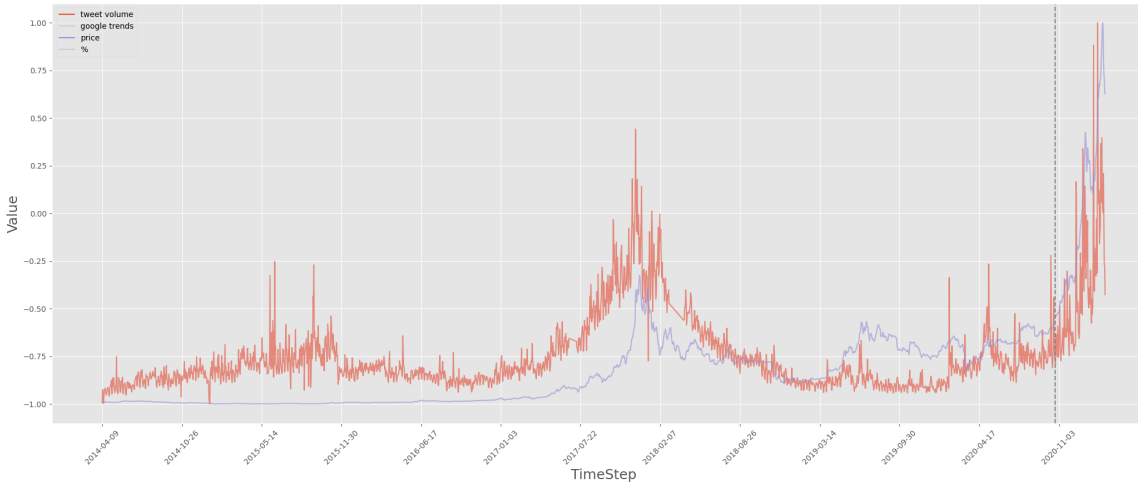

Figura 4.3: *Séries temporais (Bitcoin) do preço e google trends comparadas*

Cada *feature social* foi testada separadamente, para a moeda Bitcoin, usando junto apenas a feature de variação (%) de preço. Os resultados podem ser vistos na tabela 4.9

	MeanSquaredError	Mov. Direction Acc.	PriceUpDown Acc.
Variação preço apenas	3,97 e-4	73,42 %	75,55 %
Eventos	4,18 e-4	73,03 %	74,66 %
Volume tweets	4,05 e-4	73,40 %	75,25 %
Google trends	4,08 e-4	73,24 %	75,09 %
Variação volumeTweets	3,98 e-4	73,58 %	75,65 %
Variação googleTrends	3,97 e-4	73,68 %	75,67 %

Tabela 4.9: *Média das métricas com features sociais cada feature foi usada em conjunto da* ***variação de preço****; K-Fold(k=5) também foi utilizado*

4.3.2 Features de Indicadores Técnicos

Primeiro será avaliada, para a moeda Bitcoin, a performance ao usar **indicadores técnicos** junto da **variação (%) do preço**. Então, selecionar os indicadores com melhores resultados, e testar para o restante das criptomoedas. Iremos testar combinações do *Oscilador estocástico(%K)* e *RSI*, com t assumindo os valores $t = \{7,\ 14,\ 28,\ 56\}$.

	MSE	Mov. Acc.	UpDown Acc.
$RSI_{56} + RSI_{28}$	3,756 e-4	75,29 %	77,80 %
RSI_{56}	3,757 e-4	75,25 %	77,60 %
$RSI_{56} + RSI_{28} + RSI_{14}$	3,762 e-4	75,19 %	77,76 %
RSI_{28}	3,765 e-4	75,15 %	77,78 %
$RSI_{56} + ESTOC_{56}$	3,775 e-4	75,05 %	77,52 %
RSI_{14}	3,784 e-4	74,90 %	77,72 %
RSI_7	3,796 e-4	74,48 %	77,42 %
$RSI_7 + ESTOC_7$	3,803 e-4	74,28 %	77,38 %
$RSI_{14} + ESTOC_{14}$	3,808 e-4	74,54 %	77,46 %
$ESTOC_7$	3,845 e-4	74,04 %	75,65 %
$ESTOC_{14}$	3,855 e-4	74,52 %	76,01 %
$ESTOC_{28}$	3,859 e-4	74,62 %	76,23 %
$ESTOC_{56}$	3,869 e-4	74,64 %	76,67 %
Nenhum	3,976 e-4	73,44 %	75,53 %

Tabela 4.10: *Performance (média) ao usar Indicadores Técnicos*
tabela ordenada pelo MSE (não necessariamente representa melhor performance)

Todos os testes utilizaram K-Fold (k=5).

Os resultados, presentes na tabela 4.11, indicam que o *RSI* entrega mais ganho de performance do que o *K%*, até quando combinados. Também que, apesar do valor padrão recomendado ser $t = 14$, vimos que $t = 56$ trouxe maior ganho para o *RSI* isolado, e $t = 56, 28$ o campeão.

Sendo assim vamos escolher os seguintes conjuntos de indicadores técnicos para testar no restante das criptmoedas:

$$\{(RSI_{56} + RSI_{28}), (RSI_{56} + ESTOC_{56}), (RSI_{56})\}$$

		MSE	Mov. Acc.	UpDown Acc.
ETH	$RSI_{56} + RSI_{28}$	8,695e-4	75,00%	77,42%
	RSI_{56}	8,695e-4	74,94%	77,36%
	$RSI_{56} + ESTOC_{56}$	8,784e-4	74,52%	77,13%
	$Nenhum$	8,62 e-4	75,44%	75,44%
TRX	RSI_{56}	9,074e-4	74,66%	77,71%
	$RSI_{56} + RSI_{28}$	9,090e-4	74,84%	77,62%
	$RSI_{56} + ESTOC_{56}$	9,095e-4	73,95%	77,30%
	$Nenhum$	8,86 e-4	75,42%	78,42%
EOS	$RSI_{56} + RSI_{28}$	9,114e-4	74,67%	73,95%
	RSI_{56}	9,191e-4	74,53%	73,95%
	$RSI_{56} + ESTOC_{56}$	9,341e-4	73,95%	73,99%
	$Nenhum$	9,05 e-4	74,88%	74,44%

Tabela 4.11: *Performance (média) ao usar de Indicadores Técnicos - restante das moedas*

subtabelas ordenadas pelo MSE (não necessariamente representa melhor performance)

CAPÍTULO 5

Conclusões

Ao Utilizar a **variação (%) do preço** ao invés do preço em sí, conseguimos generalizar a informação do comportamento do preço, resultando nas seguintes observações:

- Preços muito pequenos e muito altos são convertidos para o mesmo "escopo"de valores, permitindo que o modelo aproveite todo o conjunto de dados e se adeque mais facilmente a todo ele, consequentemente aumentando a performance/acurácia.
- Junto da observação acima, preços de criptomoedas diferentes são trazidos para o mesmo "escopo"também, permitindo que o modelo saiba como se comportar com dados de criptomoedas nunca vistas antes (de volatilidade similar ou não-similar), tendo uma diferença de performance pequena.

Sendo assim, o Modelo Genérico se mostra capaz de **DomainShift** (capacidade do modelo de funcionar em outro domínio), problema conhecido pela sua dificuldade. Entretanto vale notar que isso pode ter sido alcançado apenas porque criptomoedas, em geral, possuem correlação considerável entre sí.

Também observamos que, dentre os dados sociais, *volume de tweets* e *google trends* melhoraram a performance, sendo o último quem proveu o maior melhora. Entretanto o ganho foi **mínimo/irrelevante**, de apenas +0,26% para *MovementDirection Accuracy* e +0,12% para *PriceUpDown Accuracy*.
Já o uso de indicadores técnicos, bem conhecidos por investidores do Mercado de Ações, provocou um pequeno aumento relevante na performance em todas as métricas para moedas mais estáveis (*Bitcoin* e *Ethereum*), mas apenas gerou ruído e perca de performance para as *Altcoins* (*Tron* e *EOS*).

5.1 Comparação com outros trabalhos

O problema da discrepância de escala dos preços entre o inicio e fim das séries temporais (**data drift**), é contornado por alguns trabalhos utilizando técnicas como

rolling window [Nelson, Pereira e Oliveira 2017](para cada dia, uma nova rede neural é treinada com apenas os últimos N dias), ou trabalhando com granularidade menores que a diária [Caux, Bernardini e Viterbo 2020] (permitindo um grande dataset de apenas preços recentes). Mas ainda devem abrir mão de parte dos dados.
Enquanto no nosso trabalho, o uso da variação de preço elimina este problema mudando a forma de representar informação, e consequentemente permite (ou facilita) o aproveitamento de grandes datasets de longos períodos.

O trabalho de [Jaquart, Dann e Weinhardt 2021], que usa classificação binária e granularidade de minutos, tenta prever o Bitcoin usando LSTM e obteve melhor acurácia de 60% (para previsões de 60 min.), pior de 51% (para previsões de 1min.) e média de 56%. Enquanto nosso trabalho, numa comparação indireta (pois trabalhamos com granularidade diária), obteve acurácia média de 75,55% sem features adicionais, e 77,80% com indicadores técnicos.

5.2 Futuros trabalhos

Apesar de não termos conseguido obter melhoras significativas com o uso de features sociais acredita-se que, dentre todas, as **features de Calendário de Eventos** (extraídas do coinmarketcal.com) possuem potencial como indicadores, contendo informações não presentes em outros dados sociais. Algumas sugestões pensadas foram:

1. **Filtrar "eventos ruído"**: existem muitos eventos que não possuem relevância alguma para o preço da criptomoeda, trazendo grande ruído ao conjunto de dados, dado que a série temporal de eventos é esparsa.
2. **Categorizar eventos**: pode ser melhor tornar a feature qualitativa (classificar) ao invés de quantitativa. Isso ajudaria a rede entender que tipo de eventos são relevantes ou não (por exemplo listagem da criptomoeda em uma corretora, ou lançamento de nova versão de sua blockchain).
3. **Aplicar para *Altcoins* apenas**: *Altcoins* possuem maior volatilidade, contém mais eventos e estão mais sujeitas à influência de divulgação eventos relacionados, por serem mais recentes e ativas com a comunidade.

Uma extensão e aperfeiçoamento deste trabalho também é possível, pois ele focou mais em provar que é possível criar um Modelo Genérico para Previsão de Criptomoedas, e testar algumas features extras junto da **variação do preço** (ao invés do preço, como normalmente encontrado nos outros trabalhos).
Sendo assim, para possíveis continuações deste trabalho:

1. Indicadores técnicos **também** no Modelo Genérico: aplicar os indicadores técnicos selecionados (e outros novos que mostrarem melhoria na performance) para treino e validação cruzada do Modelo Genérico (isto é, treinar e validar com moedas distintas, usando indicadores técnicos);
2. Treino cruzado do Modelo Genérico: mesclar dados de diferentes moedas (possível graças à representação por variação de preço), gerando uma base de dados maior, e alterar a arquitetura da Rede Neural para tomar proveito da quantidade de dados;
3. Testar com outras granularidades: testar a capacidade do Modelo Genérico com granularidades de dados diferentes (minuto(s), hora(s), etc);
4. Testar com o Mercado de Ações: o mercado de criptomoedas e de ações compartilham muitas similaridades. Pode ser que

Todo o código, dados, gráficos e resultados encontram-se disponíveis em https://github.com/jgabriel98/TCC

Referências Bibliográficas

[Abraham, Jethin; Higdon, Daniel; Nelson, John; e Ibarra, Juan 2018]Abraham, Jethin; Higdon, Daniel; Nelson, John; e Ibarra, Juan. Cryptocurrency Price Prediction Using Tweet Volumes and Sentiment Analysis. 2018. Disponível em: <https://scholar.smu.edu/datasciencereview/vol1/iss3/1>.

[Caux, Bernardini e Viterbo 2020]CAUX, M. de; BERNARDINI, F.; VITERBO, J. Short-term forecasting in bitcoin time series using lstm and gru rnns. In: . [S.l.: s.n.], 2020. p. 97–104.

[Felizardo et al. 2019]FELIZARDO, L. et al. Comparative study of bitcoin price prediction using wavenets, recurrent neural networks and other machine learning methods. In: . [S.l.: s.n.], 2019. p. 1–6.

[Hochreiter e Schmidhuber 1997]HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, v. 9, p. 1735–80, 12 1997.

[Hwang 2018]HWANG, T. *Computational Power and the Social Impact of Artificial Intelligence*. 2018.

[Jaquart, Dann e Weinhardt 2021]JAQUART, P.; DANN, D.; WEINHARDT, C. Short-term bitcoin market prediction via machine learning. *The Journal of Finance and Data Science*, v. 7, p. 45–66, 2021. ISSN 2405-9188. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2405918821000027>.

[Ming et al. 2017]MING, Y. et al. Understanding hidden memories of recurrent neural networks. 10 2017.

[Nelson, Pereira e Oliveira 2017]NELSON, D.; PEREIRA, A.; OLIVEIRA, R. de. Stock market's price movement prediction with lstm neural networks. In: . [S.l.: s.n.], 2017. p. 1419–1426.

[Olah 2015]OLAH, C. *Understanding LSTM Networks*. 2015. Disponível em: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.

[Rajkumar 2021]RAJKUMAR, S. *Cryptocurrency Historical Prices*. 02 2021. Disponível em: <https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory>.

[Tandon et al. 2019]TANDON, S. et al. Bitcoin price forecasting using lstm and 10-fold cross validation. In: . [S.l.: s.n.], 2019. p. 323–328.

[TSENG 2019]TSENG, Q. *Reconstruct Google Trends Daily Data for Extended Period*. 2019. Disponível em: <https://towardsdatascience.com/reconstruct-google-trends-daily-data-for-extended-period-75b6ca1d3420>.

[Weng, Ahmed e Megahed 2017]WENG, B.; AHMED, M.; MEGAHED, F. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*, v. 79, 02 2017.

APÊNDICE A

Apêndice

Resultados e estatísticas

- Comparativo pré-testes - preço "crú" VS variação do preço:
  - estatisticas em: `./src/LSTM_BTC_raw_vs_variation.ipynb`
  - plots (graficos) das predições: `./src/LSTM.ipynb` está mais para o inicio, acaba antes da metade
- Modelos genéricos:
  - Plots dos resultados: `./resultados/generic_model/<moeda baseTreino> trained - <moeda alvoTeste> forecasting.png`
  - Estatísticas dos resultados: `./src/LSTM_GenericModel_<moeda baseTreino>-<moeda alvoTeste>.ipynb`
- Features Adicionais:
  - Testes com indicadores técnicos (Bitcoin): `./src/LSTM_technicalFeatures.ipynb`
  - Teste com indicadores técnidos (restante das moedas): `./src/LSTM_technicalFeatures_with_altcoins.ipynb`
  - Testes com dados sociais: `./src/LSTM.ipynb` começe olhando pelo final, pois tem muita coisa antes

Dados

- dados do kaggle: `./data/kaggle - Cryptocurrency Historical Prices/coin_<moeda>.csv`
- dados sociais: `./data/social_data<moeda>.csv` . Esses arquivos são gerados quando a função `load_data()` (encontrada em `src/data/loades/utils.py`) é chamada e o arquivo de dados sociais desta moeda não existe (é preciso ter o selenium instalado e funcionando nesse caso, e algumas outras dependências que não lembro mais).

código fonte

o código fonte se encontra na pasta `./src`

- `./src/data` contém arquivos relacionados a obtenção e tratamento de dados.
  `→ para tratamento de dados `utils.py` é o arquivo mais relevante.
- `./src/data/loaders` contém as classes para extração de dados sociais.
- `./src/metrics/custom.py` contém as funções das métricas, as relevantes são: `mean_squared_error()` , `custom_movement_accuracy()` e `above_or_below_zero_accuracy()`

Figura A.1: *Instruções de boas vindas do repositório*
Imagem tirada no dia 12/06/2021
repositório encontrado em https://github.com/jgabriel98/TCC

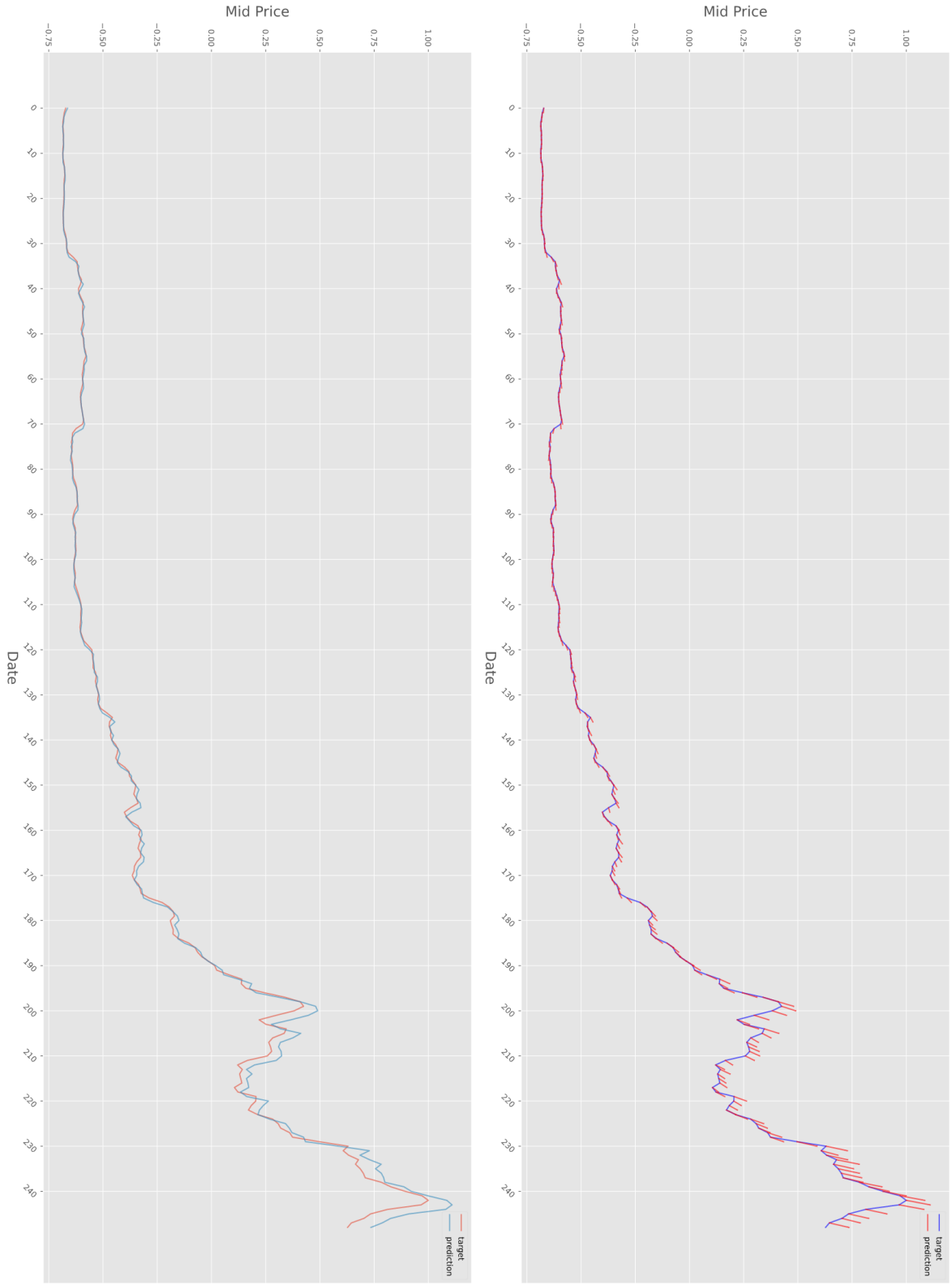

Figura A.2: *Previsão preço bitcoin. treino 90% - teste 10%, treinado com preço e variação de preço*
Sem K-Fold

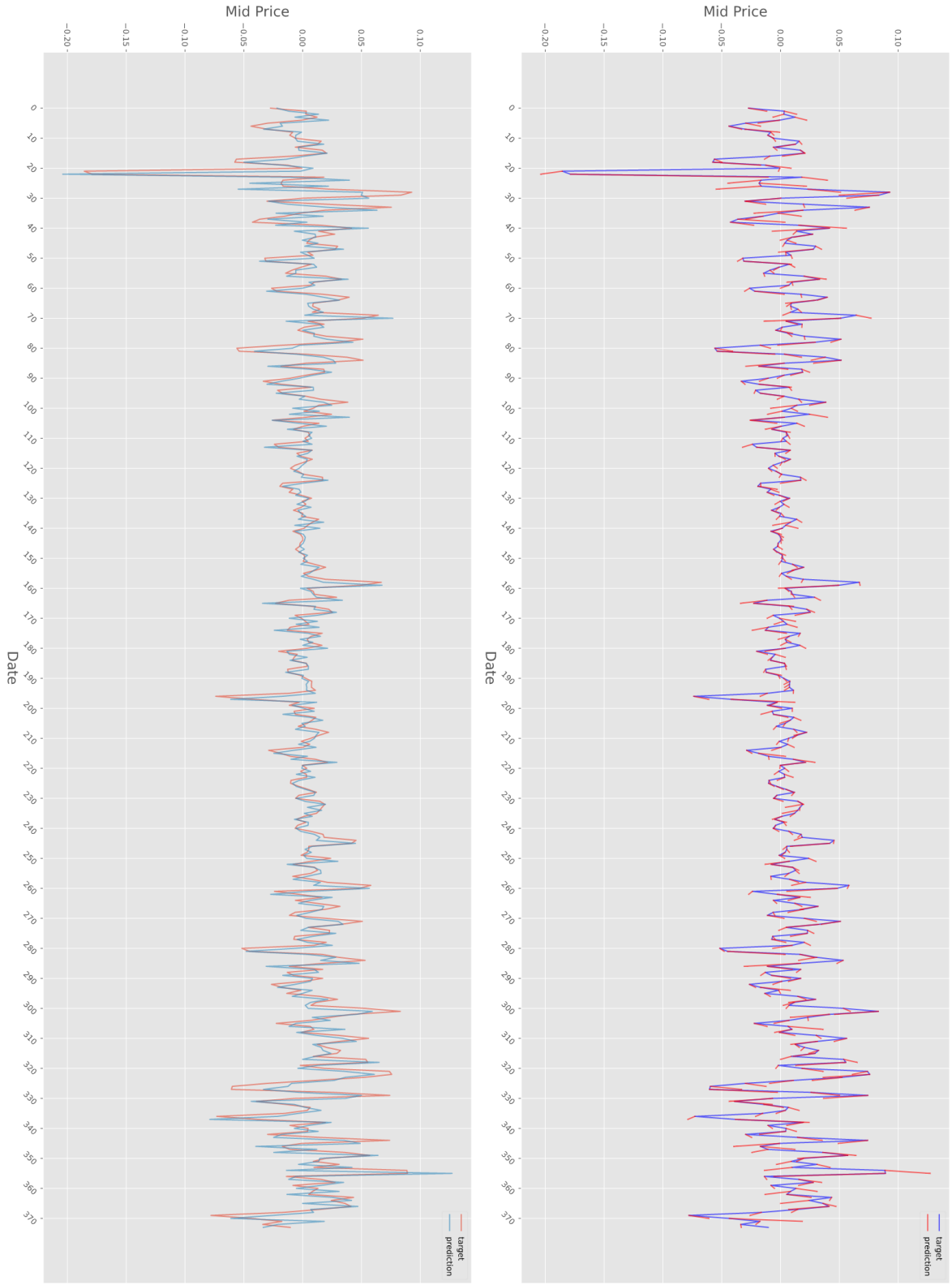

Figura A.3: *Previsão variação do preço bitcoin. treino 85% - teste 15%, treinado com variação de preço*
Sem K-Fold