isec
Politécnico de Coimbra

Inteligência Computacional

Docente
Inês Dominguês
Carlos Pereira

Alunos
Paulo Henrique Figueira Pestana de Gouveia - a2020121705
Nuno Alexandre Almeida Santos - a2019110035

November 14, 2022

Índice

1 Introdução

Este trabalho foi realizado no âmbito da Unidade Curricular de Inteligência Computacional, tem por objetivo treinar uma rede neuronal capaz de estimar o valor da Bitcoin num determinado minuto.

2 Descrição do caso de estudo e objetivos do problema

O Dataset escolhido foi Bitcoin Price USD, neste conjunto de dados os dados são gerados no intervalo de 1 minuto por uma API (Binance API) entre 1 de janeiro de 2021 a 12 de Maio de 2021. Inclui várias colunas que mostram a mudança real no preço da Bitcoin também mostra o preço Open, High, Low, Close da Bitcoin em minutos específicos.

- Features
  1. Preço de abertura num minuto específico (Open Price of particular minute);
  2. Preço alto num minuto específico (High Price of particular minute);
  3. Preço baixo num minuto específico (Low Price of particular minute);
  4. Fechar Preço num minuto específico (Close Price of particular minute);
  5. Volume total num minuto específico (Total volume of particular minute);
  6. Volume de ativos de cotação (Quote asset volume);
  7. Número de negócios para determinado minuto (Number of trades for particular minute);
  8. Volume de ativos base de compra do tomador (Taker buy base asset volume);
  9. Volume de ativos de cotação de compra do tomador (Taker buy quote asset volume).
- Exemplos: 188318

Como o Horário de abertura (Open Time) e o Horário de fecho (Close Time) são sempre iguais, retirámos essas colunas das features.

3 Descrição da implementação dos algoritmos

Para treinar o nosso modelo optámos por usar o PyTorch que é uma biblioteca Python de aprendizagem máquina de código aberto. A principal diferença entre PyTorch e TensorFlow é a escolha entre simplicidade e desempenho: o PyTorch é mais fácil de aprender (especialmente para programadores Python), enquanto o TensorFlow tem uma curva de aprendizagem, mas tem um desempenho melhor e é mais utilizado na comunidade de desenvolvedores.

Os algoritmos de treino(métricas de avaliação) usádos foram:

- MSE (Erro quadrado médio)

$$MSE : \sum_{i=1}^{D} (x_i - y_i)^2 \tag{1}$$

- MAE (Erro absoluto médio)

$$MAE : \sum_{i=1}^{D} |x_i - y_i| \tag{2}$$

- RMSE (Raíz quadrada do erro médio)

$$RMSE : \sqrt{\frac{1}{n} \Sigma_{i=1}^{n} \Big(\frac{d_i - f_i}{\sigma_i} \Big)^2} \tag{3}$$

- RSQUARED (Coeficiente de determinação)

$$RSQUARED : 1 - \frac{SS_{res}}{SS_{tot}} \tag{4}$$

Funções de ativação usadas:

- Softmax

$$Softmax(x) = \frac{\exp(x_i)}{\sum_j (x_j)} \tag{5}$$

- RReLU

$$RReLU(x) = \begin{cases} x, & \text{if } x \geq 0 \\ ax, & \text{caso contrário} \end{cases} \tag{6}$$

- Sigmoid

$$Sigmoid(x) = \sigma(x) = \frac{1}{1 + \exp(-x)} \tag{7}$$

Treinámos 4 redes com as diferentes funções de treino para testar qual teria menor erro:

1. Rede MSE
   - Nome - network MSE 2Lay 256 Tanh 8 94e 05.tar
   - Função de Treino - MSE
   - Função de Ativação - Tanh
   - Nº de Neurónios - 256
   - Nº de Camadas - 2
   - Learning Rate - 3e-4
   - Erro de teste - 8,94E-05

2. Rede RMSE
   - Nome - network RMSE 2Lay 256 Tanh 0 00228.tar
   - Função de Treino - RMSE
   - Função de Ativação - Tanh
   - Nº de Neuronios - 256
   - Nº de Camadas - 2
   - Learning Rate - 3e-4
   - Erro de teste - 0,0028

3. Rede MAE
    - Nome - network MAE 2Lay 256 Tanh 0 00173 .tar
    - Função de Treino - MAE
    - Função de Ativação - Tanh
    - Nº de Neuronios - 256
    - Nº de Camadas - 2
    - Learning Rate - 3e-4
    - Erro de teste - 0,00173

4. Rede RSQUARED
    - Nome - network R2 2Lay 256 Tanh 0 062.tar
    - Função de Treino - RSquared
    - Função de Ativação - Tanh
    - Nº de Neurónios - 256
    - Nº de Camadas - 2
    - Learning Rate - 3e-4
    - Erro de teste - 0,062

A melhor rede obtida na primeira fase, alterando apenas as funções de treino foi a rede network MSE 2Lay 256 Tanh 8 94e 05.tar (Rede MSE) cujo a função de treino foi a MSE (Erro quadrado médio).

Foram criadas novas 5 redes cujo a função de treino é a função MSE pois foi a função que obtivemos melhor resultados, nestas novas 5 redes irão variar vários parâmetros tais como a função de ativação, número de neurónios e número de camadas.

1. Rede MSE (Softmax para função de ativação)
    - Nome - network MSE 2Lay 256 Softmax 0 065.tar
    - Função de Treino - MSE
    - Função de Ativação - Softmax
    - Nº de Neurónios - 256
    - Nº de Camadas - 2
    - Learning Rate - 3e-4
    - Erro de teste - 0,065

2. Rede MSE (RReLU para função de ativação)
    - Nome - network MSE 2Lay 256 RReLU 0 0019.tar
    - Função de Treino - MSE
    - Função de Ativação - RReLU
    - Nº de Neurónios - 256
    - Nº de Camadas - 2
    - Learning Rate - 3e-4
    - Erro de teste - 0,0019

3. Rede MSE (Sigmoid para função de ativação)
    - Nome - network MSE 2Lay 256 Sigmoid 0 0348.tar
    - Função de Treino - MSE
    - Função de Ativação - Sigmoid
    - Nº de Neurónios - 256
    - Nº de Camadas - 2
    - Learning Rate - 3e-4
    - Erro de teste - 0,0348

4. Rede MSE(com uma camada):
    - Nome - network MSE 1Lay 256 Tanh 9 04E 05.tar
    - Função de Treino - MSE
    - Função de Ativação - Tanh
    - Nº de neuronios - 256
    - Nº de camadas - 1
    - Learning rate - 3e-4
    - Erro de teste - 9,04E-05

5. Rede MSE(com 50 neuronios):
    - Nome - network MSE 2Lay 50 Tanh 0 062.tar
    - Função de Treino - MSE
    - Função de Ativação - Tanh
    - Nº de neuronios - 50
    - Nº de camadas - 2
    - Learning rate - 3e-4
    - Erro de teste - 0,062

4 Análise de Resultados

4.1 Diferença nas funções de ativação(métricas de avaliação)

MSE é uma função diferenciável que facilita a execução de operações matemáticas em comparação com uma função não diferenciável como MAE. Portanto, em muitos modelos, o RMSE é usado como métrica padrão para calcular a Função de Perda, apesar de ser mais difícil de interpretar do que o MAE.

O menor valor de MAE, MSE e RMSE implica em maior precisão de um modelo de regressão. No entanto, um valor mais alto de RSquared é considerado desejável.

Para comparar a precisão entre diferentes modelos de regressão linear, RMSE é uma escolha melhor do que R Squared.

4.2 Diferença nas funções de ativação

Como podemos visualizar nos resultados acima, o função Tanh continua a ter a melhor performance, isto tudo tem haver da maneira como foram normalizados os dados.

Os nossos dados estão normalizados entre -1 e 1, sendo que a Softmax e a Sigmoid limitam o output entre 0 e 1, o que não ajuda para a performance. Para RReLU aplica a função de unidade de revestimento retificado com vazamento aleatório, elemento a elemento, podendo obter melhor resultados se o seu lower e upperbound foram bem definidos.

4.3 Diferença nas configurações da rede

Sendo um modelo linear, precisamos apenas de uma camada para rede, sendo que o aumento de camadas não afeta a performance da rede que é o que acontece nos nossos testes.

Enquanto que na diminuição da quantidade de neurónios podemos ver que a nossa performance da rede piora, pois que quantos mais neurónios para a nossa rede melhor ela será ao custo de um aumento do tempo de treino.

5 Conclusões

Podemos concluir que para este caso a melhor métrica de avaliação é a MSE com a função de ativação adequada para a maneira como foram normalizados os dados.

Ao testar com valores esta rede dá valores favoráveis, que infelizmente não podem ser levados para uso de ganhos financeiros no mercado da criptomoeda devido ao facto que a nossa rede não tem dados para prever fatores externos que possam causar por exemplo um crash no mercado.

Combinamos o interesse ao tópico das criptomoedas hoje em dia, para aprofundar o nossa compreensão no tópico de criar redes neuronais para previsão de valores de um modelo de regressão.

6 Referências

[1] BINANCE BinanceDataset.

[2] Pytorch.

[3] Pytorch.nn.