

Usando notícias para prever o movimento de ações

Felipe Lana Machado¹

¹Engenharia de Software – Universidade do Estado de Santa Catarina (UDESC)
R. Dr. Getúlio Vargas, 2822 - Bela Vista, Ibirama - SC, 89140-000

Abstract. *This article presents the initial proposal of the course completion project of Udacity Machine Learning Engineer. The first is a reference of the subject, followed by an initial analysis of the data and the detailed description of the problem. Finally, the evaluation metrics are discussed and a solution is proposed for the problem, which is fully described in the final session.*

Resumo. *Este artigo apresenta a proposta inicial do projeto de conclusão de curso de Engenheiro de Machine Learning da Udacity. Sendo que primeiro é feita uma referência do assunto, seguido por uma análise inicial dos dados e a descrição detalhada do problema. Por fim, são discutidas as métricas de avaliação e é proposta uma solução para o problema, sendo esta completamente descrita na sessão final.*

1. Histórico do Assunto

O mercado financeiro sempre foi alvo de muitos estudos, sempre houveram diversas formas de se analisar o mercado afim de tomar a melhor decisão para obter o melhor rendimento. Muitos acreditam que o mercado é influenciado por notícias e acontecimentos que ocorrem diariamente, essas notícias podem ser tanto positivas ou negativas quanto neutras para um determinado ativo na bolsa de valores.

Com o aumento da performance dos hardwares, conseguindo processamentos cada vez mais velozes por um menor custo, aliado a um volume de dados crescente e cada vez de melhor qualidade, a área de pesquisa do aprendizado de máquina tem cada vez mais se tornado central na tentativa de melhorar o desempenho em praticamente todas as atividades lucrativas.

Desta forma, aliando o crescente volume de notícias relevantes que possam impactar o mercado financeiro com o aumento da capacidade de processamento dos computadores, tem-se como objetivo encontrar um modelo que permita verificar quais notícias são relevantes para prever o movimento das ações. E com isso conseguir prever o movimento do mercado financeiro utilizando notícias.

2. Descrição do problema

Através da competição do Kaggle [Kaggle 2018] que define o objetivo prever como as ações irão mudar baseados no estado do mercado e novas notícias. Para isso, é necessário respeitar algumas regras como não utilizar dados futuros para realizar previsões do dia atual de treinamento. Ou seja, não se pode utilizar dados futuros no treinamento para prever algo no "passado".

Outras regras também foram impostas a competição a fim de minimizar as discrepâncias dos resultados, assim como tentar excluir resultados obtidos de forma ilegal

para o projeto. Assim, para o desenvolvimento do projeto tem como restrições o tempo de execução do algoritmo que não pode ser superior a 6 horas, além disso não se pode utilizar a GPU para o processamento e nem possuir conexão com a internet, para que não sejam utilizados dados fora os disponibilizados. Também, não poderá passar de 16Gb consumidos durante a execução para que seja padronizado os recursos disponíveis entre os competidores.

3. Conjuntos de dados e entradas

Existem duas fontes de dados neste problema, a primeira são os dados do mercado financeiro (a partir de 2007) que é provida pelo Intrinio [Intrinio: Data Intelligence, on demand 2018]. Estes dados contém informações financeiras como preço de abertura, preço de fechamento, volume negociado, retorno calculado, etc. Enquanto a outra fonte de dados se refere as notícias (a partir de 2007), contém informações como artigos e alertas publicados sobre os ativos, detalhes do artigo, sentimento e outros comentários. Estes dados são fornecidos pela Thomson Reuters [Thomson Reuters 2017].

Os dados do mercado financeiro incluem um conjunto de ativos americanos, mostrando variações diárias. Por ser uma competição do Kaggle [Kaggle 2018], o autor informa que poderão haver "buracos" nos dados e que apesar destes dados algumas vezes existirem, estes não poderão ser utilizados pois não estará de acordo com o critério de seleção dos dados escolhido.

- `time(datetime64[ns, UTC])` - o horário corrente (nos dados do mercado, todas as linhas foram obtidas às 22:00 UTC)
- `assetCode(object)` - id único do ativo
- `assetName(category)` - o nome que representa um grupo de códigos de ativos. Pode ser desconhecido ("Unknown") se o código do ativo correspondente não tiver em nenhuma linha nos dados de notícias.
- `universe(float64)` - um booleano indicando se o ativo, naquele dia, será incluído na pontuação ou não. O universo de ativos disponíveis em um determinado dia, são os ativos que estão disponíveis para transações. O universo de transações muda diariamente.
- `volume(float64)` - volume de transações em participações para o dia.
- `close(float64)` - o preço de fechamento para o dia (splits e dividendos não estão ajustados)
- `open(float64)` - o preço de abertura no dia (splits e dividendos não estão ajustados)
- `returnsClosePrevRaw1(float64)` - veja a explicação a baixo
- `returnsOpenPrevRaw1(float64)` - veja a explicação a baixo
- `returnsClosePrevMktres1(float64)` - veja a explicação a baixo
- `returnsOpenPrevMktres1(float64)` - veja a explicação a baixo
- `returnsClosePrevRaw10(float64)` - veja a explicação a baixo
- `returnsOpenPrevRaw10(float64)` - veja a explicação a baixo
- `returnsClosePrevMktres10(float64)` - veja a explicação a baixo
- `returnsOpenPrevMktres10(float64)` - veja a explicação a baixo

- `returnsOpenNextMktres10(float64)` - 10 dias, retorno-rezidualizado do mercado. Esta é a variável de alvo utilizada como pontuação na competição proposta. Os dados do mercado financeiro foram filtrados de forma com que `returnsOpenNextMktres10` não seja nula jamais.

Quanto aos dados referentes as notícias, temos que os dados não estão normalizados propositalmente.

- `time(datetime64[ns, UTC])` - UTC horário que a notícia ficou disponível no feed (precisão de segundos)
- `sourceTimestamp(datetime64[ns, UTC])` - UTC horário em que foi criado o item de notícia
- `firstCreated(datetime64[ns, UTC])` - UTC horário da primeira versão do item
- `sourceId(object)` - um id para cada item de notícia
- `headline(object)` - o título do item
- `urgency(int8)` - tipo de história (1: alerta, 3: artigo)
- `takeSequence(int16)` - o número da sequência para cada item de notícia, começando em 1. Para cada história, alertas e artigos tem sequencias separadas.
- `provider(category)` - identificação da organização que providenciou os itens de notícia (RTRS para Reuters News e BSW para Business Wire)
- `subjects(category)` - códigos de tópicos ou companhias que estão relacionadas ao item de notícia. Os tópicos em códigos descrevem o assunto do item de notícia, e podem cobrir classes, geografias, eventos, industrias/setores e outros tipos
- `audiences(category)` - identifica qual (is) produto (s) de notícias do desktop o item de notícias pertence. Eles são tipicamente adaptados para públicos específicos. (ex."M"para Money International News Service e "FB"para French General News Service)
- `bodySize(int32)` - o tamanho da atual história em caracteres
- `companyCount(int8)` - o número de companhias explicitamente listados nos itens de notícia nos campos de assunto
- `headlineTag(object)` - o Thomson Reuters título marcado para o item de notícia
- `marketCommentary(bool)` - booleano indicador que o item esta discutindo as condições gerais do mercado, como "After the Bell"resumo.
- `sentenceCount(int16)` - o número total de sentenças no item de notícia. Pode ser utilizado em conjunto com `firstMentionSentence` para determinar a posição da primeira menção no item.
- `wordCount(int32)` - o número total de palavras e pontuações no item de notícia.
- `assetCodes(category)` - lista de ativos mencionados na notícia
- `assetName(category)` - nome dos ativos
- `firstMentionSentence(int16)` - a primeira frase, começando pelo título.
 - 1: título
 - 2: primeira frase no corpo
 - 3: segunda frase no corpo
 - 0: o ativo sendo avaliado não foi encontrado no titulo ou no corpo do texto. Assim, a notícia inteira será usada para avaliar o valor do sentimento.

- `relevance(float32)` - um valor decimal (0 a 1) indicando a relevância do item de notícia para o ativo financeiro. Se o ativo for mencionado no título, a relevância é 1. Quando for um alerta, a relevância deverá ser obtida através do `firstMentionSentence`.
- `sentimentClass(int8)` - indica a predominância do sentimento para esse item de notícia em relação ao ativo financeiro. A classe indicada é a que representa a maior probabilidade.
- `sentimentNegative(float32)` - probabilidade do sentimento no item de notícia ser negativo para o ativo.
- `sentimentNeutral(float32)` - probabilidade do sentimento no item de notícia ser neutro para o ativo.
- `sentimentPositive(float32)` - probabilidade do sentimento no item de notícia ser positivo para o ativo.
- `sentimentWordCount(int32)` - número de palavras ou pontuações que são relevantes para o ativo. Isso pode ser utilizado juntamente com o `wordCount` para verificar a proporção da notícia que está discutindo o ativo.
- `noveltyCount12H(int16)` - A novidade de 12 horas do conteúdo de um item de notícias em um recurso específico. É calculado comparando-o com o texto específico do ativo em um cache de itens de notícias anteriores que contêm o ativo.
- `noveltyCount24H(int16)` - o mesmo que acima, mas para 24 horas
- `noveltyCount3D(int16)` - o mesmo que acima, mas para 3 dias
- `noveltyCount5D(int16)` - o mesmo que acima, mas para 5 dias
- `noveltyCount7D(int16)` - o mesmo que acima, mas para 7 dias
- `volumeCounts12H(int16)` - o volume de notícias de 12 horas para cada ativo. Um cache de notícias anteriores é mantido e o número de itens de notícias que mencionam o ativo dentro de cada um dos cinco períodos históricos é calculado.
- `volumeCounts24H(int16)` - o mesmo que acima, mas para 24 horas
- `volumeCounts3D(int16)` - o mesmo que acima, mas para 3 dias
- `volumeCounts5D(int16)` - o mesmo que acima, mas para 5 dias
- `volumeCounts7D(int16)` - o mesmo que acima, mas para 7 dias

4. Descrição da solução

O primeiro a ser realizado para se chegar a uma solução será realizar um processamento nos dados para que os mesmos sejam normalizados e com isso sejam removidos dados incorretos (faltantes), colocados em escala equivalente e outras coisas. Desta forma, teremos os dados prontos para serem trabalhados.

Como solução do problema, será proposta uma rede neural recorrente(RNN), chamada de LSTM (Long short-term memory) [Wikipedia 2018]. As redes LSTM são boas quando se deseja classificar, processar e realizar previsões baseadas em dados contínuos (como é o caso dos dados do mercado financeiro).

Ao final será gerado um arquivo com o seguinte formato: "time,assetCode,confidenceValue"sendo que o valor `confidenceValue` será entre 1 e -1, com 1 sendo esperado um grande aumento no ativo e -1 uma grande queda, para os próximos 10 dias.

5. Modelo de referência (benchmark)

Por ser um problema específico, com dados específicos, o melhor modelo de referência será um trabalho disponível desta mesma competição, intitulado de "Tuning hyper-params in LGBM achieve 0.66 in LB".

Porém, por ser uma competição, a referência utilizada será o ranking disponibilizado pelo Kaggle [Kaggle 2018]. Ou seja, o resultado será comparado com os melhores resultados obtidos na competição até o momento de entrega do projeto final. Sendo que neste momento, a melhor pontuação é 0.75022.

6. Métricas de avaliação

Para esta competição do Kaggle [Kaggle 2018], o proposto é que se preveja um valor de confiança que é multiplicado pelo retorno ajustado do mercado financeiro de um determinado ativo em uma janela de 10 dias. Para cada dia no período de avaliação calcula-se:

$$x_t = \sum_i \hat{y}_{ti} r_{ti} u_{ti},$$

onde r_{ti} é o retorno inicial de dez dias ajustado pelo mercado para o dia t para o instrumento i e u_{ti} é uma variável do universo 0/1 (consulte a descrição dos dados para obter detalhes) que controla se um determinado ativo está incluído na pontuação em um determinado dia .

A pontuação é calculada como a média dividida pelo desvio padrão de seus valores x_t diários.

$$score = \bar{x}_t / \sigma(x_t)$$

7. Design do projeto

7.1. Análise de dados exploratória

Primeiramente, será realizada uma análise exploratória para a familiarização dos dados existentes. Essa análise terá como objetivo identificar alguns padrões e verificar a existência de dados sujos. Ou seja, serão verificados dados inexistentes, dados incorretos, dados discrepantes e outros.

Para a realização da análise, serão utilizados gráficos e visualizações para maior entendimento dos dados. Apresentando de forma clara como os dados estão organizados e como isso afetará a solução de forma geral.

7.2. Pré-processamento dos dados

Após realizada a análise exploratória, neste momento serão ajustados os dados de forma a melhorar a consistência dos mesmos. Serão normalizados os dados para não haver discrepância da realidade, serão eliminados ou corrigidos dados incorretos, serão categorizados corretamente os dados sem categorias.

Por fim, serão novamente visualizados os dados através de gráficos para verificar se foi correto o pré-processamento dos dados afim de manter a melhor amostra possível para se realizar as previsões.

7.3. Treinamento da rede neural

Para o treinamento da rede neural, será utilizado a princípio um rede neural recorrente. As redes neurais recorrentes são redes que possuem conexões ponderadas dentro de uma camada [Christopher Olah 2015]. Ou seja, diferente das redes de feed-forward tradicionais em que a informação flui apenas para frente, as redes recorrentes possuem loops entre si (Figura 1). Podendo armazenar informações ao processar novas entradas, tornando ideal para séries temporais em que as entradas anteriores devem ser consideradas.

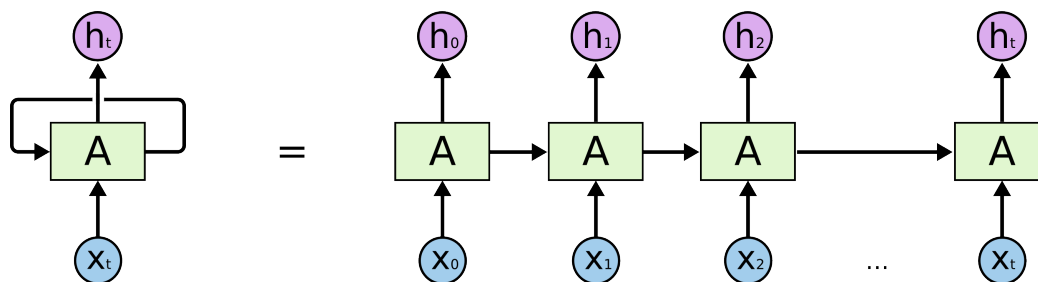


Figura 1. An unrolled recurrent neural network.

Para este problema também poderiam ser utilizadas inúmeras arquiteturas de redes recorrentes, porém foi optada uma rede recorrente bem conhecida chamada de LSTM (Long Short Term Memory) que é um tipo especial das redes neurais recorrentes. Elas foram introduzidas por Hochreiter Schmidhuber (1997) e foram melhoradas com o tempo por diversos autores, sendo utilizadas em uma grande variedade de problemas.

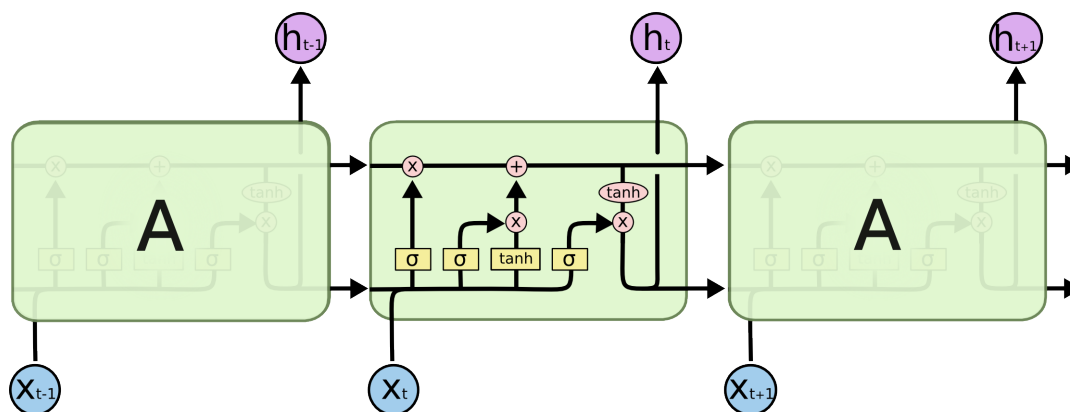


Figura 2. The repeating module in an LSTM contains four interacting layers.

A grande vantagem de se utilizar as LSTM's é que essas redes possuem a capacidade de adicionar ou remover informações nas células de estado, de forma bem regulada pelas estruturas chamadas de portões. Sendo que as LSTM possuem três destes portões para controlar as células de estado.

7.4. Calibração do modelo

Para realizar a calibração do modelo escolhido, escolhendo os melhores hiper-parâmetros para obter o melhor resultado, serão utilizados diversos parâmetros que são importantes nas LSTM-RNN, como taxa de aprendizado, taxa de abandono, "epochs" e número de neurônios.

Uma das formas mais conhecidas e utilizadas para realizar a calibração é a pesquisa de grade [Jason Brownlee 2016]. Nela você insere como argumento um mapeamento do nome do parâmetro e uma lista de possíveis valores para serem testados. Estes testes são com base na acurácia das respostas, porém pode-se definir uma nova forma de calcular as pontuações.

Essas calibrações normalmente são bem demoradas por se tratar de um sistema de tentativa e erro, assim como o problema define claramente um limite de tempo e memória de execução, será optado pela utilização de apenas uma parte dos dados ao invés de utilizar todo o conjunto de dados disponíveis.

7.5. Discussão sobre os resultados

Por fim, os resultados obtidos serão comparados com o modelo de referência, bem como a pontuação atual na competição do Kaggle [Kaggle 2018]. Sendo que serão discutidos os motivos para obtenção dos resultados, principais fatores que foram observados e será analisada outras possíveis soluções para o problema.

Observando o modelo de referência temos que a solução proposta por eles utiliza um método que agrupa diversos outros modelos para chegar a um resultado. Este modelo tem sido utilizado nas competições do Kaggle [Kaggle 2018] e tem se mostrado bastante promissor, porém a intenção é bater este método com o modelo proposto neste projeto.

Referências

- Christopher Olah (2015). Understanding lstm networks. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>. [Online: Acessado em 30 de Outubro de 2018].
- Intrinio: Data Intelligence, on demand (2018). Data market provider. <https://intrinio.com/>. [Online: Acessado em 30 de Outubro de 2018].
- Jason Brownlee (2016). How to grid search hyperparameters for deep learning models in python with keras. <https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/>. [Online: Acessado em 31 de Outubro de 2018].
- Kaggle (2018). Two sigma: Using news to predict stock movements. <https://www.kaggle.com/c/two-sigma-financial-news>. [Online: Acessado em 30 de Outubro de 2018].
- Thomson Reuters (2017). News data provider. <https://www.thomsonreuters.com/en.html>. [Online: Acessado em 30 de Outubro de 2018].
- Wikipedia (2018). Long short-term memory. <http://www.bioinf.jku.at/publications/older/2604.pdf>. [Online: Acessado em 30 de Outubro de 2018].