UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG

Marlon Rubio de Carvalho Franco

# Previsão da Soja futura na bolsa de Chicago (CBOT) utilizando modelo LSTM e relacionando a dados climáticos das regiões mais produtivas dos EUA.
# Universidade Federal do Rio Grande – FURG

Brasil

2019

UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG

Marlon Rubio de Carvalho Franco

# Previsão da Soja futura na bolsa de Chicago (CBOT) utilizando modelo LSTM e relacionando a dados climáticos das regiões mais produtivas dos EUA. Universidade Federal do Rio Grande – FURG

Trabalho acadêmico apresentado ao Curso de Engenharia de Computação da Universidade Federal do Rio Grande como requisito parcial para a obtenção do grau de Bacharel em Engenheira de Computação.

Orientador: Marcelo Rita Pias

Universidade Federal do Rio Grande – FURG

Centro de Ciências Computacionais

Curso de Engenharia de Computação

Brasil

2019

# Agradecimentos

À Deus.

À minha família, por todo esforço, apoio e confiança.

Aos meus mentores, pelos conhecimentos compartilhados e conselhos que recebi durante minha graduação.

# Resumo

A soja, devido ao seu alto teor de proteína e óleo, se tornou a mais importante fonte de alimento, sendo produzida em larga escala pelo mundo. O Brasil é o maior exportador, e o segundo maior produtor de soja, perdendo apenas para a produção dos EUA. No Brasil, os preços internos da soja estão fortemente atrelados ao referencial do mercado futuro (Bolsa de Chicago), mais precisamente à *Chicago Board of Trade* (CBOT), bolsa norte-americana em que é negociada a soja. A análise de dados permite a previsão deste referencial, utilizando técnicas de aprendizagem profunda (*Deep Learning*), escopo deste trabalho.

Foi criado um *dataset* denominado *DatasetMarlon*, que une dois *datasets*: o primeiro que contem o histórico diário das cotações de soja da Bolsa de Chicago desde 1959, e o segundo que contém dados climáticos dos EUA no mesmo período. Foram filtrados os dados climáticos utilizado um terceiro *dataset* da USDA (United States Department Of Agriculture) com a lista dos estados norte-americanos que mais produziram soja desde 2001. Estes dados foram visualizados utilizando a biblioteca *pandas*, em Python.

O *DatasetMarlon* foi utilizado no treinamento de um modelo de rede neural artificial LSTM, cujo objetivo é prever a cotação da soja. O desempenho do modelo foi avaliado utilizando a métrica RMSE.

**Palavras-chave**: soja. predição. LSTM. Deep Learning.

# Abstract

Soybeans, due to its high protein and oil content, has become the most important source of food, being produced on a large scale throughout the world. Brazil is the largest exporter, and the second largest soybean producer, losing only to US production.

In Brazil, domestic soybean prices are strongly tied to the benchmark of the future market (Chicago Stock Exchange), more precisely to the Chicago Board of Trade (CBOT), the American market in which soybeans are traded. The data analysis allows the forecast to be inferential, using deep learning techniques, the scope of this work.

A database was created called DatasetMarlon, which links two datasets: the first one that contains the daily history of the Chicago Stock Exchange soybean prices since 1959, and the second one that contains US weather data over the same period. The climate data were filtered using a third dataset from USDA (United States Department Of Agriculture) which list all the states that have produced the most soybeans since 2001. These data were visualized using the library Pandas in Python.

The DatasetMarlon was used in the training of an artificial neural network model LSTM, whose objective is to predict the soy quotation. The performance of the model was evaluated using the RMSE metric.

**Keywords**: soybeans. prediction. LSTM. Deep Learning.

# Lista de ilustrações

# Lista de abreviaturas e siglas

**Bushel** É uma unidade de volume utilizada nos EUA para comercializar grãos, como a soja.

**CBOT** Chicago Board of Trade, bolsa norte-americana em que é negociada a soja.

**Commodity** Termo para designar produtos utilizados no comércio que são intercambiáveis com outras mercadorias do mesmo tipo.

**DNN** Deep Neural Network, ou Rede Neural Profunda, é uma Rede Neural Artificial (ANN) com uma ou mais camadas ocultas.

**Hedge** Operação de "travamento" de preço da *commodity* para uma data futura.

**Liquidação de contrato** Finalização de um contrato.

**LSTM** Long Short-Term Memory, ou Rede de Longo Prazo de Memória (LSTM), é uma extensão para redes neurais recorrentes (RNN's), que basicamente estende sua memória, sendo adequada para aprendizado utilizando séries temporais com intervalo de tempo muito longo.

**Mercado futuro** Mercado em que os contratos são padronizados e negociados em bolsas organizadas.

**Prêmio de exportação** Diferença dos preços externos da bolsa com o preço dentro do navio no porto.

**PRCP** Dado referente a precipitação (em décimos de mm) vindo do *dataset* climático.

**RNN** Recurrent Neural Network, ou Rede Neural Recorrente, é uma família de redes neurais especializadas em análise sobre séries temporais.

**TAVG** Dado referente a temperatura média (em décimos de °C) vindo do *dataset* climático.

**TMAX** Dado referente a temperatura máxima (em décimos de °C) vindo do *dataset* climático.

**TMIN** Dado referente a temperatura mínima (em décimos de °C) vindo do *dataset* climático.

**USDA** United States Department Of Agriculture, também conhecido como o Departamento de Agricultura, é o departamento executivo federal dos EUA responsável pelo desenvolvimento e execução de leis federais relacionadas à agricultura, silvicultura e alimentos.

# Sumário

# 1 Introdução

A soja (Glycine max (L.) Merr.) (M.M.P.N.D., ) é uma das commodities agrícolas mais econômicas e valiosas devido a sua composição química única. Ela possui o maior teor de proteína (cerca de 40%) entre os cereais e leguminosas (que variam entre 20% e 30%), além de possuir a segunda maior concentração de óleo (cerca de 30%), perdendo apenas para o amendoim (48%) (LIU, 1997). Por estas características, a leguminosa se tornou a mais importante fonte de alimento, proteína e óleo, sendo cultivada em larga escala pelo mundo (PAGANO; MIRANSARI, 2016).

Segundo o relatório do mês de Novembro de 2018 da USDA (2018a), os EUA lideram a produção mundial de soja, com 125,2 milhões de toneladas produzidas na safra 2018/19; seguido pelo Brasil, com 120,5 milhões de toneladas, e pela Argentina, com 55,5 milhões de toneladas. Em exportações, o Brasil lidera a lista, com 77 milhões de toneladas exportadas; seguido pelos EUA com 51,7 milhões de toneladas, e pela Argentina com 8 milhões de toneladas. Em contrapartida, as importações são lideradas pela China, com 90 milhões de toneladas, seguida pela União Européia, com 15,8 milhões de toneladas, e pelo México, com 5 milhões de toneladas importadas. Estes dados evidenciam a importante presença do Brasil no mercado da soja, que por sua vez impulsiona o investimento em pesquisa e desenvolvimento de tecnologias que possam aumentar a competitividade do país no mercado internacional.

Historicamente, o desenvolvimento da agricultura no mundo passou por três principais fases, como apresenta Pham e Stack (2018). A primeira fase da produção agrícola foi caracterizada pela força de trabalho humano e pouca produtividade, e persistiu desde a era colonial até meados dos anos 1940. A segunda, chamada Agricultura Convencional, teve como principal característica o uso de fertilizantes e produtos agrícolas para aumentar a produtividade das lavouras, persistindo desde os anos 1940 até os anos 2000. Embora houvesse um aumento na produtividade, o excesso de fertilização causou sérios problemas ambientais, evidenciando a necessidade de uma maior precisão na utilização dos fertilizantes e demais produtos de modo a se atingir um equilíbrio. A terceira e atual fase da agricultura é chamada Agricultura de Precisão (AP), que consiste na análise de dados coletados na agricultura para aumentar a precisão e auxiliar na tomada de decisões.

A agricultura de precisão depende de uma série de tecnologias que trabalham em conjunto para permitir a coleta e a análise de dados, a fim de prover informações confiáveis ao produtor. Dentre estas tecnologias, destacam-se:

- GPS: tratores equipados com sensores e sistemas de orientação de direção que usam as posições registradas para navegar;

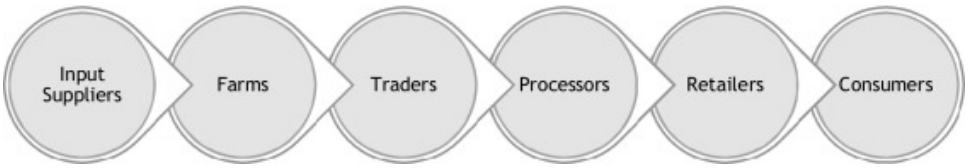
- Sensores montados em máquinas agrícolas: para a captura de dados como a umidade e temperatura do solo, direção do vento, radiação solar, etc;

- Georreferenciamento: que consiste na utilização dos dados coletados por satélite para criar mapas de produtividade, plantio e fertilidade do solo; dentre outras tecnologias.

A exemplo de empresas que atuam na Agricultura de Precisão está a John Deere, que no final dos anos 90 e início dos anos 2000 passou a conectar sensores de GPS a seus tratores e outras máquinas para realizar a coleta de dados no campo (PHAM; STACK, 2018). Embora a coleta dos dados seja fundamental, a A.P. exige também a capacidade de fazer algo valioso com estes dados. A capacidade de capitalizar os dados sendo gerados exige um conjunto muito diferente de capacidades pelos agricultores e (talvez ainda mais importante) pelos fornecedores de insumos.

O cultivo da soja, assim como de outras culturas, faz parte de apenas um nó da chamada cadeia de valor da agricultura. Na Figura 1, o cultivo propriamente dito ocorre somente após serem supridos os devidos insumos, como fertilizantes, defensivos, combústivel para as máquinas agrícolas, dentre outros. Os fornecedores de insumos (*Input Suppliers*) então, ocupam o primeiro nó desta cadeia, suprindo as fazendas (*Farms*) com os produtos necessários ao cultivo. O terceiro nó desta cadeia é ocupado pelos *Traders* (comerciantes), que compram a produção gerada no campo, e vendem para os processadores (*Processors*), que por sua vez são empresas que produzem bens a partir da matéria prima recebida. O quinto nó é ocupado pelos *Retailers* (varejistas), que repassam os produtos processados para o consumidor final (*Consumers*), nó final da cadeia de valor da agricultura.

Figura 1 – A cadeia de valor da agricultura




Fonte: Pham e Stack (2018)

O uso intensivo de Análise de dados começou a impactar todos os nós dessa cadeia, redefinindo a concorrência, as operações e as estratégias dentro e entre esses vários nós (PHAM; STACK, 2018). O foco deste trabalho, contudo, está nas conexões que ligam os agricultures aos vendedores (*traders*), tendo também impacto sobre as conexões entre os fornecedores de insumos e os agricultores.

## 1.1 Bolsa de Chicago (CBOT)

A Bolsa de Mercadorias de Chicago (CME, sigla em inglês) é a principal referência para os preços internacionais da soja. Segundo o IMEA (2017), isto se deve ao fato de que a Bolsa de Chicago possui uma alta concentração de ofertantes e demandantes dos principais países produtores e importadores da oleaginosa, como os EUA, Brasil e China. Além disso, é a primeira bolsa de futuros do mundo, fundada em 1848 (CMEGROUP, 2018b), o que a torna referência consolidada no mercado. No Brasil, os preços internos da soja estão fortemente atrelados ao referencial do mercado futuro (Bolsa de Chicago), mais precisamente à *Chicago Board of Trade* (CBOT), bolsa norte-americana em que é negociada a soja.

O mercado futuro é o tipo de mercado onde são realizadas negociações de compra e venda por meio de contratos uniformes, sendo estes de caráter agrícola ou financeiros, cuja entrega ou liquidação se dá em data futura e já estabelecida no contrato. De acordo com o IMEA (2017), o objetivo de operar no mercado futuro é fixar um preço futuro e operar na chamada *hedge*, livrando-se das oscilações do preço e com isso proteger o resultado do seu negócio. Além disso, é possível realizar outros tipos de operações neste mercado, como é o caso dos especuladores que visam ganhar com a oscilação do mercado e os arbitradores que ganham com as diferenças de preços que ocorrem entre mercados.

A soja no Brasil é colhida, transportada e armazenada a granel, tendo como unidade de referência para o mercado interno a saca de 60 kg. No mercado internacional (Bolsa de Chicago), os grãos são comercializados em *bushel*, que por sua vez é uma undade de medida de volume equivalente a 27,215 kg. As cotações na Bolsa de Chicago são dadas em *cents* de dollar (US$c)/*bushel*, e a Equação 1.1 mostra o cálculo de conversão para R$/saca de 60 kg.

$$(R\$/sc) = \frac{(US\$c/bushel) \times (cotação\_do\_dollar) \times 2,2046}{100} \tag{1.1}$$

O preço da soja no mercado interno depende também de descontos, ou acréscimos, do prêmio de exportação e dos custos de movimentação do produto na área produtora para o porto. O prêmio de exportação é a diferença dos preços externos da Bolsa de Chicago com o preço dentro do navio no porto. Somando-se a isso o dólar comercial, frete e outras variáveis impactam também consideravelmente no mercado doméstico (IMEA, 2017).

O produtor de soja no Brasil possui gastos com insumos que são produzidos fora do país, cujos preços são, em sua maioria, negociados em dollar americano. Como a principal fonte de renda do produtor é a venda de sua produção, e esta, como apresentado anteriormente, é fortemente influenciada pela cotação da Bolsa de Chicago, a rotina[1] de um

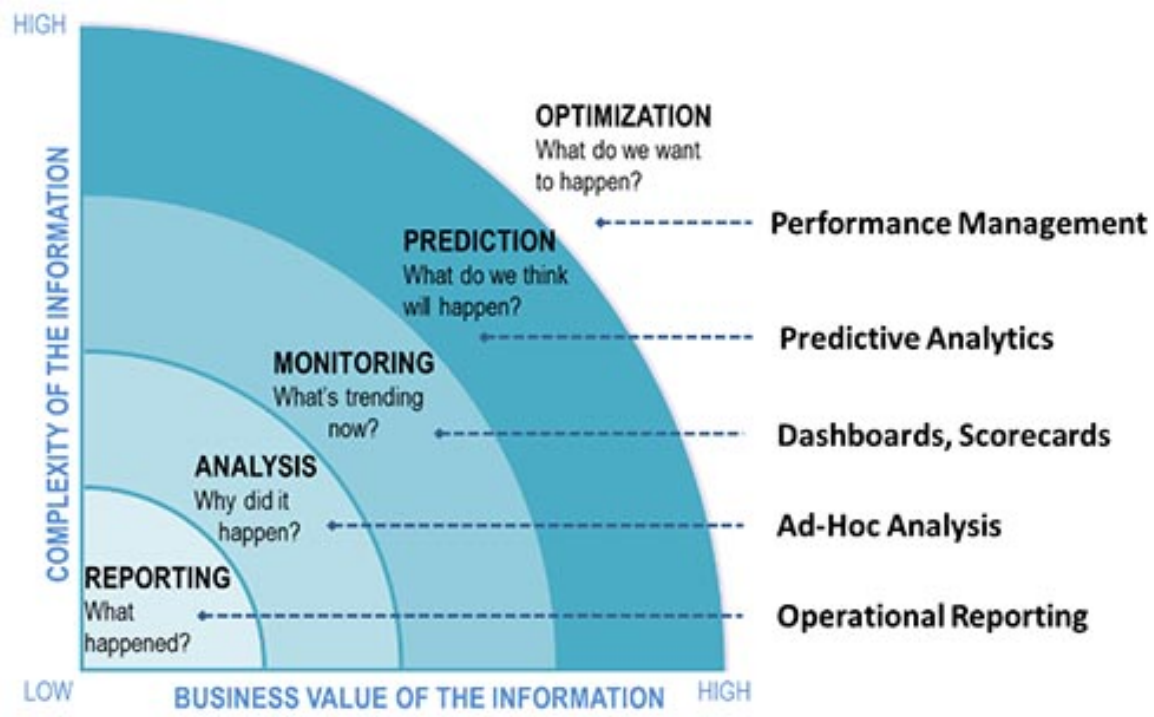
[1] Coversa em privado com produtor de soja do estado do Tocantins (TO), cujo relato revela a sua rotina diária que consiste de, entre outras atividades, verificar as cotações da soja na Bolsa de Chicago pelo

agricultor moderno engloba o monitoramento destas oscilações do mercado internacional a fim de preparar suas finanças para a compra de insumos e para a venda de sua produção.

## 1.2 Análise de dados

A análise dos dados é amplamente explorada na agricultura moderna, como indicam os recentes investimentos realizados em *Big Data* voltado à agricultura por empresas de tecnologia como o Google e IBM (PHAM; STACK, 2018). Dentro do escopo da análise de dados está a Inteligência Artificial, que por sua vez engloba tecnologias como o aprendizado de máquina (*Machine Learning* - ML) e o aprendizado profundo (*Deep Learning* - DL) (GOODFELLOW; BENGIO; COURVILLE, 2016), amplamente utilizados para a extração de informações sobre o grande volume de dados em que está imersa a agricultura moderna. O motivo pelo qual estão sendo empregadas estas tecnologias sobre os dados pode ser justificado pelo fato de que a informação possui diferentes valores para o negócio de acordo com a sua complexidade. O gráfico da Figura 2 ilustra os diferentes valores que a informação pode assumir, sendo a extremidade superior direita o maior valor.

Figura 2 – Complexidade e valor da informação




Fonte: DeVries (2018)

site do CMEGroup, além de analisar a previsão do tempo para os EUA. A justificativa dada para esta última verificação é o fato do clima influenciar na produção de soja americana, e consequentemente na sua cotação.

A primeira e menos valiosa informação está na elaboração de relatórios dos acontecimentos passados a partir dos dados coletados, que é representado no primeiro nível no canto inferior esquerdo do gráfico da Figura 2. O segundo nível é atingido quando são inferidas justificativas para os acontecimentos passados, aumentando o valor da informação. Quando os dados são coletados e monitorados em tempo real (como faz atualmente a John Deere (PHAM; STACK, 2018)), é atingido o terceiro nível, onde a informação proveniente deste monitoramento possui valor maior ao da simples análise por permitir que o comportamento do negócio seja visualizado em tempo real. O quarto passo na escala de valor da informação está em prever o futuro, i.e., realizar a Análise Preditiva sobre os dados e prover informações mais valiosas do que às do monitoramento. O último e maior valor da informação para um negócio é proveniente da predição de ações que possam ser tomadas para otimizar os resultados futuros. O valor da informação é diretamente proporcional à sua complexidade. O escopo deste trabalho está na Análise Preditiva, de modo a prover com informações sobre a futura cotação da soja para o produtor rural.

# 2 Trabalhos relacionados

A soja futura na Bolsa de Chicago (CBOT) é uma *commodity*, e isto significa que é possível utilizar estudos já realizados sobre predição de outras *commodities*, como o milho, o petróleo bruto e até mesmo o *BitCoin*. A Bolsa de Chicago também opera o mercado futuro para *BitCoin* da mesma forma como faz com a soja, o milho, o trigo e outras *commodities* (CMEGROUP, 2018a). Desta forma, foram pesquisados estudos já realizados sobre a utilização de Deep Learning para a predição de *commodities* em geral.

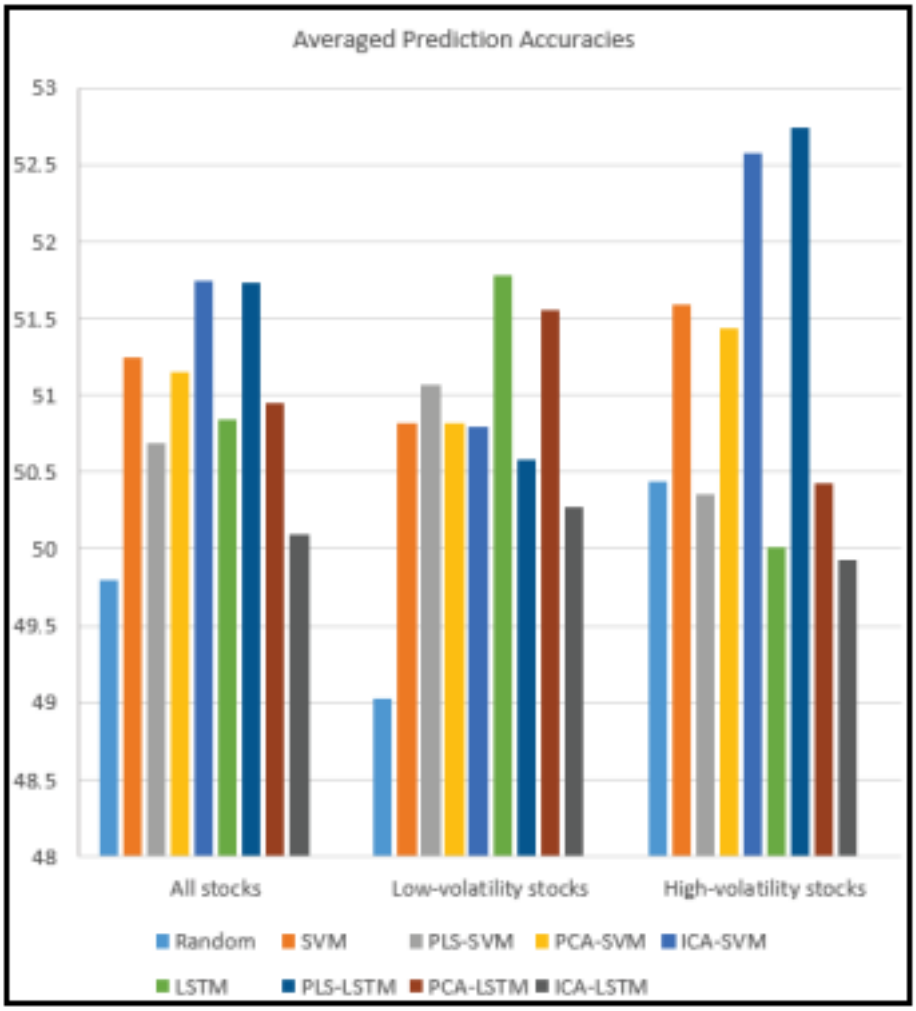
O trabalho de McNally, Roche e Caton (2018) avalia o desempenho da rede neural LSTM (*Long Short-Term Memory* ou Rede de Longo Prazo de Memória) comparado ao desempenho da rede neural recorrente (RNN) na predição da cotação de *BitCoin*. Os resultados obtidos apresentam um desempenho superior da LSTM sobre a RNN atingindo a mais alta precisão de classificação de 52% e um RMSE de 8%. O popular modelo ARIMA para previsão de séries temporais também foi implementado como uma comparação aos modelos de aprendizagem profunda, e como esperado pelos autores, os métodos não-lineares de aprendizagem profunda superam a previsão ARIMA, que teve desempenho considerado ruim. Os tempos de treinamento em GPU e CPU para ambos os modelos (LSTM e RNN) foram comparados, sendo o treinamento realizado na GPU 67,7% mais rápido do que na CPU. Este trabalho de modo geral evidencia o desempenho superior de aprendizado profundo na predição da *commodity*, destacando-se ainda o modelo LSTM.

O trabalho de Wang (2017) utiliza o modelo de aprendizado supervisionado SVM (*Support Vector Machine*, ou Máquina de vetores de suporte) para realizar predição das cotações do milho, soja e petróleo bruto, relacionando as três *commodities*. O desempenho da SVM é comparado a um modelo de regressão logística. Os resultados obtidos mostraram que os fatores técnicos dos preços futuros do milho em um mês, juntamente com outros fatores técnicos que representam as inter-relações com os produtos relacionados, podem ser um poderoso conjunto de características preditivas. O modelo SVM apresenta melhor desempenho do que o modelo de regressão logística em todas as amostras de tamanho de teste. Um outro fato observado neste trabalho foi a importância de se alimentar o modelo com os dados na forma sequencial e não aleatória, para que se preserve a tendência no comportamento dos dados.

O trabalho de (LI; TAM, 2017) aponta que o uso das técnicas SVM e LSTM são amplamente adotados para prever os movimentos de preços de ações na China e em outros países. Foram analisadas diferentes medidas de desempenho do LSTM e do SVM e comparadas em uma série de cotações exibindo volatilidades diferentes. De modo surpreendente, os resultados obtidos revelaram que o desempenho geral do SVM original

é superior na predição de todas as cotações e também cotações de baixa volatilidade no Índice Shanghai Stock Exchange 50 (SSE 50), enquanto o LSTM original consistentemente alcança o melhor desempenho geral na predição das cotações de alta volatilidade no Índice SSE 50.

Figura 3 – Comparação das médias de desempenho (*accuracy*) entre os modelos LSTM e SVM




Fonte: Li e Tam (2017)

A Figura 3 mostra a comparação do desempenho médio entre os modelos LSTM e SVM. Também foram utilizadas as técnicas de pré-processamento de dados PCA, ICA e PLS para ambos os modelos, de modo a otimizar os resultados de cada um. O PCA-LSTM e o PLS-LSTM são os modelos de melhor desempenho para as cotações de baixa volatilidade e alta volatilidade, respectivamente. O PCA-LSTM obteve uma precisão 51,55% para cotações de baixa volatilidade, enquanto o PLS-LSTM apresenta a maior precisão de 52,74% para as cotações de alta volatilidade do índice SSE 50.

# 3 Metodologia

Tanto a obtenção e pré-processamento dos dados, quanto o treinamento e avaliação do modelo foram realizados no Google Collaboratory(GOOGLE COLLAB, 2019), um ambiente de *notebook* gratuito Jupyter(JUPYTER, 2018) que não requer configuração e é executado inteiramente na nuvem. O Collaboratory disponibiliza 12 GB de RAM, 350 GB de armazenamento em disco (temporário), processamento em CPU, GPU e TPU, para executar códigos, salvar e compartilhar análises diretamente do navegador.

A análise preliminar dos *datasets* foi realizada através das funções da biblioteca *open source pandas* (MCKINNEY, 2017) e *NumPy* (SCIPY.ORG, 2019) em Python 3.6. Para a montagem e treinamento do modelo preditivo foram utilizadas as bibliotecas Keras (KERAS, 2019) e Tensorflow. (TENSORFLOW, 2018).

## 3.1 Obtenção dos dados

A primeira etapa do trabalho foi a obtenção dos dados para a composição do *dateset*. Para a obtenção do *dataset* de cotações da soja, foi utilizada a API Quandl (QUANDL, 2013), onde os dados foram extraídos diretamente do site do CME Group em formato *JSON*. As cotações são agrupadas por dia em período desde 01/07/1959, contendo as seguintes características:

```
"column_names": [
    "Date",    (Data no formato YYYY-MM-DD)
    "Open",    (Abertura em US$c)
    "High",    (Alta em US$c)
    "Low",     (Baixa em US$c)
    "Last",    (Ultimo valor em US$c)
    "Change", ( (Valor atual - Settle do dia anterior) em US$c)
    "Settle", (Fechamento em US$c)
    "Volume", (Volume de contratos negociados no dia)
    "Previous Day Open Interest"
               (Numero total de contratos em aberto que nao foram
               liquidados para um ativo)
],
```

Os *dataset* climático foi obtido do repositório GHCN (Global Historical Climatology Network) (GHCN, 2019) mantido pelo NOAA (National Oceanic and Atmospheric Administration) (NOAA, 2019) que contém dados diários de estações meteorológicas de

diversos países num período maior a 175 anos. Estes dados do repositório do GHCN (GHCN FTP, 2019) estão organizados por estações individuais em arquivos no formato .dly, e com isso foi necessário encontrar um método de download e conversão destes dados para o formato .csv. A solução foi encontrada no trabalho de Penne (2019) que fornece uma API em Python onde é possível localizar e converter dados de estações individuais. Contudo, foram realizadas adaptações no código fonte de Penne (2019) para que fossem obtidos apenas os dados das estações climáticas dos EUA e com as colunas originais.

## 3.2 Pré-processamento dos dados

O pré-processamento foi necessário para tornar possível a união dos dados climáticos com os dados das cotações. O *dataset* das cotações de soja não precisou ser tratado pois sua estrutura já estava no formato desejado, i.e. com cada linha sendo uma amostra diária e cada coluna uma característica, conforme ilustra a Figura 4.

Figura 4 – Dataset CBOT: cotações de soja da Bolsa de Chicago

| | Date | Open | High | Low | Last | Change | Settle | Volume | Previous Day Open Interest |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 2018-11-21 | 882.00 | 889.00 | 876.00 | 884.00 | 2.00 | 883.00 | 62504.0 | 292755.0 |
| 1 | 2018-11-20 | 874.25 | 885.75 | 870.50 | 880.75 | 7.25 | 881.00 | 62089.0 | 300785.0 |
| 2 | 2018-11-19 | 892.00 | 892.25 | 871.25 | 873.75 | 18.50 | 873.75 | 89125.0 | 299897.0 |
| 3 | 2018-11-16 | 889.75 | 894.75 | 881.75 | 890.25 | 3.50 | 892.25 | 71310.0 | 298062.0 |
| 4 | 2018-11-15 | 884.25 | 897.50 | 883.75 | 889.00 | 5.25 | 888.75 | 80828.0 | 302082.0 |

Fonte: Autor.

O *dataset* CBOT possui 15.105 linhas × 8 colunas, cada linha uma leitura diária sendo a mais antiga do dia 08/07/1959 e a mais recente do dia 21/06/2019.

Entretanto, para os dados climáticos, após realizado o download de 60.811 tabelas foi necessário normalizar as colunas de cada uma delas.

### 3.2.1 Normalização das tabelas

Originalmente, cada estação climática está representada por uma tabela que possui colunas a mais ou a menos em relação às demais, devido a coleta dos dados específica de cada estação. Por exemplo, a estação climática "USS0017B04S"(Figura 5) do estado de Washington apresenta uma estrutura de colunas com as temperaturas máxima (TMAX em décimos de °C), mínima (TMIN em décimos de °C) e média (TAVG em décimos de °C), além de precipitação (PRCP em décimos de mm), e demais indicadores específicos, enquanto que a estação "US1WAAD0002"(Figura 6) do mesmo estado possui apenas a coluna referente à precipitação (além de demais colunas de indicadores não numéricos).

Figura 5 – Estação climática "USS0017B04S"do estado de Washington que já apresenta uma estrutura de colunas com as temperaturas máxima (TMAX), mínima (TMIN) e média (TAVG), além de precipitação (PRCP).

| | | | | ID | TMAX | TMAX_FLAGS | TMIN | TMIN_FLAGS | TOBS | TOBS_FLAGS | TAVG | TAVG_FLAGS | PRCP | PRCP_FLAGS | WESD | WESD_FLAGS | SNWD | SNWD_FLAGS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MM/DD/YYYY | YEAR | MONTH | DAY | | | | | | | | | | | | | | | |
| 1986-06-23 | 1986 | 6 | 23 | USS0017B04S | 268.0 | __T | 138.0 | __T | 163.0 | __T | 198.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |
| 1986-06-24 | 1986 | 6 | 24 | USS0017B04S | 270.0 | __T | 159.0 | _IT | 139.0 | _IT | 206.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |
| 1986-06-25 | 1986 | 6 | 25 | USS0017B04S | 246.0 | __T | 129.0 | __T | 138.0 | __T | 185.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |
| 1986-06-26 | 1986 | 6 | 26 | USS0017B04S | 206.0 | __T | 128.0 | __T | 142.0 | __T | 159.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |
| 1986-06-27 | 1986 | 6 | 27 | USS0017B04S | 269.0 | __T | 140.0 | __T | 260.0 | __T | 192.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |
| 1986-06-29 | 1986 | 6 | 29 | USS0017B04S | 83.0 | __T | 49.0 | __T | 191.0 | _IT | 70.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |
| 1986-06-30 | 1986 | 6 | 30 | USS0017B04S | 83.0 | _IT | 49.0 | __T | 248.0 | __T | 70.0 | __T | NaN | NaN | NaN | NaN | NaN | NaN |

Fonte: Autor.

Figura 6 – Estação climática "US1WAAD0002"do estado de Washington que possui apenas a coluna referente à precipitação (além de demais colunas de indicadores não numéricos).

| | | | | ID | PRCP | PRCP_FLAGS | SNOW | SNOW_FLAGS | SNWD | SNWD_FLAGS |
|---|---|---|---|---|---|---|---|---|---|---|
| MM/DD/YYYY | YEAR | MONTH | DAY | | | | | | | |
| 2009-04-29 | 2009 | 4 | 29 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |
| 2009-04-30 | 2009 | 4 | 30 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |
| 2009-05-01 | 2009 | 5 | 1 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |
| 2009-05-02 | 2009 | 5 | 2 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |
| 2009-05-03 | 2009 | 5 | 3 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |
| 2009-05-04 | 2009 | 5 | 4 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |
| 2009-05-05 | 2009 | 5 | 5 | US1WAAD0002 | 0.0 | __N | 0.0 | __N | NaN | NaN |

Fonte: Autor.

A normalização se deu selecionando somente as colunas "TMAX"(temperatura máxima), "TMIN"(temperatura mínima), "TAVG"(temperatura média), "PRCP"(precipitação), ID (identificação da estação climática) e as colunas índices relacionadas ao tempo em que foi realizada a observação ("MM/DD/YYYY- renomeada para "Date", "YEAR", "MONTH"e "DAY") em todas as tabelas, eliminando as demais informações. Para as tabelas onde algumas destas colunas não existiam foram inseridas as respectivas colunas inexistentes com valor nulo (NaN da biblioteca *numpy* em Python). A escolha pelo valor nulo se deu pois durante o cálculo das médias entre as colunas (que será abordado na seção seguinte), os valores nulos são desconsiderados, não compondo o cálculo como fariam se tivessem o valor zero por exemplo.

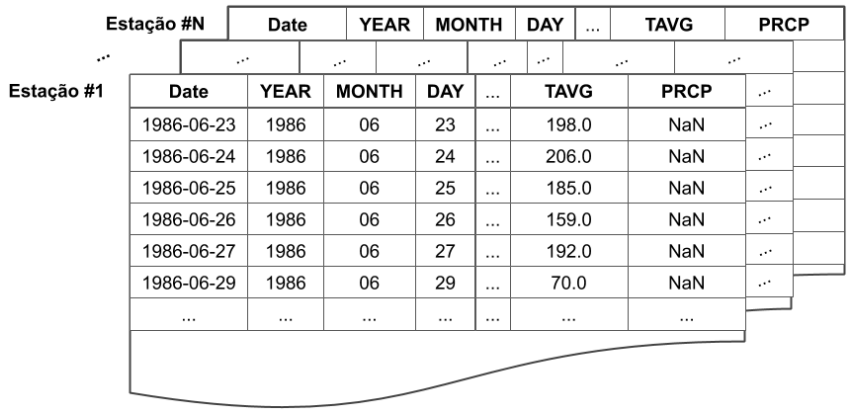
Após a normalização, cada uma das 60.811 tabelas de dados climáticos possui as mesmas colunas, conforme ilustra a Figura 7 e Figura 8

Figura 7 – Tabela normalizada da estação climática "USS0017B04S"do estado de Washington.

| | | | | ID | TMAX | TMIN | TAVG | PRCP |
|---|---|---|---|---|---|---|---|---|
| Date | YEAR | MONTH | DAY | | | | | |
| 1986-06-23 | 1986-01-01 | 6 | 23 | USS0017B04S | 268.0 | 138.0 | 198.0 | NaN |
| 1986-06-24 | 1986-01-01 | 6 | 24 | USS0017B04S | 270.0 | 159.0 | 206.0 | NaN |
| 1986-06-25 | 1986-01-01 | 6 | 25 | USS0017B04S | 246.0 | 129.0 | 185.0 | NaN |
| 1986-06-26 | 1986-01-01 | 6 | 26 | USS0017B04S | 206.0 | 128.0 | 159.0 | NaN |
| 1986-06-27 | 1986-01-01 | 6 | 27 | USS0017B04S | 269.0 | 140.0 | 192.0 | NaN |
| 1986-06-29 | 1986-01-01 | 6 | 29 | USS0017B04S | 83.0 | 49.0 | 70.0 | NaN |
| 1986-06-30 | 1986-01-01 | 6 | 30 | USS0017B04S | 83.0 | 49.0 | 70.0 | NaN |

Fonte: Autor.

Figura 8 – Esquema da estrutura das tabelas de dados climáticos.




Fonte: Autor.

### 3.2.2 Agrupamento das tabelas por estados norte-americanos

Foi decidido filtrar somente os estados norte-americanos que apresentaram produção de soja nos últimos anos. Para tanto, foram utilizados dados da United States Department of Agriculture - National Agricultural Statistics Service (USDA-NASS), Departamento de Agricultura dos EUA, que possui registros desde 2001 da produção de soja (em *bushels*) por estado norte-americano. A Figura 9 ilustra a produção da soja no ano de 2015 nos EUA.

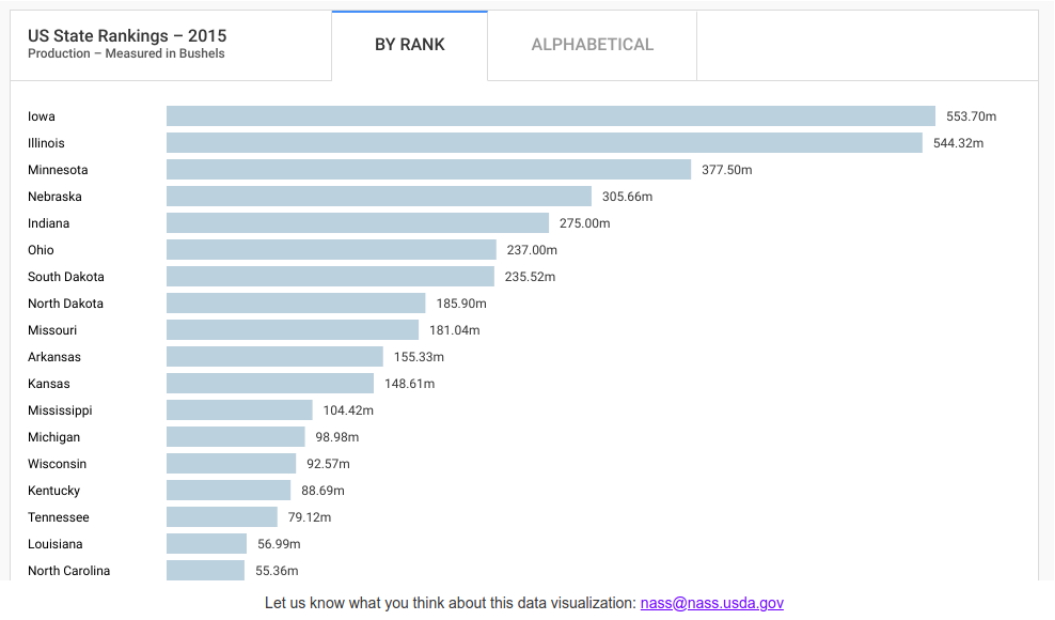
Figura 9 – USDA-NASS: produção de soja nos estados norte-americanos em 2015

| | state_name | state_alpha | Value | unit_desc | commodity_desc | year |
|---|---|---|---|---|---|---|
| 0 | ALABAMA | AL | 20090000 | BU | SOYBEANS | 2015 |
| 1 | ARKANSAS | AR | 155330000 | BU | SOYBEANS | 2015 |
| 2 | DELAWARE | DE | 6920000 | BU | SOYBEANS | 2015 |
| 3 | FLORIDA | FL | 1102000 | BU | SOYBEANS | 2015 |
| 4 | GEORGIA | GA | 13330000 | BU | SOYBEANS | 2015 |

Fonte: Autor.

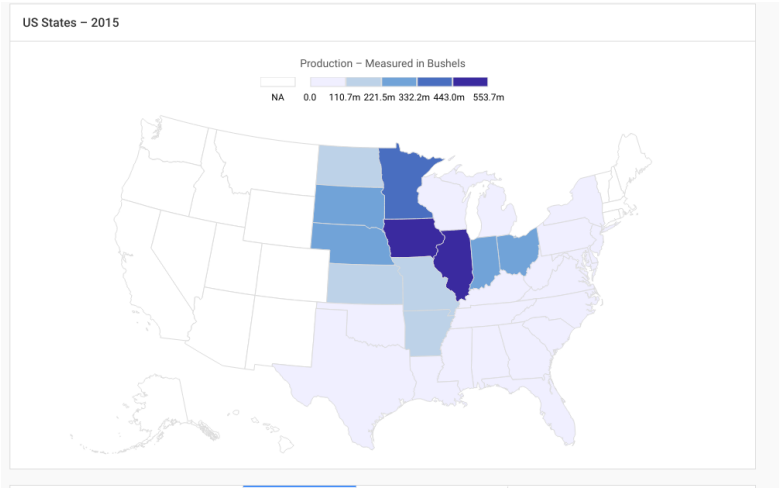
Os dados da Figura 9 podem ser melhor visualizados pelo gráfico da Figura 10 e pelo mapa da Figura 11.

Figura 10 – Ranking da produção de soja dos estados norte-americanos em 2015 (em *bushels*)




Fonte: USDA (2018b)

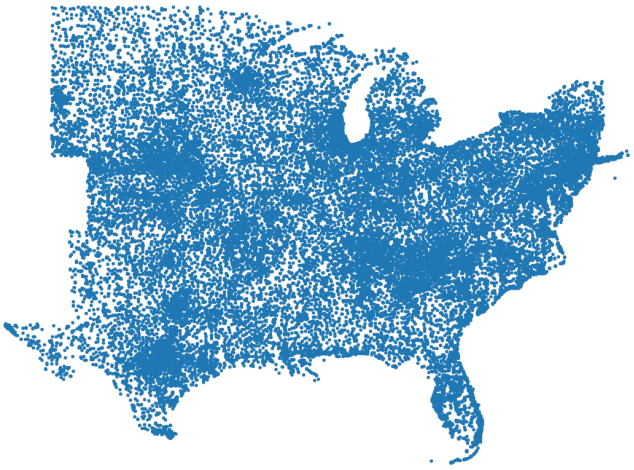
Figura 11 – Mapa da produção de soja dos estados norte-americanos em 2015 (em *bushels*)




Fonte: USDA (2018b)

Foi identificado que os estados que possuem produção de soja desde 2001 foram 'AL', 'AR', 'DE', 'FL', 'GA', 'IL', 'IN', 'IA', 'KS', 'KY', 'LA', 'MD', 'MI', 'MN', 'MS', 'MO', 'NE', 'NJ', 'NY', 'NC', 'ND', 'OH', 'OK', 'PA', 'SC', 'SD', 'TN', 'TX', 'VA', 'WV' e 'WI', totalizando 31 estados. Com esta informação, foram selecionados apenas as estações meteorológicas que estão localizadas nestes estados, resultando num total de 38.802 estações, cada uma representada por um ponto azul no mapa da Figura 12.

Figura 12 – Mapa com a localização das estações meteorológicas para os estados norte-americanos com produção de soja.



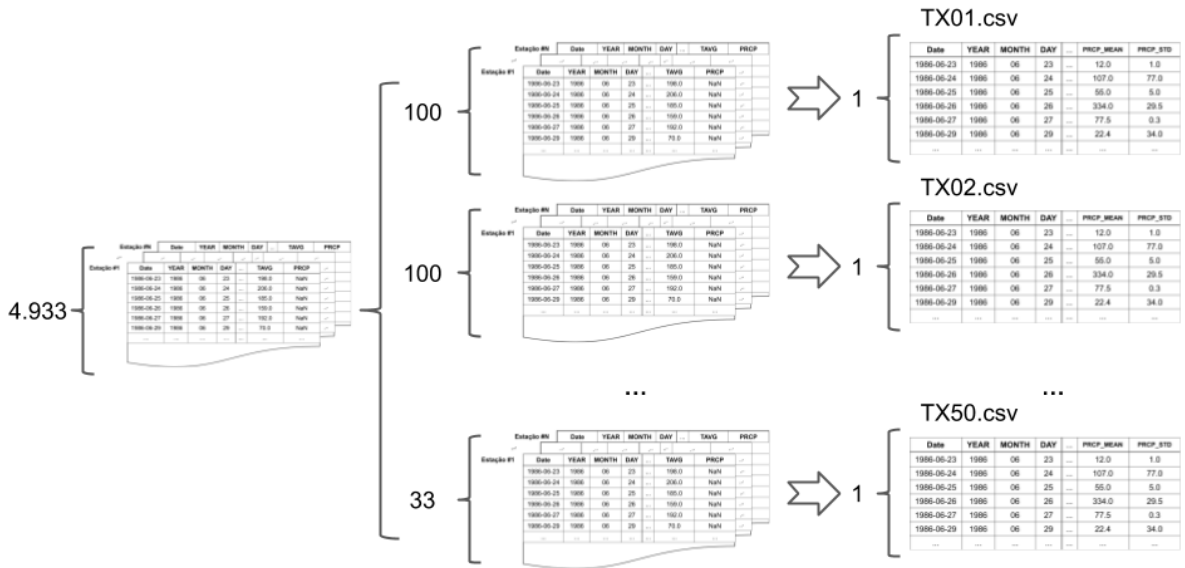
Fonte: Autor.

Cada ponto azul no mapa representa uma das 38.802 estações meteorológicas.

Fazendo um cálculo rápido: 38.802 tabelas climáticas que possuem 4 colunas de interesse cada, resulta num total de 155.208 colunas ($38.802 \times 4$). Este seria o número de colunas de uma única tabela para representar todos os dados climáticos. Como a disponibilidade de memória RAM da infra-estrutura utilizada é limitada, optou-se por agrupar estas 38.802 tabelas por estados norte-americanos, calculando a média e desvio padrão de cada uma das 4 colunas de interesse ("TMAX", "TMIN", "TAVG"e "PRCP") entre as tabelas de cada estado. O resultado deste agrupamento é uma única tabela possuindo o $numero\_de\_estados \times 8$ colunas (8 significa que para cada uma das 4 colunas de interesse agora há a média e o desvio padrão), totalizando 248 colunas ($31 \times 8$).

A relação estação-estado é disponibilizada em forma de arquivo texto no repositório GHCN (GHCN FTP, 2019), possuindo também informações de latitude e longitude de cada estação (necessárias para criar o mapa da Figura 12). Esta relação foi utilizada para separar as tabelas em diretórios distintos referentes à cada estado, resultando num total de 31 diretórios. Com este agrupamento, a quantidade de tabelas que devem ser carregadas simultaneamente em memória reduziu de 38.802 para no máximo 4.933 (referente ao estado do Texas, que possui o maior número de estações climáticas). Entretanto, não foi possível carregar as 4.933 em memória de uma só vez para calcular a média e o desvio padrão das quatro colunas de interesse, logo foi necessário realizar este processamento em lotes de 100 estações por vez, gerando para cada lote uma nova tabela, conforme ilustra a Figura 13.

Figura 13 – Processamento em lotes de 100 tabelas por vez, calculando a média e o desvio padrão para cada uma das quatro colunas de interesse ("TMAX", "TMIN", "TAVG"e "PRCP").




Fonte: Autor.

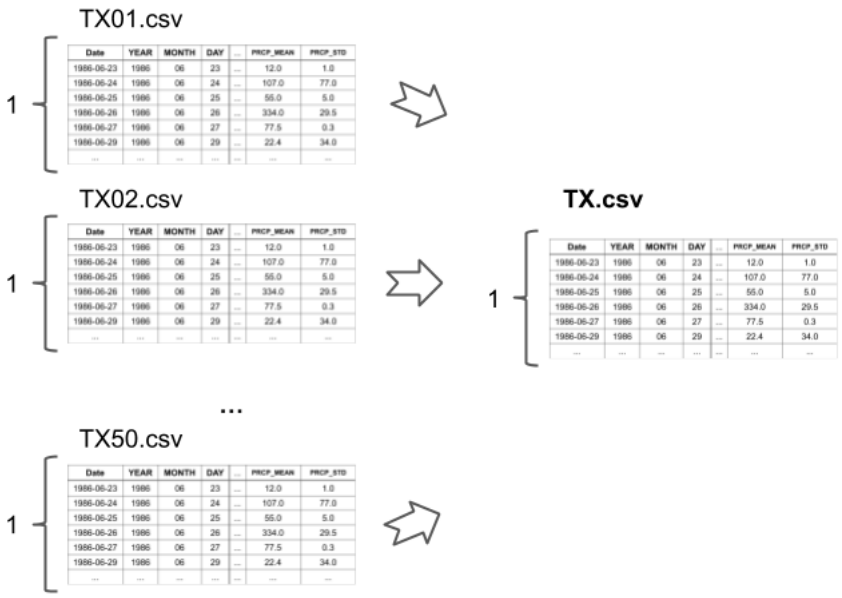
Neste exemplo, o estado do Texas agora possui 50 tabelas com 8 colunas cada,

contendo a média e desvio padrão de cada uma das 4 colunas de interesse. O próximo passo foi agrupar estas 50 tabelas, onde se repetiu o processo da Figura 13 porém com diferença na hora de calcular o desvio padrão. Para o cálculo da média o processo foi o mesmo, calculando a média das médias é o mesmo que calcular a média de todos os elementos de uma só vez, conforme mostra a Equação 3.1.

$$\mu_{X \cup Y} = \frac{N_X \times \mu_X + N_Y \times \mu_Y}{N_X \times N_Y} \tag{3.1}$$

Porém, calcular o desvio padrão dos desvios padrão não é o mesmo que calcular o desvio padrão de todos os elementos de uma só vez. Portanto, optou-se por utilizar outra abordagem: elevar os desvios padrão ao quadrado, resultando na variância, somar todas as variâncias e depois calcular a raíz quadrada desta soma. Ao término deste agrupamento, é então gerada uma única tabela que contém os dados de todas as estações climáticas do estado do Texas, conforme ilustra Figura 14.

Figura 14 – Processamento das tabelas agrupadas anteriormente.




Fonte: Autor.

Após esta etapa, obteve-se 31 tabelas referentes aos 31 estados, que foram em seguida reunidas em uma única tabela, adicionando o prefixo com a sigla do estado em cada uma das colunas de interesse, conforme ilustra a Figura 15.

Figura 15 – Dataset climático após o pré-processamento.

| Date | YEAR | MONTH | DAY | TX_TMAX_MEAN | TX_TMAX_STD | TX_TMIN_MEAN | TX_TMIN_STD | TX_TAVG_MEAN | TX_TAVG_STD | TX_PRCP_MEAN | TX_PRCP_STD | NC_TMAX_MEAN | NC_TMAX_S |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2019-04-22 | 2019 | 4 | 22 | 72.627790 | 566.42426 | 44.583332 | 354.48105 | 18.318048 | 317.19855 | 0.000000 | 0.000000 | NaN | N |
| 2019-04-18 | 2019 | 4 | 18 | 54.506924 | 582.77580 | 27.069391 | 301.68390 | 8.202712 | 267.15765 | 77.998770 | 722.597100 | 53.402460 | 434.612 |
| 2019-04-17 | 2019 | 4 | 17 | 62.617880 | 641.38715 | 37.206060 | 412.67450 | 11.757029 | 343.05920 | 0.546157 | 26.782059 | 48.102604 | 403.454 |
| 2019-04-16 | 2019 | 4 | 16 | 65.986390 | 662.37030 | 30.363200 | 340.81598 | 11.515528 | 328.71072 | 0.009583 | 1.444365 | 39.248573 | 341.688 |
| 2019-04-15 | 2019 | 4 | 15 | 59.496593 | 612.05500 | 17.168463 | 219.85052 | 10.452809 | 296.80405 | 0.042482 | 2.736362 | 41.162914 | 356.904 |
| 2019-04-12 | 2019 | 4 | 12 | 54.676180 | 586.09040 | 18.959663 | 278.85687 | 8.596012 | 259.29416 | 0.423592 | 20.780325 | 44.908268 | 384.148 |
| 2019-04-11 | 2019 | 4 | 11 | 69.903590 | 715.77690 | 28.139763 | 354.06543 | 10.630163 | 314.85098 | 0.041998 | 4.000743 | 46.239610 | 391.194 |

Fonte: Autor.

O *dataset* climático possui 64.044 linhas × 248 colunas, cada linha uma leitura diária sendo a mais antiga do dia 01/07/1836 e a mais recente do dia 22/04/2019.

### 3.2.3 União dos dados climáticos e cotações para a criação do DatasetMarlon

O propósito de se unir os dados climáticos aos dados das cotações em uma única tabela foi o de facilitar a alimentação do modelo preditivo durante o treinamento e o teste. Os dados climáticos já tratados possuem registros diários no período de 01/07/1836 a 22/04/2019, enquanto que os dados do *dataset* das cotações possuem dados no período de 08/07/1959 a 21/06/2019, logo, foi necessário realizar uma operação de *inner join* entre as duas tabelas, mantendo apenas os dias em comum em ambas. O resultado desta união é o DatasetMarlon, disponibilizado no GitHub (FRANCO, 2019) e ilustrado na Figura 16.

Figura 16 – DatasetMarlon, contendo dados diários de cotações de soja e dados climáticos dos estados norte-americanos produtores de soja num período de 10/07/1959 à 22/04/2019.

| Date | YEAR | MONTH | DAY | Open | High | Low | Last | Change | Settle | Volume | Previous Day Open Interest | TX_TMAX_MEAN | TX_TMAX_STD | TX_TMIN_MEAN | TX_TMIN_STD | TX_TAVG_MEAN | TX_TAVG_STD | TX_PRCP_MEAN | TX_PRCP_STD |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2019-04-22 | 2019 | 4 | 22 | 881.50 | 883.25 | 876.25 | 876.75 | 3.50 | 877.00 | 62527.0 | 205572.0 | 72.627790 | 566.42426 | 44.583332 | 354.48105 | 18.318048 | 317.19855 | 0.000000 | 0.000000 |
| 2019-04-18 | 2019 | 4 | 18 | 878.75 | 882.00 | 876.50 | 880.75 | 1.50 | 880.50 | 63485.0 | 214732.0 | 54.506924 | 582.77580 | 27.069391 | 301.68390 | 8.202712 | 267.15765 | 77.998770 | 722.597100 |
| 2019-04-17 | 2019 | 4 | 17 | 887.75 | 890.50 | 878.50 | 879.00 | 9.00 | 879.00 | 89706.0 | 219956.0 | 62.617880 | 641.38715 | 37.206060 | 412.67450 | 11.757029 | 343.05920 | 0.546157 | 26.782059 |
| 2019-04-16 | 2019 | 4 | 16 | 898.25 | 899.00 | 886.25 | 888.00 | 10.75 | 888.00 | 92852.0 | 221960.0 | 65.986390 | 662.37030 | 30.363200 | 340.81598 | 11.515528 | 328.71072 | 0.009583 | 1.444365 |
| 2019-04-15 | 2019 | 4 | 15 | 895.00 | 902.00 | 894.75 | 898.75 | 3.50 | 898.75 | 91118.0 | 232341.0 | 59.496593 | 612.05500 | 17.168463 | 219.85052 | 10.452809 | 296.80405 | 0.042482 | 2.736362 |
| 2019-04-12 | 2019 | 4 | 12 | 895.00 | 898.50 | 893.75 | 894.50 | NaN | 895.25 | 69411.0 | 242526.0 | 54.676180 | 586.09040 | 18.959663 | 278.85687 | 8.596012 | 259.29416 | 0.423592 | 20.780325 |
| 2019-04-11 | 2019 | 4 | 11 | 901.25 | 904.00 | 893.50 | 895.75 | 6.75 | 895.25 | 85699.0 | 253700.0 | 69.903590 | 715.77690 | 28.139763 | 354.06543 | 10.630163 | 314.85098 | 0.041998 | 4.000743 |

Fonte: Autor.

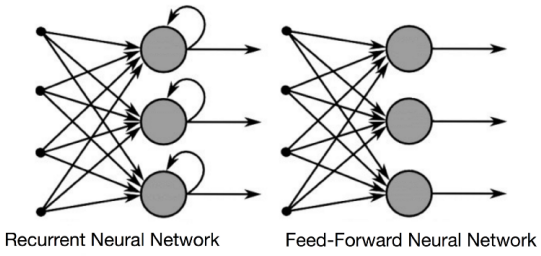
O DatasetMarlon possui 15.062 linhas × 256 colunas, e encontra-se disponível no GitHub: <https://github.com/marlonrcfranco/soyforecast/>

## 3.3 LSTM

Para realizar a predição das cotações de soja, utilizou-se o modelo Long Short Term Memory (LSTM), proposto por (HOCHREITER; SCHMIDHUBER, 1997). O LSTM é um tipo de rede neural pertencente à família de Redes neurais recorrentes (RNN) (RUMELHART; HINTON; WILLIAMS, 1986), e, segundo Goodfellow, Bengio e Courville (2016), é extremamente bem sucedido em muitas aplicações, tais como reconhecimento de manuscrito, reconhecimento de fala, geração de caligrafia, tradução automática e legendagem de imagens. RNN's são redes neurais para processamento de dados sequenciais que, diferente de uma rede convolucional (CNN) que é especializada no processamento de uma grade de valores $X$, como uma imagem, uma rede neural recorrente (RNN) é uma rede neural especializada no processamento de uma seqüência de valores $x^1..., x^\tau$. Segundo Géron (2017), RNN's podem analisar dados de séries temporais, como preços de ações, e informar quando comprar ou vender. Em sistemas de condução de automóveis, eles podem antecipar trajetórias de carros e ajudar a evitar acidentes.

A principal característica que define uma RNN é o fato de seus neurônios armazenarem o valor processado anteriormente em uma pequena memória. Em um exemplo prático, basta imaginar um problema hipotético onde deve-se prever qual a próxima letra na sequência "neuron". Para uma rede neural tradicional (como Rede Neural de Feed-Forward (FNN)), no momento em que a letra "u"está sendo processada, as letras "n"e "e"já foram esquecidas pela rede, tornando virtualmente impossível a predição da próxima letra. Para uma RNN, no entanto, a medida que o processamento avança na cadeia de caracteres, é mantida uma memória sobre o que já foi processado pelo neurônio, permitindo que seja predito qual o próximo caractere da sequência. A Figura 17 ilustra a diferença entre o funcionamento de uma Rede Neural Recorrente (RNN) e uma Rede Neural de Feed-Forward

Figura 17 – Diferença no fluxo de informações entre uma RNN e uma Rede Neural de Feed-Forward.

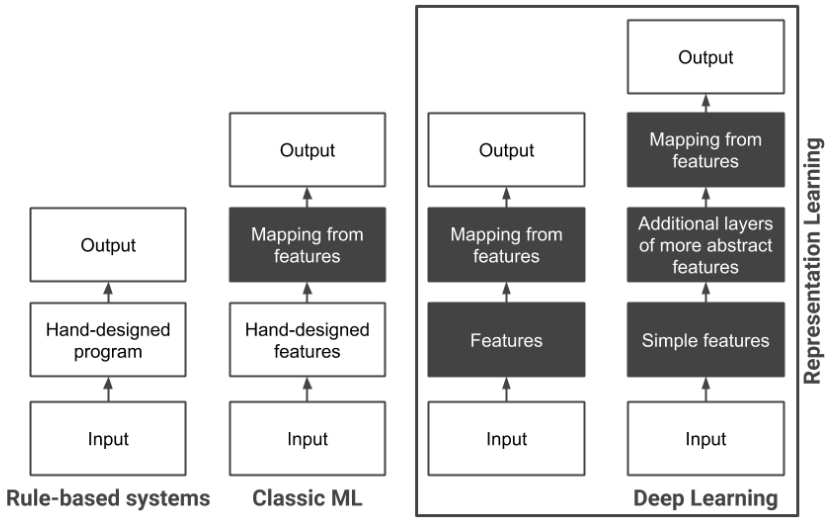



Fonte: Donges (2018)

A RNN faz parte do nicho de redes neurais profundas (Deep Neural Network - DNN),

que segundo Goodfellow, Bengio e Courville (2016), são redes que realizam o aprendizado de representação, i.e. a aprendizagem profunda permite que o computador construa conceitos complexos a partir de conceitos mais simples. Como ilustrado na Figura 18 os sistemas baseados em regras (*Rule-based systems*) são sistemas onde o desenvolvedor deve projetar as regras do sistema à priori, o que demanda alto conhecimento do problema e das variáveis que o compõe. No Aprendizado de Máquina Clássico (*Classic ML*), a máquina aprende as regras que regem o problema analisando uma série de características (*features*) que foram previamente elencadas pelo desenvolvedor. Esta técnica pode induzir à baixa acurácia nos resultados do aprendizado, caso as *features* não tenham sido suficientemente elencadas pelo desenvolvedor. Dentro do escopo do aprendizado profundo (*Deep Learning*), a própria máquina decide quais são as características que devem ser levadas em consideração para a elaboração das regras que regem o problema. Deixar esta etapa à cargo da inteligência artificial permite que a precisão nos resultados seja muito superior ao aprendizado de máquina tradicional.

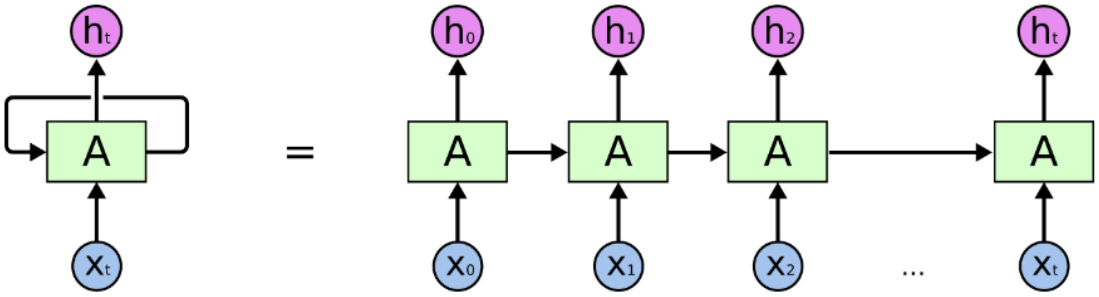
Figura 18 – Como as partes de um sistema de IA se relacionam entre si dentro de diferentes técnicas




Fonte: Goodfellow, Bengio e Courville (2016)

Uma RNN pode ser pensada em uma sequência de múltiplas cópias da mesma rede, conforme ilustra a Figura 19 proposta por Olah (2015).

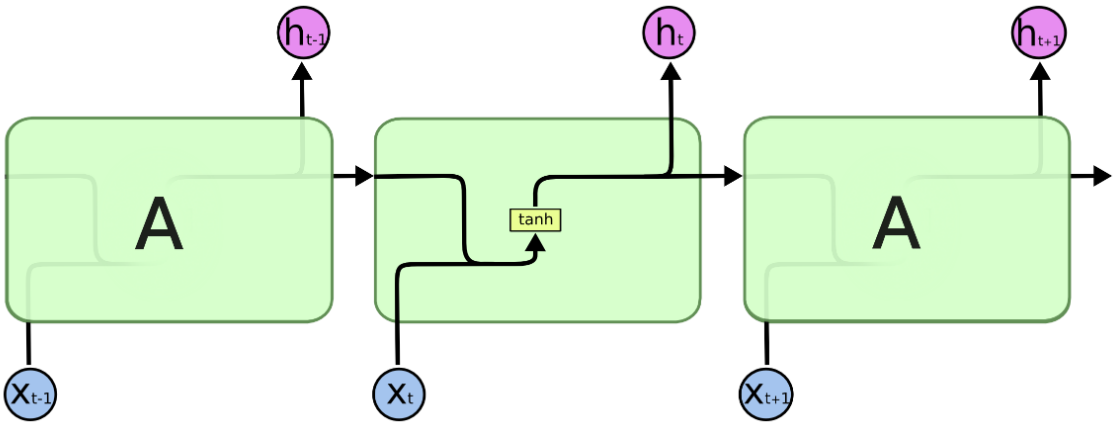
Figura 19 – Uma RNN visualizada como uma sequência de múltiplas cópias da mesma rede.




Fonte: Olah (2015)

Segundo Olah (2015), todas as redes neurais recorrentes têm a forma de uma cadeia de módulos repetitivos da rede neural. Este módulo de repetição (quadrado verde na Figura 19) possui uma estrutura muito simples, como uma única camada com a função de ativação *tanh*, conforme ilustrado na Figura 20.

Figura 20 – Uma célula RNN possui apenas uma função de ativação *tanh*.

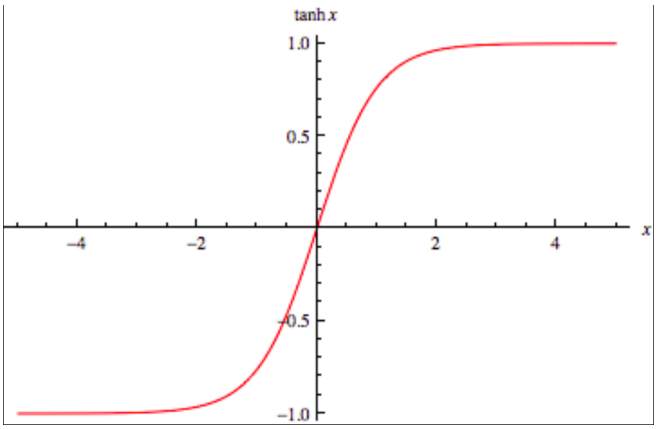



Fonte: Olah (2015)

A função de ativação *tanh* (Equação 3.2) possui a forma de "S"(Figura 21) e seu valor varia de –1 a 1, o que tende a fazer com que a saída desta camada seja mais ou menos normalizada (i.e. centrada ao redor do zero) no início do treinamento. Segundo Géron (2017), isso geralmente ajuda a acelerar a convergência.

$$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})} \tag{3.2}$$

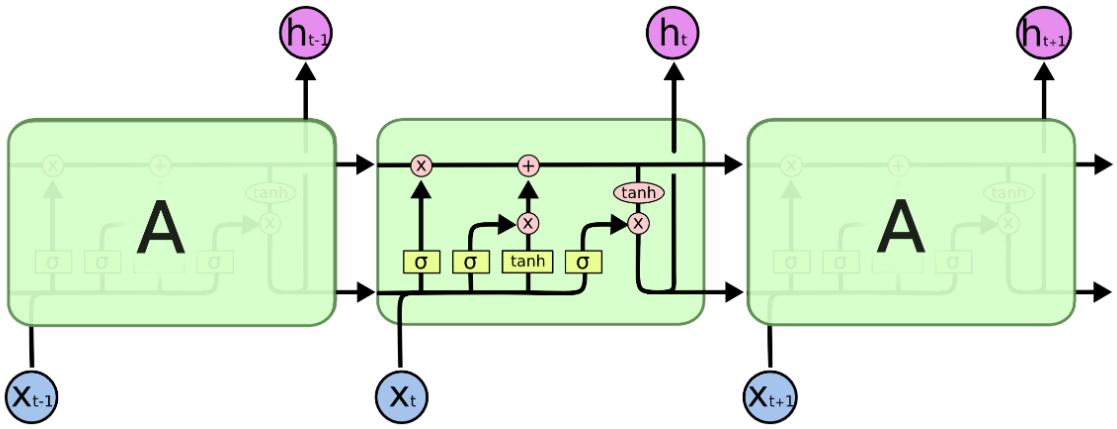
Figura 21 – Função de ativação *tanh*.




Fonte: <http://mathworld.wolfram.com/HyperbolicTangent.html>

As LSTM's, por outro lado, possuem o módulo de repetição com uma estrutura diferente, onde ao invés de ter uma única camada de rede neural, existem quatro, ligadas entre si como mostra a Figura 22.

Figura 22 – Uma célula LSTM possui quatro camadas internas interagindo entre si.




Fonte: Olah (2015)

A Figura 23 é uma legenda para os símbolos utilizados a partir de agora para auxiliar no entendimento sobre LSTM's.

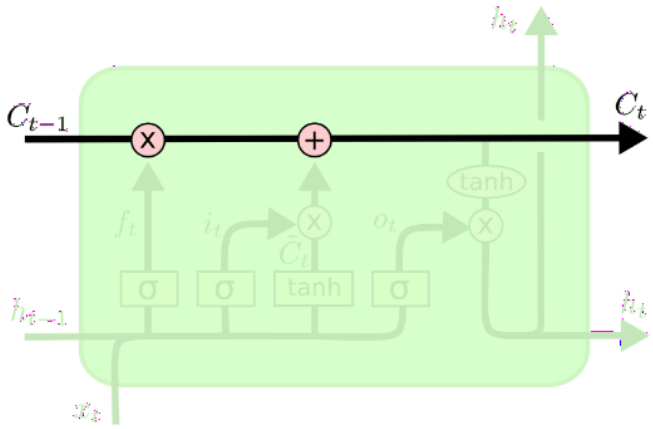
Figura 23 – Legenda dos símbolos.


Fonte: Olah (2015)

Os retângulos verdes são chamados "Camada de rede neural", funções de ativação descritas pelo símbolo dentro de cada retângulo. Os círculo vermelhos são "Operações ponto-a-ponto", como adição de elementos no vetor. As setas unidirecionais representam o caminho entre dos vetores ("Transferência de vetor"). As setas que convergem representam concatenação, enquanto as setas que divergem significam que o vetor foi copiado e as cópias vão para destinos diferentes.

Segundo Olah (2015), a chave para os LSTMs é o estado da célula (*cell state*), que é sua memória interna representada pela linha horizontal na parte superior do diagrama (Figura 24). O *cell state* transmite informações com apenas algumas interações lineares menores (operações ponto-a-ponto), sendo muito fácil que as informações fluam sem alterações.

Figura 24 – *cell state* (estado da célula)

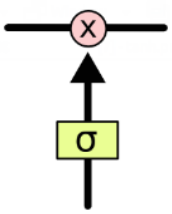



Fonte: Olah (2015)

As LSTM's possuem a habilidade de remover ou adicionar informação no *cell state*, sendo esta regulagem controlada pelas estruturas denominadas *Gates* (ou portas).

Cada *Gate* (ou porta) definem se uma informação deve ou não ser transmitida. São compostas por uma camada de rede neural com a função *Sigmoid* ligada a uma operação ponto-a-ponto de multiplicação, conforme ilustra a Figura 25.

Figura 25 – *Gate* (ou porta), consiste de uma camada de rede neural com a função *Sigmoid* ligada a uma operação ponto-a-ponto de multiplicação, cuja função é definir se uma informação deve ou não ser transmitida.

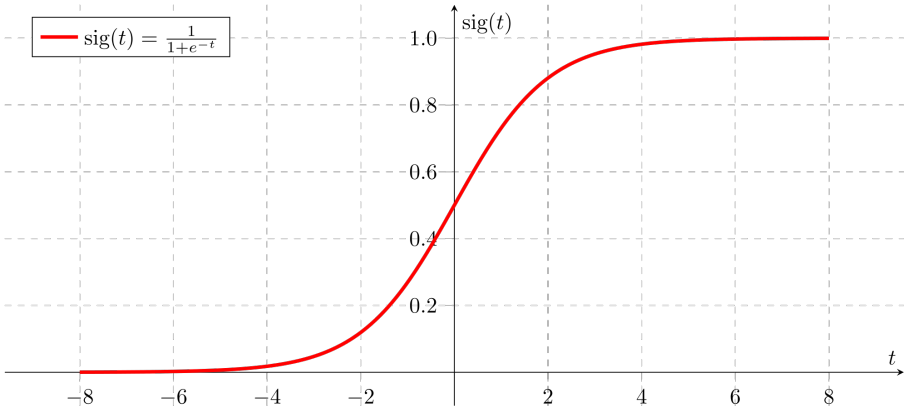



Fonte: Olah (2015)

Segundo Olah (2015), a função *Sigmoid* (Equação 3.3) retorna números entre 0 e 1, determinando o quanto cada sinal deve ser transmitido. Um valor 0 significa que o sinal não deve ser transmitido, enquanto o valor 1 significa que deve ser transmitido totalmente. A função *Sigmoid* possui a forma de "S", conforme ilustra a Figura 26.

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.3}$$

Figura 26 – Função de ativação *Sigmoid*.



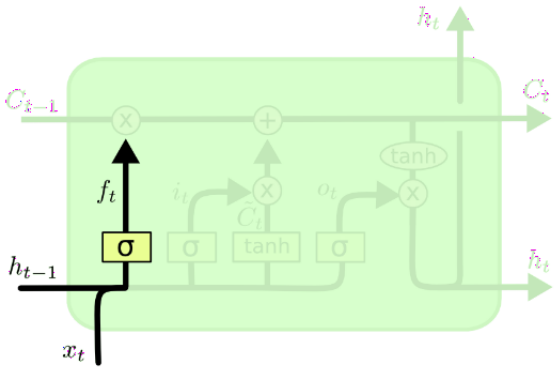

Fonte: <https://towardsdatascience.com/derivative-of-the-sigmoid-function-536880cf918e>

As LSTM's possuem três tipos de *Gates*: *forget gate*, *input gate* e *output gate*, cada um deles será descrito a seguir.

### 3.3.1 Forget gate

Segundo Olah (2015), a primeira etapa dentro de uma célula LSTM é decidir qual informação deve ser descartada do *cell state*. Esta decisão é feita pela camada de rede neural com função *Sigmoid* denominada "*forget gate*" $f_t$. Esta camada aplica a função *Sigmoid* $\sigma$ sobre a concatenação do vetor de saída anterior $h_{t-1}$ e o vetor da entrada atual $x_t$, retornando um número entre 0 e 1. Este número é então multiplicado a cada elemento no *cell state* anterior $C_{t-1}$. Se o valor multiplicado for 1, significa que o elemento deve ser mantido totalmente no *cell state*, mas se for 0, significa que deve ser totalmente "esquecido". O fluxo de informações do *forget gate* é apresentado na Figura 27.

Figura 27 – *Forget gate* consiste de uma camada de rede neural com a função *Sigmoid* ligada a uma operação ponto-a-ponto de multiplicação, cuja função é definir se uma informação deve ou não ser mantida no *cell state*.




$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

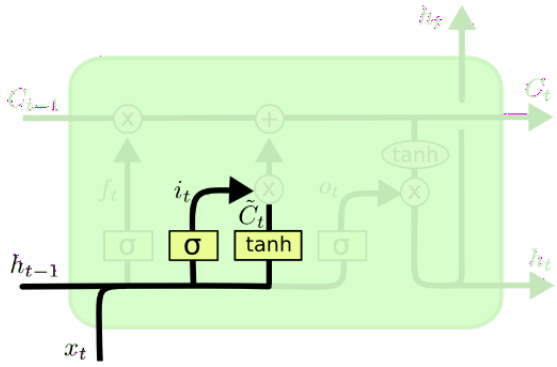
Fonte: Olah (2015)

$f_t$ forget gate; $\sigma$: função de ativação Sigmoid; $W_f$: peso; $h_{t-1}$: vetor de saída do passo anterior; $x_t$: vetor de entrada do passo atual; $b_f$: bias (ou viés).

### 3.3.2 Input gate

Para calcular os novos valores do *cell state*, é utilizada uma camada de rede neural *tanh*, que calcula valores entre -1 e 1 sobre a concatenação do vetor de saída anterior $h_{t-1}$ e o vetor da entrada atual $x_t$. O resultado desta operação é o vetor com os novos valores para *cell state* ($\widetilde{C}_t$). Porém, estes novos valores não são atualizados automaticamente, mas passam por uma operação de multiplicação ponto-a-ponto com o resultado de outra célula de rede neural com função de ativação *Sigmoid* (similar ao *forget gate*). Esta segunda célula do *input gate* é responsável por decidir o quanto cada valor de $\widetilde{C}_t$ deve ser somado ao *cell state*, após passar por uma operação de soma ponto-a-ponto.

Figura 28 – *Input gate* consiste de uma camada de rede neural com a função *Sigmoid* e uma camada com a função *tanh*, cujas saídas estão ligadas por uma operação de multiplicação ponto-a-ponto. Sua função é definir o quanto cada informação deve ser atualizada no *cell state*.



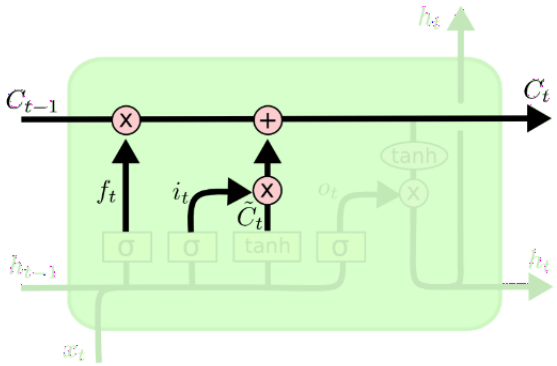

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Fonte: Olah (2015)

$i_t$ input gate; $\sigma$: função de ativação Sigmoid; $W$: peso; $h_{t-1}$: vetor de saída do passo anterior; $x_t$: vetor de entrada do passo atual; $b$: bias (ou viés); $\tilde{C}_t$: novos valores para *cell state*; *tanh*: função de ativação tangente hiberbólica.

O valor atual de *cell state* então a soma de cada valor do *cell state* anterior que permaneceu após o *forget gate* e os novos valores ponderado pelo *input gate*, ilustrado na Figura 29.

Figura 29 – Cálculo dos valores atuais para o *cell state*.




$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

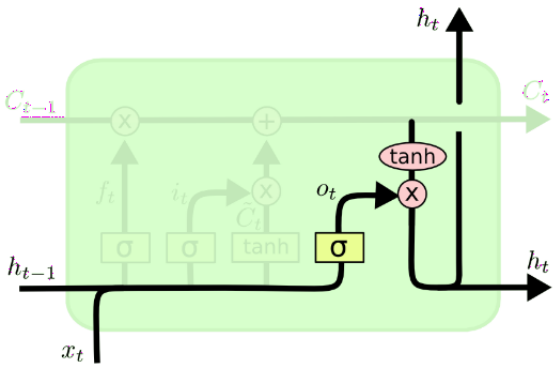
Fonte: Olah (2015)

$C_t$ *cell state* atual; $f_t$: forget gate; $C_{t-1}$: *cell state* anterior; $i_t$: input gate; $\tilde{C}_t$: novos valores para *cell state*;

### 3.3.3 Output gate

A última etapa numa célula LSTM é a decisão sobre a saída $h_t$. Segundo Olah (2015), esta saída é baseada no *cell state*, porém de uma forma filtrada. Primeiro, uma camada de rede neural com função de ativação *Sigmoid* aplica a função *Sigmoid* $\sigma$ sobre a concatenação do vetor de saída anterior $h_{t-1}$ e o vetor da entrada atual $x_t$, retornando um número entre 0 e 1 denominado *output gate* $o_t$. Em seguida, é aplicado sobre cada elemento do *cell state* atual a função ponto-a-ponto *tanh* (para normalizar os valores entre $-1$ e 1). O resultado desta operação é então multiplicado ponto-a-ponto com $o_t$, gerando finalmente a saída $h_t$, que é propagada para frente e para o próximo ciclo.

Figura 30 – *Output gate* consiste de uma camada de rede neural com a função *Sigmoid* e uma operação *tanh* ponto-a-ponto sobre o *cell state* atual, ligadas por uma operação de multiplicação ponto-a-ponto. Sua função é definir o quais valores devem fazer parte da saída $h_t$.




$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

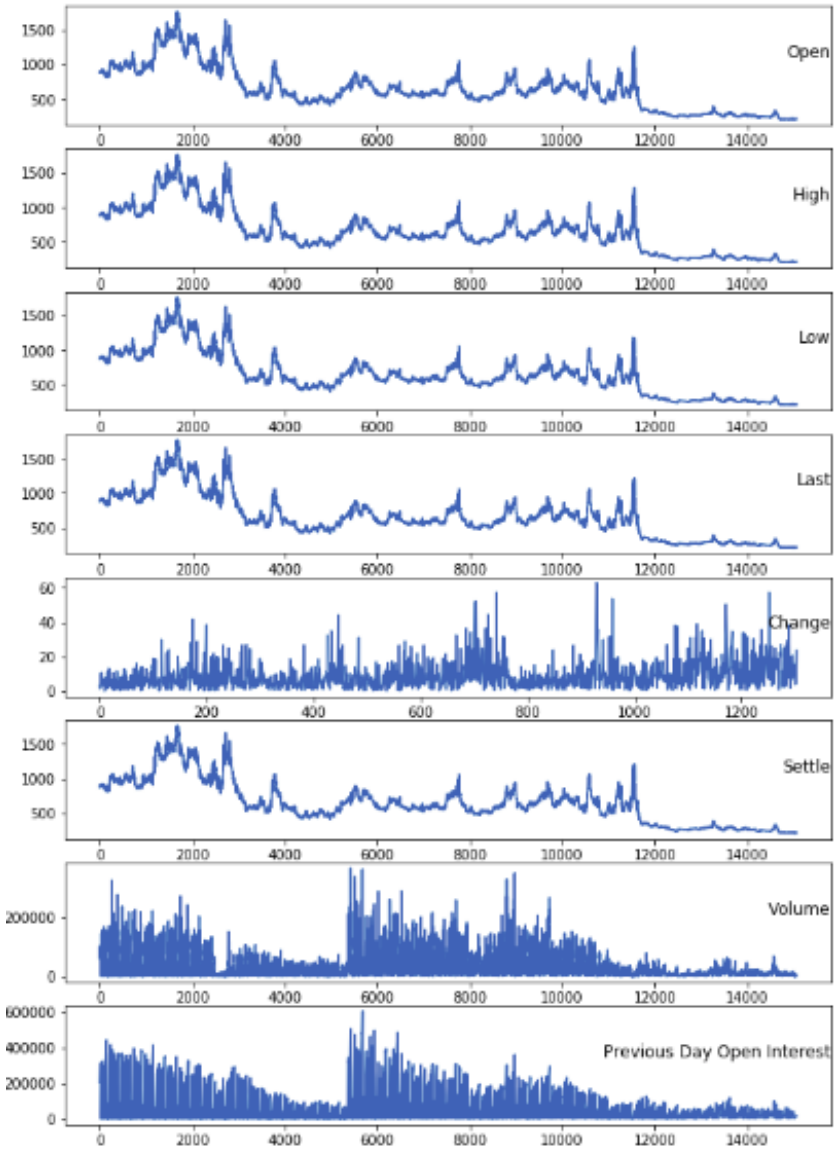
Fonte: Olah (2015)

$o_t$ output gate; $\sigma$: função de ativação Sigmoid; $W_o$: peso; $h_{t-1}$: vetor de saída do passo anterior; $x_t$: vetor de entrada do passo atual; $b_o$: bias (ou viés); $h_t$: saída atual; *tanh*: função de ativação tangente hiberbólica; $C_t$: *cell state* atual.

# 4 Resultados

Após serem reunidos e organizados os dados climáticos e as cotações em um único *dataset* denominado DatasetMarlon, foi iniciada a etapa de criação e treinamento do modelo LSTM. Para a criação e treinamento do modelo, foi utilizada a biblioteca *Keras* (KERAS, 2019), que disponibiliza uma implementação de camadas LSTM, *Sigmoid*, *tanh*, entre outras, otimizadas para o uso de GPU's.

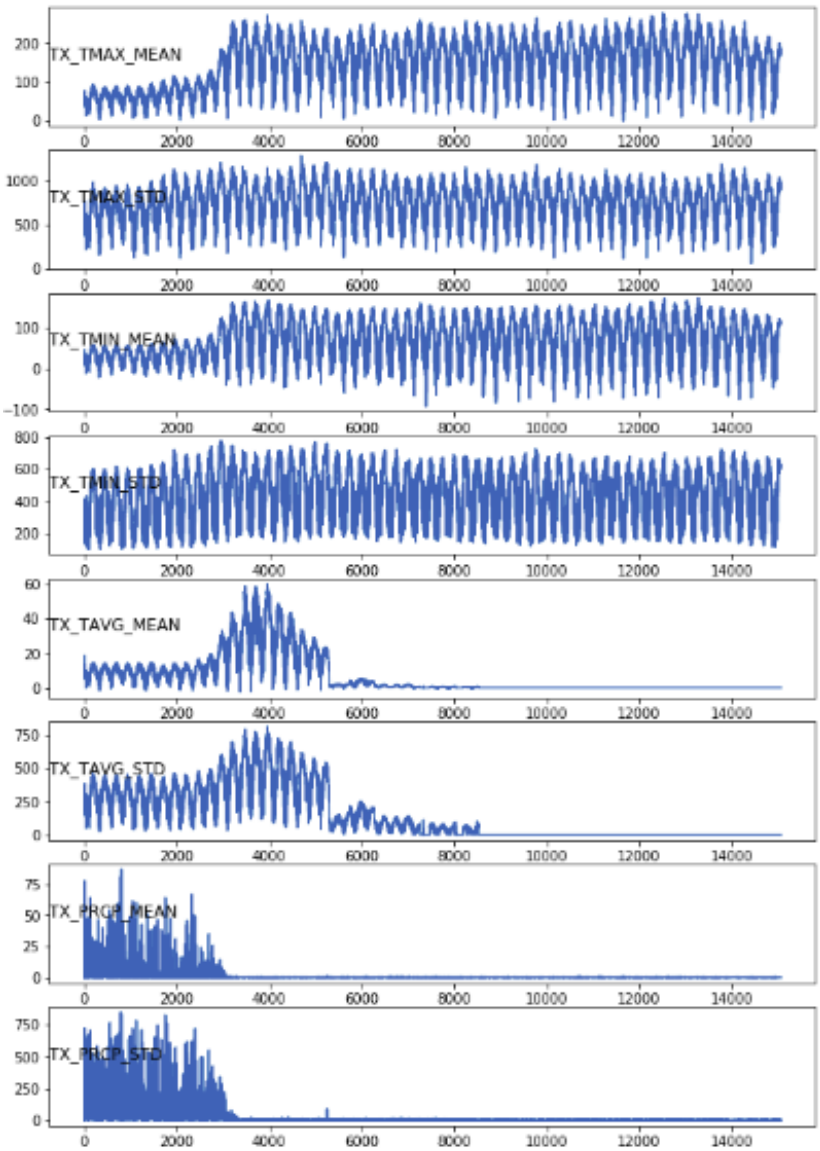
Figura 31 – DatasetMarlon: gráfico das variáveis vindas do *dataset* das cotações no período de 10/07/1959 à 22/04/2019, sendo *Settle* a variável alvo a ser predita pelo modelo.




Fonte: Autor.

A Figura 31 mostra as variáveis presentes no DatasetMarlon que vieram do *dataset* CBOT Soybean Futures (cotações), sendo "Settle"a variável alvo que é a última cotação da soja no dia. As variáveis do DatasetMarlon que vieram do *dataset* de dados climáticos (Figura 32) são ao todo 248 ($31 estados \times 8 colunas$). A mostra o comportamento das 8 variáveis climáticas para o estado do Texas no período de 10/07/1959 à 22/04/2019.

Figura 32 – DatasetMarlon: gráfico das variáveis vindas do *dataset* de dados climáticos no período de 10/07/1959 à 22/04/2019 filtrado para o estado do Texas. Ao todo, são 248 variáveis: 8 para cada um dos 31 estados (incluindo o Texas).




Fonte: Autor.

Após carregado o DatasetMarlon em memória no formato de *DataFrame* da biblioteca *pandas*, foi então necessário eliminar os registros que possuem alguma das colunas nula (NaN), e normalizar os dados para estarem no intervalo entre 0 e 1. Para a normalização,

foi utilizada a classe *MinMaxScaler* da biblioteca *pandas*, que realiza para cada *feature* o cálculo da Equação 4.1.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4.1}$$

Após a normalização dos dados, foi realizado um deslocamento nos dados para que fossem criadas colunas referentes a cada um dos 7 dias anteriores. Desta forma, é alimentado ao modelo valores referentes a uma semana antes da cotação a ser predita. A Figura 33 mostra como ficaram os dados após a normalização e o deslocamento. Na Figura 34 é importante observar que a única variável que possui o indicador "t"(i.e. no tempo atual) é a "var6_t", que por sua vez é a variável alvo *Settle*. Após a normalização e o deslocamento, o DatasetMarlon passou a ter 10.991 linhas e 1.793 colunas.

Figura 33 – DatasetMarlon após a normalização e o deslocamento dos dados em 1 semana.

| | var1(t-7) | var2(t-7) | var3(t-7) | var4(t-7) | var5(t-7) | var6(t-7) | var7(t-7) | var8(t-7) | var9(t-7) | var10(t-7) | var11(t-7) | var12(t-7) | var13(t-7) | var14(t-7) | var15(t-7) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 0.430163 | 0.431369 | 0.429128 | 0.431731 | 0.000359 | 0.431572 | 0.171927 | 0.353242 | 0.201763 | 0.431541 | 0.456176 | 0.300841 | 0.169949 | 0.328450 | 0.896115 |
| 9 | 0.435911 | 0.436790 | 0.430404 | 0.430614 | 0.002942 | 0.430614 | 0.242938 | 0.361835 | 0.230660 | 0.479189 | 0.494639 | 0.463921 | 0.226693 | 0.421765 | 0.006275 |
| 10 | 0.442617 | 0.442212 | 0.435350 | 0.436359 | 0.003516 | 0.436359 | 0.251458 | 0.365132 | 0.242662 | 0.496247 | 0.468674 | 0.358339 | 0.222838 | 0.404125 | 0.000110 |
| 11 | 0.440542 | 0.444126 | 0.440775 | 0.443221 | 0.001148 | 0.443221 | 0.246762 | 0.382209 | 0.219540 | 0.455343 | 0.418607 | 0.180602 | 0.205872 | 0.364898 | 0.000488 |
| 12 | 0.440542 | 0.441893 | 0.440137 | 0.440508 | 0.669824 | 0.440987 | 0.187976 | 0.398964 | 0.202366 | 0.434235 | 0.425404 | 0.267301 | 0.176228 | 0.318783 | 0.004867 |

Fonte: Autor.

Figura 34 – DatasetMarlon após a normalização e o deslocamento dos dados em 1 semana. A coluna "var6_t"é o *Settle* a ser predito.

| (t-1) | var245(t-1) | var246(t-1) | var247(t-1) | var248(t-1) | var249(t-1) | var250(t-1) | var251(t-1) | var252(t-1) | var253(t-1) | var254(t-1) | var255(t-1) | var256(t-1) | var6(t) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 911 | 0.245567 | 0.173112 | 0.000214 | 0.004264 | 0.462003 | 0.397564 | 0.608633 | 0.127016 | 0.442540 | 0.142258 | 0.037625 | 0.254125 | 0.443221 |
| 970 | 0.247504 | 0.203999 | 0.055373 | 0.176851 | 0.512180 | 0.490143 | 0.616917 | 0.128333 | 0.470187 | 0.265425 | 0.000550 | 0.006923 | 0.443221 |
| 457 | 0.246859 | 0.204921 | 0.017327 | 0.068066 | 0.502198 | 0.469681 | 0.620976 | 0.137802 | 0.477043 | 0.300327 | 0.036553 | 0.085632 | 0.443381 |
| 184 | 0.238309 | 0.096205 | 0.001507 | 0.026990 | 0.425961 | 0.281899 | 0.615545 | 0.133919 | 0.452272 | 0.191550 | 0.024786 | 0.053430 | 0.448168 |
| 619 | 0.243130 | 0.150464 | 0.000000 | 0.000000 | 0.420924 | 0.232492 | 0.589131 | 0.153001 | 0.443218 | 0.145499 | 0.070249 | 0.187655 | 0.443221 |

Fonte: Autor.

Os dados então são separados em dois *datasets* numa proporção de 0,87, resultando num *dataset* de treino com 9.562 linhas e um de teste com 1.429 linhas.

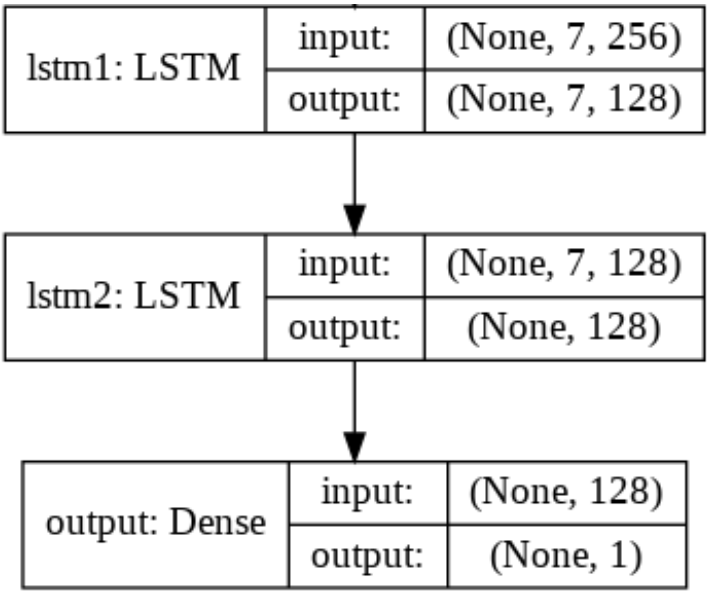
O próximo passo então foi montar o *design* do modelo preditivo. Foi arbitrado ao modelo duas camadas com 128 células (neurônios) LSTM cada, e uma camada de saída com apenas uma única célula com função de ativação *linear*, que não altera o valor.

Figura 35 – Sumário do modelo de predição montado, contendo duas camadas LSTM ($lstm1$ e $lstm2$) com 128 células (neurônios) cada, e uma camada de saída ($output$) com uma única célula.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm1 (LSTM)                 (None, 7, 128)            197120
_________________________________________________________________
lstm2 (LSTM)                 (None, 128)               131584
_________________________________________________________________
output (Dense)               (None, 1)                 129
=================================================================
Total params: 328,833
Trainable params: 328,833
Non-trainable params: 0
```

Fonte: Autor.

Figura 36 – Esquema do modelo preditivo montado.




Fonte: Autor.

Assim como um modelo de *feed-foward* (vide Figura 17), o modelo criado atualiza os valores de peso $W$ e $bias$ (ou viés) $b$ após cada $epoch$ (época, i.e. fluxo da informação da camada de entrada até a camada de saída). A cada $epoch$, o modelo deve comparar o valor de cotação predito $\hat{y}_i$ com o valor da cotação real $y_i$. A função escolhida para realizar esta comparação e calcular o erro foi a métrica Root-Mean-Square Error (RMSE), que

calcula a magnitude média do erro entre a curva de cotação estimada e a curva real,

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{4.2}$$

onde $n$ é o numero de cotações, $y_i$ é a cotação real, e $\hat{y}_i$ é a cotação prevista pelo modelo. O $RMSE$ é a raíz quadrada da média da soma das diferenças entre a cotação prevista e a cotação real. Quanto menor este valor, mais precisa (do inglês *accurate*) é a previsão realizada pelo modelo.

A função de otimização escolhida para realizar a atualização do peso $W$ e *bias* $b$ ao término de cada *epoch* foi a Adam - *Adapted Moment Estimation* (KINGMA; BA, 2014), com taxa de aprendizado $\alpha = 0.001$. A Figura 37 mostra o algorítmo proposto por Kingma e Ba (2014), que combina o efeito do gradiente descendente $m_t$ com o gradiente descendente vindo do RMSprop $v_t$.

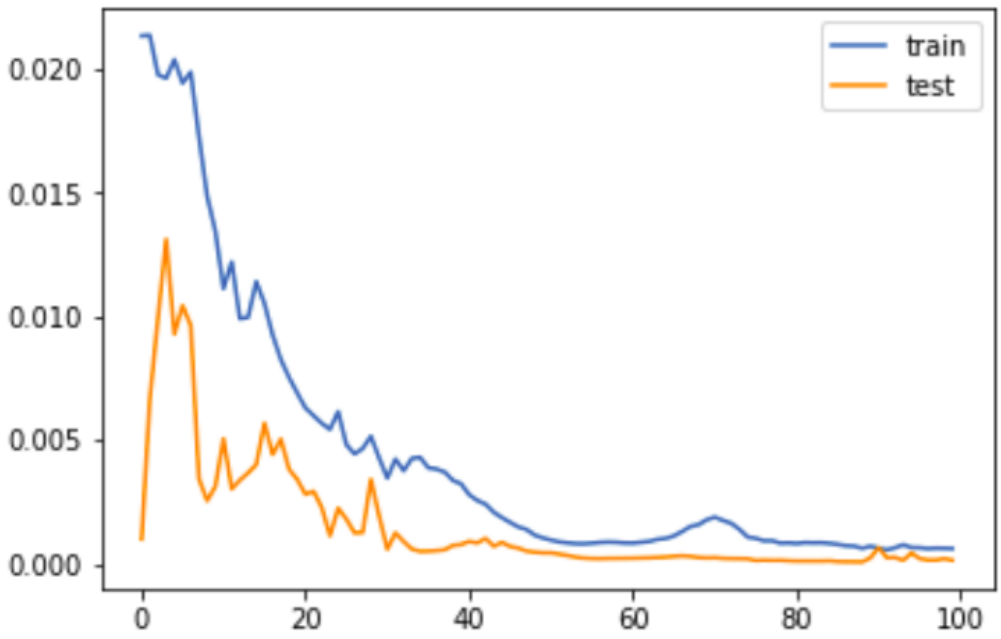
Figura 37 – Algoritmo Adam proposto por Kingma e Ba (2014).

**Require:** $\alpha$: Stepsize
**Require:** $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
**Require:** $f(\theta)$: Stochastic objective function with parameters $\theta$
**Require:** $\theta_0$: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize $1^{st}$ moment vector)
  $v_0 \leftarrow 0$ (Initialize $2^{nd}$ moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  **while** $\theta_t$ not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep $t$)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  **end while**
  **return** $\theta_t$ (Resulting parameters)

Fonte: (KINGMA; BA, 2014)

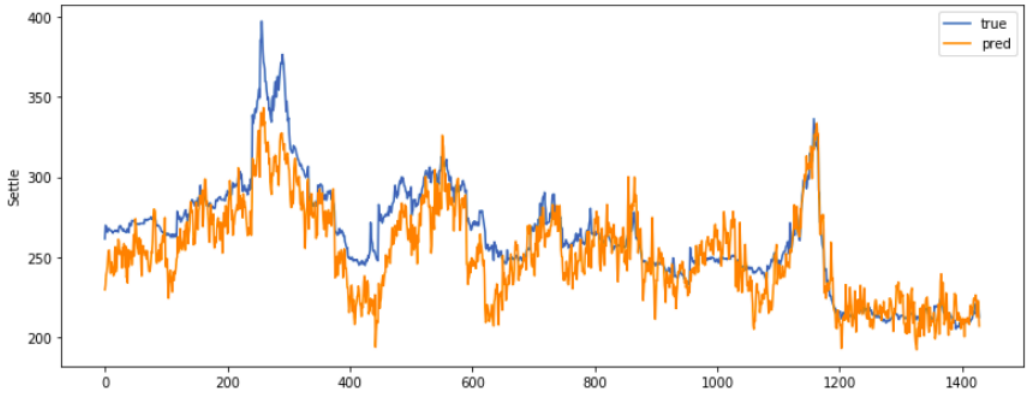
O modelo foi então treinado durante 100 *epochs* em lotes (*batch size*) de tamanho 77. A evolução dos erros de treinamento e de teste são apresentados na Figura 38, enquanto que na Figura 39 é apresentado uma comparação entre a cotação predita para dos dados de teste, e a cotação real.

Figura 38 – Evolução dos erros de teste e treinamento ao longo das épocas.




Fonte: Autor.

Figura 39 – Comparação entre a cotação predita para os dados de teste e a cotação real.



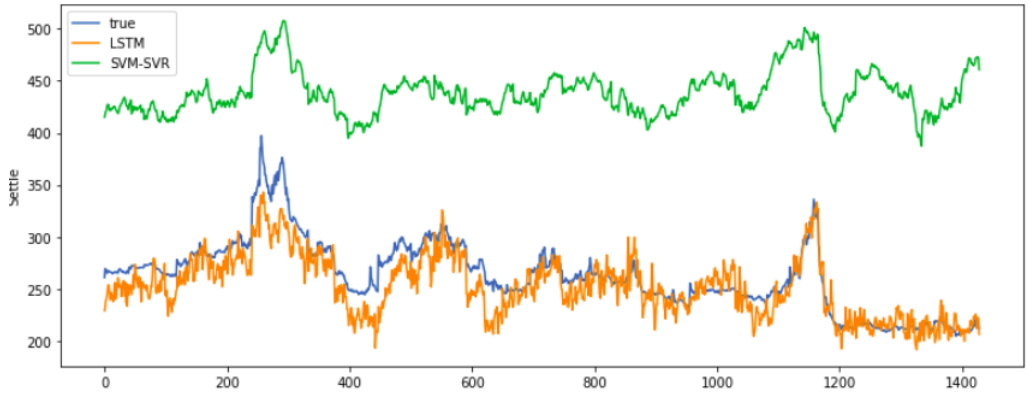

Fonte: Autor.

O RMSE calculado ao término da predição foi 18,789, o que significa que o erro médio entre a cotação predita e a cotação real é de US$ 0,18789.

Um modelo de *Machine Learning* clássico SVM para regressão também foi treinado utilizando a biblioteca *sklearn* (SKLEARN. . . , 2019) e sobre o DatasetMarlon, com a mesma separação entre dados de treino e teste. O resultado da predição do modelo SVM foi avaliado seguindo a métrica RMSE, obtendo um erro médio de 178,381. Comparando com o RMSE do modelo LSTM proposto neste trabalho, pode-se verificar que o modelo LSTM

de *Deep Learning* chegou mais próximo da predizer a cotação da soja do que o modelo de *Machine Learning* clássico SVM. A Figura 40 mostra a cotação real e as cotações previstas pelo modelo LSTM e pelo modelo SVM-SVR para o *dataset* de teste.

Figura 40 – Comparação entre a cotação real e as cotações preditas pelos modelos LSTM e SVM-SVR.




Fonte: Autor.

# 5 Conclusão

A significância do Brasil no cenário mundial da produção e comercialização de soja assim como o fato das cotações de soja seguirem o mesmo padrão de séries temporais, motivou o desenvolvimento de um modelo preditivo capaz de prever cotações na Bolsa de Chicago (CBOT). A relação entre o clima e a produção da soja, assim como a forte influência dos EUA nas cotações fez com que fosse explorada a união de dados históricos da Bolsa de Chicago com dados climáticos dos estados norte-americanos produtores de soja, a fim de ser criado um *dataset* reunindo estas informações denominado DatasetMarlon.

O modelo preditivo escolhido foi o LSTM (Long Short-Term Memory), um tipo de Rede Neural Recorrente (RNN) com uma memória interna que permite uma janela de aprendizado longa sobre uma série temporal. Cada célula LSTM possui três *Gates* (portas) internas responsáveis por decidir o que deve ser esquecido, incorporado à memória ou transmitido para a saída a cada ciclo de aprendizado. A escolha pela utilização da LSTM se deu pelo fato de ser uma Rede Neural Profunda (DNN), possuindo alto desempenho comparada a tecnicas de Aprendizado de Máquina Clássico (*Machine Learning* clássico), conforme evidencia o trabalho de Li e Tam (2017).

O modelo foi desenvolvido possuindo duas camadas com 128 células (neurônios) LSTM cada, e uma camada de saída com apenas uma única célula com função de ativação *linear*, que não altera o valor. Os dados de entrada foram separados em *dataset* de treino (com 9.562 amostras) e de teste (com 1.429 amostras), normalizados e organizados de modo a permitirem que o modelo realize a predição da cotação (variável "Settle") com uma semana de antecedência.

A predição realizada sobre os dados de teste foi avaliada utilizando a métrica Root-Mean-Square Error (RMSE), que calculou um erro médio entre a curva de cotação estimada e a curva real de US$ 0,18789. Este resultado foi comparado ao resultado da predição realizada por um modelo de *Machine Learning* clássico SVM para regressão, que também foi treinado sobre o DatasetMarlon, com a mesma separação entre dados de treino e teste. Comparando o RMSE entre os modelos LSTM e SVM, pôde-se verificar que o modelo LSTM chegou mais próximo de predizer a cotação da soja do que o modelo SVM, evidenciando a melhor performance dos modelos de *Deep Learning* em relação aos de *Machine Learning* clássico.

# Referências


CHICAGO MERCANTILE EXCHANGE  CHICAGO BOARD OF TRADE. *Now Available: Bitcoin Futures*. 2018. Disponível em: <https://www.cmegroup.com/trading/bitcoin-futures.html>. Acesso em: 26 nov 2018.  Citado na página 22.

CHICAGO MERCANTILE EXCHANGE  CHICAGO BOARD OF TRADE. *Timeline of CME Achievements*. 2018. Disponível em: <https://www.cmegroup.com/company/history/timeline-of-achievements.html>. Acesso em: 09 dez 2018.  Citado na página 19.

DEVRIES, H. *Breaking Down the Silos*. 2018. Ellucian. Disponível em: <https://www.ellucian.com/emea-ap/Blog/Breaking-Down-the-Silos/>. Acesso em: 25 nov 2018. Citado na página 20.

DONGES, N. *Recurrent Neural Networks and LSTM*. Towards Data Science, 2018. Disponível em: <https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5>. Acesso em: 26 nov 2018. Citado na página 33.

FRANCO, M. R. C. *marlonrcfranco/soyforecast*. 2019. Disponível em: <https://github.com/marlonrcfranco/soyforecast>. Acesso em: 23 jun 2019.  Citado na página 32.

GLOBAL HISTORICAL CLIMATOLOGY NETWORK. *Data File Access (FTP)*. 2019. Disponível em: <ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/>. Acesso em: 02 jun 2019.  Citado 2 vezes nas páginas 25 e 30.

GLOBAL HISTORICAL CLIMATOLOGY NETWORK. *Global Historical Climatology Network (GHCN)*. 2019. Disponível em: <https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/global-historical-climatology-network-ghcn>. Acesso em: 02 jun 2019.  Citado na página 24.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.  Citado 3 vezes nas páginas 20, 33 e 34.

GOOGLE COLABORATORY. *Welcome to Colaboratory!* 2019. Disponível em: <https://colab.research.google.com/notebooks/welcome.ipynb>. Acesso em: 02 jun 2019. Citado na página 24.

GOOGLE TENSORFLOW. *Recurrent Neural Networks*. 2018. Disponível em: <https://www.tensorflow.org/tutorials/sequences/recurrent#lstm>. Acesso em: 26 nov 2018.  Citado na página 24.

GéRON, A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2017. ISBN 1491962291, 9781491962299.  Citado 2 vezes nas páginas 33 e 35.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, v. 9, n. 8, p. 1735–1780, 1997. Disponível em: <https://doi.org/10.1162/neco.1997.9.8.1735>.  Citado na página 33.

INSTITUTO MATO-GROSSENSE DE ECONOMIA AGROPECUáRIA. Entendendo o mercado da soja. In: . 2017. v. 3, p. 01–48. Disponível em: <http://www.imea.com.br/upload/pdf/arquivos/2015_06_13_Paper_jornalistas_boletins_Soja_Versao_Final_AO.pdf>. Acesso em: 25 nov. 2018. Citado na página 19.

KERAS. *Keras: The Python Deep Learning library*. 2019. Disponível em: <https://keras.io/>. Acesso em: 22 jun 2019. Citado 2 vezes nas páginas 24 e 42.

KINGMA, D. P.; BA, J. *Adam: A Method for Stochastic Optimization*. 2014. Citado 2 vezes nas páginas 10 e 46.

LI, Z.; TAM, V. A comparative study of a recurrent neural network and support vector machine for predicting price movements of stocks of different volatilites. In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. [S.l.: s.n.], 2017. p. 1–8. Citado 3 vezes nas páginas 22, 23 e 49.

LIU, K. Chemistry and nutritional value of soybean components. In: _____. *Soybeans: Chemistry, Technology, and Utilization*. Boston, MA: Springer US, 1997. p. 25–113. ISBN 978-1-4615-1763-4. Disponível em: <https://doi.org/10.1007/978-1-4615-1763-4_2>. Citado na página 17.

MCKINNEY, W. *Python for Data Analysis*. 2th. ed. [S.l.]: O'Reilly Media, 2017. 550 p. Citado na página 24.

MCNALLY, S.; ROCHE, J.; CATON, S. Predicting the price of bitcoin using machine learning. In: *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*. [S.l.: s.n.], 2018. p. 339–343. ISSN 2377-5750. Citado na página 22.

MULTILINGUAL MULTISCRIPT PLANT NAME DATABASE. *Sorting Glycine names*. [S.l.]. Disponível em: <http://www.plantnames.unimelb.edu.au/Sorting/Glycine.html#max>. Acesso em: 25 nov. 2018. Citado na página 17.

NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION. *About our agency*. 2019. Disponível em: <https://www.noaa.gov/about-our-agency>. Acesso em: 02 jun 2019. Citado na página 24.

OLAH, C. *Understanding LSTM Networks*. 2015. Disponível em: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Acesso em: 23 jun 2019. Citado 8 vezes nas páginas 34, 35, 36, 37, 38, 39, 40 e 41.

PAGANO, M. C.; MIRANSARI, M. 1 - the importance of soybean production worldwide. In: MIRANSARI, M. (Ed.). *Abiotic and Biotic Stresses in Soybean Production*. San Diego: Academic Press, 2016. p. 1 – 26. ISBN 978-0-12-801536-0. Disponível em: <http://www.sciencedirect.com/science/article/pii/B9780128015360000013>. Citado na página 17.

PENNE, A. *Get NOAA GHCN Data*. 2019. Disponível em: <https://github.com/aaronpenne/get_noaa_ghcn_data>. Acesso em: 02 jun 2019. Citado na página 25.

PHAM, X.; STACK, M. How data analytics is transforming agriculture. *Business Horizons*, v. 61, n. 1, p. 125 – 133, 2018. ISSN 0007-6813. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0007681317301325>. Citado 4 vezes nas páginas 17, 18, 20 e 21.

PROJECT JUPYTER. *The Jupyter Notebook*. 2018. Disponível em: <http://jupyter.org/index.html>. Acesso em: 26 nov 2018. Citado na página 24.

QUANDL API FOR COMMODITY DATA. *API for Commodity Data*. 2013. Disponível em: <https://blog.quandl.com/api-for-commodity-data>. Acesso em: 26 nov 2018. Citado na página 24.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, Nature Publishing Group SN -, v. 323, p. 533 EP –, Oct 1986. Disponível em: <https://doi.org/10.1038/323533a0>. Citado na página 33.

SCIPY. *NumPy*. 2019. Disponível em: <https://www.numpy.org/>. Acesso em: 22 jun 2019. Citado na página 24.

SKLEARN.SVM.SVR. 2019. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html>. Acesso em: 23 jun 2019. Citado na página 47.

UNITED STATES DEPARTMENT OF AGRICULTURE. *Oilseeds: World Markets and Trade*. 2018. Foreign Agricultural Service/USDA - Office of Global Analysis. Disponível em: <https://apps.fas.usda.gov/psdonline/circulars/oilseeds.pdf>. Acesso em: 25 nov 2018. Citado na página 17.

UNITED STATES DEPARTMENT OF AGRICULTURE. *Production – Measured in Bushels – United States*. 2018. National Agricultural Statistics Service. Disponível em: <https://www.nass.usda.gov/Data_Visualization/index.php>. Acesso em: 26 nov 2018. Citado 2 vezes nas páginas 28 e 29.

WANG, F. Forecasting agricultural commodity prices through supervised learning. *month*, v. 2016, p. 11–11, 2017. Citado na página 22.