Universidade Federal
do Rio de Janeiro
Escola Politécnica

ANÁLISE DE NOTÍCIAS DO MERCADO FINANCEIRO UTILIZANDO PROCESSAMENTO DE LINGUAGEM NATURAL E APRENDIZADO DE MÁQUINA PARA DECISÕES DE SWING TRADE

Lucas Gama Canto

Projeto de Graduação apresentado ao Curso de Engenharia de Controle e Automação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador: Heraldo Luís Silveira de Almeida

Rio de Janeiro
Março de 2020

ANÁLISE DE NOTÍCIAS DO MERCADO FINANCEIRO UTILIZANDO PROCESSAMENTO DE LINGUAGEM NATURAL E APRENDIZADO DE MÁQUINA PARA DECISÕES DE SWING TRADE

Lucas Gama Canto

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE AUTOMAÇÃO.

Examinado por:

Prof. [TODO]Nome do Primeiro Examinador Sobrenome, D.Sc.

Prof. [TODO]Nome do Segundo Examinador Sobrenome, Ph.D.

Prof. [TODO]Nome do Terceiro Examinador Sobrenome, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2020

Gama Canto, Lucas

Análise de Notícias do Mercado Financeiro Utilizando Processamento de Linguagem Natural e Aprendizado de Máquina Para Decisões de Swing Trade/Lucas Gama Canto. – Rio de Janeiro: UFRJ/ Escola Politécnica, 2020.

XIII, 28 p.: il.; 29, 7cm.

Orientador: Heraldo Luís Silveira de Almeida

Projeto de Graduação – UFRJ/ Escola Politécnica/ Curso de Engenharia de Controle e Automação, 2020.

Referências Bibliográficas: p. 19 – 21.

1. Aprendizado de Máquina. 2. Processamento de Linguagem Natural. 3. Mercado Financeiro. I. Silveira de Almeida, Heraldo Luís. II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia de Controle e Automação. III. Título.

Ao povo brasileiro, pela total contribuição em minha graduação.

Agradecimentos

Gostaria de agradecer a todas as pessoas e situações que tornaram este momento possível. Em especial, meus pais Benedita e Manoel, pelo suporte e esforço incondicional em apoiar minha decisão de vir estudar engenharia no Rio de Janeiro, aos professores da graduação, que me fizeram evoluir no âmbito acadêmico, profissional e pessoal, em especial ao meu orientador e professor Heraldo, que não mediu esforços para me ajudar neste trabalho, e aos amigos que me apoiaram e participaram do meu processo de graduação.

Resumo do Projeto de Graduação apresentado à Escola Politécnica/ UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Automação.

ANÁLISE DE NOTÍCIAS DO MERCADO FINANCEIRO UTILIZANDO PROCESSAMENTO DE LINGUAGEM NATURAL E APRENDIZADO DE MÁQUINA PARA DECISÕES DE SWING TRADE

Lucas Gama Canto

Março/2020

Orientador: Heraldo Luís Silveira de Almeida

Curso: Engenharia de Controle e Automação

Com o objetivo de automatizar análises fundamentalistas de mercado, o uso de tecnologia para processamento de texto vem sendo utilizado constantemente no meio acadêmico[1] e profissional[2]. De forma a contribuir para este campo em crescimento, este trabalho discorre um estudo acerca da criação de modelos preditivos sobre a valorização ou desvalorização de ações na bolsa de valores do Brasil (B3, antiga Bovespa) a partir de notícias sobre o mercado brasileiro de forma a auxiliar decisões de Swing Trade, ou seja, compra e venda de ações dentro de uma janela de tempo maior que um dia.

Para isto, o presente projeto utiliza o framework PyText, que se baseia em conceitos de Aprendizado de Máquina, Redes Neurais e Processamento de Linguagem Natural de forma a desenvolver modelos preditivos com a tarefa de classificação textual.

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Engineer.

FINANCIAL MARKET NEWS ANALYSIS USING NATURAL LANGUAGE PROCESSING AND MACHINE LEARNING FOR SWING TRADE DECISIONS

Lucas Gama Canto

March/2020

Advisor: Heraldo Luís Silveira de Almeida

Course: Automation and Control Engineering

In order to automate fundamental market analysis, the use of text processing technology has been constantly used in academic[1] and professional[2] means. To contribute to this growing field, this paper discusses a study about the creation of predictive models regarding the valuation or devaluation of shares on the Brazilian stock exchange (B3, former Bovespa) based on news about the Brazilian market in order to assist Swing Trade decisions, that is, buying and selling stocks within a time window longer than one day.

To this end, the present project uses the PyText framework, which is based on Machine Learning, Neural Networks and Natural Language Processing concepts in order to develop predictive models with the task of textual classification.

Sumário

Lista de Figuras **x**

Lista de Tabelas **xi**

Lista de Símbolos **xii**

Lista de Abreviaturas **xiii**

1 Introdução **1**

1.1 Tema 1
1.2 Delimitação 1
1.3 Justificativa 2
1.4 Objetivos 2
1.5 Metodologia 2
1.6 Descrição 3

2 Fundamentação Teórica **4**

2.1 Bolsa de Valores e Ações 4
2.1.1 Preços de Ações 4
2.1.2 Índice de Bolsa de Valores 6
2.2 Aprendizado de Máquina 6
2.2.1 Aprendizado Supervisionado 7
2.2.2 Aprendizado Não Supervisionado 8
2.2.3 Avaliação de Desempenho 8
2.3 Redes Neurais 10
2.3.1 Redes Neurais Convolucionais 11
2.3.2 Redes Neurais Recorrentes 12
2.3.3 *Dropout* 13
2.4 Processamento de Linguagem Natural 13
2.4.1 Pré-processamento de Sintaxe 13
2.4.2 Representação Vetorial 14
2.5 PyText 14

2.5.1 Configuração 14
2.5.2 Uso 14

3 Obtenção e Pré-processamento de Dados **15**
3.1 Estratégia 15
3.2 Pré-processamento 15

4 Treinamento **16**
4.1 Configuração 16
4.2 Rotina de Treinamento 16
4.3 Análise 16

5 Considerações Finais **18**
5.1 Conclusão 18
5.2 Trabalhos Futuros 18

Referências Bibliográficas **19**

A Rotinas e Arquivos de Configuração **22**
A.1 Rotina de Pré-processamento e Criação de Conjuntos de Dados . . . 22
A.2 Rotina de Criação de Arquivos de Configuração e Execução de Treinamentos 26
A.3 Modelo de Arquivo de Configuração 27

Lista de Figuras

2.1 Os três níveis da HME, cada nível adiciona um tipo de informação cujo com o qual não seria possível prever um movimento de preço no mercado[3]. 5
2.2 Exemplos das estratégias de *Undersampling* e *Oversampling* em um problema de classificação de 2 classes desbalanceadas[4]. 8
2.3 Exemplo de rede neural com duas camadas ocultas[5]. 10
2.4 Neurônio de uma rede neural[6]. 10
2.5 Exemplo de um dado de entrada e um *kernel*[7]. 11
2.6 Exemplo de convolução da imagem anterior[7]. 12
2.7 Exemplo de *max-pooling* utilizando uma submatriz de tamanho 2x2 e passo 2[7]. 12
2.8 Exemplo de aplicação do *Bag-of-Words*[8]. 14

Lista de Tabelas

2.1 Os 5 ativos com o maior volume e participação na B3, associados ao Ibovespa[9]. 6

4.1 Acurácia para o caso de 3 classes. 16
4.2 Precisão média para o caso de 3 classes. 16
4.3 Cobertura média para o caso de 3 classes. 17
4.4 Medida F1 média para o caso de 3 classes. 17
4.5 Acurácia para o caso de 2 classes. 17
4.6 Precisão média para o caso de 2 classes. 17
4.7 Cobertura média para o caso de 2 classes. 17
4.8 Medida F1 média para o caso de 2 classes. 17

Lista de Símbolos

$\emptyset$ EXEMPLO-Conjunto vazio, p. 1

Lista de Abreviaturas

iBovespa	Índice Bovespa, p.2
HME	Hipótese do Mercado Eficiente, p.4
AT	Análise Técnica, p.4
AF	Análise Fundamentalista, p.4
MLP	*Multilayer Perceptron*, p.11
CNN	*Convolutional Neural Network*, p.11
ReLU	*Rectified Linear Unit*, p.12
PNL	Processamento de Linguagem Natural, p.TODO
BOW	*Bag-of-Words*, p.TODO

Capítulo 1

Introdução

1.1 Tema

O tema deste trabalho se resume no estudo da criação de modelos preditivos de modo que estes possam prever a valorização ou desvalorização de ações da bolsa de valores por meio do processamento de notícias do mercado brasileiro.

Deste modo, o problema a ser abordado é a identificação de quando uma notícia pode impactar positivamente ou negativamente a variação de preço de ações de forma automatizada.

1.2 Delimitação

Este trabalho se restringe ao processamento de texto em português brasileiro, tendo como foco a predição da variação de preço das ações que fazem parte da bolsa de valores do Brasil, a B3. Pela indisponibilidade de dados sobre notícias brasileiras contendo a informação do horário de lançamento da notícia, o projeto mira em predições dentro de uma janela de tempo maior que um dia, de forma a auxiliar decisões de Swing Trade, isto é, operações de compra e venda de ações numa janela de tempo maior que um dia.

Além disso, o estudo se baseia na ferramenta PyText, um framework recentemente desenvolvido pelo Facebook que providencia modelos de processamento de linguagem natural de última geração através de uma interface simples e extensível[10].

1.3 Justificativa

Diante do crescente número de investidores na bolsa de valores no Brasil, nota-se uma maior preocupação da população brasileira acerca da busca por independência financeira e fontes alternativas de renda com o intuito de contribuir à economia familiar, previdência, ou mesmo utilizar este método como fonte principal de renda[11].

Ao mesmo tempo, estudos associados à inteligência artificial, aprendizado de máquina e processamento de linguagem natural continuam emergindo no meio acadêmico e auxiliando o meio profissional como nunca antes, incluindo o mercado financeiro[12].

Através destes dois fatores, o presente trabalho busca contribuir para a difusão do estudo e uso de algumas destas tecnologias sobre um assunto que gradualmente se encontra dentro do interesse da população brasileira e que colabora para uma possível instauração de uma cultura de economia e independência financeira no Brasil.

1.4 Objetivos

O objetivo geral do presente trabalho é de analisar modelos preditivos associados ao mercado financeiro que possam ser construídos a partir do framework PyText, tendo como objetivos específicos, apresentar: (1) A busca por dados de notícias e do histórico da bolsa de valores; (2) A lógica utilizada para a união destes dados de forma a construir os conjuntos de dados utilizados no treinamento dos modelos; (3) O pré-processamento dos conjuntos de dados; (4) As possíveis configurações do framework utilizado de forma a obter a melhor performance; (5) O detalhamento e a análise dos modelos finais encontrados.

1.5 Metodologia

O trabalho teve início a partir da procura por bases de dados de notícias associadas ao mercado brasileiro e escritas em português do Brasil, seguida pela obtenção do histórico das variações de preço dos ativos que compõem o iBovespa. Após isto, o histórico foi filtrado de forma a manter as informações dos 5 ativos mais significativos e das varições destes ativos que ocorreram dentro da mesma janela de tempo das notícias obtidas. Em seguida, estes dados foram unidos de forma a obter 5 conjuntos de dados para cada ativo, cada um levando em consideração uma diferente janela de tempo para indicar a valorização: de 1 a 5 dias.

Logo após, houve a etapa de pré-processamento do corpo das notícias de forma a remover possíveis ruídos e facilitar a etapa de treinamento, sem perda de contexto do conteúdo. Com os conjuntos de dados prontos, foram feitos testes no PyText com o objetivo de definir a melhor configuração possível para a natureza dos dados, e assim obter a melhor performance.

Por fim, os testes finais de cada modelo gerado foi detalhado e analisado para permitir uma conclusão e avaliação do processo como um todo.

1.6 Descrição

O capítulo 2 apresenta toda a fundamentação teórica utilizada como base para o projeto a partir de uma breve descrição de como a bolsa de valores funciona seguida de explicações sobre Aprendizado de Máquina, Processamento de Linguagem Natural, Redes Neurais e o framework Pytext.

No capítulo 3 é detalhado todo o processo executado para obtenção do conjunto de notícias e do histórico da B3, seguido do pré-processamento realizado nestes dois conjuntos e a criação dos conjuntos de dados finais utilizados para o treino, cada um associado a um ativo e uma janela de tempo específica.

Os detalhes das configurações utilizadas no PyText e o treinamento em si é especificado no capítulo 4, onde há uma discussão acerca dos parâmetros encontrados para a geração de modelos mais performáticos, além das métricas finais encontradas para cada modelo gerado.

Por fim, o capítulo 5 apresenta uma conclusão acerca dos modelos encontrados seguido por sugestões que futuramente podem ser aplicadas para a evolução do tema e uma possível melhora de desempenho dos modelos preditivos.

O código desenvolvido para o pré-processamento e geração dos conjuntos de dados e arquivos de configurações do PyText utilizados para a geração dos modelos podem ser encontrados no repositório do github referenciado em [13].

Capítulo 2

Fundamentação Teórica

2.1 Bolsa de Valores e Ações

A Bolsa de Valores é um lugar centralizado onde, além de abranger outros tipos de investimento, se negociam ações (também chamados de ativos ou papéis), isto é, parcelas do capital social de empresas de capital aberto. Atualmente a B3 (Brasil, Bolsa, Balcão) é a Bolsa de Valores oficial do Brasil que em 2017 atingiu a 5ª posição das maiores bolsas de mercados de capitais do mundo em valor de mercado, com um patrimônio de US$ 13 bilhões[14].

As ações são negociadas diariamente a partir das ordens de compra e venda emitidas pelas corretoras durante o pregão eletrônico, que na B3, funciona em dias úteis das 10:00 às 17:00.

2.1.1 Preços de Ações

O preço de um ativo na Bolsa de Valores pode ser determinado por diversas razões que podem se relacionar entre si, entre essas, pode-se destacar a lei da oferta e demanda, perspectivas de crescimento da empresa associada ao papel e especulação. A previsibilidade acerca de movimentações no mercado de ações normalmente pode ser baseada em Análise Técnica (AT - estudo dos movimentos do mercado baseado em métricas como preço, volume e taxa de juros[15]), Análise Fundamentalista (AF - estudo feito a partir de resultados financeiros e operacionais, indicando a saúde da empresa[16]) ou numa junção destes dois conceitos.

A validade da previsibilidade destas movimentações são questionadas por críticas com base na Hipótese do Mercado Eficiente (HME) e seus três níveis definidos em [17]:

- HME fraca: Afirma que os preços refletem totalmente a informação contida na sequência histórica dos preços. Ou seja, a AT não consegue prever os movimentos futuros pois os preços passadas só podem descrever o presente.
- HME semi-forte: Afirma que os preços presentes não só refletem toda a sequência histórica de preços mas também toda informação pública sobre as organizações associadas ao ativo em questão. Neste nível de eficiência, a AF também não seria capaz de prever movimentos futuros, pois toda informação como demonstrativos de resultados ou análises orçamentárias refletiria apenas o preço presente.
- HME forte: Neste nível, é afirmado que *toda* informação conhecida sobre as organizações é totalmente refletida pelo preço presente, logo, nem mesmo aqueles com informações privilegiadas podem utilizar isto como ferramenta para prever preços futuros.

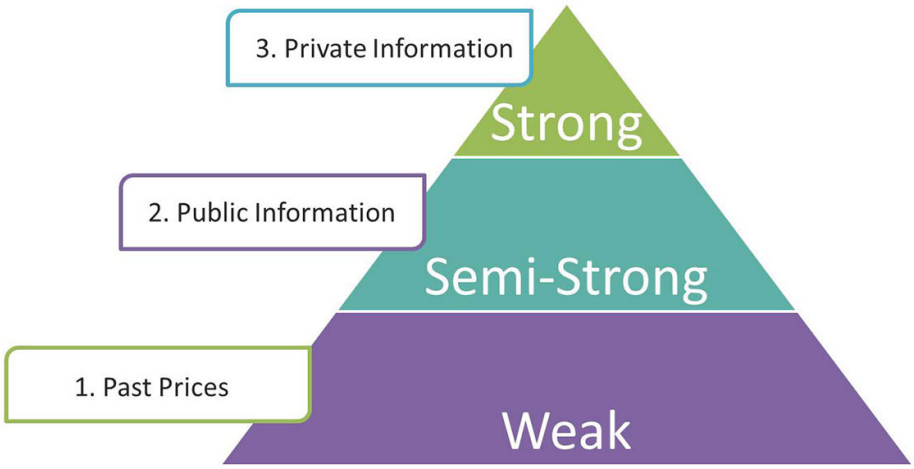

Figura 2.1: Os três níveis da HME, cada nível adiciona um tipo de informação cujo com o qual não seria possível prever um movimento de preço no mercado[3].

Não há uma reposta correta perante a validade da HME. Porém, muitos acadêmicos acreditam, pelo menos, na HME fraca[18], fazendo com que, em algumas ocasiões, seja preferível a utilização da AF, ou seja, a análise de resultados financeiros, relatórios anuais e notícias divulgadas acerca do mercado financeiro.

2.1.2 Índice de Bolsa de Valores

Com objetivo de parametrizar algumas informações intrínsecas às bolsas, estas disponibilizam diversos índices. O principal índice da B3 é o Ibovespa, que é formado pelos ativos com maior volume negociado na bolsa nos últimos meses e indica, de forma resumida, o desempenho das ações negociadas na B3. Por ser um indicador principal, muitos fundos de investimento baseados no mercado de ações estão atrelados ao Ibovespa, contribuindo para a atratividade destes ativos de maneira geral.

Tabela 2.1: Os 5 ativos com o maior volume e participação na B3, associados ao Ibovespa[9].

Código	Ação	Qtde. Teórica	Part.(%)
ITUB4	Itaú Unibanco	4.738.562.684	9,095
PETR4	Petrobras	4.520.185.835	7,038
BBDC4	Bradesco	3.873.597.664	7,028
VALE3	Vale S.A.	3.147.743.563	8,414
ABEV3	AMBEV	4.344.066.764	4,173

2.2 Aprendizado de Máquina

Pode-se definir Aprendizado de Máquina como o campo de estudo de algoritmos com o objetivo de fazer com que computadores possam agir sem serem explicitamente programados para fazer determinada tarefa. São algoritmos que analisam dados e aprendem com eles, gerando um modelo preditivo que pode fornecer uma predição de algo on mundo.

Outra definição dada por Tom M. Mitchell[19] fala que o campo de Aprendizado de Máquina busca responder a pergunta: "Como podemos construir sistemas de computadores que possam automaticamente melhorar através de experiência e quais são as leis fundamentais que governam todo este processo de aprendizado?". Outra definição do mesmo autor[20] diz que "Um programa de computador é dito aprender com a experiência E em respeito a uma tarefa T e medida pelo desempenho P se o seu desempenho em T, medido por P, melhora com a experiência E". Neste conceito, se fosse desejado um programa de computador que aprendesse a classificar e-mails como spam ou não, por exemplo, poderíamos fazer a seguinte associação:

- E = A experiência de ver o usuário classificar emails como spam ou não.
- T = A tarefa de classificar os emails.
- P = O número ou fração de emails corretamente classificados como spam/não spam

Geralmente, os algoritmos de Aprendizado de Máquina podem ser divididos em dois tipos: Aprendizado Supervisionado e Aprendizado Não Supervisionado.

2.2.1 Aprendizado Supervisionado

Neste tipo, o algoritmo é inicialmente servido por uma série de dados rotulados cujo resultado já é conhecido. A ideia e que o algoritmo aprenda a criar uma estratégia para chegar ao resultado baseando-se nesses dados de modelo inicial. O aprendizado supervisionado pode ser dividido em problemas de regressão ou classificação.

Problema de Regressão

Neste tipo de problema, os dados de entrada (parâmetros) são mapeados em uma função contínua. Por exemplo, um algoritmo cujo objetivo fosse prever o preço dos imóveis na cidade do Rio de Janeiro baseando-se em dados como área útil, bairro, número de vagas na garagem, etc é considerado como um problema de regressão, pois o resultado final será um número contínuo, neste caso, o preço dos imóveis.

Problema de Classificação

Neste caso, os parâmetros são mapeados de forma a classificar os dados em categorias distintas. Por exemplo, um algoritmo utilizado para prever se um tumor é benigno ou maligno a partir de dados como o tamanho, rugosidade do tumor e idade do paciente é considerado um problema de classificação, pois o resultado final será a categoria na qual o tumor pertence.

Os problemas de classificação muitas vezes apresentam desbalanceamento de classes no conjunto de dados utilizado para o treinamento do modelo, ou seja, o conjunto de dados pode apresentar poucas amostras de uma determinada classe em relação às outras envolvidas, o que pode ocasionar falhas na predição da classe minoritária.

De forma a corrigir tal problema, existem algumas técnicas que podem ser baseadas em dois conceitos: *Undersampling* e *Oversampling*. A primeira se resume na remoção de amostras das classes majoritárias, e a segunda, no acréscimo de amostras da classe minoritária a partir das amostras já existentes no conjunto, de forma a se obter um conjunto de dados balanceados.

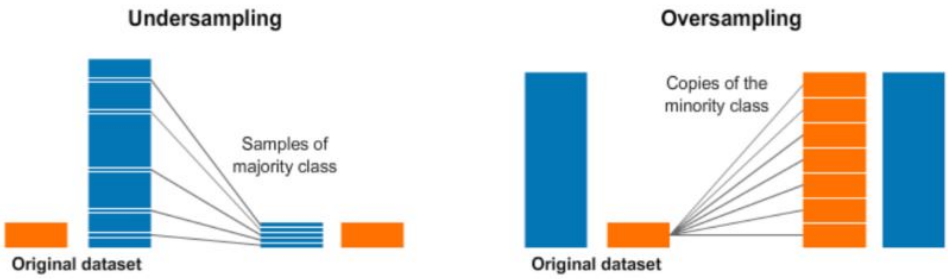

Figura 2.2: Exemplos das estratégias de *Undersampling* e *Oversampling* em um problema de classificação de 2 classes desbalanceadas[4].

Entretanto, estas técnicas podem gerar alguns efeitos negativos, como por exemplo, a demasiada diminuição do conjunto de dados como um todo no *Undersampling*, o que prejudica o aprendizado. No caso do *Oversampling*, pode-se gerar um maior efeito de sobre-ajuste no modelo preditivo, isto é, pela falta de generalização, o modelo consegue prever muito bem amostras do conjunto de dados de treino, mas se mostra ineficaz ao prever dados de teste.

2.2.2 Aprendizado Não Supervisionado

Neste tipo de aprendizado, não existe um conjunto inicial de dados e resultados, ou seja, nos permite abordar problemas onde temos pouca ou nenhuma ideia do que nossos resultados devem aparentar. Um exemplo, seria um algoritmo onde, utilizando uma coleção de 1000 artigos publicados por uma universidade, fizesse um agrupamento desses artigos, baseando-se em diferente variáveis como frequência de palavras semelhantes, número de paginas, etc.

2.2.3 Avaliação de Desempenho

A avaliação de desempenho de um modelo preditivo pode ser realizada através de diversas métricas que são medidas diante de uma previsão do modelo sobre um conjunto de dados de teste, logo, existe uma necessidade em dividir o conjunto de dados inicial em dados de treino e dados de teste.

Não existe um modo ideal de dividir o conjunto de dados, o tamanho do conjunto de treino normalmente é maior que o de teste, de modo que este consiga abranger mais generalizações acerca dos parâmetros do modelo. Assim, algumas proporções são mais comumente usadas, como 60/40, 75/25 e 80/20, proporção baseada no Principio de Pareto que afirma que, 80% das saídas/consequências vem de 20%

das entradas/causas[21]. Levando em consideração o escopo de Aprendizado de Máquina, podemos dizer que 20% pode mapear 80% do conjunto de dados.

Além disso, em modelos mais simples, também existe a possibilidade de utilizar a Validação Cruzada, uma técnica que separa o conjunto de dados em subconjuntos exclusivos e diferentes e alguns destes subconjuntos são utilizados para treino e outros para teste, de forma iterativa. Um dos métodos de validação cruzada mais famosos é o *k-fold*, onde o conjunto de dados é separado em k subconjuntos e o treino é realizado k vezes, cada vez utilizando um subconjunto diferente para teste e o resto para treino[22]. No final, as métricas de avaliação são definidas como a média diante dos k subconjuntos.

Métricas

As métricas utilizadas para avaliação dependem do tipo de problema. Por exemplo, em problemas de regressão é comum utilizar o erro quadrático médio[23]. Nos problemas de classificação, é comum utilizar as seguintes medidas: Acurácia, Precisão, Cobertura e a Medida F1:

- Acurácia: É a medida que define a assertividade do modelo em geral, se resume na porcentagem de acertos dentre todas as previsões feitas no conjunto de teste.

$$A = \frac{\text{Número total de acertos}}{\text{Número total de palpites}}$$

- Precisão: Medida de assertividade referente a uma classe específica. É a porcentagem de acertos dentre todos os palpites de uma classe.

$$P_X = \frac{\text{Número total de acertos da classe X}}{\text{Número total de palpites da classe X}}$$

- Cobertura: Porcentagem de palpites certos dentro do número de amostras de uma classe específica.

$$C_X = \frac{\text{Número total de acertos da classe X}}{\text{Número total de amostras da classe X}}$$

- Medida F1: Média harmônica entre Precisão e Cobertura.

$$F1_X = 2\frac{P_X C_X}{P_X + C_X}$$

2.3 Redes Neurais

Redes neurais são estruturas matemáticas baseadas no funcionamento do cérebro humano. O campo que estuda a aplicação de redes neurais com várias camadas de processamento em métodos de Aprendizado de Máquina é chamado de Aprendizagem Profunda (do inglês, *Deep Learning*). Em sua forma mais simples, uma rede neural contém três camadas: entrada (*input layer*), camada oculta (*hidden layer*) e saída (*output layer*), onde a camada oculta pode ser única ou múltipla. Cada camada é composta por neurônios, também chamados de nós.

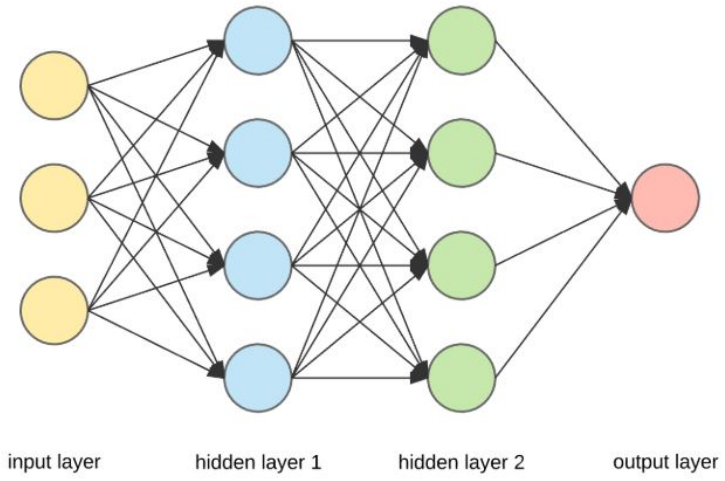

Figura 2.3: Exemplo de rede neural com duas camadas ocultas[5].

Cada nó de uma rede neural representa uma abstração matemática, esta é definida pela função de ativação (*Activation Function*) que recebe o somatório das entradas (*Inputs*) multiplicadas por seus respectivos pesos e mais um viés (*Bias*).

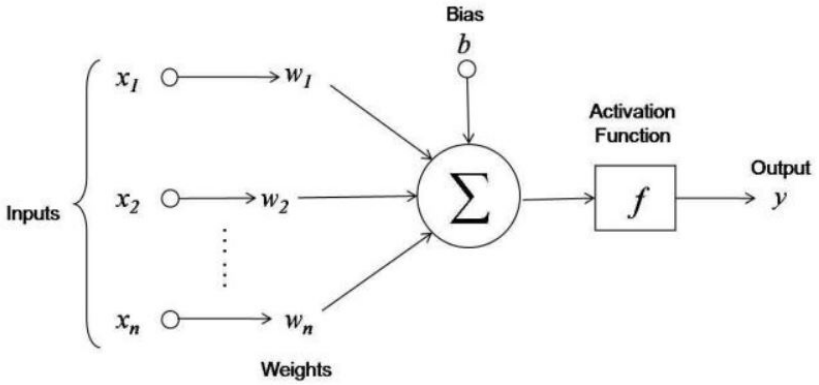

Figura 2.4: Neurônio de uma rede neural[6].

Durante a etapa de treinamento, estes pesos são ajustados de forma que se obtenha saídas iguais ou suficientemente próximas às saídas do conjunto de dados utilizado para treino, isso pode acontecer através de diversas técnicas, sendo a *backpropagation* umas das mais conhecidas, que utiliza descida de gradiente[24] para minimizar a função custo determinada pelas entradas, viés, função de ativação e saída dos neurônios.

Este modelo simples de rede neural com apenas uma camada oculta se chama *Perceptron* e os modelos com múltiplas camadas são chamados de *Multilayer Perceptron* (MLP). Apesar de terem significante poder preditivo, estas redes podem apresentar falhas, principalmente quando o número de camadas ocultas alcança um valor muito grande ou quando os dados de entrada apresentam uma alta dimensão. Mesmo após algumas tentativas de solucionar estas falhas das redes MLP, outras arquiteturas de redes neurais foram propostas.

2.3.1 Redes Neurais Convolucionais

As Redes Neurais Convolucionais (*Convolutional Neural Network - CNN*) foram desenvolvidas com o intuito de facilitar a classificação de dados de alta dimensão, como imagens e textos. Em um problema de classificação de imagem com uma rede MLP, cada pixel está diretamente ligado a uma entrada, ocasionando um vetor de entrada de alta dimensão. Em uma rede CNN, normalmente, este vetor de entrada sofre diversas reduções de dimensionabilidade dentro de três camadas: camada de convolução, camada de função de ativação e camada de *pooling*.

Na primeira camada, ocorre a convolução do dado de entrada com um filtro (*kernel*), que também é treinado ao longo do treinamento de toda a rede.

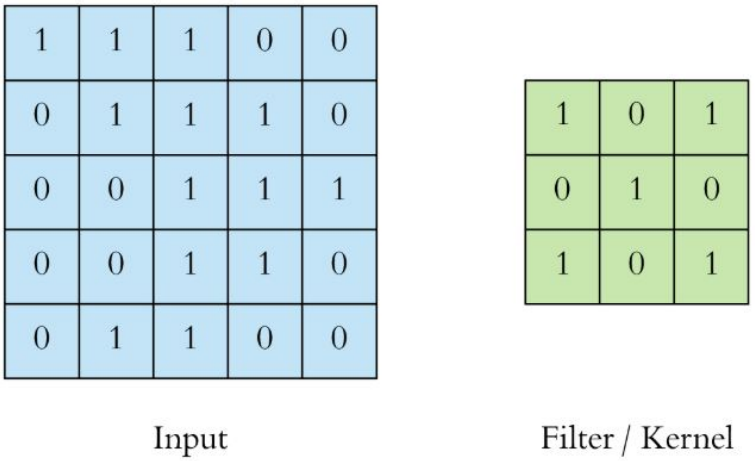

Figura 2.5: Exemplo de um dado de entrada e um *kernel*[7].

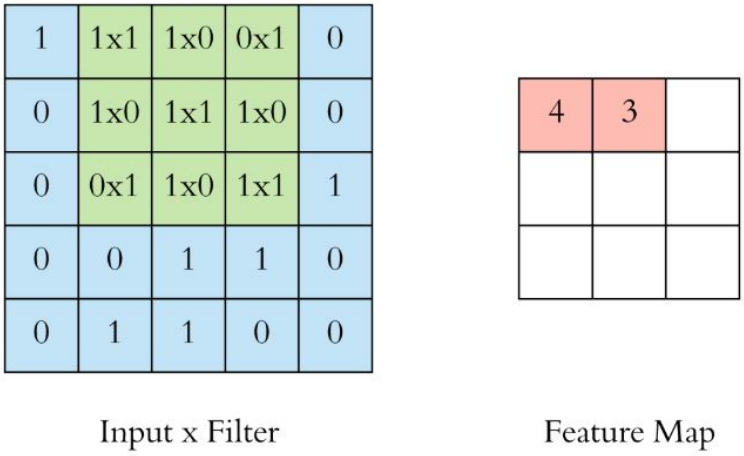

Figura 2.6: Exemplo de convolução da imagem anterior[7].

Após esta etapa, o resultado da camada convolução passa por funções de ativação. Nestas, normalmente são atribuídas funções ReLU (*Rectified Linear Unit*), análoga à função rampa, ou seja, valores menores que zero são zerados. Por final, o resultado da camada de função de ativação passa pelo *pooling* onde a redução de dimensibilidade pode ser feita através de diversas técnicas, sendo a *max-pooling* a mais utilizada. Esta consiste em selecionar os maiores valores dentro de submatrizes de tamanho e espaçamento (passos) específicos, como indicado na figura abaixo.

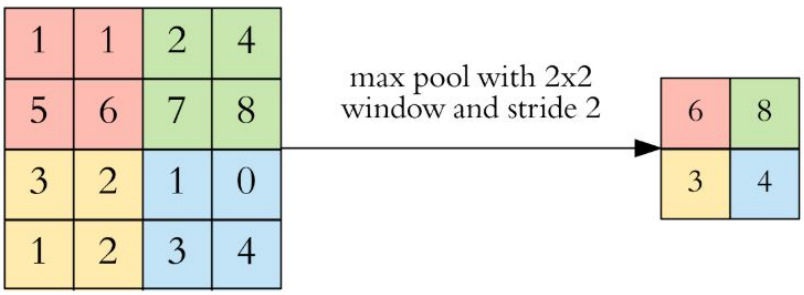

Figura 2.7: Exemplo de *max-pooling* utilizando uma submatriz de tamanho 2x2 e passo 2[7].

Após estas camadas, o resultado pode ser utilizado na camada de entrada de uma rede comum, normalmente, uma rede MLP.

2.3.2 Redes Neurais Recorrentes

As Redes Neurais Recorrentes (*Recurrent Neural Networks - RNN*) são redes que foram desenvolvidas principalmente para problemas associados à predição de sequência de dados, como a predição de palavras em um texto de entrada. TODO

2.3.3 *Dropout*

TODO

2.4 Processamento de Linguagem Natural

Processamento de Linguagem Natural (PNL) se resume ao campo de estudo das tecnologias utilizadas para ajudar computadores a entenderem a linguagem natural dos humanos, é também considerado uma subárea da Inteligência Artificial. Pode ser usado em diversas aplicações[25], como por exemplo:

- Aplicativos de tradução de idioma, como o Google Translator
- Processamento de palavras, que empregam PNL para a correção gramática
- Resposta de Voz iterativa em call centers, de forma a responder adequadamente conforme a requisição de usuários
- Assistentes pessoais, como OK Google, Siri, Cortana e Alexa

Geralmente, o PNL abrange um pré-processamento de texto antes deste ser transformado em uma forma inteligível por computadores, de forma a remover ruídos ou facilitar o processamento, e isto pode ocorrer de diversas formas.

2.4.1 Pré-processamento de Sintaxe

A sintaxe se refere à forma de como as palavras se organizam em uma sentença para que se obtenha sentido gramatical. Estas são algumas técnicas de sintaxe que podem ser utilizadas no pré-processamento de texto:

- Stemização: É a transformação de palavras flexionadas para sua forma radical. Por exemplo, as palavras “estudos”, “estudar” e “estudando” se transformariam apenas em “estud”, mas a palavra “tiver"se transformaria em “tiv” e “tenho"se transformaria em “tenh”.
- Lematização: Semelhante à Stemização, porém, a palavra é resumida para seu lema, fazendo com que se alcance um nível maior de abstração. Neste caso, tanto a palavra “tiver” como “tenho” se transformaria no lema “ter”.
- Remoção de *stopwords*: A remoção de “palavras de parada”, ou seja, palavras como “a”, “de”, “o”, “da”, “que”, “e”, “do” é útil pois, na maioria das vezes, não são informações importantes para construção do modelo.

Além destas, outra técnicas mais simples são utilizadas, como a transformação de caracteres maiúsculos para minúsculos.

2.4.2 Representação Vetorial

Após o pré-processamento textual, diversas técnicas podem ser utilizadas para a transformação do texto em números. Uma das formas mais simples de se fazer isso é o *Bag-of-Words* (BOW), que consiste em simbolizar textos de um conjunto de dados através de uma matriz na qual cada coluna é associada a uma palavra existente no conjunto de dados. Cada linha da matriz representa um texto e as variáveis indicam o número de ocorrências de uma palavra específica.

Figura 2.8: Exemplo de aplicação do *Bag-of-Words*[8].

Apesar da praticidade e simplicidade, o BOW apresenta alguns problemas, entre eles, a perda da informação de ordem das palavras e a alta dimensão da representação vetorial, fazendo com que muitos busquem outra alternativa.

Word2Vec

TODO

2.5 PyText

TODO

2.5.1 Configuração

TODO

2.5.2 Uso

TODO

Capítulo 3

Obtenção e Pré-processamento de Dados

TODO

3.1 Estratégia

TODO

3.2 Pré-processamento

TODO

Capítulo 4

Treinamento

TODO

4.1 Configuração

TODO

4.2 Rotina de Treinamento

TODO

4.3 Análise

TODO

Tabela 4.1: Acurácia para o caso de 3 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	57,34	57,78	58,15	57,32	58,34
2	57,63	57,82	X	56,67	55,43
3	57,96	57,25	X	58,27	57,97
4	58,77	60,21	57,84	58,32	59,55
5	57,57	59,93	58,41	59,42	59,20

Tabela 4.2: Precisão média para o caso de 3 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	38,39	38,54	38,85	54,86	38,91
2	55,02	38,61	X	60,12	53,78
3	49,65	38,28	X	38,82	38,72
4	55,64	40,01	38,56	72,04	39,70
5	53,45	39,92	39,04	39,66	39,41

Tabela 4.3: Cobertura média para o caso de 3 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	39,04	38,69	38,81	40,04	38,72
2	39,66	38,73	X	40,86	39,25
3	40,27	38,26	X	39,08	38,78
4	41,15	39,80	38,66	40,62	39,78
5	41,29	39,82	39,11	39,70	39,60

Tabela 4.4: Medida F1 média para o caso de 3 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	38,55	38,59	38,63	41,07	38,43
2	40,28	38,64	X	42,73	40,62
3	41,39	38,13	X	38,94	38,64
4	42,58	39,67	38,61	42,18	39,74
5	42,71	39,53	38,97	39,63	39,50

Tabela 4.5: Acurácia para o caso de 2 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	58,74	56,71	57,72	56,12	57,16
2	56,16	57,61	57,83	56,71	57,54
3	59,46	59,25	58,25	57,48	57,07
4	57,71	58,32	60,08	58,26	58,17
5	58,97	60,52	59,61	59,33	59,37

Tabela 4.6: Precisão média para o caso de 2 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	58,97	56,93	57,73	56,10	57,04
2	56,16	57,62	57,91	56,79	57,58
3	59,38	59,24	58,30	57,49	57,08
4	57,73	58,04	60,35	58,28	58,16
5	58,43	60,62	59,61	59,25	59,31

Tabela 4.7: Cobertura média para o caso de 2 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	58,97	56,80	57,67	55,99	56,82
2	56,18	57,62	57,89	56,76	57,57
3	58,97	59,21	58,25	57,50	57,07
4	57,76	59,28	59,78	58,30	58,16
5	58,32	60,64	59,51	59,24	59,12

Tabela 4.8: Medida F1 média para o caso de 2 classes.

Dias \ Ativo	ABEV3	BBDC4	ITUB4	PETR4	VALE3
1	58,74	56,54	57,61	55,84	56,63
2	56,12	57,61	57,82	56,68	57,53
3	58,76	59,19	58,18	57,47	57,05
4	57,67	57,83	59,39	58,23	58,15
5	58,33	60,52	59,45	59,25	59,03

Capítulo 5

Considerações Finais

TODO

5.1 Conclusão

TODO

5.2 Trabalhos Futuros

TODO

Referências Bibliográficas

[1] LIU, Z., ZHU, H., CHONG, T. Y. "An NLP-PCA Based Trading Strategy On Chinese Stock Market", *Advances in Social Science and Education and Humanities Research*, v. 334, n. 2, pp. 80–89, jul. 2019.

[2] SEDLAK, M. "How Natural Language Processing is transforming the financial industry". https://www.ibm.com/blogs/watson/2016/06/natural-language-processing-transforming-financial-industry-2/, 2016. Acessado em Dezembro/2019.

[3] SMITH, D. J. "Efficient-Market Hypothesis (EMH) and Random-Walk Theory". https://stockmarketsupertrader.com/theory/efficient-market-hypothesis-emh-and-random-walk-theory/, 2018. Acessado em Dezembro/2019.

[4] VAZ, A. L. "Como lidar com dados desbalanceados em problemas de classificação". https://medium.com/data-hackers/como-lidar-com-dados-desbalanceados-em-problemas-de-classifica%C3%A7%C3%A3o-17c4d4357ef9, 2019. Acessado em Dezembro/2019.

[5] OGNJANOVSKI, G. "Everything you need to know about Neural Networks and Backpropagation — Machine Learning Easy and Fun". https://towardsdatascience.com/everything-you-need-to-know-about-neural-networks-and-backpropagation-mach 2019. Acessado em Dezembro/2019.

[6] PRATEEK, N. "Statistics is Freaking Hard: WTF is Activation function". https://towardsdatascience.com/statistics-is-freaking-hard-wtf-is-activation-function-df8342cdf292, 2017. Acessado em Dezembro/2019.

[7] DERTAT, A. "Applied Deep Learning - Part 4: Convolutional Neural Networks". https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2, 2017. Acessado em Dezembro/2019.

[8] AMEISEN, E. "How to solve 90 of NLP problems: a step-by-step guide". https://blog.insightdatascience.com/how-to-solve-90-of-nlp-problems-a-step-by-step-guide-fda605278e4e, 2018. Acessado em Dezembro/2019.

[9] "Índice Bovespa (Ibovespa)". http://www.bmfbovespa.com.br/pt_br/produtos/indices/indices-amplos/indice-ibovespa-ibovespa-composicao-da-carteira.htm. Acessado em Dezembro/2019.

[10] ALY, A., LAKHOTIA, K., ZHAO, S., et al. "PYTEXT: A SEAMLESS PATH FROM NLP RESEARCH TO PRODUCTION", dez. 2018.

[11] DO PAVINI, A. "Cresce número de pessoas físicas como profissionais na Bolsa". https://exame.abril.com.br/seu-dinheiro/cresce-numero-de-pessoas-fisicas-como-profissionais-na-bolsa/, 2019. Acessado em Dezembro/2019.

[12] BACHINSKIY, A. "The Growing Impact of AI in Financial Services: Six Examples". https://towardsdatascience.com/the-growing-impact-of-ai-in-financial-services-six-examples-da386c0301b2, 2019. Acessado em Dezembro/2019.

[13] CANTO, L. G. "Stock Market Predictor". https://github.com/lgcanto/stock-market-predictor/, 2019. Acessado em Dezembro/2019.

[14] MOREIRA, M. "Fusão entre BM&FBovespa e Cetip cria a B3, 5ª maior bolsa de valores do mundo". http://agenciabrasil.ebc.com.br/economia/noticia/2017-03/fusao-entre-bmfbovespa-e-cetip-cria-b3-5a-maior-bolsa-de-valores-do-mundo, 2017. Acessado em Dezembro/2019.

[15] WAWRZENIAK, D. "O Que É Análise Técnica?" https://www.bussoladoinvestidor.com.br/o-que-e-analise-tecnica/, 2018. Acessado em Dezembro/2019.

[16] WAWRZENIAK, D. "O Que É Análise Fundamentalista?" https://www.bussoladoinvestidor.com.br/o-que-e-analise-fundamentalista/, 2018. Acessado em Dezembro/2019.

[17] MALKIEL, B. G. "Efficient Market Hypothesis", *Finance*, pp. 127–134, 1989.

[18] "Análise Técnica x Análise Fundamentalista". https://www.tororadar.com.br/investimento/analise-tecnica/analise-tecnica-x-fundamentalista. Acessado em Dezembro/2019.

[19] JORDAN, M. I., MITCHELL, T. M. "Machine learning: Trends, perspectives, and prospects", *Science*, v. 349, pp. 80–89, jul. 2015.

[20] MITCHELL, T. M. *Machine Learning*. McGraw Hill, 1997.

[21] GULIPALLI, P. "The Pareto Principle for Data Scientists". https://www.kdnuggets.com/2019/03/pareto-principle-data-scientists.html, 2019. Acessado em Dezembro/2019.

[22] KOHAVI, R. "A Study of CrossValidation and Bootstrap for Accuracy Estimation and Model Selection", *AInternational joint Conference on artificial intelligence*, v. 14, pp. 1137––1145, 1995.

[23] GROVER, P. "5 Regression Loss Functions All Machine Learners Should Know". https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0, 2018. Acessado em Dezembro/2019.

[24] ROJAS, R. *Neural Networks - A Systematic Introduction*. Springer, 1996.

[25] GARBADE, M. J. "A Simple Introduction to Natural Language Processing". https://becominghuman.ai/a-simple-introduction-to-natural-language-processing-ea66a1747b32, 2018. Acessado em Dezembro/2019.

Apêndice A

Rotinas e Arquivos de Configuração

A.1 Rotina de Pré-processamento e Criação de Conjuntos de Dados

```
import re
import numpy as np
import pandas as pd
import csv
import os
from datetime import timedelta
from unicodedata import normalize
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.snowball import SnowballStemmer

OUT_DIR = './dataset_out'
MAX_DAYS = 5
TRAIN_PERCENTAGE_SIZE = 80/100
TEST_PERCENTAGE_SIZE = 20/100
REGEXP_REMOVE_SPECIAL = re.compile('[^a-zA-Z0-9 ]+')
ONLY_ONE_CODE = False
ONLY_ONE_CODE_NAME = 'VALE3'
STOPWORDS = stopwords.words('portuguese')
STEMMER = SnowballStemmer('portuguese')
ARTIGOS_TEXT_COLUMN = 'text'
ARTIGOS_UNUSED_COLUMNS = ['title', 'category', 'subcategory', 'link']
BOVESPA_UNUSED_COLUMNS = ['open', 'company', 'typereg', 'bdicode',
↪  'markettype', 'spec', 'prazot', 'currency', 'max', 'min', 'med',
↪  'preofc', 'preofv', 'totneg', 'quatot']
```

```
25
26  df_companies = pd.read_csv('../datasets/company-codes.csv')
27  df_bovespa = pd.read_csv('../datasets/kaggle/bovespa.csv')
28  df_articles = pd.read_csv('../datasets/kaggle/articles.csv')
29
30  def getAppreciation(before, after):
31    if after > before:
32      return 1
33    if before > after:
34      return -1
35    else:
36      return 0
37
38  def getEffectDate(date):
39    effectDate = date
40    while (df_bovespa[df_bovespa.date == effectDate].size < 1):
41      effectDate = effectDate + timedelta(days=1)
42    return effectDate
43
44  def exportToTSV(dataframe, filename):
45    if not os.path.exists(OUT_DIR):
46      os.mkdir(OUT_DIR)
47    fullpath = '%s/%s' % (OUT_DIR, filename)
48    dataframe.to_csv(fullpath, sep='\t', quoting=csv.QUOTE_NONE,
      ↪  index=False, header=False)
49
50  def getCleanText(text):
51    finalTextArray = []
52    lowerText = text.lower()
53    for word in lowerText.split():
54      if word not in STOPWORDS:
55        finalTextArray.append(STEMMER.stem(word))
56    finalText = ' '.join(finalTextArray)
57    finalText = normalize('NFKD', finalText).encode('ASCII',
      ↪  'ignore').decode('ASCII')
58    finalText = REGEXP_REMOVE_SPECIAL.sub('', finalText)
59    finalText = re.sub(' +', ' ', finalText)
60    return finalText
61
62  df_articles = df_articles[df_articles.category == 'mercado']
63  df_articles.drop(ARTIGOS_UNUSED_COLUMNS, inplace=True, axis=1)
```

```
df_articles['date'] = df_articles.apply(lambda row:
↪  np.int64(row['date'].replace('-', '')), axis=1)
df_articles['date'] = pd.to_datetime(df_articles['date'].astype(str),
↪  format='%Y%m%d')

df_bovespa.columns = map(str.lower, df_bovespa.columns)
df_bovespa =
↪  df_bovespa[df_bovespa.codneg.str.strip().isin(df_companies.code)]
df_bovespa['date'] = pd.to_datetime(df_bovespa['date'].astype(str),
↪  format='%Y%m%d')
df_bovespa = df_bovespa[df_bovespa.date >= df_articles.date.min()]
df_bovespa.drop(BOVESPA_UNUSED_COLUMNS, inplace=True, axis=1)
df_bovespa = df_bovespa.sort_values('date')

df_articles['date'] = df_articles.apply(lambda row:
↪  getEffectDate(row.date), axis=1)
df_articles[ARTIGOS_TEXT_COLUMN] = df_articles.apply(lambda row:
↪  getCleanText(row[ARTIGOS_TEXT_COLUMN]), axis=1)
df_articles = df_articles.sort_values('date')

df_analysis = pd.DataFrame(columns=['dataset','1s','0s', '-1s'])

for index, row in df_companies.iterrows():
  if not ONLY_ONE_CODE or (ONLY_ONE_CODE and row['code'] ==
  ↪  ONLY_ONE_CODE_NAME):
    print('Generating for ' + row['code'])
    df_full = df_bovespa[df_bovespa.codneg.str.strip() == row['code']]
    df_full =
    ↪  df_full.assign(close_before=df_full['close'].transform(lambda
    ↪  group: group.shift(1)))
    df_full = df_full[~np.isnan(df_full.close_before)]
    for d in range(MAX_DAYS):
      interval = d + 1
      df_interval =
      ↪  df_full.assign(close_after=df_full['close'].transform(lambda
      ↪  group: group.shift(-interval)))
      df_interval = df_interval[~np.isnan(df_interval.close_after)]
      df_interval.drop('close', inplace=True, axis=1)
      df_company = pd.merge(df_articles, df_interval, on='date',
      ↪  how='inner')
      if df_company.size > 0:
```

```
93          df_company.drop(['date', 'codneg'], inplace=True, axis=1)
94          df_company['label'] = df_company.apply(lambda row:
            ↪  getAppreciation(row['close_before'], row['close_after']),
            ↪  axis=1)
95          df_company.drop(['close_before', 'close_after'], inplace=True,
            ↪  axis=1)
96          df_company = df_company[['label', ARTIGOS_TEXT_COLUMN]]
97
98          df_company_positive = df_company[df_company.label ==
            ↪  1].sample(frac=1)
99          df_company_neutral = df_company[df_company.label ==
            ↪  0].sample(frac=1)
100         df_company_negative = df_company[df_company.label ==
            ↪  -1].sample(frac=1)
101
102         analysis = pd.Series({"dataset": row['code'] + '_' + str(interval)
            ↪  + 'd.tsv', "1s": len(df_company_positive), "0s":
            ↪  len(df_company_neutral), "-1s": len(df_company_negative)})
103         df_analysis = df_analysis.append(analysis, ignore_index=True)
104
105         trainPositiveSize =
            ↪  round(len(df_company_positive)*(TRAIN_PERCENTAGE_SIZE))
106         testPositiveSize =
            ↪  round(len(df_company_positive)*(TEST_PERCENTAGE_SIZE))
107
108         trainNeutralSize =
            ↪  round(len(df_company_neutral)*(TRAIN_PERCENTAGE_SIZE))
109         testNeutralSize =
            ↪  round(len(df_company_neutral)*(TEST_PERCENTAGE_SIZE))
110
111         trainNegativeSize =
            ↪  round(len(df_company_negative)*(TRAIN_PERCENTAGE_SIZE))
112         testNegativeSize =
            ↪  round(len(df_company_negative)*(TEST_PERCENTAGE_SIZE))
113
114         df_company_train = df_company_positive.head(trainPositiveSize)
115         df_company_positive = df_company_positive.iloc[trainPositiveSize:]
116         df_company_train =
            ↪  df_company_train.append(df_company_negative.head(trainNegativeSize))
117         df_company_negative = df_company_negative.iloc[trainNegativeSize:]
118
```

```
        df_company_test = df_company_positive.head(testPositiveSize)
        df_company_positive = df_company_positive.iloc[testPositiveSize:]
        df_company_test =
        ↪  df_company_test.append(df_company_negative.head(testNegativeSize))
        df_company_negative = df_company_negative.iloc[testNegativeSize:]

        exportToTSV(df_company_train, row['code'] + '_2c_' + str(interval)
        ↪  + 'd_train.tsv')
        exportToTSV(df_company_test, row['code'] + '_2c_' + str(interval)
        ↪  + 'd_test.tsv')

        df_company_train =
        ↪  df_company_train.append(df_company_neutral.head(trainNeutralSize))
        df_company_neutral = df_company_neutral.iloc[trainNeutralSize:]
        df_company_test =
        ↪  df_company_test.append(df_company_neutral.head(testNeutralSize))
        df_company_neutral = df_company_neutral.iloc[testNeutralSize:]

        exportToTSV(df_company_train, row['code'] + '_3c_' + str(interval)
        ↪  + 'd_train.tsv')
        exportToTSV(df_company_test, row['code'] + '_3c_' + str(interval)
        ↪  + 'd_test.tsv')
```

A.2 Rotina de Criação de Arquivos de Configuração e Execução de Treinamentos

```
import os
import pandas as pd
import subprocess
import glob

MAX_DAYS = 5
df_companies = pd.read_csv("../datasets/company-codes.csv")

for index, row in df_companies.iterrows():
  asset = row['code']
  for d in range(MAX_DAYS):
    interval = d + 1
    # configName = asset + "_2c_" + str(interval) + "d"
    configName = asset + "_3c_" + str(interval) + "d"
    fileName = "configs/" + configName + ".json"
```

```
    with open("templateconfig_XXXXX_Yc_Zd.json") as inputFile,
    ↪  open(fileName, "w") as outputFile:
      for line in inputFile:
        outputFile.write(line.replace("XXXXX_Yc_Zd", configName))

    print("Executing for " + configName)

    bashCommand = "pytext train < " + fileName
    result = subprocess.run(bashCommand, shell=True,
    ↪  stdout=subprocess.PIPE)
    result.stdout.decode('utf-8')

    print("Renaming run folder")

    runFoldersList = glob.glob("runs/*")
    latestRunFolder = max(runFoldersList, key=os.path.getctime)
    os.rename(latestRunFolder, "runs/" + configName)

    print("Execution done for " + configName)
```

A.3 Modelo de Arquivo de Configuração

```
{
  "version": 18,
  "task": {
    "DocumentClassificationTask": {
      "data": {
        "source": {
          "TSVDataSource": {
            "field_names": ["label", "text"],
            "train_filename":
            ↪  "../preprocessor/dataset_out/XXXXX_Yc_Zd_train.tsv",
            "test_filename":
            ↪  "../preprocessor/dataset_out/XXXXX_Yc_Zd_test.tsv",
            "eval_filename":
            ↪  "../preprocessor/dataset_out/XXXXX_Yc_Zd_test.tsv"
          }
        }
      },
      "model": {
        "DocModel": {
```

```
          "representation": {
            "DocNNRepresentation": {}
          }
        }
      },
      "trainer": {
        "epochs": 15
      },
      "metric_reporter": {
        "output_path": "metric_reports/XXXXX_Yc_Zd.txt",
        "model_select_metric": "accuracy",
        "target_label": null,
        "text_column_names": [
          "text"
        ]
      }
    }
  },
  "export_torchscript_path": "torchscripts/XXXXX_Yc_Zd.pt1",
  "export_caffe2_path": "caffe2_exports/XXXXX_Yc_Zd.caffe2.predictor"
}
```