FUNDAÇÃO GETULIO VARGAS
ESCOLA DE ADMINISTRAÇÃO DE EMPRESAS DE SÃO PAULO

Programa Institucional de Bolsas de Iniciação Científica (PIBIC)

FAKE NEWS NO BRASIL
INSIGHTS EMPÍRICOS SOBRE O FENÔMENO E AVALIAÇÃO DAS MÍDIAS DIVULGADORAS DE NOTÍCIAS

ALEX AKIRA OKUNO
ANDRÉ SAMARTINI

São Paulo – SP
2019

Resumo

Este artigo teve como objetivo encontrar insights sobre notícias divulgadas no meio online pelos portais de divulgação brasileiros, com principal foco nas *fake news*, dada a grande relevância do tema no ano de eleição de 2018. Foi feita uma coleta automatizada de títulos de notícias em diversos websites (web scraping) para a confecção de uma base de dados de notícias falsas e verdadeiras. Com esses dados, foram treinados modelos de classificação para distinguir entre as notícias *fake* e verdadeiras e os modelos treinados permitiram classificar notícias em fakes ou não com acurácia acima de 92%. Também utilizou-se um modelo para interpretabilidade chamado LIME (*Locally Interpretable Model-Agnostic Explanations*), de modo que foi possível identificar as palavras mais impactantes e seus respectivos efeitos para a classificação das notícias. Por fim, utilizou-se os resultados do modelo de regressão logística para propor uma métrica inspirada no índice de Sharpe para avaliação de mídias divulgadoras de notícias inspirada no índice de Sharpe.

Palavras chave: fake news, notícias, web scraping, machine learning, processamento de linguagem natural, LIME, interpretabilidade, classificação, índice de sharpe.

1. Introdução

A utilização de notícias de conteúdo falso não é uma invenção recente, e há evidências da ocorrência deste fenômeno há mais de 2 mil anos, mas com a utilização em escala da tecnologia, esse conteúdo tem um potencial de disseminação nunca antes visto [Posetti, Matthews, 2018].[1] O alcance desse fenômeno pode tomar proporções consideráveis como no caso da eleição nos EUA em 2016, que pode ter tido seus resultados afetados por conta das *fake news* [Pogue, 2017]. No Brasil, o tema se tornou particularmente relevante no ano de 2018, dado o contexto da eleição presidencial bastante polarizada, que é um cenário que pode favorecer difusão de informação de cunho duvidoso.

Utilizando um *framework* teórico de Gentzkow, Shapiro e Stone (2016), pode-se entender os incentivos para o surgimento das *fake news* e potenciais danos à sociedade. Esse tipo de conteúdo surge pois existem incentivos para a formação de mercado de viés, cuja (i) oferta vem do fato de que a mídia pode ter interesse em ganhos políticos, ao custo de diminuição de seus lucros; e (ii) demanda vem de consumidores que apreciam conteúdo viesado. O equilíbrio desse mercado não é necessariamente ruim: do lado da oferta, competição tende a diminuir o viés e aumentar o bem-estar; do lado da demanda, a competição tende a acentuar ainda mais os vieses. Mesmo o segundo caso é benéfico partindo da hipótese de que os agentes são racionais e, principalmente, pesquisam sobre a informação que consomem. Nesse *framework*, o fato de que os agentes não necessariamente pesquisam sobre a informação é o que leva às distorções no mercado à diminuição do bem-estar.

A literatura acerca desse tema vem se expandindo, e diversos trabalhos têm surgido à medida que o tema toma relevância global. Por exemplo, Gentzkow e Allcott (2016) oferecem aspectos teóricos sobre o fenômeno, bem como base empírica para sustentação do debate no que tange à eleição americana em 2016. Na mesma direção, Zhou e Zafarani (2018) mostram um panorama geral do fenômeno, abordando desde métodos de fact-checking (manuais e automáticos) e detecção baseada em estilo de escrita à modelagem da propagação das *fake news* e credibilidade das fontes de notícias.

Embora o tema tenha relevância global e a literatura seja crescente, no que tange especificamente ao Brasil, pouco se trabalha no tema com abordagens empíricas e computacionais como a de Gentzkow e Allcott (2016) e Zhou e Zafarani (2018). Nesse sentido, um primeiro objetivo deste

[1] A short guide to the history of Fake News and Disinformation, ICFJ (International Center for Journalists). Julie Posetti and Alice Matthews.

trabalho é de contribuir com um melhor entendimento do fenômenos das *fake news* no Brasil. Para isso, crio uma base de dados de notícias do Brasil e utilizo (i) diversos modelos de classificação e (ii) um método chamado *LIME* para extrair *insights* provenientes da estrutura textual do corpo das notícias. Chamarei esses *insights* de *conclusões* durante o desenvolvimento do texto, e defino *conclusão* como um resultado decorrente de evidências nos dados utilizados.

Após essa primeira etapa, tenho o objetivo de contribuir com o problema do *framework* proposto em Gentzkow, Shapiro e Stone (2016), no sentido de mitigar a assimetria de informação que o leitor de notícias tem para com o conteúdo consumido. Para isso, proponho uma métrica para avaliação das fontes divulgadoras de notícias inspirada no índice de Sharpe para avaliação do retorno de ativos financeiros, tomando como hipótese que exista variabilidade no nível de confiabilidade de diferentes fontes de notícias.

2. O conceito de *fake news*

Não existe um conceito universal ou definição formal de *fake news*, dada a variabilidade de formas em que esse fenômeno pode aparecer na prática. Entretanto, existem diversos conceitos que rondam o espectro de *fake news*, e irei definir esses diversos conceitos que aparecem frequentemente quando se fala de *fake news.*

Existem três principais características que descrevem diferentes conceitos relacionados a *fake news:* (i) autenticidade (falsa ou não), (ii) intenção (má ou não), e (iii) se a informação é, de fato, uma notícia ou não [Zhou e Zafarani, 2018]. Da interação dessas 3 características, temos vários conceitos: *notícias maliciosamente falsas* [Allcott e Gentzkow 2017; Shu et al. 2017; Waldrop 2017], *notícias falsas* [Vossoughi et al. 2018], notícias sátira [Berkowitz e Schwartz, 2016], *desinformação* [Kshetri e Voas 2017], *misinformação* [Kucharski, 2016] e *rumor* [Buntain e Golbeck 2017]. A seguinte tabela sumariza esses seis conceitos:

	Autenticidade	**Intenção**	**Notícia**
Notícia maliciosamente falsa	Falsa	Má	Sim
Notícia Falsa	Falsa	Desconhecida	Sim
Notícia sátira	Desconhecida	Não má	Sim

Desinformação	Falsa	Má	Desconhecida
Misinformação	Falsa	Desconhecida	Desconhecida
Rumor	Desconhecida	Desconhecida	Desconhecida

Tabela 1: comparação de conceitos relacionados a *fake news.* [Adaptado de Zhou e Zafarani, 2018].

3. Dados

Foram coletados, de maneira automatizada, dados de três websites, de maneira a consolidar uma base de dados com notícias falsas e verdadeiras.

Para os websites que representam as *fake news*, foram os utilizados o *newsatual* e *boatos.org*. O primeiro é um site predominantemente político e que foi dito, em matéria de 2018 da Folha de SP, como um dos maiores divulgadores de fake news no Facebook. O segundo, é um projeto de *fact-checking* feito por uma equipe de jornalistas que divulgam notícias comprovadas como *fake news*. Para o website confiável, foi utilizado o *El País*, que tem uma reputação de profissionalidade confiável, segundo a citada matéria da Folha de São Paulo.

O conteúdo do *newsatual* se adequa mais aos conceitos que abrangem ou más intenções ou autenticidade desconhecida. Já o conteudo do *boatos.org* pode se encaixar em qualquer um dos seis conceitos, pela própria natureza do projeto que é de checar informações de qualquer natureza duvidosa. Desta forma, a definição de *fake news* utilizada no presente trabalho é qualquer conteúdo de cunho informativo na *web* e que se encaixa em algum dos seis conceitos explicitados na Tabela 1.

Foram coletados, através de *web scraping*, 20.827 títulos de notícias do *website El país*, 2.817 títulos do *boatos.org* e 2.452 títulos do *newsatual*, totalizando 26.096 títulos, dos quais 79,8% são considerados confiáveis e 21,2% são considerados *fake*.

Para que estas notícias estivessem prontas para as etapas posteriores do trabalho, foi necessário fazer alguns ajustes a fim de não obter resultados espúrios. Isto acontece porque diversas notícias

continham palavras indicativas e altamente sugestivas de que se seguia uma *fake news*, por exemplo:

- ❑ **Denúncia grave:** delegado revela que esfaqueador do Bolsonaro se reuniu com Deputado Federal antes do crime.
- ❑ **URGENTE:** Juíza quebra sigilo de dados de celulares do agressor de Bolsonaro
- ❑ E a boba da corte foi Mirian Leitão.. **Que fiasco! Que tragédia!**
- ❑ **Cuidado –** Link "Brincadeira acaba em morte em prédio de Recife" esconde vírus

Como pode-se observar, existem alguns padrões de escrita nessas palavras sugestivas, e estes foram retirados utilizando-se de *Regular Expressions*. Assim, apenas palavras pertencentes ao corpo do título da notícia podem denunciá-la como fake. Observou-se, portanto, que é comum encontrar no título de fake news na web palavras chamativas que não fazem parte do conteúdo da notícia.

Assim, é razoável assumir que existe um público consumidor de informação que aprecia consumir viés, como proposto no mercado de viés de Gentzkow, Shapiro e Stone (2016). E estes próprios autores definem que *fake news* contém essencialmente sinais distorcidos não correlacionados com os sinais que descrevem a *verdade*. Assim, temos evidência de que essas palavras chamativas fazem sentido com o *framework* teórico e são indicativos significativos de *fake news*. Essa discussão tem o objetivo de validar o uso das notícias coletadas, mas como dito anteriormente, essas palavras indicativas foram removidas a fim de evitar resultados espúrios na etapa dos modelos de classificação.

4. Definição e formalização matemática da modelagem do problema

4.1. Modelos de Classificação

Tomemos a seguinte notação para generalizar o conceito de classificação:

- ❑ □representa nossa variável dependente, no caso, um vetor de □ entradas, cuja entrada □□representa uma classe à qual o título de notícia □pertence, sendo assim, $y_j \in S$, sendo □ o conjunto das possíveis classes que uma observação pode assumir;

- □ representa as variáveis independentes, e é uma matriz de dimensão □ × □, sendo □ o número de observações na amostra e □ a quantidade de atributos (*features*) que descrevem cada observação na amostra;

O objetivo de um modelo de classificação é encontrar uma função $f : \mathbb{R}^k \rightarrow [0, 1]^s$, que mapeia o conjunto de □ atributos de uma determinada observação em um vetor de probabilidade □ de □entradas, cujas entradas representam a probabilidade de □□pertencer a cada uma das □classes. Formalmente, temos $p_j = P(y_j \in i) \forall i \in S$, sendo $\sum_{s \in S} p_s = 1$. A classificação predita para cada observação □□que tem um vetor de probabilidade predita □será dada por $\hat{y_j} = argmax_s p_s$ para $s \in S$.

Para encontrar a função f mais adequada, temos que escolher os parâmetros $\theta \in \Theta$, sendo Θ o espaço de parâmetros. Essa escolha é dada pela minimização de uma função de custo $J(y, X; \theta)$, que por sua vez, é uma função dos erros $\epsilon = y - f(y, X; \theta)$. Um exemplo de função de custo é o erro médio quadrático (MSE), que é simplesmente a soma dos erros de cada observação ao quadrado, ou seja: $MSE(y, x; \theta) = \sum_{i=1}^{n} \epsilon_i^2$, ou em notação matricial, $\epsilon' \epsilon$.

No caso de uma classificação binária, que é o caso deste trabalho, uma função de custo muito utilizada é a *Log Loss* ou Entropia Cruzada (*Binary Cross Entropy Loss,* ou BCE). Essa função é definida por: $BCE(y, X; \theta) = -\frac{1}{n} \sum_{j=1}^{n} y_j log(\hat{p_j}) + (1 - y_j) log(1 - \hat{p_j})$, sendo aqui $\hat{p_j}$ a probabilidade associada à classe correta de y_j. Essa função é obtida pela minimização da divergência de Kullback-Leibler entre as distribuições de y e $\hat{y}$.

Uma característica muito importante dessa função de custo para o problema de classificação é que ela não apenas penaliza predições de classe incorretas, mas como também penaliza, para observações corretamente classificadas, a incerteza associada a essa determinada classificação. Exemplificando, uma predição que acerta a classe certa ($y_j = \hat{y_j}$), mas com probabilidade de 60% será mais penalizada que aquela que acerta com probabilidade de 90% (ou seja, quanto maior a distância entre $max_j p_j$ e y_j).

Em conclusão, os parâmetros escolhidos, $\hat{\theta}$, serão aqueles que minimizam essa função de custo, ou seja, $\hat{\theta} = argmin_{\theta} J(y, X; \theta)$. Para determinadas classes de funções, existem soluções analíticas para $\hat{\theta}$, e caso não exista solução analítica, o processo para encontrar $\hat{\theta}$ é utilizar um algoritmo de otimização. A maioria dos otimizadores utilizados são métodos de primeira ordem, como SGD, Adam e RMSProp [Nocedal e Wright, 1999].

Neste trabalho, utilizei diversos classificadores, com diferentes características de complexidade dos modelos e de interpretabilidade. Discutirei posteriormente quais serão os critérios para comparar os resultados desses classificadores.

Foram implementados os seguintes classificadores :

- ❏ **Regressão Logística**
- ❏ **Naïve Bayes**
- ❏ **K Nearest Neighbors (KNN)**
- ❏ **Support Vector Classifier (SVC)**
- ❏ **Decision Tree**
- ❏ **Random Forest**
- ❏ **Adaptative Boost (AdaBoost)**

4.2. Métricas de Avaliação e Validação

Para poder comparar objetivamente diferentes classificadores, existem algumas métricas de performance, que têm diferentes interpretações e serão peça chave para o trabalho posteriormente. Para questões de simplificação, assumamos que amostra utilizada tenha N observações, das quais a foram corretamente classificadas e $e \equiv 1 - a$ foram erroneamente classificadas. Também vamos condicionar o problema a uma classificação de binárias, cujas classes serão verdadeiro (1) e falso (0).

- ❏ **Verdadeiros Positivos (TP):** quantidade de observações de classe 1 corretamente classificadas.

- **Verdadeiros Negativos (TN):** quantidade de observações de classe 0 corretamente classificadas.
- **Falsos Positivos (FP):** quantidade de observações de classe 0 classificadas como 1.
- **Falsos Negativos(FN):** quantidade de observações de classe 1 classificadas como 0.
- **Acurácia:** representa a porcentagem de acerto na amostra, ou seja, $\frac{a}{N} = \frac{TP+FP}{TP+FP+TN+FN}$.
- ***Precision*:** Quantidade de acertos dentre as observações classificadas como verdadeiras $\frac{TP}{TP+FP}$
- ***Recall*:** Quantidade de acertos dentre as observações que cuja classe real é verdadeiro $\frac{TP}{TP+FN}$
- ***F1 Score*:** Média harmônica entre *precision* e *recall* : $2 \times \frac{precision \times recall}{precision + recall}$
- **Matriz de confusão:** Matriz que mostra os acertos e erros acima descritos classe a classe. Compara-se as classes verdadeiras e preditas nessa matriz, como visto a seguir:

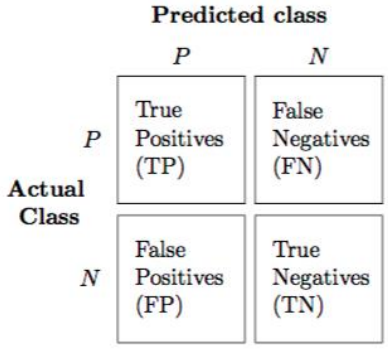

Figura 1: Matriz de Confusão.

Podemos perceber, desde já, que são necessárias outras métricas além da taxa de acerto de um classificador para poder avaliar sua performance. Da argumentação seguinte, seguem exemplos da necessidade de métricas como *precision*, *recall* e *F1 Score*.

I. **Desbalanceamento de amostra:** é bastante comum encontrar amostras classificadas em que o número de observações de uma classe é muito maior que da outra, por exemplo, bases de dados de transações fraudulentas. Dessa forma, um classificador que apenas retorna a classe majoritária tem naturalmente alta performance em questão de acurácia.

II. **Diferenciação de classes:** com apenas a métrica da acurácia, não podemos saber se o classificador está errando mais em determinadas classes, o que pode ser crucial para a

análise pois acertar determinada classe pode ter uma consequência prática mais importante comparativamente. Por exemplo, num caso de detecção de câncer, existe maior preocupação em corretamente diagnosticar um paciente que tem a doença do que aquele que não possui a doença dado que no primeiro caso existe risco de vida.

Além das métricas acima citadas, outras 2 métricas se mostram bastante relevantes:

- **Area under the curve (AUC):** essa métrica nasce do fato de que geralmente existe um *tradeoff* entre errar em falsos positivos e verdadeiros positivos em problemas de classificação, ou seja, a partir de certo ponto para aumentar a taxa de verdadeiros positivos (*tpr*) é necessário um incremento na taxa de falsos positivos (*fpr*). Esse *tradeoff* é retratado na chamada curva ROC (*Receiver Operating Characteristic*), que mostra, para um determinado conjunto de dados num problema de classificação como varia a tupla (*fpr, tpr*) à medida que variamos o limite a probabilidade estimada a partir do qual consideramos que a classe será verdadeira. E a métrica AUC nada mais é que a área debaixo da curva ROC. Também é interessante pontuar que essa métrica varia de 0 (pior caso) a 1 (melhor caso). No melhor caso, a AUC será 1 pois para uma taxa de falsos positivos de 0, conseguimos ter uma taxa de verdadeiros positivos de 1. Segue uma imagem com exemplos da curva ROC:

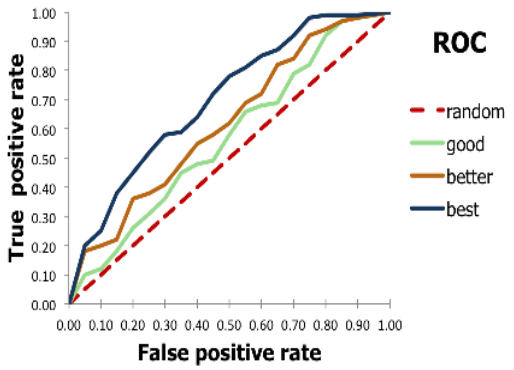

Figura 2: Exemplos de curva ROC

- **Log Loss:** Essa métrica foi mencionada quando falou-se sobre funções de perda, mas ela também é muito importante em questão de avaliação de classificadores. Como já foi dito, essa métrica irá bonificar o classificador conforme acerta as classes com maior grau de certeza.

Para que esse conjunto de métricas seja confiável, precisamos, ainda tratar de outro potencial problema chamado ***overfitting*** (ou sobreajuste), ou seja, quando o critério de classificação dos modelos se ajusta perfeitamente aos dados, mas tem desempenho bastante inferior em novos conjuntos de dados. Em resumo, o modelo é incapaz de generalizar características dos dados, insumo essencial para boa performance em dados ainda não vistos durante o treinamento.

Utilizaremos um processo de validação cruzada (*k-fold cross validation*). Neste processo, o conjunto de dados é dividido em k grupos e ocorrerão k treinamentos. Por exemplo para $k = 5$, o modelo será treinado primeiramente nos grupos $\{2, 3, 4, 5\}$ (amostra de teste) e as métricas serão calculadas na amostra de treino, o grupo $\{1\}$, aquele que não foi visto durante o treinamento. Posteriormente, a amostra de treino serão os grupos $\{1, 3, 4, 5\}$ e o teste será $\{2\}$, e assim por diante. A métrica final será a média entre as métricas dos 5 conjuntos de teste. Deste modo, mitigamos o *overfitting*, pois todos as observações do conjunto de dados foram utilizadas em algum momento tanto no treinamento quanto no teste, além de que as métricas se tornam muito mais confiáveis pois foram calculadas em amostras fora do conjunto de treinamento nas etapas da validação cruzada.

Fig. 3: Esquematização da validação cruzada

3.3. Features Textuais

Para que possamos efetivamente treinar modelos de classificação, precisamos extrair de cada observação (notícia) variáveis numéricas. De maneira geral, essa etapa pode ser sumarizada em 3 passos [Gentzkow, 2017]:

I. Representar o texto "cru" $\mathcal{D}$ como um vetor numérico C.
II. Mapear C em valores preditos $\hat{y}$ de uma variável de resposta y.
III. Usar $\hat{y}$ na análise descritiva/causal subsequente.

No problema deste trabalho, $\mathcal{D}$ representa o conjunto de títulos de notícias coletado e tratado, y representa a variável de resposta que diz se as notícias são ou não *fake*, $\hat{y}$ são as estimativas de y utilizando métodos de classificação tendo C como variáveis explicativas. Aqui, cada variável de resposta y_i é descrita apenas pela respectiva notícia $\mathcal{D}_i$, de forma que, então, $\mathcal{D}$ é dividido num conjunto de documentos $\{\mathcal{D}_i\}_{i=1}^{n}$. Ademais, métodos de obtenção de C, ou seja, representações matemáticas de $\mathcal{D}$, serão discutidos nesta seção.

É comum realizar, também, uma etapa de preprocessamento opcional em $\mathcal{D}$, que consiste em retirar as chamadas *stopwords*, que são palavras extremamente comuns (ex: artigos, conjunções, formas dos verbos ser e estar, etc.). No geral, essas palavras são importantes para a estrutura gramatical, mas não tem um significado intrínseco. Por esse motivo, é comum retirar esse conjunto de *stopwords* [Gentzkow, 2017]. Neste trabalho, as *stopwords* utilizadas são de uma lista pré-definida do *package* de processamento de texto NLTK (*Natural Language Toolkit*).

Existem duas principais abordagens para a tarefa de construir C, que serão discutidas em seguida.

3.3.1. TF-IDF

Para o primeiro caso, construiremos a chamada matriz TF-IDF (*Term Frequency - Inverse Document Frequency*). Supondo que temos um vocabulário V na amostra de r palavras distintas, e que cada palavra $v \in V$ é mapeada em um natural do conjunto $R = \{1, \ldots, r\}$. Dessa forma, podemos denotar tf_{ij} (*term frequency*) como sendo a quantidade de vezes que a palavra $v_j, j \in R$ aparece no documento $\mathcal{D}_i$. Para uma amostra de n documentos, podemos definir idf_j (*inverse document frequency*) como $\log\left(\frac{n}{d_j}\right)$, o logaritmo de n sobre a quantidade de documentos em que a palavra v_j aparece ($d_j = \sum_i 1_{[tf_{ij}>0]}$).

Em resumo, para um determinado documento (no nosso caso, cada notícia), o vetor TF-IDF será um vetor de tamanho r em que cada entrada do vetor é não nula nas posições que representam palavras que aparecem naquele documento específico. Para posições não nulas no vetor, o valor é composto por 2 partes: uma delas é a frequência com aquela palavra específica apareceu no documento. Porém, apenas com isso, estaríamos inflando o efeito de algumas palavras que naturalmente podem aparecer mais naquela língua ou contexto.

Desta forma, multiplicamos esse valor da frequência da palavra por uma espécie de deflator, que vai ponderar o peso dessa palavra de acordo com a quantidade de vezes que ela aparece também nos outros documentos, de forma que se ela aparece muito em diversos documentos, o valor TD-IDF dessa palavra será penalizado, e vice-versa. Ao final desse processo, teremos a primeira representação de C, C_{tfidf}, que será uma matriz de dimensão $n \times r$. Também poderíamos permitir que n-gramas, combinações de n palavras ($n > 1$), sejam incluídas, o que pode ser bastante relevante, mas ao custo de aumentar ainda mais a dimensão de C_{tfidf}

É importante observar que C_{tfidf} é uma matriz de dimensão muito elevada (muitas colunas) e também muito esparsa, pois o conjunto de palavras numa notícia é muito menor que o conjunto de palavras do vocabulário, e isso pode acabar sendo um empecilho para modelos que não lidam bem com essas características. Outra característica importante é que esse método não captura aspectos da mesma palavra em diferentes contextos, justamente por representar cada palavra, para cada

documento, como um valor único, e não um vetor, que tem diferentes dimensões que podem explicar essas variações de sentido, o que pode ser benéfico ou não dependendo do problema.

3.3.2. Word Embedding

Simplificadamente, *word embeddings* são representações vetoriais de palavras e documentos que permitem capturar características como contexto, interações entre palavras, similaridade sintática, etc. Com a representação descrita em 3.3.1., cada dimensão do vetor do documento representa uma palavra, e essas dimensões são totalmente independentes, no sentido em que não há diferença relevante para a representação vetorial entre trocar uma palavra qualquer por um sinônimo ou uma palavra totalmente diferente em significado. Essa propriedade advém da própria construção da representação, pois cada dimensão representa uma palavra, e não uma espécie de *feature* abstrata de significado ou contexto.

Dessa forma, o que procuramos aqui é que os vetores que representam documentos similares sejam também similares. Em termos práticos, para dois vetores u e v que representam sentenças muito similares, queremos a propriedade que o ângulo θ entre u e v seja próximo de 0. Definiremos, portanto, uma métrica de similaridade entre dois documentos representados por u e v pelo cosseno do ângulo entre eles, ou seja, $sim(u, v) = \cos(\theta) = \frac{u \cdot v}{||u|| \cdot ||v||}$. Quanto mais similares forem esses documentos, $\theta \to 0$ e, portanto, $\cos(\theta) \to 1$, e quanto menos similares forem os documentos, $\theta \to \pi$ e $\cos(\theta) \to -1$. É trivial ver que $sim(u, v) \in [-1, 1] \forall u, v$.

O método de geração de *word embeddings* a ser utilizado neste trabalho é o Word2Vec CBOW (*Word to Vector Continuous Bag of Words*). A ideia deste método é tomar uma palavra como *input* numa arquitetura de rede neural e tentar prever seu contexto, ou seja, a palavra ou as palavras que estão na vizinhança da palavra *input*.

Dado um corpus de texto $\mathcal{D}$, temos uma sequência de palavras $\{w_1, w_2, \ldots, w_t, \ldots\}$, em que o subíndice refletirá apenas a posição da palavra no texto. Suponhamos que a palavra vizinha/contexto de uma palavra w_{t+1} qualquer seja w_t, ou seja, a palavra anterior. Suponhamos

também um vocabulário de V palavras. Para a estimação dos *word embeddings*, teríamos a seguinte arquitetura de rede neural:

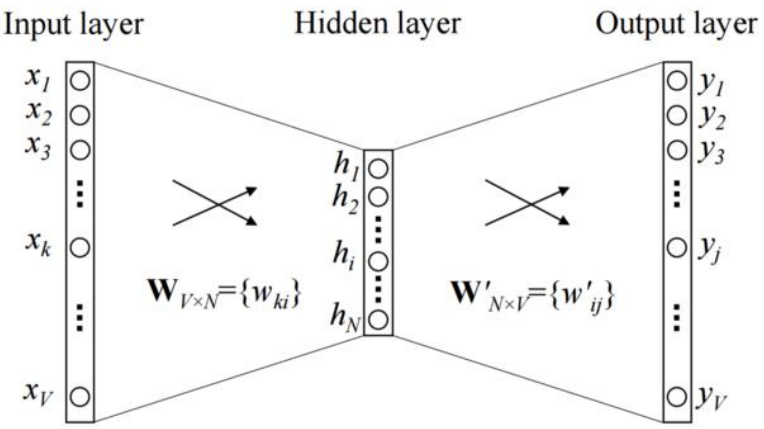

Fig. 4: Esquematização da rede neural para word2vec

Como *input*, temos cada palavra contexto em representação *one-hot*, ou seja, num vetor de tamanho V em que cada dimensão representa uma palavra, e todas as entradas são 0 exceto a que representa a palavra w_t. Também há um hidden layer sem função de ativação e de dimensão N, o que necessita de uma matriz de pesos de dimensão $V \times N$, denominada W na figura. Como variável de resposta, temos a representação *one-hot* da palavra *target*. Como esse *output* tem dimensão V, precisamos de outra matriz de pesos de dimensão $N \times V$, definida como W' na figura. A camada de *output* terá uma função de ativação softmax.

Os *embeddings* são os pesos aprendidos contidos na matriz W, de forma que o *embedding* da palavra indexada por k na representação *one hot* é a linha k da matriz W. É fácil visualizar isso tendo em mente que o vetor do *hidden layer* $h = Wx$. Como todas as entradas em x são nulas exceto para a palavra indexada por k (w_k), a representação h para w_k será a linha k da matriz W. Também é importante pontuar que, portanto, a quantidade de neurônios N no *hidden layer* é a dimensão da representação das palavras nesse modelo. Essa representação, ao contrário da definida em 3.3.1., é densa e de dimensão bem reduzida comparativamente, dado que a escolha de N é um número bem menor que R. Os pesos da segunda matriz, W' nos dará informação sobre como a

palavra representada por h se relaciona ao seu contexto, dado que o *output* da rede nos dá a distribuição de probabilidades das palavras vizinhas de h.

Neste modelo, apenas utilizamos uma palavra de contexto como *input*, mas poderíamos escolher C palavras de contexto, tal que $C > 1$. Assim, apenas teríamos uma estrutura de rede neural levemente modificada, com C inputs de dimensão V que compartilham os mesmos pesos W, como pode ser visto na figura abaixo. Detalhes e propriedades mais específicas sobre o Word2Vec podem ser vistas em [Mikolov et al., 2013] e [Rong, 2016].

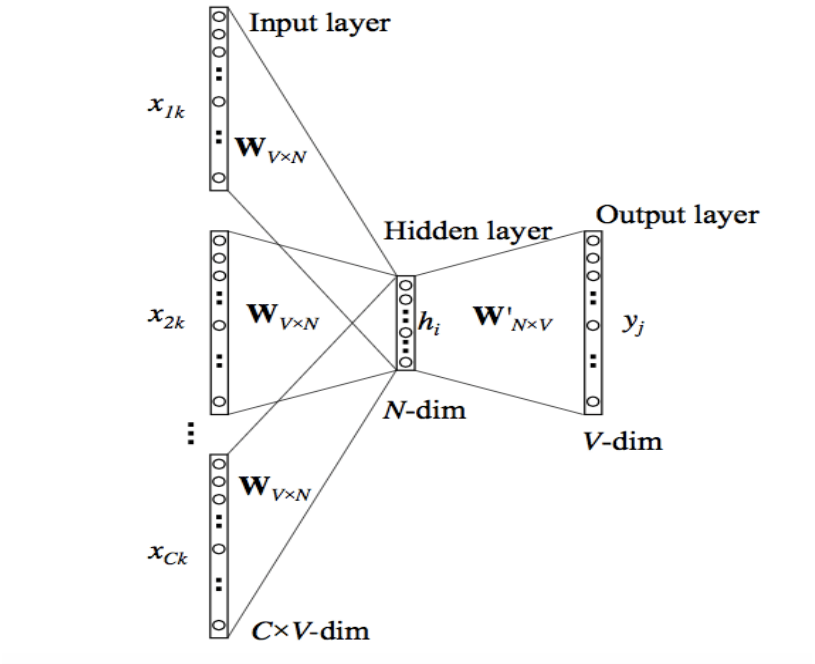

Fig. 5: Esquematização da rede neural para word2vec

O processo de treinamento dos pesos para embeddings pode ser muito custoso em termos de tempo, principalmente ao utilizar conjuntos de textos muito extensos (que são desejáveis por uma questão de poderem generalizar melhor na representação das palavras, pelo fato de muitos contextos terem sido parte do treinamento). Uma possibilidade para contornar esse problema é utilizar embeddings pré-treinados e disponibilizados para download. Utilizarei os embeddings pré-treinados disponibilizados pelo NILC (Interinstitutional Center for Computational Linguistics - USP), com as especificações Word2Vec CBOW dimensão 300. Os embeddings do NILC foram treinados em

17 conjuntos de textos, incluindo Wikipedia, Google News, G1, obras literárias de domínio público, Revista Mundo Estranho, livros texto, entre outros.

4. *Leakage*

Como foram utilizadas 2 fontes de notícias *fake*, é possível que hajam notícias repetidas, o que causaria o problema de leakage, que consiste em vazar informação do conjunto de treino para o conjunto de teste. Isso pode ocorrer pois há uma probabilidade de, na etapa de cross validation, 2 notícias iguais caírem no conjunto de teste e no conjunto de treino, o que traria resultados espúrios. Iremos investigar a existência desse problema mais a fundo, ou descartar essa possibilidade.

Para tanto, precisamos de uma métrica de semelhança entre 2 textos baseado nas letras contidas nestes. O conceito de distância de Levenshtein será utilizado nesta investigação. Essa distância nada mais é do que o número mínimo de caracteres que temos que mudar em um texto para que este fique idêntico ao outro texto que está sendo comparado. Por exemplo, a distância entre $a\,cara$ e $caro$ é 3, pois temos que fazer a retirada de a, do espaço, e a substituição do último a por o.

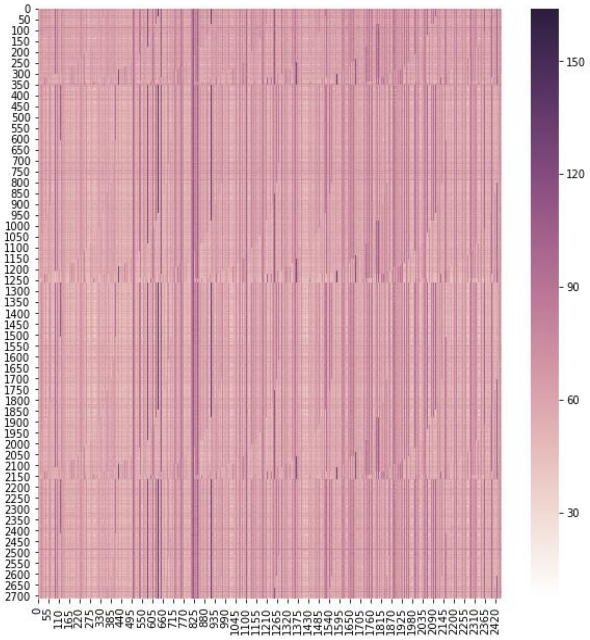

Fig 6: Heatmap das distâncias de Levenshtein

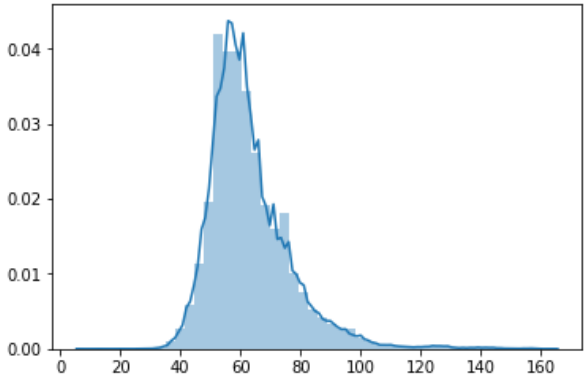

Fig 7: Distribuição das distâncias de Levenshtein

HIPÓTESE: *O problema de leakage é inexistente ou irrelevante neste caso, sendo que, uma metodologia que arrisque excluir notícias que potencialmente possam não ser repetidas em prol de retirar notícias possivelmente repetidas traria mais custos do que benefícios para o modelo.*

A primeira figura (esquerda) é um mapa de calor das distâncias de *Levenshtein* entre as 2 fontes *fake*. Em termos simples é uma matriz $n \times k$, onde n é a quantidade de notícias na fonte do eixo y e k a quantidade de notícias da fonte no eixo x. Cada entrada i, j da matriz é a distância entre a notícia i da fonte do eixo y e a notícia j da fonte do eixo x. Após montar a matriz, apenas colorimos de acordo com a escala de cor explicitada ao lado do mapa de calor. Visualmente, é possível ter indícios de que o problema de *leakage* não seja relevante, pois áreas muito claras parecem inexistir.

Para uma noção mais precisa da hipótese acima, vamos ver a distribuição das distâncias de Levenshtein. A distância média é de 62,96 caracteres. Ao mesmo tempo, o tamanho médio de uma notícia é de 69,97 caracteres. Assim, em média, teríamos que mudar 90% de uma notícia para que esta fique idêntica a outra, e é razoável assumir, portanto, que o problema de *Leakage* é inexiste ou irrelevante neste caso.

5. Resultados dos modelos de classificação

Dadas as considerações dos capítulos anteriores, podemos ter 2 variações de *features* a serem utilizadas:

I. Incluindo ou não *stopwords*
II. Utilizar TF-IDF ou *Word Embeddings*

Primeiramente, exploraremos o uso de *stopwords* utilizando *features* TF-IDF, para então decidir quanto ao uso de *embeddings*. Assim, seguem os resultados dessa primeira etapa:

Model	Accuracy	TN	TP	FN	FP	Precision	Recall	F1-score	AUC	Logloss
LogisticRegression	0.925919	677	4135	354	31	0.921141	0.992559	0.955517	0.960937	0.208940
BernoulliNB	0.926304	806	4008	225	158	0.946846	0.962074	0.954399	0.940003	0.458769
KNeighborsClassifier	0.825861	137	4155	894	11	0.822935	0.997360	0.901791	0.740783	1.963788
SVC	0.801616	0	4166	1031	0	0.801616	1.000000	0.889886	0.944863	0.249439
DecisionTreeClassifier	0.938041	794	4081	237	85	0.945113	0.979597	0.962046	0.884940	1.888025
RandomForestClassifier	0.833750	168	4165	863	1	0.828361	0.999760	0.906026	0.916872	0.385947
AdaBoostClassifier	0.896864	591	4070	440	96	0.902439	0.976956	0.938220	0.831551	0.653651

Tabela 2: TF-IDF sem *stopwords*

Model	Accuracy	TN	TP	FN	FP	Precision	Recall	F1-score	AUC	Logloss
LogisticRegression	0.928613	721	4105	310	61	0.929785	0.985358	0.956765	0.963639	0.197565
BernoulliNB	0.926688	845	3971	186	195	0.955256	0.953193	0.954223	0.945476	0.420767
KNeighborsClassifier	0.825284	134	4155	897	11	0.822447	0.997360	0.901497	0.763432	1.467211
SVC	0.801616	0	4166	1031	0	0.801616	1.000000	0.889886	0.955914	0.218955
DecisionTreeClassifier	0.922263	726	4067	305	99	0.930238	0.976236	0.952682	0.842503	2.623544
RandomForestClassifier	0.841062	206	4165	825	1	0.834669	0.999760	0.909786	0.923882	0.375069
AdaBoostClassifier	0.896286	571	4087	460	79	0.898834	0.981037	0.938138	0.890352	0.656690

Tabela 3: TF-IDF com *stopwords*

Primeiramente, para realizar as análises, iremos escolher um critério para eleger o melhor classificador para tomar como base. Apenas por uma questão de acurácia, e tomando 80.16% como benchmark inicial (pois essa é a acurácia de um classificador naive que resulta classe 0 sempre), poderíamos eliminar KNN, SVC e Random Forest e Adaboost. Os mesmos classificadores também são os que menos performam em termos de falsos negativos (*precision*) também. Como o trabalho tem um motivador muito forte de entender os *drivers* da classificação das *fake news*, é muito importante que o modelo tenha boa performance em termos de *precision*. Em termos de F1-score, os 3 classificadores restantes têm performances muito parecidas.

O segundo critério mais importante na escolha do classificador é a *logloss*. Por um motivo de explicabilidade, é muito interessante obter um modelo com a menor variância possível dentro daqueles que performam bem em termos de *precision, recall* e F1. E num contexto estatístico, um

maior grau de certeza nas probabilidades preditas se reflete numa menor *logloss*. Assim, para ser utilizado como classificador e com melhores propriedades para posterior análise de explicabilidade, optamos pelo modelo de regressão logística.

CONCLUSÃO 2: *O modelo de regressão logística é o que tem melhores propriedades para explicabilidade.*

Um fato que merece destaque das primeiras tabelas de resultados é que as métricas de acurácia, de modo geral, variaram muito pouco com a utilização ou não de *stopwords*, porém a composição da matriz de confusão teve composições diferentes e bastante notáveis no classificador escolhido (regressão logística). Ao adicionar *stopwords* no modelo, falsos negativos diminuíram e os falsos positivos aumentam, mas estes últimos aumentaram a uma taxa menor do que a diminuição dos falsos negativos, levando a um aumento geral da acurácia do modelo.

CONCLUSÃO 3: *Existe um tradeoff entre falsos positivos e falsos negativos no que tange à utilização de stopwords.*

Assim, vamos escolher utilizar *stopwords* para o modelo que utiliza *Word Embeddings* como variável explicativa. Duas observações importantes dessa etapa são que a variância da acurácia dos classificadores diminui e que o melhor modelo (regressão logística) apresenta menos falsos negativos.

Model	Accuracy	TN	TP	FN	FP	Precision	Recall	F1-score	AUC	Logloss
LogisticRegression	0.922455	733	4061	298	105	0.931636	0.974796	0.952727	0.948557	0.210502
BernoulliNB	0.875505	712	3838	319	328	0.923262	0.921267	0.922264	0.903459	0.446823
KNeighborsClassifier	0.915336	625	4132	406	34	0.910533	0.991839	0.949449	0.951592	0.375540
DecisionTreeClassifier	0.883202	592	3998	439	168	0.901059	0.959674	0.929443	0.766937	4.034133
RandomForestClassifier	0.878391	403	4162	628	4	0.868894	0.999040	0.929433	0.979816	0.253915
AdaBoostClassifier	0.878199	619	3945	412	221	0.905440	0.946952	0.925730	0.900446	0.665157

Tabela 4: *Word Embeddings* com *stopwords*

CONCLUSÃO 4: *Word Embeddings têm uma vantagem em relação ao TF-IDF na qualidade da representação numérica das palavras, o que se reflete no fato de que o desempenho dos classificadores do primeiro tem menor variância, e desempenho mínimo de acurácia é de 87,55% enquanto que no TF-IDF é de 80,16% (naive), o que pode ser produto da propriedade de vetores densos e não esparsos dos vetores de embedding.*

6. LIME: Locally Interpretable Model-Agnostic Explanations

Com o objetivo de explicabilidade do modelo, mais especificamente de encontrar as palavras mais importantes para a classificação, usarei o modelo LIME, que funciona como um aproximador local do classificador.

Para formalizar melhor a metodologia do LIME, vamos definir G como uma família de classificadores interpretáveis, e para cada $g \in G$, podemos definir uma medida de complexidade do classificador $\Omega(g)$, em oposição à explicabilidade. Suponha que uma amostra é classificada pelo classificador f, sendo $f(x)$ a probabilidade de que x pertença a uma certa classe. Também definimos $\pi_x(z)$, que nos dá a distância entre uma observação z e x, podendo definir então uma vizinhança ao entorno de x. Finalmente, suponha uma função $\mathcal{L}(f, g, \pi_x)$ que nos dá uma medida de quão mal g aproxima f na localidade definida por π_x.

Finalmente, a o modelo escolhido pelo LIME é $\xi(x) = argmin_{g \in G}\ \mathcal{L}(f, g, \pi_x) + \Omega(g)$, de modo que escolhemos g que minimiza tanto $\mathcal{L}$, garantindo uma boa aproximação local e $\Omega(g)$, garantindo explicabilidade .

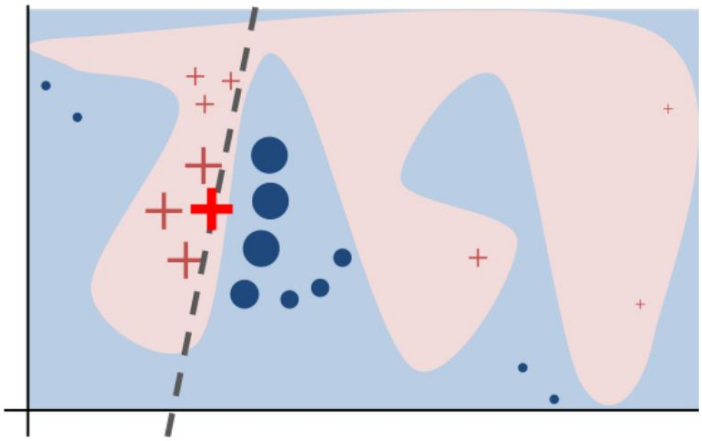

Fig 8: exemplo intuitivo do LIME. Fonte: Singh, S., Guestrin, C., Ribeiro, M. (2016).

Na figura acima, vemos que a fronteira de decisão do modelo mais complexo f é representado pelo fundo rosa e azul. A cruz vermelha mais grossa é a observação x sendo explicada. A partir daí, tira-se uma amostra de observações a serem classificadas usando f, cujas importâncias são ponderadas pela distância a x, a observação sendo explicada. A linha pontilhada é o modelo explicativo aprendido, que é fiel localmente a f, mas não globalmente.

Dessa forma, ao utilizar o LIME, podemos ter uma medida de importância de cada palavra ao classificar um título de notícia, ainda sabendo se seu impacto colaborou na direção *fake* ou não. Por exemplo, para a notícia "*Indústria brasileira reage com melhora do comércio internacional*", temos o seguinte output:

Indústria brasileira reage com melhora do comércio internacional

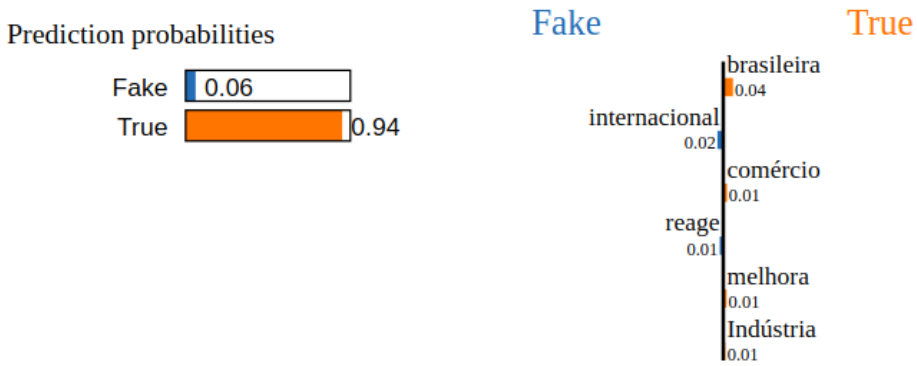

Fig 9: Palavras mais significativas obtidas com o LIME (1)

A palavra "brasileira" foi a mais significativa em termos de valores absolutos de impacto, seguida por "internacional", "comércio", "reage", "melhora" e "indústria". No entanto, as palavras "internacional" e "reage" impactam a probabilidade da notícia ser verdadeira para baixo. Ainda, a palavra de maior impacto teve um impacto de aproximadamente 4% na probabilidade da notícia ser verdadeira. Neste caso em específico, não temos palavras muito marcantes pois seus impactos são baixos em termos de magnitude.

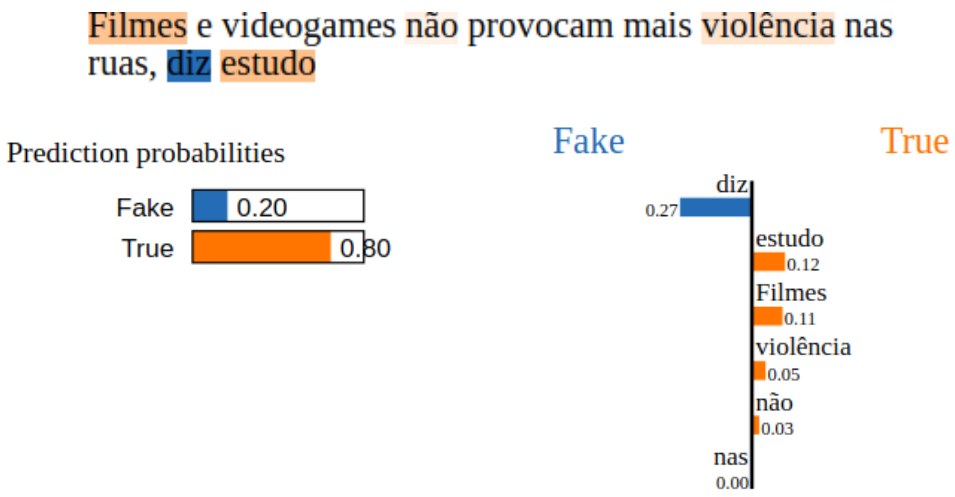

Fig 10: Palavras mais significativas obtidas com o LIME (2)

Já neste outro caso, temos a notícia "*Filmes e videogames não provocam mais violência nas ruas, diz estudo*". É possível ver que agora temos palavras bem mais impactantes em termos de magnitude. A palavra "diz", por exemplo, reduziu em 27 pontos percentuais a probabilidade da notícia ser verdadeira, ao passo que uma das palavras "estudo" e "filmes" aumenta em mais de 10

pontos percentuais a probabilidade da notícia ser verdadeira. As figuras acima mostram o impacto de cada palavra na notícia específica, porém, para ter um entendimento global, também irei reportar os impactos médios por palavra.

Podemos segmentar as palavras de maior impacto naquelas com impacto na direção de diminuir a probabilidade da notícia ser verdadeira e aquelas que aumentam essa probabilidade.

#	**Palavras Fake**	**Efeito**	**Palavras True**	**Efeito**
1	Lula	-0.432823	Petrobrás	0.199730
2	Bolsonaro	-0.415960	Marquéz	0.155084
3	Boatos	-0.393998	Pé	0.149210
4	Temer	-0.351613	Neutralidade	0.130236
5	Dilma	-0.337519	Nazistas	0.119001
6	Menina	-0.325579	Perfeita	0.109333
7	Bandidos	-0.323430	Mexicana	0.109016
8	Gilmar	-0.322311	Telefônica	0.104287
9	Boato	-0.317396	Ouro	0.102923
10	PT	-0.314835	Mercado	0.102211
11	Diz	-0.305873	Espécie	0.100704
12	Encontrada	-0.305011	Awards	0.098492
13	Moro	-0.300776	Namorado	0.096771
14	Causa	-0.299047	Células	0.095265
15	Dando	-0.294733	Principal	0.092039
16	Pastor	-0.293603	Fifa	0.091120
17	Gleisi	-0.291079	Gás	0.090782
18	Policiais	-0.286279	Cérebro	0.090649
19	Morreu	-0.282236	Cinema	0.089352
20	Falsa	-0.278480	Twitter	0.087616

Tabela 5: Palavras mais importantes obtidas com o LIME

CONCLUSÃO: *As palavras fake mais significativas têm impacto em termos de magnitude consideravelmente maiores em relação às palavras mais significativas para verdadeiro. Além*

disso, as palavras fake, no geral, se tratam de um tema muito claro: política e pessoas relacionadas, enquanto que nas palavras de efeito True os temas são dispersos.

7. Índice para ranking de fontes de notícias

Nesta etapa, usarei o modelo de classificação para criar um índice de qualidade das fontes de notícia. Para tal, foram coletadas 200 notícias, 20 notícias em 10 portais de notícias de diferentes origens geográficas no Brasil, como vê-se na tabela a seguir:

Portal	Origem
Zero Hora	Porto Alegre
Gazeta do Povo	Curitiba
Folha de São Paulo	São Paulo
Tribuna do Planalto	Goiás e Tocantins
Correio da Paraíba	Paraíba
A crítica de Manaus	Amazonas
Estado de Minas	Minas Gerais
O Globo	Rio de Janeiro
Diário do Nordeste	Ceará
Correio da Bahia	Bahia

Tabela 6: Lista de portais de notícias e origens geográficas

7.1. O Índice de Sharpe

O Índice de Sharpe é um índice muito utilizado em finanças para medir o desempenho de ativos financeiros. Esse índice irá medir o retorno médio de um ativo acima de um ativo acima de um benchmark, e.g. risk-free rate, ajustado pelo risco do ativo, representado pelo desvio padrão dos retornos.

$$S = \frac{R_p - R_f}{\sigma_p}$$

7.2. O Índice de Sharpe aplicado às notícias

Cada notícia pode ser representada por $N_{ij}, i = 1, \ldots, 10, j = 1, \ldots, 20$, onde i é o índice que indica o portal de 1 a 10, e j indica a notícia de 1 a 20. Para cada notícia, podemos calcular a probabilidade de ser verdadeira, ou seja, $P(N_{ij} = True)$.

Vamos considerar que cada portal seja uma espécie de ativo, cujo retorno vem é dado por ter notícias verdadeiras, ou com alta probabilidade de serem verdadeiras. Deste modo, podemos definir um análogo de retorno médio para um portal i como $R_i = \frac{1}{20} \sum_{j=1}^{20} P(N_{ij} = True)$, ou seja, a média das probabilidades das 20 notícias deste portal serem verdadeiras. Como uma medida de risco, podemos usar o desvio padrão dessas probabilidades, pois se o portal tem notícias com muita variabilidade de fake ou verdadeira, trata-se de um portal mais instável e mais arriscado de ser consumido.

Portanto, definamos $SD_i = M_i = (\frac{1}{20} \sum_{j=1}^{20} (P(N_{ij} = True) - M_i)^2)^{\frac{1}{2}}$. Como uma medida de "performance do mercado", vamos tirar a média de probabilidade da notícia ser verdadeira em todas as notícias de todos os websites, ou seja, $M = \frac{1}{200} \sum_{i=1}^{10} \sum_{j=1}^{20} P(N_{ij} = True)$. Desta forma, uma primeira proposição de índice para classificar os portais seria $S_i = \frac{R_i - M}{SD_i}$. A tabela a seguir mostra esses resultados.

Fonte	Local	Ri	SDi	Si
O Globo	Rio de Janeiro	0,8	0,15	0,33333333
Gazeta do Povo	Curitiba	0,81	0,19	0,31578947
Folha de São Paulo	São Paulo	0,8	0,19	0,26315789
Correio da Paraíba	Paraíba	0,79	0,18	0,22222222
Zero Hora	Porto Alegre	0,78	0,21	0,14285714
O Estado de Minas	Minas Gerais	0,76	0,27	0,03703704
Conjunto Completo		0,75	0,22	0
Tribuna do Planalto	Goiás e Tocantins	0,74	0,23	-0,0434783
G1	Nacional	0,72	0,17	-0,1764706
Diário do Nordeste	Ceará	0,7	0,23	-0,2173913
Correio da Bahia	Bahia	0,69	0,25	-0,24
A Crítica de Manaus	Manaus	0,66	0,21	-0,4285714

Tabela 7: Primeira versão dos índices de Sharpe para os portais de notícia

Um potencial problema com esse approach é que ao considerar o benchmark como a média geral das fontes, podemos estar sobre penalizando fontes que apesar de não terem retorno tão alto, tem baixa variância, como é o caso do G1. Calculando um novo índice dado por $S_i = \frac{R_i}{SD_i}$, temos:

Fonte	Local	Ri	SDi	Si
O Globo	Rio de Janeiro	0,8	0,15	5,33333
Correio da Paraíba	Paraíba	0,79	0,18	4,38889
Gazeta do Povo	Curitiba	0,81	0,19	4,26316
G1	Nacional	0,72	0,17	4,23529
Folha de São Paulo	São Paulo	0,8	0,19	4,21053
Zero Hora	Porto Alegre	0,78	0,21	3,71428
Conjunto Completo		0,75	0,22	3,40909
Tribuna do Planalto	Goiás e Tocantins	0,74	0,23	3,21739
A Crítica de Manaus	Manaus	0,66	0,21	3,14286
Diário do Nordeste	Ceará	0,7	0,23	3,04347
O Estado de Minas	Minas Gerais	0,76	0,27	2,81481
Correio da Bahia	Bahia	0,69	0,25	2,76000

Tabela 8: Segunda versão dos índices de Sharpe para os portais de notícia

8. Conclusão

Neste artigo, foram utilizadas ferramentas de econometria, machine learning e processamento de linguagem natural para análise de dados de notícias verdadeiras e *fake*. Primeiramente, foram implementados modelos de classificação de texto com base nos títulos das notícias e, para tal, foram utilizados dois métodos para transformação de texto em vetores numéricos: o TF-IDF e word2vec.

O modelo com melhor performance foi a regressão logística em ambos os casos, sendo essa performance medida pela acurácia. Ainda, a acurácia da regressão logística foi bastante parecida nos dois casos: 92,86% para o TF-IDF e 99,25% para o word2vec.

No entanto, considerou-se que melhor opção seria o word2vec, pois apesar de uma acurácia marginalmente menor, apresentou muito menos variabilidade na acurácia dos diversos classificadores utilizados, sendo que a menor acurácia foi de 87,55%, que é bem razoável quando comparada aos 80,16% do TF-IDF. Isso sugere que há uma maior qualidade da representação vetorial do texto com o word2vec. Além disso, com o word2vec houve menos falsos negativos (*fakes* classificados como verdadeiros) na classificação quando comparado ao TF-IDF.

Também foi possível verificar nessa etapa de classificação que a remoção ou não de stopwords apresenta um t*radeoff* em termos de Falsos Positivos e Falsos Negativos ainda mais acentuado. Enquanto que ao utilizar *word2vec* diminuiu-se 12 falsos positivos, a utilização das stopwords diminuiu 44 falsos positivos. Possivelmente, isso se deve ao fato de que as stopwords, apesar de não carregarem palavras com significados expressivos, denunciam o modo de escrita *fake*, que pode conter mais conectivos, pronomes, etc., sugerindo que o modo como se escreve tem suma importância para entender as *fake news*, quando olhamos algo além do próprio conteúdo.

Posteriormente, utilizando o modelo de classificação escolhido e o modelo LIME para interpretabilidade, é possível calcular o impacto positivo ou negativo que cada palavra tem sobre a probabilidade de uma determinada notícia ser verdadeira. Calculou-se, utilizando todas as notícias da base de dados, o efeito médio associado a cada palavra. Dessa forma, foi possível encontrar evidências de que política foi um alvo temático dos sites não confiáveis pois, dentre as palavras com maior impacto negativo para a probabilidade da notícia ser verdadeira, uma grande parcela é diretamente associada à política ou figuras políticas. Por exemplo, dentre as 10 palavras mais *fake*, estão inclusas Lula, Bolsonaro, Temer, Dilma, Gilmar e PT. Contrariamente, as palavras que mais influenciam para verdadeiro tem temas muito mais diversos, não sendo possível identificar um alvo temático tão evidente. Ainda, verificou-se que o efeito em termos absolutos das palavras mais *fake* são consideravelmente maiores que os das palavras mais verdadeiras.

Finalmente, foi coletada uma amostra de 20 notícias de 10 portais diferentes de maneira a criar um ranking de confiabilidade que utilizasse as etapas anteriores do artigo. Este índice criado foi inspirado no índice de Sharpe. Observou-se que o índice, de modo geral, premiou portais de notícia grandes como o Globo e Folha de São Paulo, o que é bastante razoável e animador, dado que notícias desses portais não entraram no treinamento do modelo e esses portais foram reconhecidos como de melhor qualidade.

No entanto, o trabalho apresenta limitações e existem pontos a serem melhorados em pesquisas futuras. Foram coletadas notícias de 3 portais apenas, assim, o trabalho poderia ser expandido para lidar com uma diversidade muito maior de notícias e temas ao coletar dados de mais fontes. Além disso, a base de dados foi coletada apenas para este artigo, portanto, não há um benchmark de comparação com outros modelos. Assim, a metodologia deste trabalho poderia ser testada com dados de texto amplamente utilizados pela literatura como benchmark. A pesquisa também poderia ser expandida ao utilizar outros métodos de classificação utilizando *deep learning*, que não foi o enfoque do trabalho. Por exemplo, é possível utilizar a família modelos de atenção [Vaswani, A. et al. (2017)] ou BERT [Devlin, J. et al. (2019)] para classificação de texto. Finalmente, pesquisas futuras também poderiam explorar métricas mais robustas para rankeamento dos portais de notícia.

Por fim, vemos que mesmo com técnicas avançadas de classificação de texto para o problema de fake news, ainda existe uma porção de notícias bastante considerável que não é devidamente classificada. Assim, é necessária muita cautela quando se abre a questão de implementação de algoritmos de detecção e bloqueio de *fake news* na internet dado que a linha que separa *fake news* de notícias verdadeiras pode ser bastante tênue para uma parcela significativa de notícias.

9. Bibliografia

Posetti, J., Matthews, A. (2018). A short guide to the history of ‘fake news’ and disinformation. International Center for Journalists (ICFJ).

Zhou, X., Zafarani, R. (2018). Fake News: A Survey of Research, Detection Methods, and Opportunities. arXiv:1812.00315v1

Wu, Alice (2017). Gender Stereotyping in Academia: Evidence from Economics Job Market Rumors Forum.

Gentzkow, M., Allcott, H. (2017). Social Media and Fake News in the 2016 Election. Journal of Economic Perspectives—Volume 31, Number 2, Spring 2017, 211–236.

Brummette, J., DiStaso, M., Vafeiadis, M., & Messner, M. (2018). Read All About It: The Politicization of "Fake News" on Twitter. Journalism & Mass Communication Quarterly, 95(2), 497–517.

Gentzkow, M., Shapiro, J., Stone, D. (2014). Media Bias in the Marketplace: Theory. Handbook of Media Economics, Volume 1, 2015, 623-645.

Gentzkow, M., Kelly, B., Taddy, M. (2017). Text as Data. National Bureau of Economic Research (NBER) Working Paper No. 23276.

T. Hastie, R. Tibshirani, and J. Friedman. Springer Series in Statistics Springer New York Inc., New York, NY, USA, (2001).

Mikolov, T. et al. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781v3.

Hartmann, N. et al. (2017). Portuguese Word Embeddings: Evaluating on Word Analogies and Natural Language Tasks. arXiv:1708.06025v1.

Rong, X. (2016). Word2vec Parameter Learning Explained. arXiv:1411.2738v4.

Deep Learning (Ian J. Goodfellow, Yoshua Bengio and Aaron Courville), MIT Press, 2016.

Numerical Optimization (Jorge Nocedal and Stephen J. Wright), Springer, 2006.

Singh, S., Guestrin, C., Ribeiro, M. (2016). "Why Should I Trust You?" Explaining the Predictions of Any Classifier. arXiv:1602.04938v3.

Vaswani, A. et al. (2017). Attention Is All You Need. arXiv:1706.03762v5.

Devlin, J. et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805v2.

10. Apêndice

10.1. 100 Palavras com maior impacto para fake

'lula', 'bolsonaro', 'boatos', 'temer', 'dilma', 'menina', 'bandidos', 'gilmar', 'boato', 'pt', 'diz', 'encontrada', 'moro', 'causa', 'dando', 'pastor', 'gleisi', 'policiais', 'morreu', 'falsa', 'deputado', 'grátis', 'exército', 'assaltos', 'rj', 'vídeo', 'texto', 'jesus', 'muçulmanos', 'flagrado', 'diabo', 'professora', 'telefone', 'aécio', 'general', 'link', 'aparece', 'stf', 'cadeia', 'congresso', 'vagas', 'farsa', 'video', 'rosário', 'pm', 'lúcia', 'mata', 'igreja', 'amarela', 'falso', 'vai', 'joesley', 'políticos', 'org', 'web', 'aponta', 'pf', 'cobra','juízes', 'lei', 'povo', 'sendo', 'odebrecht', 'cabral', 'ministros', 'senado', 'praia', 'cura', 'pokémon', 'ladrão', 'doação', 'vírus', 'sergio', 'câmara', 'intervenção', 'criança', 'governo', 'fátima', 'acaba', 'lava', 'globo', 'bebê', 'jean', 'wyllys', 'suicídio', 'filha', 'mensagem', 'bandido', 'mostra', 'gay', 'adriana', 'marisa', 'jato', 'preso', 'cunha', 'mil', 'deputados', 'veneno', 'juiz', 'compartilhe', 'baleia', 'pesquisa', 'urgente', 'lucía', 'cão', 'fazenda', 'militar', 'hospital', 'carta', 'supremo', 'janeiro', 'circula', 'militares', 'escolas', 'golpe', 'mst', 'polícia', 'francisco', 'vermelha', 'site', 'cármen', 'brasília', 'dodge', 'avião', 'foto', 'amanhã', 'armadas', 'programa', 'advogados', 'neves', 'carmen', 'maduro', 'tsunami', 'matou', 'lúcia', 'bebê', 'mega', 'palocci', 'alerta', 'mendes', 'eleições', 'exército', 'públicos', 'crianças', 'leia', 'presidente', 'protesto', 'assalto', 'delação', '7', 'corruptos','friboi', 'islâmico', 'batista', 'carros', 'jovem', 'ministra', 'usam', 'senador', 'celular', 'renan', 'depoimento', 'caso', 'dentro', 'pokémon', 'informação', 'pra', 'coca', 'república', 'íntegra', 'será', 'população', 'psdb', 'grávida', 'morto', 'cão', 'esquerda', 'juca', 'presos',

'whatsapp', 'federal', 'áudio', 'eduardo', 'falsos', 'usá', '1º', 'mal', 'vêm', 'petista', 'após','calheiros', 'bíblia', 'carro', 'marido', 'hoje', 'mãe', 'mulher','mg', 'mãos'.

10.2. 100 Palavras com maior impacto para verdadeiro

'petrobrás', 'marquéz', 'pe', 'neutralidade', 'nazistas', 'perfeita', 'mexicana', 'telefônica', 'ouro', 'mercado', 'espécie', 'awards', 'namorado', 'células', 'principal', 'gás', 'fifa', 'cérebro', 'cinema', 'twitter', 'ma', 'ondas', 'oms', 'oprah', 'flip', 'óscar', 'software', 'mudar', 'dólar', 'inimigo', 'machismo', 'banda', 'riscos', 'oscar', 'técnica', 'italia', 'sono', 'assim', 'estudo', 'note', 'mensagens', 'jagger', 'escritor', 'google', 'iphone', 'liberdade', '55', 'felicidade', 'muita', 'desafio', 'resistência', 'futebol', 'neymar', 'george', 'bem', 'lo', 'grandes', 'teme', 'notícias', '000', 'carlsen', 'mudança', 'adeus', 'fiscal', 'jazz', 'pornô', 'ryan', 'ajuste', 'ciência', 'orgulho', 'apple', 'hollywood', 'pessôa', 'guia', 'comida', 'dalí', 'gelo', '1986', 'redes', '2022', 'primeira', 'fmi', 'amazon', 'dna', 'acesso', 'tempos', 'polêmica', 'planeta', 'esperança', 'cozinha', 'tecnologia', 'argentina', 'carnaval', 'continua', 'vida', 'jogador', 'notícias', 'apoia', 'misterioso', 'primeiro', 'conhece', 'coisa', 'jogos', 'voz', 'messi', 'humanidade', 'atropelada', 'arte', 'depp', 'ue', 'reconhecer', 'petrobras', 'economia', 'princesa', 'méxico', 'plutão', 'pele', 'bolsas', 'sinais', 'instagram', 'qualidade', '2020', 'longe', 'velha', 'gene', 'la', 'oliver', 'crescer', 'scolari', 'outras', 'brown', 'espanha', 'john', 'rainha', 'retrato', 'medidas', 'bruno', 'celulares', 'gesto', 'reais', 'periferia', 'personagens', 'último', 'alonso', 'viajar', 'barça', 'você', 'minutos', 'sede', 'zika', 'desconhecido', 'movimento', 'nobel', 'acredita', 'rei', 'recuperar', 'costas', 'tempo', 'fome', 'tão', 'memória', 'felipão', 'munique', 'capítulo', 'tela', 'lições', 'apesar', 'snapchat', '99', 'especialista', '90', 'el', 'associação', 'espero', 'nunca', 'máximo', 'david', 'bolas', 'silício', 'chegada', 'nadal', 'férias', 'nada', 'brasileira', 'espanhola', 'dunga', 'the', 'mirror', 'álcool', 'drones', 'telefónica', 'feminista', 'toró', 'shakespeare', 'cultura', 'johnny', 'séries','silêncio', 'pódio', 'paris'.