Análise de Fake News

Andreza Fernandes de Oliveira, Manuel Edvar Bento Filho
Departamento de Computação
Universidade Federal do Ceará
andrezafernandes@alu.ufc.br
edvarfilho1234@gmail.com

Abstract

O presente artigo visa mostrar o processo de elaboração de modelos para classificar notícias em verdadeira ou falsa. O conjunto de dados utilizado se encontra disponível no Kaggle, sendo composto de notícias de diferentes fontes. Os modelos utilizados neste artigo são o Regressão Logística, Análise Discriminante Gaussiano, K-Nearest Neighbors, Árvore de Decisão, Support Vector Machine e Random Forest. Por fim, é realizado uma comparação e análise dos resultados dos modelos.

1 Introdução

O atual cenário político-social mundial tem se mostrado fortemente impactado pela disseminação de Fake News, as quais vem ganhando bastante notoriedade com o crescente uso de redes sociais e divulgação das mesmas nesses meios.

Em [1] há um exemplo claro do impacto de fake news, onde há um pequeno relato do processo eleitoral ocorrido em 2016 nos Estados Unidos, onde hackers russos impulsionaram divulgação de conteúdo que prejudicou a campanha de um dos candidatos a presidência, favorecendo a vitória do candidato Donald Trump. No cenário brasileiro, no processo eleitoral do ano de 2018, o WhatsApp se tornou a principal ferramenta para a disseminação de dados falsos envolvendo as eleições [3].

Em [6] Friedman cita que para criar uma história convincente, mentirosos falam de certas maneiras na tentativa de enganar o receptor da mensagem. Já que essas formas de expressão são intrínsecas de quando contamos histórias que sabemos não ser verdade, são bastante difíceis de serem modificadas e se torna algo passível de ser detectado via Machine Learning.

Publicações subsequentes à de Friedman indicam que podemos aprender muito sobre as emoções e motivações de alguém ao analisar as palavras usadas na comunicação. Métodos tão simples quanto contar a quantidade de palavras usadas podem ajudar a julgar se o fato comunicado é ou não verdadeiro. O uso de determinados tipos de palavras também é um bom indicador para modelos classificadores de mentiras.

Visto que as fakes news acabam servindo como um meio de alienar a população, o presente artigo possui o objetivo de elaborar um modelo que possa classificar se determinada notícia é verdadeira ou falsa. Esse modelo seria treinado com um conjunto de notícias, de diferentes fontes, que possuam notícias verdadeiras e falsas. Após a validação e teste do modelo elaborado, o mesmo seria utilizado para classificar demais noticias em fake ou não.

2 Fundamentação Teórica

O problema trabalhado neste artigo trata-se de uma classificação binária. O conjunto de dados utilizado foi extraído do Kaggle [2], sendo fruto de um web crawler de diferentes fontes americanas, consistindo ao todo em 4009 registros de notícias, sendo 2137 notícias falsas e 1872 notícias verdadeiras,

Os atributos originais desse conjunto de dados consistem:

- **URL:** refere-se a fonte da notícia;
- **Headline:** refere-se ao título da notícia;
- **Body:** refere-se ao corpo da notícia;
- **Label:** efere-se a classe da notícia.

As features trabalhadas neste artigo são **Headline** e **Label**.

2.1 Trabalhos Relacionados

Pesquisas sobre detecção de notícias falsas ainda estão em um estágio inicial, já que este é um fenômeno relativamente recente, pelo menos no que diz respeito ao interesse levantado pela sociedade. Analisamos dois trabalhos:

Em **[7]** foi desenvolvida uma plataforma, que integra o projeto *Detecção Automática de Notícias Falsas para o Português*, com o objetivo de detectar notícias falsas com o uso de um modelo elaborado a partir da técnica Linear SVC, com parâmetros pré-definidos, e com o uso de um conjunto de dados com notícias brasileiras no período de Janeiro de 2016 à Janeiro de 2018, com tamanho de 7.200 notícias, onde cada notícia verdadeira corresponde a uma falsa. Para avaliar o algoritmo foi utilizado as métricas Precision, Recall, F-measure e a Acurácia. No artigo é citado que um dos parâmetros relevantes para identificar notícias falsas na plataforma é a quantidade de erros ortográficos e advérbios, além de outros parâmetros como o número médio de substantivos, adjetivos e pronomes nos textos.

Em **[5]** há a descrição da construção de um modelo que detecta fake news usando análise de n-gramas. As técnicas utilizadas foram K-Nearest Neighbors, Support Vector Machine, Regressão Logística, Máquina Linear de Vetores de Suporte, Árvore de Decisão e Descendente de Gradiente Estocástico. Nesse trabalho também foi utilizado algumas das mesmas técnicas para limpeza dos dados como remoção das stop words, técnica de stemming e extração de features utilizando Term Frequency (TF) and Term Frequency-Inverted Document Frequency(TF-IDF). A avaliação com melhor desempenho foi obtida usando TF-IDF e Linear Support Vector Machine (LSVM) como classificador, obtendo com uma precisão de 0.92.

3 Metodologia

O processo de elaboração dos modelos consistem em 3 fases, que estão descritas logo abaixo, e podem ser encontradas no repositório Github [4].

3.1 Pré-processamento dos Dados

Os dados trabalhados tratam-se de strings, o que tornou necessário a tomada dos procedimentos abaixo:

- **Lower Case:** Processo de transformação das textos para sua forma minúscula/caixa baixa. Este processo é importante devido ser necessário deixar as palavras em um mesmo patamar, e para mais tarde evitar duplicação de features, no processo de Bag of Word, por diferenciar as mesmas palavras por questões de estarem com letras minúsculas ou maiúsculas;
- **Remoção de pontuação:** Processo de remoção dos sinais de pontuação dos textos;
- **Remoção de Stop Words:** Stop Words referem-se a palavras de paradas, como preposições, sujeitos e etc. Neste processo foi efetuado a remoção dessas palavras. Esta fase é importante devido aos textos contarem muitas preposições, advérbios e demais palavras que não influem e/ou interferem no real sentido da frase. Além disso para reduzir a dimensão do dado;
- **Remoção das palavras mais e menos frequentes:** Processo de contagem e identificação das 10 palavras mais e menos frequentes e remoção das mesmas;
- **Stemming/Lemmarization:** Processo de extração do núcleo de uma palavra. A importância desta fase é a mesma citada no Lower Case, evitar criar features desnecessárias no processo de Bag of Word;

- **Bag of Words:** Processo de geração de uma matriz, onde as suas colunas são todas as palavras que ocorrem no dataset e suas linhas as mesmas do dataset original. A matriz irá armazenar valores 1s nas colunas em que tal palavra compõe a sentença daquela linha, e 0s nas demais. Esse procedimento é essencial para que possamos gerar modelos e trabalhar com nosso dataset, visto que os algoritmos de aprendizagem esperam por valores numéricos.

3.2 Redução de Dimensionalidade

Após o pré-processamento inicial houve um grande aumento no número de atributos do conjunto de dados, sendo 6333. Para contornar este problema foi utilizado algumas técnicas de redução de dimensionalidade, o **Fisher Score**, que trata-se de um filtro, e o **Análise de Componentes Principais (PCA)**, que trata-se de um algoritmo não-supervisionado que combina os dados.

Ao buscar 90% da variância explicada total dos dados, temos um dataset com 1443 atributos e ao querer diminuir este novo conjunto, utilizamos o fisher score e removemos os 1000 dados menos relevantes para o modelo.

3.3 Modelos

Após a limpeza dos dados, separamos os conjuntos de treino e teste, com um porcentagem de 25% para teste. Os algoritmos utilizados, já citados anteriormente, foram:

- **Regressão logística:** semelhante a regressão linear, visa classificar os dados em duas classes por meio de um função polinomial que mapeie os dados, nas duas classes a partir da função sigmóide, isto, pois temos um conjunto de dados com uma enormidade de atributos. Possui como hiperparâmetros o **passo de aprendizagem** e o **número de épocas do treinamento** do modelo;
- **Análise Discriminante Gaussiano:** a partir de uma visualização e análise dos dados estatísticos dos atributos é possível realizar o cálculo das probabilidades para as classes a serem preditas, sendo a que obtiver maior probabilidade a que será retornada;
- **K-Nearest Neighbors:** a partir da observação dos K vizinhos mais próximos é possível predizer a classe do registro. Possui como hiperparâmetros o valor de **K**, como a **métrica de distância** a ser utilizada;
- **Árvore de Decisão:** trata-se de um método capaz de gerar um fluxograma para predizer a classe de um registro. Possui como hiperparâmetros o **índice de impureza**, a **profundidade da árvore** e o **número de atributos** a serem levados em consideração;
- **Suppport Vector Machine:** trata-se de um classificador binário não-probabilístico que apodera-se dos vetores de suporte para predizer uma classe ao registro desejado. Possui como hiperparâmetros a **função de kernel**, **termo de regularização C**, e dependendo da função de kernel podemos ter **gama** e **grau do polinômio**;
- **Random Forest:** sendo um comitê de árvores de decisão, combina-as para melhorar o resultado predito, através de "pontos de vistas"diferentes sob os dados, além de adicionar uma certa aleatoriedade ao modelo. Como hiperparâmetros temos o **número de árvores** e a **profundidade máxima das árvores**.

Realizamos o procedimento de Grid Search em busca dos hiperparâmetros que possam maximizar a acurácia, para aqueles modelos que possuam estes.

3.4 Métricas de Avaliação

As métricas utilizadas aqui foram:

- **Acurácia:** a acurácia é capaz de medir o quão próximo estão os valores reais dos valores preditos pelo modelo, obtendo uma relação entre as assertivas e o número total de classes;
- **Precisão:** verifica dado as classes classificadas como verdadeira, quais realmente são verdadeiras, dando assim uma fração de instâncias recuperadas que são relevantes.
- **Revocação:** nos da a fração de instâncias relevantes que são recuperadas.

- **F1-Score:** utlizado em problemas de classificação binária o F1-score é uma medida harmônica entre a precisão e a revocação.
- **Matriz de Confusão:** é um tabela que mostra as frequências de classificação para cada classe do modelo. Mostrando os verdadeiros positivos e negativos e falsos positivos e negativos;

As implementações descritas até aqui encontram-se disponíveis no Github [4].

4 Experimentos

Em nossos experimentos, realizamos uma divisão conjunto de dados para treinamento e para teste. Para treino temos um total de 3009 (75% dos dados) registros e para teste o valor de 1002 (25% dos dados) registros.

Em seguida, realizamos o Grid Search para escolher os melhores hiperparâmetros para os modelos propostos, com exceção da Regressão Logística, no qual foi pré-definido o valor do número de épocas como 1000 e o passo de aprendizagem sendo 0.0001, e do Análise de Discriminante Gaussiano que não possui hiperparâmetros. Para cada um dos algoritmos temos os seguintes hiperparâmetros testados e em seguida o que obteve melhor resultado após a execução do Grid-Search.

- **KNN:**
  Hiperparâmetro: K.
  - **Valores testados:** Valor de K: 3 a 11, variando em duas unidade.
  - **Valor escolhido:** Valor de K: 3.
- **Árvore de Decisão:**
  Hiperparâmetros: Profundidade máxima da árvore e índice de pureza.
  - **Valores testados:** Profundidade máxima: 1 a 50, variando em apenas uma unidade. Índice de pureza: entropia e índice de Gini
  - **Valores escolhidos:** Profundidade máxima: 49. Índice de pureza: entropia.
- **Support Vector Machine:**
  Hiperparâmetros: Função de kernel, valor de C, gama dependendo da função de kernel, assim como o grau do polinômio.
  - **Valores testados:** Função de kernel: gaussiano, polinomial, linear. Valor de C: 2 com potência de -5 a 16 variando-as em duas unidade. Valor de gama para kernel gaussiano: 2 com potência de -15 a 4 variando em duas unidades. Grau do polinômio para kernel polinomial: 2 a 6, variando em uma unidade.
  - **Valores escolhidos:** Função de kernel: gaussiano. Valor de C: 8. Gama: 0.125.
- **Random Forest:**
  Hiperparâmetros: Número de estimadores e profundidade máxima para as árvores.
  - **Valores testados:** Número de estimadores: 5, 10 e 15. Profundidade máxima: 1 a 4, variando em apenas uma unidade. Índice de pureza: entropia ou índice de Gini.
  - **Valores escolhidos:** Índicce de pureza: entropia. Número de estimadores: 146. Profundiade máxima: 9.

Após o treinamento com os melhores hiperparâmetros selecionados anteriormente, realizamos os testes e obtivemos os resultados ilustrados na Tabela 1, para cada uma de nossas métricas, além das matrizes de confusão obtidas por cada modelo.

Com relação a Tabela de resultados 1 podemos inferir que no que diz respeito a todas as métricas avaliadas o melhor modelo foi o SVM, pelo simples fato de que o SVM busca encontrar a maior margem que separa as notícias falsas das verdadeiras, mesmo sendo suscetível a erros ocasionados por outliers e exemplos rotulados errado, com o auxílio da variável de folga este problema pode ser minimizado.

Como os nossos dados não são linearmente separáveis a Regressão Logística, considerado o pior modelo avaliado, não é capaz de traçar uma reta que separe liearmente as duas classes. Estimamos

Tabela 1: Avaliação dos Algoritmos

Algoritmos	**Acurácia**	**Precisão**	**Revocação**	**F1-Score**
Regressão Logística	0.52	0.26	0.50	0.34
Análise de Discriminante Gaussiano	0.83	0.84	0.84	0.83
K-NN	0.65	0.69	0.64	0.62
Árvore de Decisão	0.75	0.75	0.74	0.74
SVM	**0.89**	**0.89**	**0.89**	**0.89**
Random-Forest	0.82	0.83	0.82	0.82

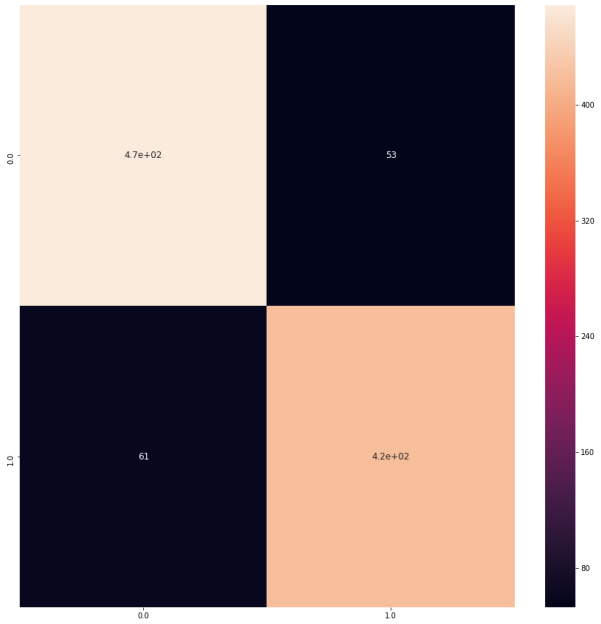

Figura 1: Matriz de Confusão - SVM

que por esse fato, a função de kernel escolhida no Grid-Search para o SVM foi a gaussiana, que permite o mapeamento por meio de uma função não-linear os dados para um plano que se torne liearmente separável uma classe da outra.

Para melhor visualização desta diferença, basta anaisarmos os resultados obtidos nas matrizes de confusão de ambos os modelos na Figura 1 e na Figura 2, pois é perceptível que mesmo a função custo da regressão logística ir convergindo ela prediz que todos os dados de teste não notícias verdadeiras.

Em relação ao resultado do K-NN é possível dizer que qualquer variância em uma palavra do título da notícia pode movê-la para uma região de notícias de outra classe diferente da sua, principalmente quando a notícia é falsa pelo fato de que foi removido o núcleo de todas as palavras, a escrita errada não pôde ser notada por este modelo, pois as notícias falsas têm essa característica em particular. O dito, pode ser obeservado na Figura 3 ao ser visto que muitas notícias falsas foram preditas como verdadeiras.

Sobre a Análise Discrimante Gaussiano o resultado foi ótimo, até mesmo antes da execução do Grid-Search para os modelos, ela foi caracterizada melhor que o SVM, pois ela mapeia as probabilidades das classes com base nos dados estatísticos dos atributos em questão, mas caso não fosse adicionado uma espécie de ruído em um dos fatores calculados ocorreria que no cálculo da determinante da matriz de covarância das classes o resultado seria 0. O resultado positivo é ilustrado na Figura 4, onde é possível notar que a grande maioria dos dados de teste foram preditos corretamente.

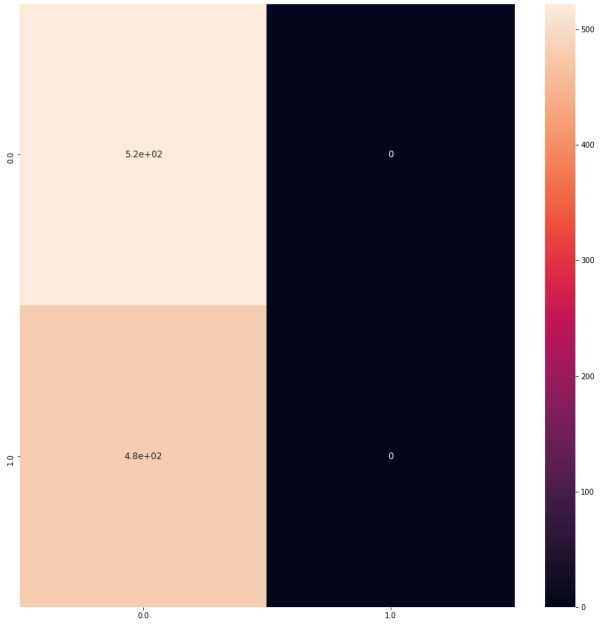

Figura 2: Matriz de Confusão - Regressão Logística

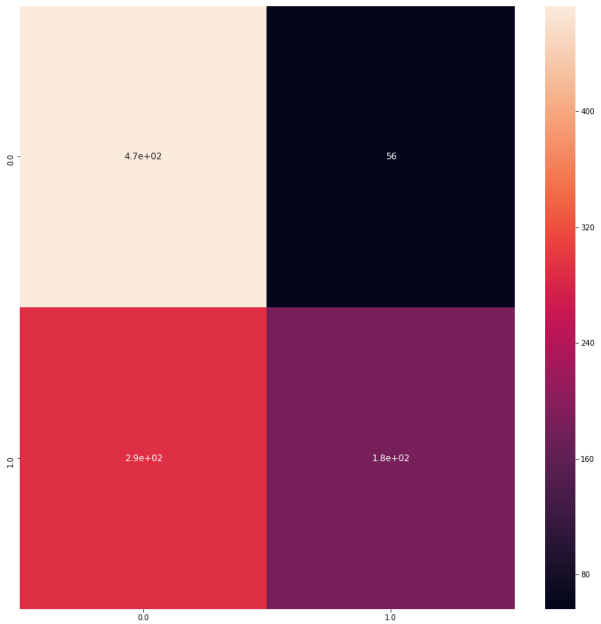

Figura 3: Matriz de Confusão - KNN

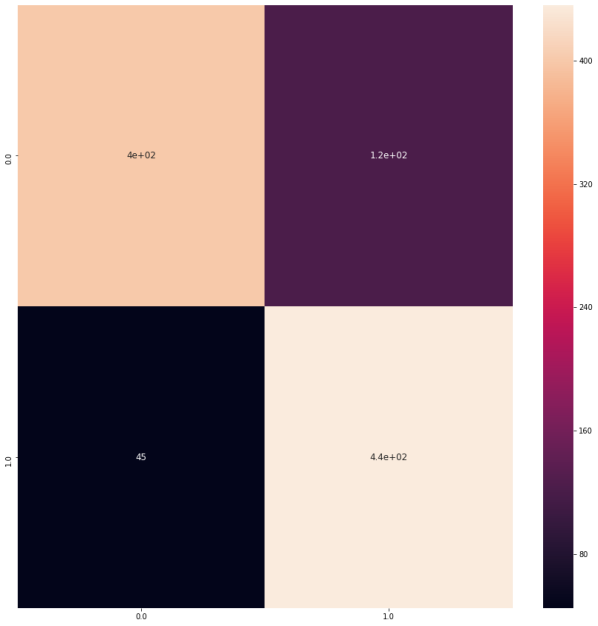

Figura 4: Matriz de Confusão - Análise de Discriminante Gaussiano

Sobre a Árvore de Decisão, possivelmente a profundiade da árvore pode ter permitido esse resultado razoavelmente ótimo, caso a árvore possuísse uma profundidade menor, não seria possível separar de forma tão primorosa as notícias em falsas e verdadeiras. Na Figura 5 é possível observar que o que ocasionou esse resultado foi uma taxa de notícias falsas que foram preditas como verdadeiras, possivelmente pelo fato de que em algum momento um dado atributo não foi tratado ou até mesmo algum registro que ficou no limiar da classificação e não foi positivamente classificado.

SNo que diz respeito ao Random Forest, que é uma combinação de diversas árvores de decisão, o resultado positivo foi garantido principalmente pelo número de estimadores e pelo índice de pureza utilizado. O resultado positivo pode ser observado na análise da matriz de confusão deste modelo na Figura 6.

5 Conclusão

Por fim, como já comentado o melhor modelo obtido foi o SVM com função de kernel gaussiana, parâmetro regularizador C igual a 8 e o gamma igual a 0.125. O nosso trabalho atendeu os objetivos requeridos de avaliar diversos modelos para classificação de notícias falsas. O que dificulta a classificação de algumas notícias é a semelhança das notícias falsas e verdadeiras.

5.1 Trabalhos Futuros

Em relação as expectativas para o futuro e melhoramento do projeto, pretendemos utilizar novos algoritmos, tal como o Multilayer Perceptron para elaboração de um novo modelo; moldar novos modelos para notícias brasileiras, fazendo uso assim de datasets brasileiros; explorar e mitigar mais features dos dados, como visto em [5] fazer uso de n-gramas, trazendo assim mais informações.

Referências

[1] A ciência da detecção de fake news. `https://medium.com/@irio/a-ciencia-da-deteccao-de-fake-news-d4faef2281aa`. Acessado em 06/06/2019.

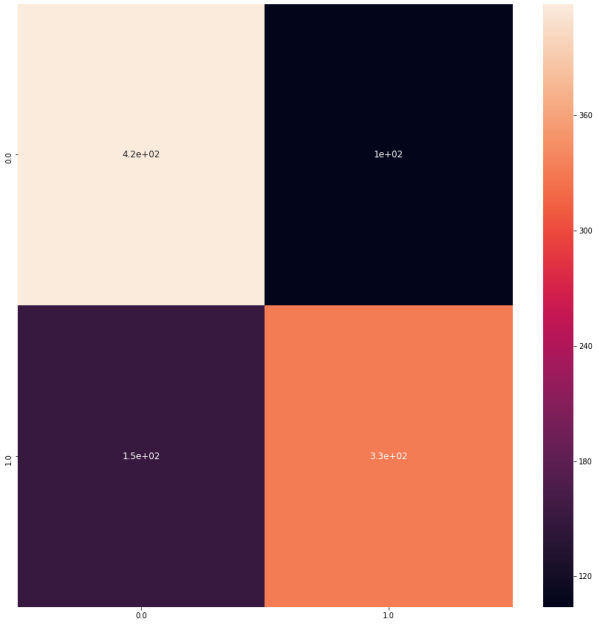

Figura 5: Matriz de Confusão - Árvore de Decisão

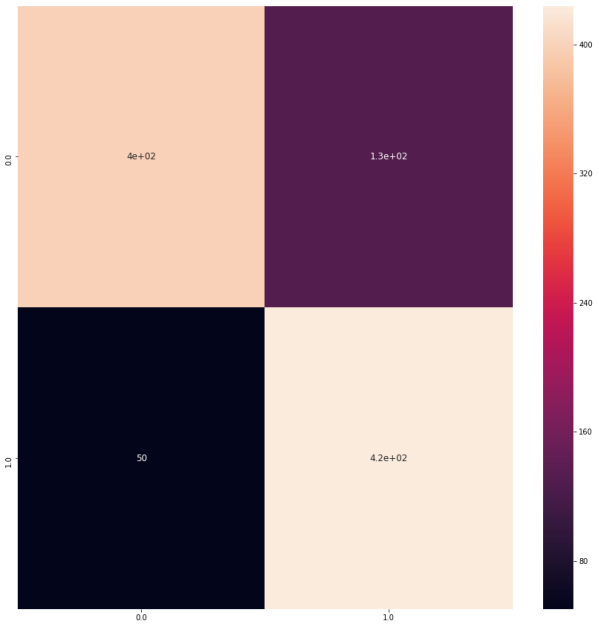

Figura 6: Matriz de Confusão - Random Forest

[2] Fake news detection. https://www.kaggle.com/jruvika/fake-news-detection. Acessado em 06/06/2019.

[3] O impacto das fake news nas eleições 2018. https://politica.estadao.com.br/blogs/fausto-macedo/o-impacto-das-fake-news-nas-eleicoes-2018/. Acessado em 27/06/2019.

[4] Repositório do github. https://github.com/EdvarFilho/TrabalhoAprendizagemDeMaquina/.

[5] Hadeer Ahmed, Issa Traore, and Sherif Saad. Detection of online fake news using n-gram analysis and machine learning techniques. In *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, pages 127–138. Springer, 2017.

[6] Tucker J. S. Tomlinson-Keasey C. Schwartz J. E. Wingard D. L. Criqui M. H Friedman, H. S. Does childhood personality predict longevity? journal of personality and social psychology. *Journal of Neuroscience*, 65(1):176–185, 1993.

[7] Rafael A Monteiro, Roney LS Santos, Thiago AS Pardo, Tiago A de Almeida, Evandro ES Ruiz, and Oto A Vale. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In *International Conference on Computational Processing of the Portuguese Language*, pages 324–334. Springer, 2018.