UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Análise de textos por meio de processos estocásticos na representação word2vec

Gabriela Massoni

Dissertação de Mestrado do Programa Interinstitucional de Pós-Graduação em Estatística (PIPGEs)

ICMC USP SÃO CARLOS

Gabriela Massoni

Análise de textos por meio de processos estocásticos na representação word2vec

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP e ao Departamento de Estatística – DEs-UFSCar, como parte dos requisitos para obtenção do título de Mestre em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística. *VERSÃO REVISADA*

Área de Concentração: Estatística

Orientador: Prof. Dr. Rafael Bassi Stern

USP – São Carlos
Março de 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

M421a

Massoni, Gabriela
Análise de textos por meio de processos estocásticos na representação word2vec / Gabriela Massoni; orientador Rafael Bassi Stern. -- São Carlos, 2021.
58 p.

Dissertação (Mestrado - Programa Interinstitucional de Pós-graduação em Estatística) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. Representação vetorial de palavras. I. Bassi Stern, Rafael, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Gabriela Massoni

Text mining with stochastic process in word2vec representation

Master dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP and to the Department of Statistics – DEs-UFSCar, in partial fulfillment of the requirements for the degree of the Master Interagency Program Graduate in Statistics. *FINAL VERSION*

Concentration Area: Statistics

Advisor: Prof. Dr. Rafael Bassi Stern

USP – São Carlos
March 2021

Aos meus pais, Ademir e Zenira, e ao meu irmão, Giovani,
para que continuem a me incentivar e ser luz na minha vida.

AGRADECIMENTOS

Agradeço a Deus pelo dom da vida e do conhecimento, por me proporcionar saúde e força, especialmente mediante a pandemia do COVID-19.

Agradeço aos meu pais, Ademir e Zenira, e ao meu irmão, Giovani, pelo apoio incondicional.

Agradeço ao meu orientador, Rafael Stern, pela dedicação, companheirismo e confiança na partilha dos conhecimentos na ciência e na vida. Ao professor Rafael Izbicki por estar próximo e dar completo apoio nesta jornada e aos demais professores do DEs-UFSCar e ICMC-USP, por me transmitirem conhecimento.

Agradeço aos meus amigos pelo convívio, conselhos e apoio nos momentos de dificuldade, dentro e fora da universidade.

Agradeço à banca pelo tempo disponibilizado e pelas críticas construtivas que enriqueceram esta pesquisa.

Agradeço ao programa PIPGEs, pela infra-estrutura e oportunidades oferecidas e ao CNPQ, pelo auxílio financeiro.

“O saber a gente aprende com os mestres e com os livros. A sabedoria a gente aprende com a vida e com os humildes.”

(Cora Coralina)

RESUMO

MASSONI, G. **Análise de textos por meio de processos estocásticos na representação word2vec**. 2021. 59 p. Dissertação (Mestrado em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Dentro do campo de Processamento de Linguagem Natural (NLP), o modelo *word2vec* vêm sendo bastante explorado no campo da representação vetorial de palavras. Ele é uma rede neural que se baseia na hipótese de que palavras semelhantes tem contextos semelhantes. Na literatura em geral, o texto é representado pelo vetor de médias das representações das suas palavras, que, por sua vez, é utilizado como variável explicativa em modelos preditivos. Um alternativa é, além da médias, utilizar outras medidas, como desvio-padrão e medidas de posição. Porém, o uso destas medidas supõe que a ordem das palavras não importa. Assim, nesta dissertação exploramos o uso de processos estocásticos, em particular, Modelos de Série Temporal e Modelos Ocultos de Markov (HMM), para incorporar a ordem "cronológica" das palavras na construção das variáveis explicativas a partir da representação vetorial dada pelo *word2vec*. O impacto desta abordagem é medido com a qualidade dos modelos preditivos aplicados à dados reais e comparado às abordagens usuais. Para os dados analisados, as abordagens propostas tiveram um resultado superior ou equivalente às abordagens usuais na maioria dos casos.

Palavras-chave: representação vetorial de palavras, modelos de predição, processamento de linguagem natural, processos estocásticos..

ABSTRACT

MASSONI, G. **Text mining with stochastic process in word2vec representation**. 2021. 59 p. Dissertação (Mestrado em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Within the field of Natural Language Processing (NLP), the *word2vec* model has been extensively explored in the field of vector representation of words. It is a neural network that is based on the hypothesis that similar words have similar contexts. In the literature in general, the text is represented by the mean vector of the representations of its words, which, in turn, is used as an explanatory variable in predictive models. An alternative is, in addition to averages, to use other measures, such as standard deviation and position measures. However, the use of these measures assumes the order of the words does not matter. Thus, in this dissertation we explore the use of stochastic processes, in particular, Time Series Models and Hidden Markov Models (HMM), to incorporate the "chronological" order of words in the construction of explanatory variables from the vector representation given by *word2vec*. The impact of this approach is measured with the quality of the predictive models of real data and compared to the usual ones. For the analysed data, the proposed approaches have a result that is superior to or equivalent to the usual approaches in most cases.

Keywords: word vector representation, prediction models, natural language processing, stochastic process..

LISTA DE ILUSTRAÇÕES

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

O Processamento de Linguagem Natural (NLP, do inglês *Natural Language Processing*) é um campo de estudos que permite que os computares "entendam" ou reproduzam o comportamento da linguagem natural (humana) para executar alguma tarefa. O primeiro passo comum as tarefas de NLP, e possivelmente mais importante, é encontrar maneiras de representar as palavras de forma eficiente, para que possam servir de entrada para qualquer modelo de *machine learning*. O desafio é transformar uma observação do conjunto de dados, que está no formato de texto, em valores numéricos.

A seguir, temos algumas aplicações da análise de textos:

- **Detecção de spam (FRIEDMAN; HASTIE; TIBSHIRANI, 2001):** tem o objetivo de detectar se um e-mail é spam ou não com base no seu texto;
- **Sentença de processos:** tem o objetivo de prever qual será o resultado de um processo judicial com base na petição inicial[1];
- **Avaliação de produtos:** tem o objetivo de prever a nota dada ao produto pelo consumidor, com base no comentário da avaliação;
- **Classificação de notícias:** tem o objetivo de classificar uma notícia segundo seu tema, com base no seu próprio texto;

Para analisar um texto, consideramos que cada texto, t, é uma sequência de palavras $X_{t,1}, X_{t,2}, \ldots, X_{t,n(t)}$, em que $n(t)$ é o número de palavras deste texto. Intuitivamente, poderíamos considerar que cada palavra, X_{ti}, é uma covariável categórica que pode ser utilizada para predizer a variável resposta de interesse. Porém, essa abordagem gera pelo menos duas dificuldades: (i) os

[1] A petição inicial consiste no primeiro passo a ser dado por quem deseja acionar o judiciário e garantir o seu direito, feito através de um texto.

textos tem número de palavras diferentes, e portanto, não teriam o mesmo número de covariáveis, e (ii) as palavras assumem valores categóricos em um dicionário D, que pode ser grande.

Em geral, a metodologia utilizada para transformar dados textuais em variáveis quantitativas, driblando os problemas citados, é denominada por *bag-of-words* (HARRIS, 1954), que consiste basicamente em identificar o aparecimento de cada palavra no texto. Trata-se essencialmente de codificar o aparecimento de todas as palavras em cada observação. Ou seja, esse método utiliza como covariáveis palavras que aparecem no texto. Essas covariáveis são binárias, e assumem valor um, se aquela palavra apareceu no texto de determinada observação, ou assumem o valor zero, caso contrário.

Como exemplo, considere as duas seguintes sentenças:

Texto 1: "Ser ou não ser, eis a questão";

Texto 2: "Quanto mais inteligente é o ser, mais ele sofre".

A representação *bag-of-words* seria como na Tabela 1.

Tabela 1 – Representação *bag-of-words* dos textos.

	ser	**ou**	**não**	**eis**	**questão**	**quanto**	**mais**	**inteligente**	**sofre**	**ele**
Texto 1	1	1	1	1	1	0	0	0	0	0
Texto 2	1	0	0	0	0	1	1	1	1	1

Existem variações de como utilizar o *bag-of-words* de forma mais eficiente, como considerar a frequência em que cada palavra aparece no texto. Outra possibilidade é, além dos unigramas, pode-se considerar n-gramas (sequência de n palavras) na análise. No texto 2, por exemplo, para n = 2, teríamos o n-grama "mais_inteligente" e "ele_sofre", entre outros.

Entretanto, o *bag-of-words* possui limitações, como não conseguir capturar relações semânticas entre as palavras ou ainda não capturar a inversão de sentido causado por palavras como "não". Além disso, a ordem em que as palavras aparecem no texto, não afeta sua representação. Dessa forma, algumas metodologias de **representação vetorial de palavras**, que não possuem esse tipo de limitação, vem sendo desenvolvidas.

O *word2vec* (MIKOLOV *et al.*, 2013) é um procedimento baseado em redes neurais artificiais, que surgiu como alternativa para a captura de relações semânticas entre as palavras. Com este método, cada palavra do texto passa a ser representada por um vetor numérico, ou seja, um texto passa a ser representado por uma matriz de valores. Então, para aplicar modelos estatísticos ao problema, é necessário resumir a informação de forma que cada observação (texto) possa ser representado por um vetor de variáveis.

Usualmente, o que se faz é tomar a média dos vetores que representam as palavras. Entretanto, por traz disso existe a suposição de que a ordem das palavras não influencia no significado do texto, o que não é verdade. Desta forma, neste trabalho, buscamos por métodos que capturem a ordem "cronológica" das palavras em um texto e consiga sumarizar a representação matricial sem supor a independência entre as palavras. Em particular, trabalharemos com Séries Temporais e com Modelos Ocultos de Markov aplicados à representação vetorial dada pelo *word2vec*. Nestes métodos, consideramos que a palavra atual depende da palavra anterior, ou seja, englobamos a dependência "cronológica". Assim, o objetivo desta pesquisa é comparar os métodos de sumarização da representação vetorial dada pelo *word2vec* que consideram a dependência entre as palavras com aqueles que não a consideram.

É sabido que existem métodos computacionais envolvendo redes neurais complexas que tem um ótimo desempenho para problemas com bancos de dados com milhões de observações. Esta pesquisa está voltada para banco de dados de tamanhos menores, onde os métodos computacionais complexos não possuem um bom poder preditivo.

Neste texto, no Capítulo 2 é discutido com maiores detalhes a metodologia *word2vec* para análise de textos. Nos Capítulos 3 e 4 são apresentados os modelos de séries temporais e modelos que utilizam Cadeias de Markov (*Hidden Markov Models*), respectivamente, bem como sua utilização em análise de textos. Em seguida, no Capítulo 5 são apresentados alguns métodos de predição utilizados nesta pequisa. Por fim, no Capítulo 6 serão apresentadas aplicações para os métodos sugeridos.

CAPÍTULO

2

REPRESENTAÇÃO VETORIAL DE PALAVRAS

A representação vetorial de palavras consiste em técnicas que possibilitem a transformação de uma palavra em um vetor numérico. Os vetores de palavras são essenciais para ter bom desempenho nas tarefas de NLP, visto que facilitam o cálculo de similaridade entre as palavras. A forma mais simples de realizar esta tarefa é representar cada palavra do vocabulário [1] por um vetor *one-hot*, ou seja, um vetor de dimensões $|V| \times 1$ com valor 1 na posição correspondente a palavra representada, e zero nas demais. Por exemplo, se trabalharmos com o vocabulário $V = \{\text{gosto, mineração, textos, estatística}\}$, teríamos as representações:

$$\mathbf{v}_{\text{gosto}} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{v}_{\text{mineração}} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{v}_{\text{textos}} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{v}_{\text{estatística}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Note que o *bag-of-words* em sua formulação mais simples consiste em representar um texto pela soma dos vetores *one-hot* que aparecem nele. Esta é uma representação **discreta**, que não possibilita quantificar similaridade entre as palavras, já que todos os vetores são independentes/ortogonais. Outra possibilidade é a representação **semântica distributiva**, na qual o significado da palavra é representado de acordo com o contexto em que ela aparece. Algumas formas de realizar este tipo de representação serão discutidas nesta Seção.

[1] Conjunto de todas as palavras que aparecem nos dados em estudo.

2.1 Word2Vec

A abordagem chamada busca por representações vetoriais que aumentem a probabilidade de observação de pares de palavras e contextos através do uso de redes neurais. O modelo *word2vec* (MIKOLOV *et al.*, 2013) é uma rede neural que se baseia na importante hipótese linguística de que palavras semelhantes têm contexto semelhante, chamada de similaridade distributiva. Como o *word2vec* utiliza redes neurais, o poder preditivo é substancialmente aumentado quando comparado ao *bag-of-words*.

Definiremos como **contexto** (c) de ordem **m** de uma palavra as $2m$ palavras mais próximas à ela. A Figura 1 mostra exemplos de contextos de ordens 1 para a sentença *"Eu gosto de mineração de textos"*. À direita, apresentam-se os pares de palavra e contexto, de acordo com a palavra central.

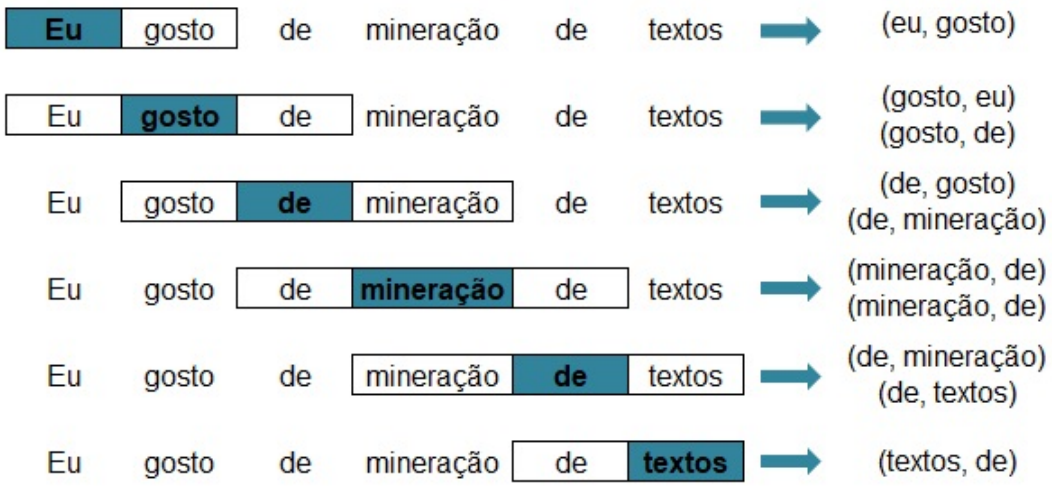

Figura 1 – Exemplos de pares de palavras e contextos de ordem 1.

Existem dois tipos de arquitetura para esta rede neural (representados na Figura 2):

- ***Continuous bag-of-words* (CBOW):** utiliza os contextos (entrada) para prever a palavra central (saída).
- ***Skip-gram* (SG):** utiliza como entrada de uma rede neural a palavra central, para prever os contextos (saída);

Neste trabalho, detalharemos a metodologia por trás da arquitetura SG.

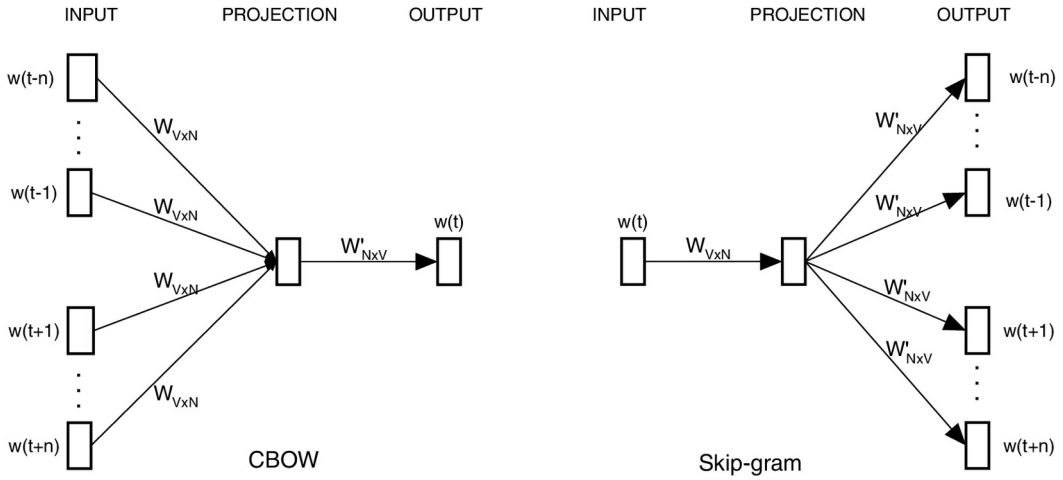

Figura 2 – Representação gráfica do CBOW e do SG (Fonte: (MIKOLOV *et al.*, 2013)).

A ideia do *word2vec* é passar por cada posição no texto, identificar a palavra central (w) e o contexto (c), e usar a similaridade entre cada par (w, c) para calcular a probabilidade da ocorrência de c dado w, ou vice e versa, para SG e CBOW, respectivamente. Neste trabalho, abordaremos a arquitetura Skip-Gram.

O modelo estima as probabilidades condicionais dos contextos dado uma palavra através da função objetivo (GOLDBERG; LEVY, 2014):

$$\begin{aligned} \arg\max_{\theta} L(\theta) &= \arg\max_{\theta} \prod_{t=1}^{T} \prod_{-m \leq j \leq m; j \neq 0} p(w_{t+j}|w_t, \theta) \\ &= \arg\max_{\theta} \prod_{(w,c) \in \mathbf{D}} p(c|w, \theta), \end{aligned} \tag{2.1}$$

em que $\mathbf{D}$ é o conjunto formado por todos os pares palavra-contexto do dicionário. Para uma palavra central w e um contexto c, a formulação básica da arquitetura SG define:

$$p(c|w, \theta) = \frac{\exp\left[\mathbf{u}(c) \cdot \mathbf{v}(w)\right]}{\sum_{c' \in \mathbf{C}} \exp\left[\mathbf{u}(c') \cdot \mathbf{v}(w)\right]}, \tag{2.2}$$

em que $\mathbf{v}(w), \mathbf{u}(c) \in \mathbb{R}^d$ e são as representações vetoriais de w e c, respectivamente, $\mathbf{C}$ é o conjunto com todos os contextos possíveis, e $\theta = \{\mathbf{u}(c), \mathbf{v}(w) : w \in \mathbf{V}, c \in \mathbf{C}\}$. Assim, θ tem dimensão $d \times |V| \times |C|$. A Equação 2.2 é conhecida como função *softmax*.

Note que cada palavra recebe duas representações vetoriais: uma quando está em posição de contexto, denotado por $\mathbf{u}$, e outra quando é a palavra central, denotado por $\mathbf{v}$. Isso é necessário pois, intuitivamente, a probabilidade de uma palavra ser contexto de si própria, $p(w|w)$ é baixa. Porém, se considerarmos a mesma representação vetorial para as duas situação, teríamos que $\mathbf{v}(w) \cdot \mathbf{v}(w)$ deveria ser baixo, o que é impossível (GOLDBERG; LEVY, 2014).

A Figura 3 mostra com mais detalhes a arquitetura SG em questão. Como entrada da rede neural, temos um vetor vetor *one-hot*. A camada oculta contém d neurônios, definindo a dimensão das representações vetoriais. Por fim, a saída da rede contém as probabilidades associadas a cada palavra pertencente ao conjunto de contextos, ou seja, uma distribuição de probabilidades. Embora a saída da rede neural seja um vetor de probabilidades, temos interesse na representação vetorial calculada na camada oculta.

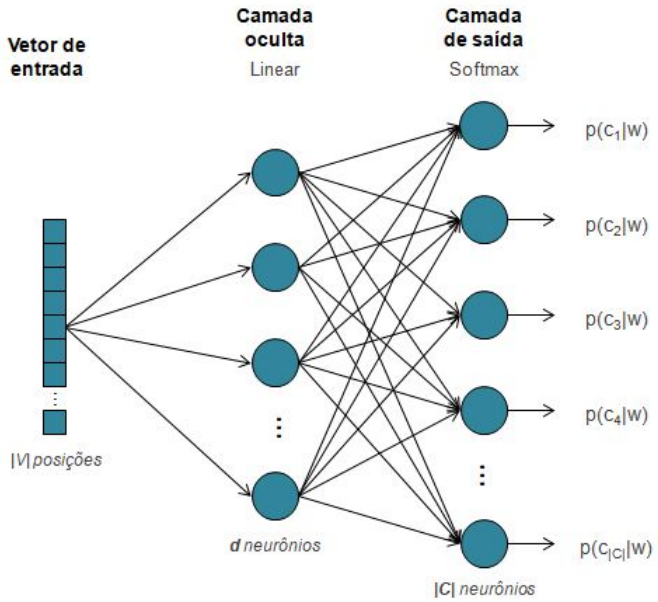

Figura 3 – Arquitetura da rede neural *word2vec* - Skip-Gram.

O funcionamento deste modelo pode ser descrito nos seguintes passos (MANNING *et al.*, 2017):

1. Gerar um vetor $x \in \mathbb{R}^{|V|}$ de zeros e um para a palavra central (*one-hot encoding*);
2. Obter o vetor de representação da palavra central $\mathbf{v}(w) = \mathbb{V}x \in \mathbb{R}^d$, em que $\mathbb{V} \in \mathbb{R}^{d \times |V|}$ é a matriz de palavras, sendo a i-ésima coluna (v_i) a representação vetorial de w_i;
3. Calcular o vetor de escores $\mathbf{z} = \mathbb{U}\mathbf{v}(w)$, onde $\mathbb{U} \in \mathbb{R}^{|C| \times d}$ é a matriz de contextos, sendo a j-ésima coluna (u_j) a representação vetorial do contexto c_j (camada oculta);
4. Transformar o vetor de escores em probabilidades fazendo $\mathbf{y} = softmax(\mathbf{z}) \in \mathbb{R}^{|C|}$ (camada de saída).

A finalidade do modelo é aprender as matrizes $\mathbb{V}$ e $\mathbb{U}$ visando maximizar a função objetivo já discutida (Equação 2.1). Com isto, pode-se calcular os gradientes com respeito aos parâmetros desconhecidos e atualiza-los utilizando Gradiente Descendente Estocástico (BOTTOU, 2010).

Note que o *word2vec* é motivado a gerar representações vetoriais semelhantes para palavras que tem um contexto semelhante. Esperamos, por exemplo que os sinônimos "competente" e "eficiente" tenham contextos parecidos, e portanto, representações próximas.

Cada palavra, por fim, é representada por um vetor de tamanho d. Assim, cada texto é representado por uma matriz em que cada linha é composta pelo vetor associado a cada palavra deste. Considerando o texto t, temos:

$$\mathbb{W}^t = \begin{bmatrix} \mathbf{v}(w_1^t) \\ \vdots \\ \mathbf{v}\left(w_{n(t)}^t\right) \end{bmatrix}_{n(t)\times d}, \tag{2.3}$$

em que $\mathbf{v}(w_i^t)$ é a representação da i-ésima palavra do texto t e $n(t)$ é o total de palavras deste texto. Note que a matriz de representação tem dimensão diferente para textos com número de palavras diferente. Desta forma, para o ajuste de modelos estatísticos, faz-se necessário transformar esta matriz em um vetor linha que tenha a mesma dimensão para todos os textos. Pode-se pensar em métricas a partir dos valores para alcançar este objetivo:

$$T(\mathbb{W}^t, g) = \left[g(W_{\cdot,1}^t), \ldots, g(W_{\cdot,d}^t)\right]_{1\times d}, \tag{2.4}$$

em que $g : \mathbb{R}^{n(t)} \to \mathbb{R}$ e $W_{\cdot,j}^t$ é a j-ésima coluna de $\mathbb{W}^t$. Note que $T(\mathbb{W}^t, g)$ será o vetor de covariáveis utilizadas como entrada nos modelos de predição (discutidos no Capítulo 5).

Em geral, usa-se a média como função de transformação $g(.)$. Além da média, pode-se utilizar outras medidas que sumarizam a informação, como o desvio-padrão, curtose e todos os quantis (STERN *et al.*, 2020). Formalmente, temos:

$$\mathbf{T}(\mathbb{W}_t, \mathbf{g}) = [T(\mathbb{W}_t, g_1), \ldots, T(\mathbb{W}_t, g_k)]_{1\times(d\cdot p)}, \tag{2.5}$$

em que $\mathbf{g} = (g_1, \ldots, g_p)$ e g_i é a i-ésima medida de sumarização utilizada.

Estes métodos de sumarização consideram que a ordem em que as palavras aparecem no texto **não importa**, já que estas medidas consideram as observações independentes. No entanto, sabemos que a ordem das palavras em um texto determina totalmente o sentido dele. Assim, esta dissertação propõe trabalhar com modelos de processos estocástico, que consideram a dependência existente entre as observações, para sumarizar as informações contidas na representação vetorial das palavras. Em particular, trabalhamos com Modelos Ocultos de Markov, que será apresentado no Capítulo 4, bem como seu uso neste contexto.

2.1.1 Redução de dimensão

Podemos notar que o número de covariáveis dos modelos de predição (dimensão de $T(\mathbb{W}^t, g)$) é diretamente proporcional à dimensão do *word2vec* (d). Além disso, é sabido que valores altos de d trazem melhor ajuste à representação vetorial das palavras. Porém, se considerarmos um valor alto de d, teremos um grade número de covariáveis, podendo dificultar computacionalmente o ajuste dos modelos estatísticos de predição. Desta forma, faz-se necessário o uso de algum método de **redução de dimensão** para as representações vetoriais do *word2vec*.

O t-SNE (*t-Distributed Stochastic Neighbor Embedding*) é uma técnica para redução de dimensionalidade adequada para dados de alta dimensão (MAATEN; HINTON, 2008). Ao contrário da técnica dos Componentes Principais (PCA) que é determinística, o t-SNE é um método probabilístico. Em resumo, o primeiro passo desta técnica é construir uma distribuição de probabilidades sobre os pares de observações em alta dimensão, fazendo com que pontos diferentes tenham probabilidade baixa e pontos próximos tenham probabilidades altas. Depois, de forma semelhante calcula-se a distribuição de probabilidades sobre os pontos em baixa dimensão. Por fim, minimiza-se a distância de Kullback-Leibler entre estas distribuições de probabilidade. Esta minimização é feita utilizando *gradient descent*.

O PCA é um método de redução de dimensão linear, e portanto não é capaz de captar relações complexas entre as dimensões. É um método que se concentra em preservar a estrutura global dos dados, ajustando todos os grupos como um todo. Já o t-SNE é baseado em distribuições de probabilidade que possibilitam encontrar tais estruturas complexas e é baseado em buscar **localmente** os pontos próximos na representação de baixa dimensão. Assim, ao reduzir a dimensão utilizando o t-SNE existe menor perda de informação em relação aos dados originais.

Optamos por ajustar o modelo *word2vec* com dimensão 100 e posteriormente utilizar o método t-SNE, reduzindo para dimensão 10 ao invés de ajustar diretamente o *word2vec* com dimensão 10 pois concluímos que há um ganho preditivo. Naturalmente, utilizar o *word2vec* com d=100 é a opção que tem maior poder preditivo, porém oferece dificuldades computacionais, enquanto que utilizá-lo com d=10 é a opção que oferece menor poder preditivo. Optamos então pelo equilíbrio entre poder preditivo e viabilidade computacional.

Nesta pesquisa, quando necessário, utilizamos o t-SNE para reduzir a dimensão dos dados de saída do *word2vec*.

CAPÍTULO

3

SÉRIES TEMPORAIS

Uma série temporal é qualquer evento que pode ser ordenado pelo tempo, como a temperatura na cidade de São Paulo ao longo dos dias, índices diários da inflação de um país, entre outros (MORETTIN; TOLOI, 2006). Neste trabalho, consideraremos que a representação vetorial de cada texto segue um modelo de série temporal.

Formalmente, uma série temporal é um conjunto de observações $\{W_i\}$ em que cada observação é realizada em um tempo específico, i. Neste trabalho, o tempo é a ordem cronológica em que as palavras aparecem no texto, ou seja, $i \in \{1,\ldots,n(t)\}$, em que $n(t)$ é o número de palavras do text t. Os modelos de séries temporais consideram que o conjunto de observações de interesse é um processo estocástico (PINSKY; KARLIN, 2010).

Aqui, trabalharemos com dois modelos de séries temporais: (i) Autorregressivos, para análises univariadas e (ii) Vetores Autorregressivos, para análises multivariadas.

3.1 Modelos Autorregrassivos (AR)

Um modelo Autorregressivo de ordem p - AR(p) - é dado por:

$$W_i = \phi_0 + \phi_1 W_{i-1} + \ldots + \phi_p W_{i-p} + \varepsilon_i, \tag{3.1}$$

em que ε_i é um ruído, com $\mathbb{E}(\varepsilon_i) = 0, \forall i \in \{1,...,n\}$, $E(\varepsilon_i^2) = \sigma^2$ e $E(\varepsilon_i \varepsilon_s) = 0$. Note que a observação no tempo presente (i) depende das p-ésimas observações anteriores.

Para facilitar a notação, retiramos o índice t, que representa o texto, mas todos os procedimentos descritos a seguir devem ser repetidos para todos os textos da base de dados. Como o modelo AR é um modelo univariado, para aplicá-lo a representação vetorial das palavras do texto, consideraremos que as dimensões do vetor dados pelo *word2vec* são independentes e

cada uma delas será modelada por um AR(p) independente. Na Equação 3.1, a variável W_i indica a i-ésima palavra de um texto fixo, dado uma posição do vetor de representação. Assim, para cada texto teremos $p \times d$ parâmetros a serem estimados. Assim, temos os seguintes parâmetros:

$$\phi_0 = (\phi_{01}, ..., \phi_{0d}), \quad \mathbb{F} = \begin{pmatrix} \phi_{1,1} & \dots & \phi_{1,p} \\ \vdots & \ddots & \vdots \\ \phi_{d,1} & \dots & \phi_{d,p} \end{pmatrix} \text{e} \quad \Sigma = \begin{pmatrix} \sigma_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_d^2 \end{pmatrix}. \tag{3.2}$$

Para o ajuste de modelos de *machine learning*, consideraremos o vetor de covariáveis dado na Equação 2.5 sendo:

$$T(\mathbb{W}, g) = [\phi_0, \mathbb{F}, \Sigma]. \tag{3.3}$$

3.2 Modelos Vetores Autorregrassivos (VAR)

Enquanto os modelos AR(p) são univariados, os modelos Vetores Autorregressvos de ordem p - VAR(p) - são ditos multivariados. No contexto da representação vetorial de palavras isso significa dizer que as dimensões dos vetores dados pelo *word2vec* **não** são independentes. Formalmente, um modelo VAR(p) é dado por:

$$\mathbf{W}_i = \mathbf{A}_0 + \mathbb{A}_1 \mathbf{W}_{i-1} + \ldots + \mathbb{A}_p \mathbf{W}_{i-p} + \varepsilon_i, \tag{3.4}$$

em que $\varepsilon_{i_{d\times 1}}$ é um vetor de ruídos com $E(\varepsilon_i) = 0$ e $Cov(\varepsilon_i) = \Sigma$, $\mathbf{A}_{0_{d\times 1}}$ é um vetor de interceptos e $\mathbb{A}_1, \ldots, \mathbb{A}_p$ são matrizes de coeficientes de dimensão $d \times d$. É importante ressaltar que, ao contrário do modelo AR, existe covariância entre as dimensões, de forma que a matriz de covariâncias Σ não é diagonal. Aqui, $\mathbf{W}_i$ representa o vetor *word2vec* da i-ésima palavra de um texto fixo.

Por fim, para o ajuste de modelos de *machine learning*, consideraremos o vetor de covariáveis dado na Equação 2.5 sendo:

$$T(\mathbb{W}, g) = [\mathbf{A}_0, \mathbb{A}_1, \ldots, \mathbb{A}_p, \Sigma]. \tag{3.5}$$

Note que o número de parâmetros do VAR(p) é significativamente maior em comparação ao AR(p). Enquanto o AR(p) tem $2d + d \times d$ parâmetros, o VAR(p) tem $d + (d \times d)(p + 1)$.

CAPÍTULO

4

MODELOS MARKOVIANOS OCULTOS

O Modelo Markoviano Oculto (HMM, do inglês *Hidden Markov Models*) é composto por duas sequências de variáveis aleatórias, uma observável, e a outra não observável (oculta), e pode ser aplicado em muitas áreas de pesquisa, como reconhecimento de fala e sequenciamento genético (RABINER; JUANG, 1986). Usualmente, considera-se que a sequência não observável é uma Cadeia de Markov (MC) e a variável observável é condicionalmente independente, dado os estados ocultos, como mostra a Figura 4. Este modelo pode ser visto como um modelo de mistura com uma estrutura de dependência entre as observações.

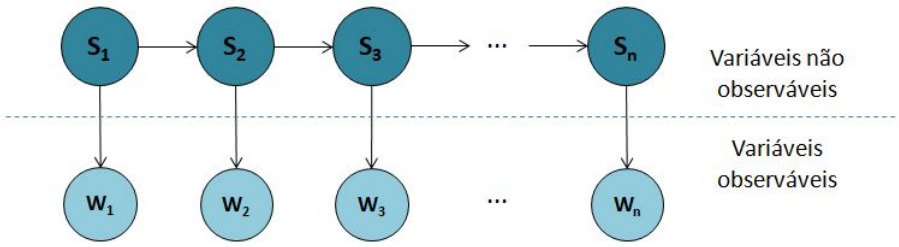

Figura 4 – Estrutura de um Modelo Markoviano Oculto

Neste trabalho, consideraremos que a representação vetorial de cada texto segue um HMM, em que os estados ocultos podem representar o sentimento do texto ou algum conjunto de tópicos latentes não observáveis, e as variável observáveis são a representação vetorial de cada palavra. Novamente, para facilitar a notação, retiraremos o índice t, que representa o texto, mas todos os procedimentos descritos a seguir devem ser repetidos para todos os textos da base de dados.

Considere que a sequência $\mathbf{S} = (S_1, ..., S_n)$ de estados ocultos assume valores $S_i \in \{1, 2, ..., K\}$, para $i = 1, ..., n$ e é uma Cadeia de Markov, ou seja,

$$\mathbb{P}(S_i = k|S_1, S_2, ..., S_{i-2}, S_{i-1}) = \mathbb{P}(S_i = k|S_{i-1}), \text{ para } k = 1, ..., K. \tag{4.1}$$

Um cadeia de Markov é caracterizada pelo vetor de probabilidades iniciais, $\mathbf{p}_0 = (p_{01}, ..., p_{0K})$, com $p_{0j} = \mathbb{P}(S_1 = j)$, e pela matriz de transição dos estados, $\mathbb{A} = \{p_{jk}\}$, em que $p_{jk} = \mathbb{P}(S_{i+1} = k|S_i = j, \mathbb{A})$, com $p_{jk} \geq 0$, para $j, k = 1, ..., K$, e $\sum_{k=1}^{K} p_{jk} = 1$.

Considere também que $\mathbb{W} = (\mathbf{W}_1, ..., \mathbf{W}_n)$ é a sequência de palavras do texto, representada por seus respectivos vetores, ou seja, $\mathbf{W_i} = \mathbf{v}(w_i)$, com $\mathbf{v}(w_i)$ definido na equação 2.3. Definimos que a distribuição de $\mathbf{W}_i$ é condicional ao resultado do estado oculto S_i, como é observado na Figura 4. A função de distribuição $F(.)$ deve ser multivariada de dimensão d, já que esta é a dimensão da representação vetorial dada pelo *word2vec*.

Em resumo, definimos o modelo HMM pelos elementos:

1. $S_i \sim MC(\mathbb{A})$, e
2. $\mathbf{W}_i|S_i = k \sim F_{\mathbf{W}_i|S_i}(w_i|k)$, para $i = 1, ..., n$.

Neste estudo, assumiremos que $W_i|S_i = k \sim N_d(\mu_k, \Sigma_k)$.

A estrutura de dependência descrita nos permite observar as seguintes relações:

$$\begin{aligned} S_i &\perp \{S_1, W_1, ..., S_{i-2}, W_{i-2}, W_{i-1}\}|S_{i-1}, \text{ para } i = 2, ..., n, \\ W_i &\perp \{S_1, W_1, ..., S_{i-1}, W_{i-1}\}|S_i, \text{ para } i = 2, ..., n. \end{aligned} \tag{4.2}$$

Desta forma, estamos considerando que a representação vetorial da i-ésima palavra do texto t depende das palavras anteriores através do i-ésimo estado oculto, que por sua vez depende apenas do estado oculto imediatamente anterior. Esta relação de dependência pode facilmente ser observada na Figura 4 e garante que a ordem das palavras está sendo considerada na sumarização da representação vetorial dada pelo modelo *word2vec*. Com esta suposição, pretende-se melhorar o poder preditivo de modelos de *machine learning*, o que será analisado no Capítulo 6.

Para um número fixo de estados K e um texto t, teremos interesse nas seguintes medidas do HMM:

1. $\pi = (\pi_1, ..., \pi_K)$: distribuição estacionária da Cadeia Oculta de Markov, tal que $\pi = \pi\mathbb{A}$;
2. $\mu_1, ..., \mu_K$: média da variável observável associada a cada estado oculto;
3. $\Sigma_1, ..., \Sigma_K$: matriz de variâncias da variável observável associada a cada estado oculto;

Considerando $\theta_i = (\mu_i, \Sigma_i), i = 1, ..., n,\ \theta = (\theta_1, ..., \theta_n)$ e a estrutura de dependência descrita na Equação 4.2, a função de verossimilhança do HMM é dada por:

$$\begin{aligned} L(\theta, \mathbf{p}_0, \mathbb{A}|\mathbf{y}, \mathbf{s}) &= \mathbb{P}\left[\mathbf{W}_1 = \mathbf{w}_1, ..., \mathbf{W}_n = \mathbf{w}_n, S_1 = s_1, ..., S_n = s_n | \theta, \mathbf{p}_0, \mathbb{A}\right] \\ &= \prod_{i=1}^{n} \mathbb{P}(\mathbf{W}_i = \mathbf{w}_i | \mathbf{S}_i = s_i, \theta_i) \mathbb{P}(\mathbf{S}_i = s_i | \mathbf{S}_{i-1} = s_{i-1}, \mathbf{p}_0, \mathbb{A}) \\ &= \left[\prod_{i=1}^{n} f_{\mathbf{W}_i|S_i}(w_i|s_i, \theta_i)\right] \left[\prod_{i=1}^{n} p_{s_{i-1}, s_i}\right] p_{0s_1}. \end{aligned} \tag{4.3}$$

Assim, para o ajuste de modelos de *machine learning*, consideraremos o vetor de covariáveis dado na Equação 2.5 sendo:

$$T(\mathbb{W}, g) = \left[\mu_1^1, ..., \mu_1^d, ..., \mu_K^1, ..., \mu_K^d, \sigma_1^{11}, ..., \sigma_1^{dd}, ..., \sigma_K^{11}, ..., \sigma_K^{dd}, \pi_1, ..., \pi_K\right]. \tag{4.4}$$

Como as matrizes Σ_k são simétricas, utilizamos apenas os elementos da diagonal superior não nulos. Se considerarmos que as dimensões do *word2vec* são **não correlacionadas**, utilizamos apenas a diagonal (variância de cada dimensão) para cada estado oculto.

Alguns métodos inferenciais para o HMM estudados por (BAUM; PETRIE, 1966) permitem a estimação de máxima verossimilhança dos parâmetros na Equação 4.4, obtidas usando o algoritmo EM, quando o número de estados, K, é conhecido (algoritmo Baum-Welch).

4.1 Não identificabilidade

Um problema comum aos modelos de mistura é a não identificabilidade dos rótulos dos estados ocultos. Isso ocorre devido a permutabilidade dos modelos (STEPHENS, 2000). Seja τ_K o conjunto de permutações dos índices $\{1, \ldots, K\}$ e alguma permutação $\tau = (i_1, ..., i_K) \in \tau_K$ consideramos o vetor de parâmetros $\tau(\theta, \mathbf{p}_0, \mathbb{A}) = (\mu_{i_1}, ..., \mu_{i_K}, \Sigma_{i_1}, ..., \Sigma_{i_K}, p_{0i_1}, p_{0i_K}, \tau(\mathbb{A}))$, em que

$$\tau(\mathbb{A}) = \begin{pmatrix} p_{i_1 i_1} & p_{i_1 i_2} & \cdots & p_{i_1 i_K} \\ p_{i_2 i_1} & p_{i_2 i_2} & \cdots & p_{i_2 i_K} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i_K i_1} & p_{i_K i_2} & \cdots & p_{i_K i_K} \end{pmatrix}.$$

Note que a função de verossimilhança dada na Equação 4.3 é invariante à permutação dos componentes dos estados ocultos do modelo. Em outras palavras, temos que $L(\theta, \mathbf{p}_0, \mathbb{A}|\mathbf{y}, \mathbf{s}) = L(\tau(\theta, \mathbf{p}_0, \mathbb{A})|\mathbf{y}, \mathbf{s})$. Este fenômeno é conhecido como não identificabilidade do HMM, ou *label switching*.

No estudo em questão, suponha que existam dois estados: um que indica que as palavras tem uma conotação positivas e outro indicando que as palavras tem uma conotação negativa. O problema de não identificabilidade do HMM faz com que, para um texto, as palavras com conotação positiva sejam provenientes de um estado rotulado como 1, enquanto que para outro texto, o estado rotulado como 1, na verdade indique palavras com conotação negativa.

Em geral, este problema é resolvido trabalhando-se com restrições sobre os parâmetros, ou seja, um subespaço do problema (SPEZIA, 2009). Em HMMs Gaussianos, a restrição pode ser feita, por exemplo, acerca das médias, assumindo-se que $\mu_1 < \mu_2 < ... < \mu_K$, ou ainda, sobre as variâncias, assumindo-se que $\sigma_1 < \sigma_2 < ... < \sigma_K$. Fazendo alguma destas restrições, os estados passam a ser completamente identificáveis. Porém, neste estudo, estamos trabalhando com distribuições multivariadas, já que cada palavra é representada por um vetor, tornando inviável alguma restrição neste sentido. Desta forma, trabalhamos com duas abordagens que serão discutidas a seguir.

Distância euclidiana

Considere um texto t_1 fixado. Para este texto, ajustamos o HMM Gaussiano multivariado (discutido acima) com K estados e concatenamos os vetores de médias estimados, ou seja,

$$(\hat{\mu}_{11}, ..., \hat{\mu}_{1d}, \hat{\mu}_{21}, ..., \hat{\mu}_{2d}, ..., \hat{\mu}_{K1}, ..., \hat{\mu}_{Kd}).$$

Depois, considere um novo texto t_i. Para este texto, consideramos as $K!$ permutações diferentes nos rótulos dos estados ocultos e, para cada permutação, concatenamos o vetor de médias estimado na ordem sugerida. Por exemplo, se considerarmos $K = 3$ estados ocultos, teríamos as permutações $(1,2,3)$; $(1,3,2)$; $(2,1,3)$; $(2,3,1)$; $(3,1,2)$ e $(3,2,1)$. Para a permutação $(3,1,2)$, em particular, o vetor de médias concatenadas é dado por:

$$(\hat{\mu}_{31}, ..., \hat{\mu}_{3d}, \hat{\mu}_{11}, ..., \hat{\mu}_{1d}, \hat{\mu}_{21}, ..., \hat{\mu}_{2d}).$$

Para cada permutação, calculamos a distância euclidiana entre o respectivo vetor de médias estimadas do texto t_i e o vetor de médias concatenadas do texto t_1. A permutação que apresentou menor distância foi escolhida como a ordem correta dos estados e os estados foram renomeados de acordo com ela. Este processo foi repetido para todos os textos $t_i, i = 2, ..., T$.

Análise de cluster

Métodos de agrupamentos (cluster) tem a finalidade de dividir as observações da base de dados em grupos, de forma que, dentro de cada grupo, as observações sejam próximas, mas os grupos sejam diferentes. Neste estudo, utilizamos a análise de cluster para dividir os textos

em $K!$ grupos. Para cada grupo formado, os estados ocultos foram rotulados conforme uma permutação.

Formalmente, buscamos uma partição $C_1, ..., C_{K!}$ dos textos $\{1, ..., T\}$, ou seja,

$$C_1 \cup ... \cup C_{K!} = \{1, ..., T\} \text{ e } C_i \cap C_j = \phi, \forall i \neq j.$$

Em particular, utilizamos o método K-Médias (MACQUEEN *et al.*, 1967), em que fixamos previamente o número de grupos, equivalente ao número de permutações possíveis dos rótulos dos estados K ocultos do HMM, dado por $K!$. Seja $d^2(\mathbf{x}_i, \mathbf{x}_j)$ e distância euclidiana entre os vetores $\mathbf{x}_i$ e $\mathbf{x}_j$, neste método, busca-se pela partição na qual

$$\sum_{k=1}^{K!} \frac{1}{K!} \sum_{i,j \in C_k} d^2(\hat{\mu}_{t_i}, \hat{\mu}_{t_j})$$

seja o menor possível. Lembre que $\hat{\mu}_{t_i}$ é o vetor de médias concatenadas do texto t_i.

Suponha que temos $K = 3$ estados ocultos. Assim, os textos pertencentes ao grupo 1 foram rotulados segundo a permutação $(1,2,3)$, já os pertencentes ao grupo 2, foram rotulados segundo a permutação $(1,3,2)$, e assim sucessivamente, até os textos do grupo 6, que foram rotulados segundo a permutação $(3,2,1)$.

CAPÍTULO

5

MODELOS DE PREDIÇÃO

Após a representação vetorial dos textos ser obtida o problema da análise de texto se resume à um problema de predição comum. Assim, a representação vetorial se torna as covariáveis que serão utilizadas para a predição de uma característica de interesse.

Neste Capítulo apresentamos os modelos de predição utilizados nesta pesquisa. Formalmente, seja $(\mathbf{X}_1, Y_1), ..., (\mathbf{X}_T, Y_T)$ os dados observados, em que $\mathbf{X}_i = \mathbf{T}(\mathbb{W}_i, \mathbf{g})$, dado na Equação 2.5 é o vetor de parâmetros estimados com base na representação vetorial das palavras do texto t_i, e Y_{t_i} é a característica de interesse deste texto. Assim, $\mathbf{X}_i$ será o vetor de variáveis de entrada do modelo de predição, enquanto que Y_i é a saída. Assim, queremos encontra uma função $g(\mathbf{x})$ que se aproxime dos valores de $\mathbf{y}$. Esta função (preditor linear) é dada por:

$$\eta_i := g(\mathbf{x}_i) = \beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}, \tag{5.1}$$

em que p é o número de variáveis de entrada.

5.1 LASSO

O *least absolute shrinkage and selection operator*, LASSO (TIBSHIRANI, 1996), é um método paramétrico para estimação de modelos de regressão linear que possua risco menor do que a estimação por mínimos quadrados. Consiste em buscar valores de β tais que:

$$\beta = \arg\min_{\beta} \left[\sum_{i=1}^{T} l(y_i, \eta_i) - \lambda \sum_{j=1}^{p} |\beta_j| \right], \tag{5.2}$$

em que $l(y_i, \eta_i)$ é o negativo do logaritmo da função de verossimilhança para a i-ésima observação.

Note que, se $\lambda = 0$, o método LASSO é equivalente ao método de mínimos quadrados. Por outro lado, se λ é muito grandes, todos os parâmetros serão estimados como nulo.Desta forma, este método, além de proporcionar a estimação dos parâmetros da função $g(\mathbf{x})$, também possibilita a seleção de variáveis. O parâmetro λ é encontrado via validação cruzada.

Em particular, quando a variável de interesse (Y) é binária, ou seja, $y \in \{0,1\}$, ajustamos a **regressão logística**. Neste caso, consideramos que $Y_i \sim Bernoulli(\theta_i)$, em que $\theta_i = \mathbb{P}(Y_i = 1)$, e que:

$$\log\left(\frac{\theta_i}{1-\theta_i}\right) = \eta_i. \tag{5.3}$$

Note que a restrição colocada na Equação 5.3 é necessária para garantir que o valor de θ_i (que é uma probabilidade) estará no intervalo $(0,1)$. Com isso, temos que o negativo do logaritmo da função de verossimilhança é dada por:

$$l(y_i, \eta_i) = -\left\{y_i\eta_i - \log\left[1+\exp(\eta_i)\right]\right\}. \tag{5.4}$$

Outro caso particular ocorre quando a característica de interesse é uma contagem, como o número de eventos que acontecem em um espaço de tempo. Neste caso, uma solução é considerar que $Y_i \sim Poisson(\theta_i)$, com $Y_i \in \{0,1,2,...\}$ e $\theta_i > 0$, e considere que:

$$\log(\theta_i) = \eta_i. \tag{5.5}$$

Neste caso, a restrição da Equação acima é necessária para garantir que θ_i será positivo. Com isso, temos que o negativo do logaritmo da função de verossimilhança é dado por:

$$l(y_i, \eta_i) = -\left\{y_i\eta_i - \exp(\eta_i) + \log\left(\frac{1}{y_i!}\right)\right\}. \tag{5.6}$$

Após encontrar o negativo do logaritmo da função de verossimilhança considerando as particularidades da característica de interesse estudada, basta substituí-lo na Equação 5.2 e encontrar as estimativas dos parâmetros do preditor linear.

5.2 Árvores e Florestas aleatórias

Ás árvores de decisão são métodos não paramétricos de predição. Formalmente, as árvores particionam o espaço das covariáveis nas regiões disjuntas $R_1, ..., R_j$. Em um problema

de regressão, a predição em cada região é dada pela média dos valores de y pertencentes à ela, ou seja, para a região R_k, temos que:

$$g(\mathbf{x}) = \frac{1}{|\{i : \mathbf{x}_i \in R_k\}|} \sum_{i:\mathbf{x}_i \in R_k} y_i. \tag{5.7}$$

A árvore deve ser construída buscando a partição mais pura, ou seja, que os $Y's$ sejam homogêneos dentro de cada região, e heterogênico comparado aos valores de outras regiões. Isso é realizado através de uma heurística:

1. Inicialmente, particiona-se o espaço nas regiões R_1 e R_2. Isto é feito buscando entre todas as covariáveis x_i e todos os cortes t_i, àqueles que levam ao menor erro quadrático, dado por:

$$\sum_{\{i:\mathbf{x}_i \in R_1\}} (y_i - \hat{y}_{R_1})^2 + \sum_{\{i:\mathbf{x}_i \in R_2\}} (y_i - \hat{y}_{R_2})^2, \tag{5.8}$$

   em que $\hat{y}_{R_k}$ é a predição dada na região R_k;

2. Após escolhida a primeira partição, esta é fixada e busca-se particionar a região R_1 ou a R_2 da mesma forma que no passo anterior, escolhendo também qual região será particionada;

3. Repete-se este procedimento até que obter uma partição com poucas observações em cada região.

Este procedimento produz árvores "profundas" com super-ajuste aos dados, ou seja, alta variância. Assim, faz-se a "poda", adicionando um pouco de viés para diminuir a variância. A poda retira um nó por vez da árvore, analisando o comportamento do erro de predição no conjunto de validação. Com base nesse comportamento, decide-se quando parar a poda.

Para problemas de classificação, ao invés da média, calcula-se a moda de y em cada região para se fazer a predição, ou seja:

$$g(\mathbf{x}) = moda\{y_i : y_i \in R_k\}. \tag{5.9}$$

Além disso, no processo de particionamento, ao invés do erro quadrático, calcula-se o índice de Gini. Suponha uma etapa com j regiões e que $Y \in \{s_1, s_2, ..., s_c\}$, então o índice Gini é dado por:

$$\sum_{k=1}^{j}\sum_{i=1}^{c}\hat{p}_{R_k,s_i}(1-\hat{p}_{R_k,s_i}), \tag{5.10}$$

em que $\hat{p}_{R_k,s_i}$ é a proporção de observações da região R_k que pertencem à categoria s_i.

Apesar de serem extremamente interpretáveis, as árvores tem resultados preditivos ruins quando comparado aos demais estimadores, como o LASSO. Por isso, a fim de melhorar o poder preditivo, surgiram formas de combinar predições de árvores, como o *bagging*, *boosting* e as florestas aleatórias (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

As florestas aleatórias, em particular, é uma combinação de B árvores de decisão. Cada uma destas árvores são ajustadas com m covariáveis, em que m é menor que o número total de covariáveis. Com isso, busca-se construir árvores não correlacionadas. Além disso, são criadas amostras *bootstrap* diferentes para o ajuste de cada uma das árvores e não é feita a poda, para que cada ávore seja não viesada. Por fim, a função de predição da floresta é dada por:

$$g(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} g_b(\mathbf{x}), \tag{5.11}$$

em que $g_b(\mathbf{x})$ é a predição dada pela b-ésima árvore não viesada e não correlacionada. Pode-se provar que a floresta produz um preditor com menor risco do que as árvores individuais.

5.3 Extreme Gradient Boosting

Como já dito, árvores de decisão são um método de predição conhecido pela sua fácil interpretação, porém tem resultados preditivos ruins quando comparado aos demais estimadores. Em particular, o *gradient boosting* é uma técnica na qual o estimador $g(\mathbf{x})$ combina estimadores "fracos" de forma incremental, assim como qualquer método de *boosting*, até se obter um estimador com bons resultados preditivos. Seja $(x_1,y_1),...,(x_n,y_n)$ o conjunto de dados de treinamento, $L(y_i,g(\mathbf{x}_i))$ uma função de perda e M o número de interações, então o *gradient boosting* é dado pelo algoritmo:

1. Inicialize o modelo com o valor:

$$g_0(\mathbf{x}) = \arg\min_{g}\sum_{i=1}^{n} L(y_i,g(\mathbf{x}_i)).$$

2. Para $m = 1,...,M$:

a) Calcule os pseudo-resíduos:

$$r_{im} = \left[\frac{\partial L(y_i, g(\mathbf{x}_i))}{\partial g(\mathbf{x}_i)}\right]_{g(\mathbf{x}_i)=g_{m-1}(\mathbf{x}_i)}, \quad i = 1, ..., n.$$

b) Ajuste um modelo fraco (como uma árvore rasa), $h_m(\mathbf{x})$ considerando os dados de treino $(x_1, r_{1m}), ..., (x_n, y_{nm})$.

c) Calcule:

$$\lambda_m = \arg\min_{\lambda} \sum_{i=1}^{n} L(y_i, g_{m-1}(\mathbf{x}_i) + \lambda h_m(\mathbf{x}_i)).$$

d) Atualize o modelo: $g_m(\mathbf{x}) = g_{m-1}(\mathbf{x}_i) + \lambda h_m(\mathbf{x}_i)$.

3. O modelo final é dado por $g_M(\mathbf{x})$.

Por fim, o *Extreme Gradient Boosting*, XGB (CHEN; GUESTRIN, 2016) é uma aprimoramento do *gradient boosting* que lida melhor com as restrições computacionais. Detalhes deste método de estimação podem ser visto no artigo citado. É importante destacar que se o número de interações M for muito alto, o modelo fica super-ajustado, logo, este é um parâmetro que deve ser escolhido via validação cruzada.

CAPÍTULO

6

APLICAÇÕES

Para aplicar as metodologias abordadas anteriormente em uma base de dados textual, podemos seguir os passos:

1. Limpeza do texto: retirar caracteres especiais, *stopwords* (de, da, do, e, a, o, etc.), deixar todas as letras minúsculas, tratar os números conforme o problema, entre outros;
2. Transformar o texto em vetor:
   a) Caso opte pelo *bag-of-words*, ajuste-o e vá ao passo seguinte;
   b) Caso opte pelo *word2vec*, ajuste-o e:
      i. Calcule a média ou outras medidas simples no resultado do *word2vec*, ou
      ii. Ajuste um modelo de Série Temporal, ou
      iii. Ajuste um HMM e resolva a não identificabilidade com distância euclidiana ou análise de cluster;
3. Ajuste um modelo estatístico de predição.

Neste Capítulo aplicaremos a metodologia para representação vetorial dos textos descrita anteriormente, e posterior ajuste de modelos de regressão e classificação. Para isso, trabalhamos com as seguintes bases de dados:

- **Avaliação de filmes:** predizer o sentimento do telespectador sobre um filme com base na sua avaliação;
- **Processos Judiciais:** predizer a sentença de um processo judicial a partir da petição inicial;
- **Tweets da Copa do Mundo:** predizer o número de “RT’s” a partir do texto do *tweet*.

Todas as bases de dados trabalhadas tem pelo menos 50 mil observações. Além disso, com exceção dos dados referentes aos *tweets*, os textos analisados podem ser grandes, contendo um vocabulário volumoso. Para agilizar o processo de análise destes dados, trabalhamos com computação paralela aplicada em linguagem *python*.

As aplicações foram feitas considerando modelos *word2vec* com arquitetura *skip-gram* de dimensões 100 juntamento com o t-SNE para reduzir a dimensão para 10. Ajustamos modelos considerando (i) as abordagens com HMM de 2, 3, 4 e 5 estados, (ii) as abordagens com Séries Temporais AR(p) e VAR(p) com p = 1, 2 e 3, (iii) apenas as médias e (iv) medidas de ordem, média e desvio-padrão.

As bases de dados foram aleatoriamente divididas em treinamento, validação e teste. A primeira parte foi utilizada para estimação dos modelos, a segunda para ajuste dos *tunning parameters* e a última para cálculo de medidas de qualidade do modelo treinado. Para comparação da metodologia, usamos o **risco**, **área sob a curva ROC** e **F1** como medida de qualidade dos modelos de classificação e apenas o **risco** como medida de qualidade para os modelos de regressão (análise dos *tweets*).

Para modelos de classificação, o risco é dado por:

$$R[g(\mathbf{x})] := \sum_{i=1}^{T} \mathbb{I}(y_i \neq \hat{y}_i). \tag{6.1}$$

Para modelos de regressão, utilizamos o risco absoluto, dado por:

$$R[g(\mathbf{x})] := \sum_{i=1}^{T} |y_i - \hat{y}_i|. \tag{6.2}$$

A área sob a curva ROC (AUC - *area under curve*) é uma medida que sumariza a capacidade do modelo decidir entre duas classes. Quanto maior a AUC, melhor o modelo. Aos comparar curvas ROC de diferentes modelos, àquela que estiver sempre acima das demais representa o melhor modelo. Mais detalhes sobre estas medidas podem ser vistas em (FAN; UPADHYE; WORSTER, 2006).

Por fim, utilizamos também o **F1**. Supondo que $Y \in \{0,1\}$, o $F1$ é dado por:

$$F1 = \frac{2Precision \times Recall}{Precision + Recall}, \tag{6.3}$$

em que $Precision = P(Y = 1|\hat{Y} = 1)$ e $Recall = P(\hat{Y} = 1|Y = 1)$. Note que, quanto maior o F1 melhor a qualidade do modelo.

Nas aplicações utilizamos as seguintes nomenclaturas para as abordagens:

- **w2v + HMM (versão 1):** uso do *word2vec* para representação vetorial das palavras, seguido do ajuste do HMM para construção das covariáveis, utilizando a **análise de cluster** para identificabilidade;

- **w2v + HMM (versão 2):** uso do *word2vec* para representação vetorial das palavras, seguido do ajuste do HMM para construção das covariáveis, utilizando a **distância euclidiana** para identificabilidade;

- **w2v + AR:** uso do *word2vec* para representação vetorial das palavras, seguido do ajuste do modelo AR de séries temporais para construção das covariáveis;

- **w2v + VAR:** uso do *word2vec* para representação vetorial das palavras, seguido do ajuste do modelo VAR de séries temporais para construção das covariáveis;

- **w2v + Média:** uso do *word2vec* para representação vetorial das palavras, seguido do cálculo da média para construção das covariáveis;

- **w2v + Quantil:** uso do *word2vec* para representação vetorial das palavras, seguido do cálculo da média, desvio-padrão, mínimo, primeiro, segundo e terceiro quartis e do máximo para construção das covariáveis.

6.1 Avaliação de filmes

A base de dados contém 50 mil análises de filmes e a classificação do sentimento do telespectador em "positivo" (50%) ou "negativo" (50%) e foi retirada da plataforma *Kaggle* (Lakshmipathi N, 2019). A proporção de sentimentos positivos e negativos se manteve na base de treinamento.

A seguir, discutiremos os resultados obtidos ao aplicar a técnica t-SNE para redução de dimensão sobre o *word2vec* de dimensão 100.

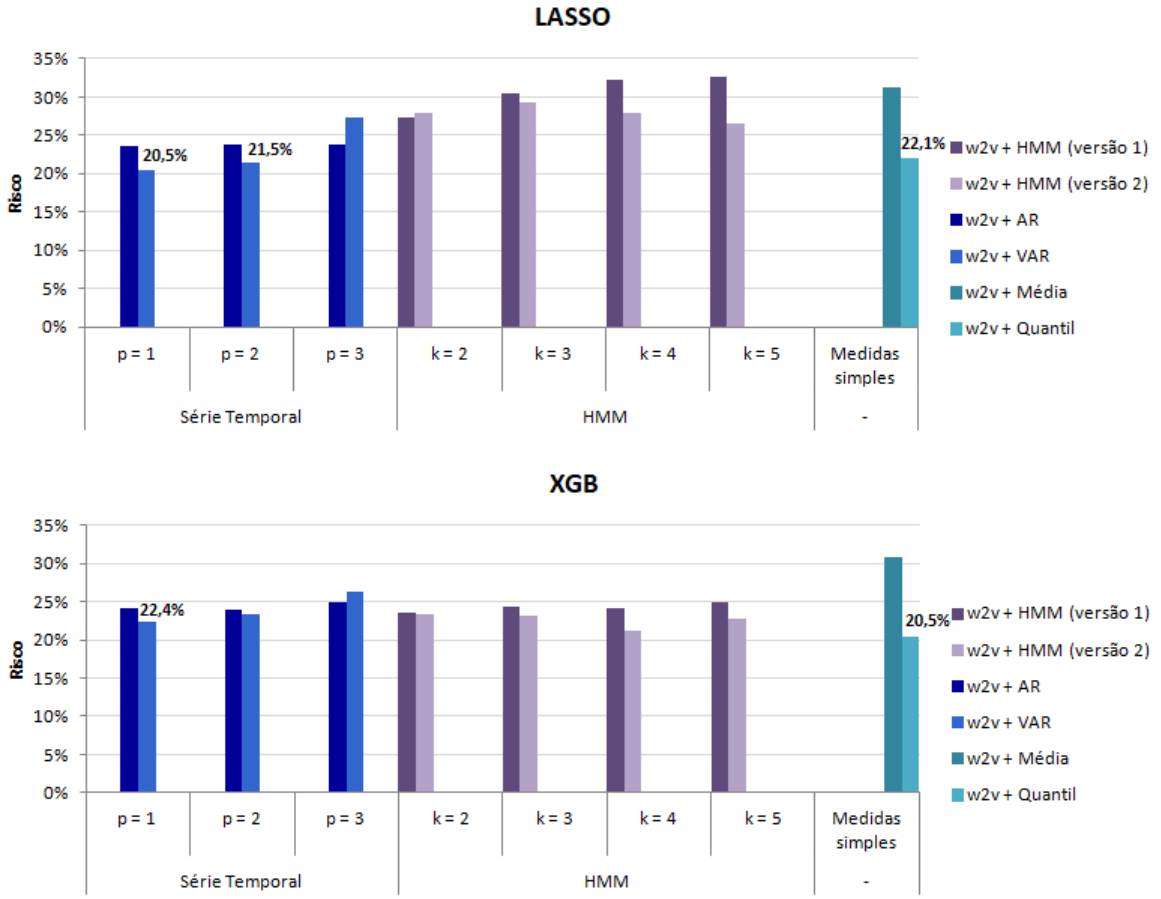

Figura 5 – Risco dos modelos de classificação para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dados de avaliação de filmes.

Na Figura 5 vemos o risco calculado para diferentes abordagens de construção de covariáveis. Notamos que, no LASSO, o risco das abordagem w2v+VAR para p = 1 e 2 são menores do que o risco considerando a abordagem w2v+quantil, porém o menor risco é atingido com o XGB considerando w2v+quantil. Já nas Figuras 6 e 7 vemos o AUC e o escore F1 dos modelos, respectivamente. O comportamento é semelhante, sendo o modelo XGB considerando w2v+quantil o que apresenta maior AUC e F1 (88,5% e 79,4%), porém com valor bastante próximo do atingido com o modelo LASSO considerando w2v+VAR com p = 1 (88% e 79%). Por fim, na Figura, chegamos a mesma conclusão ao analisar o escore F1.

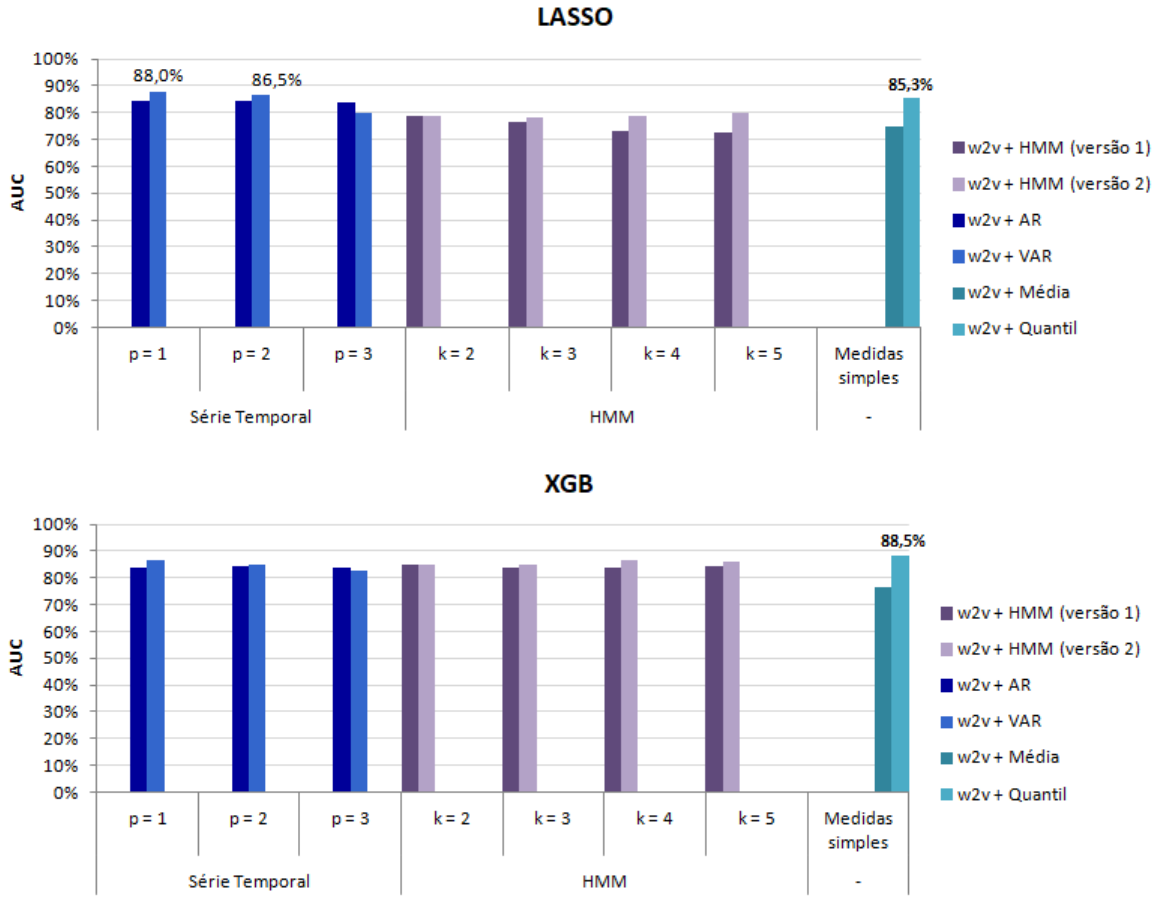

Figura 6 – AUC dos modelos de classificação para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dados de avaliação de filmes.

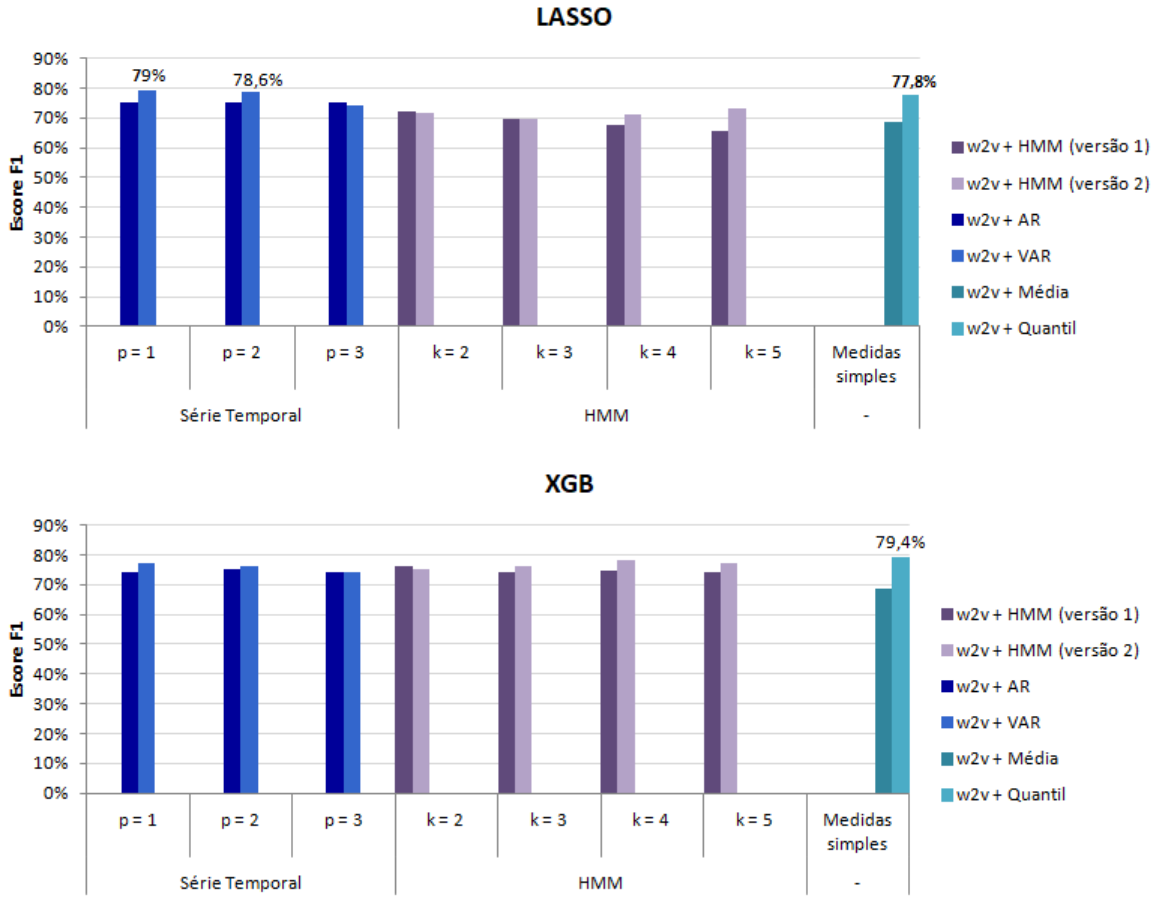

Figura 7 – Escore F1 dos modelos de classificação para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dados de avaliação de filmes.

Além dos modelos envolvendo o *word2vec*, metodologia em foco nesta pesquisa, ajustamos o modelo *bag-of-words* considerando bigramas e obtivemos um risco de 14%, valor menor do que o obtido com os demais modelos, indicando que a junção das metodologias (*bag-of-words* e *word2vec*) pode obter resultados de maior qualidade.

6.2 Processos judiciais

A base de dados foi extraída do site do Tribunal de Justiça de São Paulo (e-SAJ - TJSP) e conta o texto do recurso de cerca de 60 mil processos judiciais de segunda instância. Na Figura 8 vemos que cerca de 60% dos processos foram negados, ou seja, improcedentes, e 40% deles foram aceitos (procedentes).

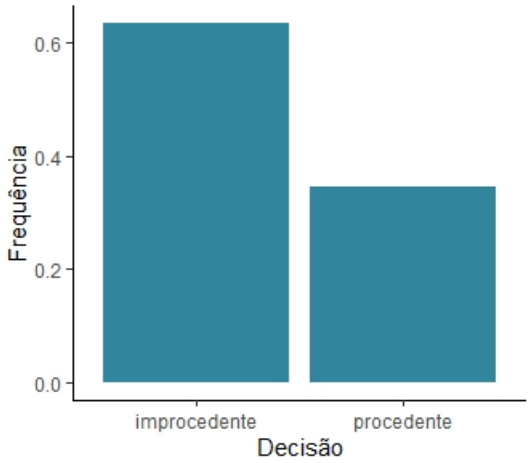

Figura 8 – Gráfico de barras da frequência de processos procedentes e improcedentes para dados de processos judiciais.

A seguir, discutiremos os resultados obtidos ao aplicar a técnica t-SNE para redução de dimensão sobre o *word2vec* de dimensão 100.

Na Figura 9 vemos que, ao considerar o LASSO, o risco dos modelo com 4 e 5 estados da abordagem w2v+HMM (versão 2) é menor do que o risco das abordagens que envolvem medidas simples (w2v+Média e w2v+Quantil). Ao analisar o XGB, vemos que o ganho se da também nas abordagens de série temporal (w2v+VAR) e nos modelos com 3, 4 e 5 estados ocultos da abordagem w2v+HMM (versões 1 e 2).

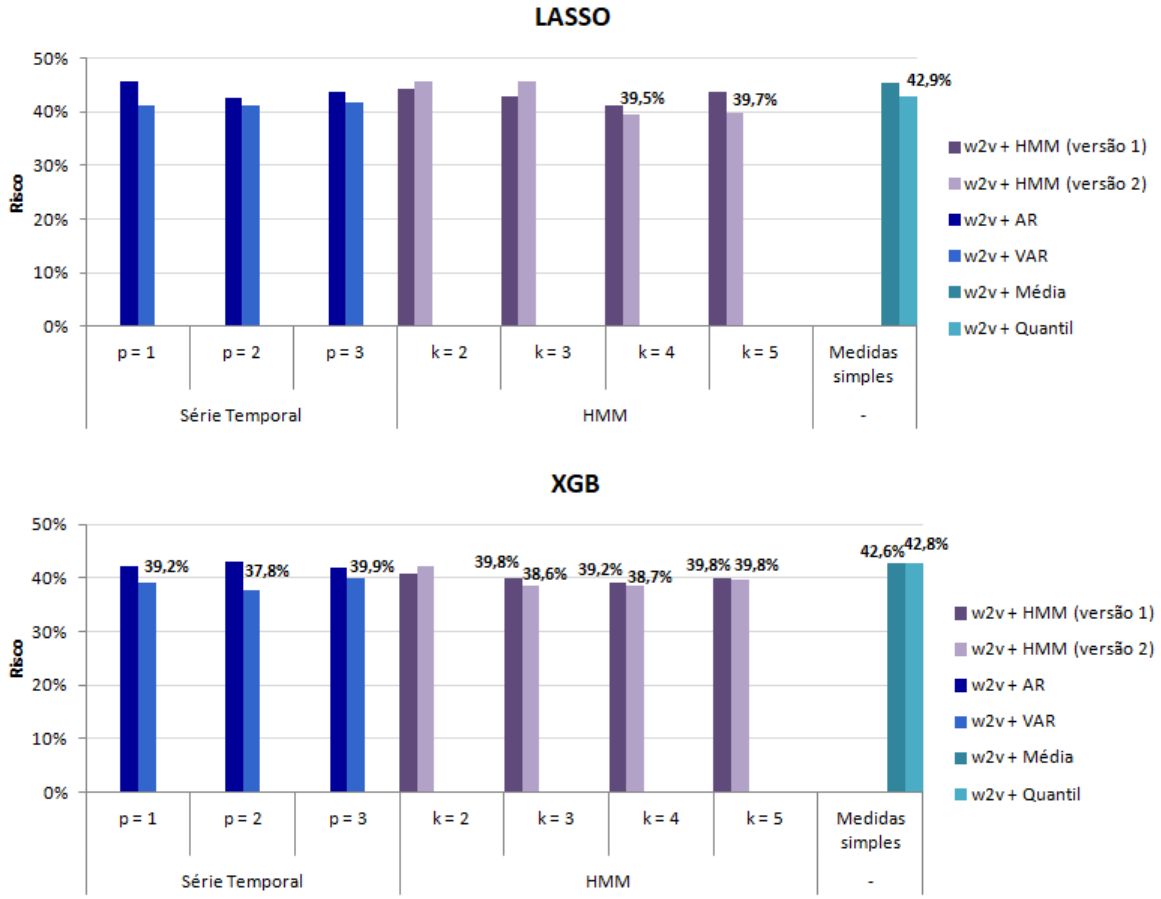

Figura 9 – AUC dos modelos de classificação para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dados de processos judiciais.

Já nas Figuras 10 vemos que o AUC das abordagens propostas é maior do que das abordagens usuais tanto no LASSO quando no XGB, sendo que o modelo com maior AUC é obtido com o XGB na abordagem w2v+VAR com p = 3 (66,2%). Por fim, analisando o escore F1 na Figura 10 as abordagens propostas se sobressaem, porém o maior F1 é atingido com o modelo LASSO considerando abordagem w3v+HMM - versão 2 - com 4 estados ocultos (74%).

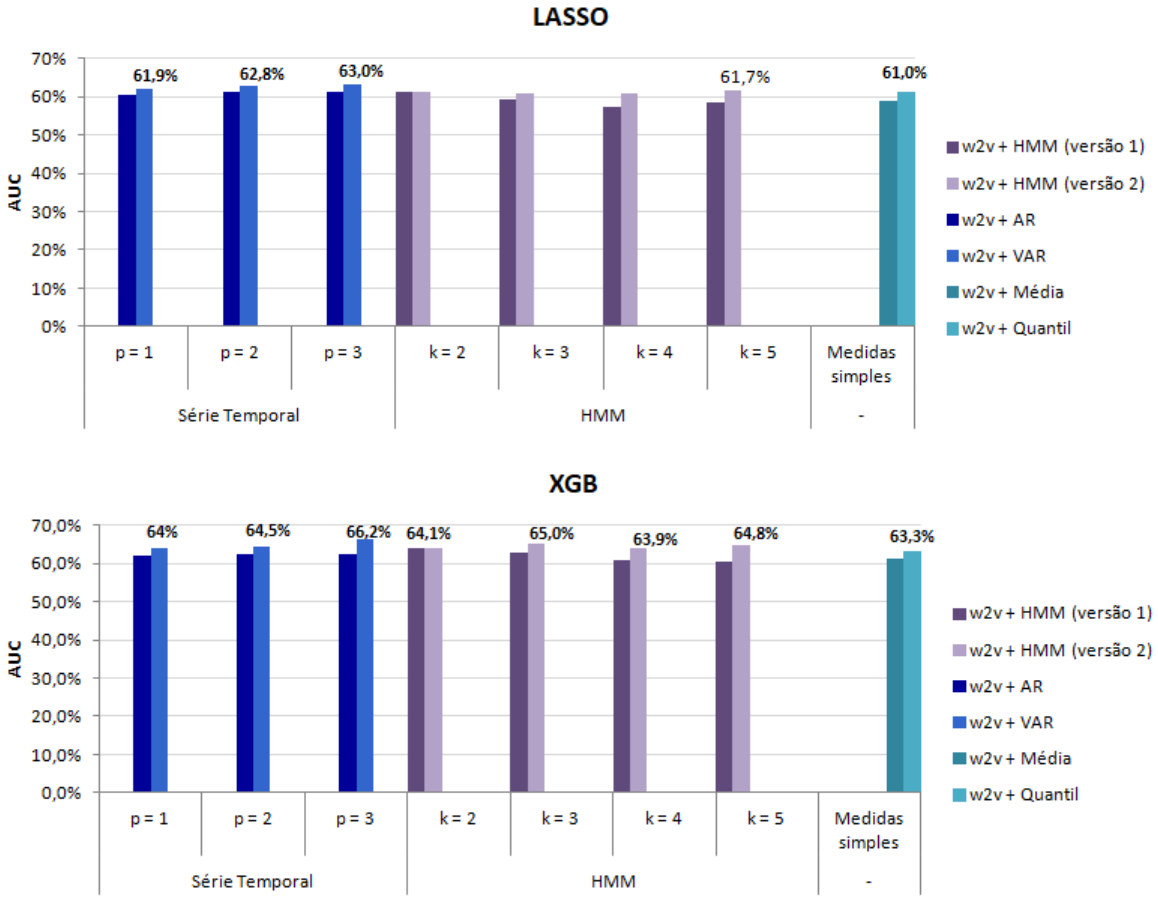

Figura 10 – AUC dos modelos de classificação para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dados de processos judiciais.

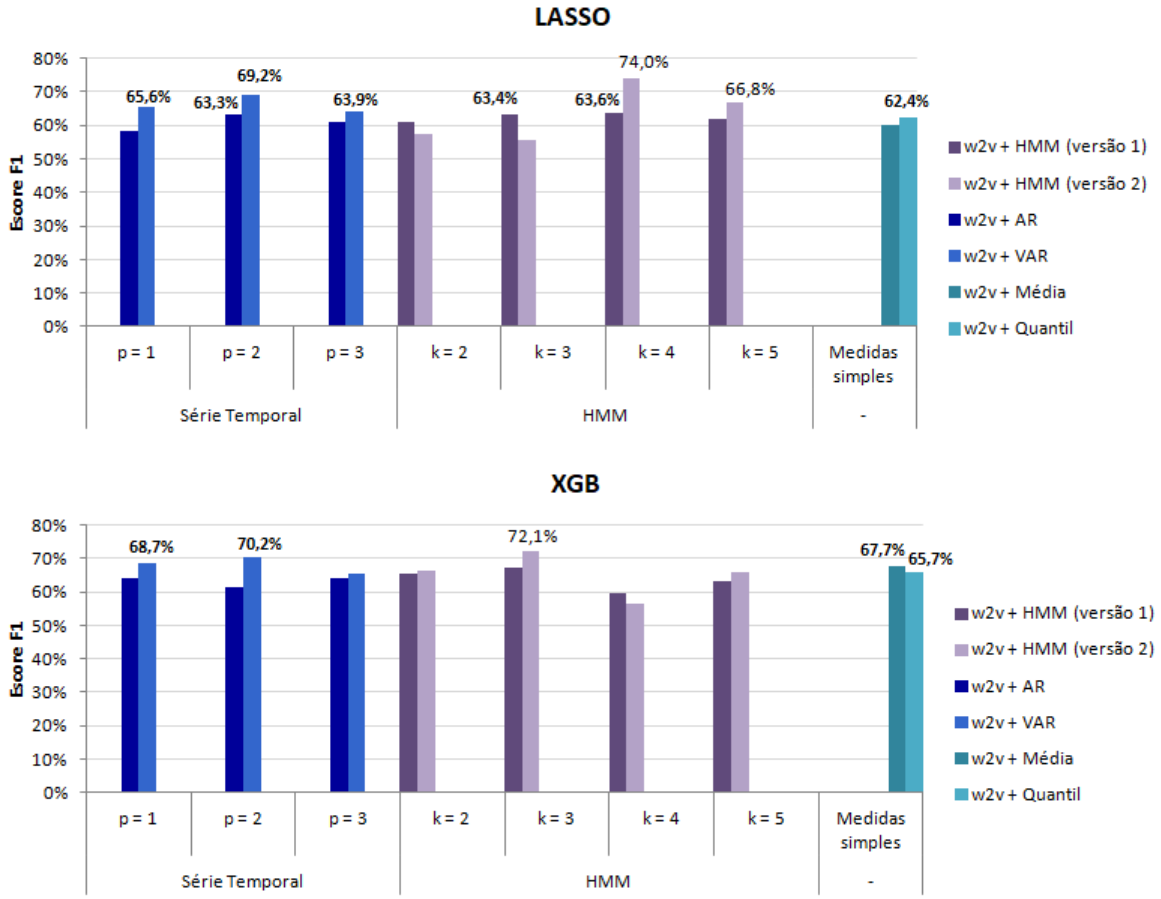

Figura 11 – Escore F1 dos modelos de classificação para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dpara dados de processos judiciais.

O modelo *bag-of-words* considerando bigramas e ajustado com o XGB retornou um risco de 35%, ou seja, resultados melhor dos que os demais modelos, porém tem F1 de 71%, valor menor do que os atingidos com a abordagem proposta. Assim como na análise de filmes, isso pode ser um indicativo de que combinar os modelos *bag-of-words* e *word2vec* pode ser benéfico.

6.3 Tweets da Copa do Mundo

A base de dados contém cerca de 80 mil *tweets* sobre a Copa do Mundo de Futebol de 2018 e foi retirada da plataforma *Kaggle* (Riturpana, 2018). O objetivo desta aplicação é, através do texto do *tweet*, prever o número de *Retweets* (RTs) que ele terá. Cerca de 80% dos *tweets* analisados não tiveram RTs.

Abaixo, discutiremos os resultados obtidos ao aplicar a técnica t-SNE para redução de dimensão sobre o *word2vec* de dimensão 100. Nesta aplicação, analisamos as abordagens usuais do *word2vec* e as abordagens envolvendo HMM.

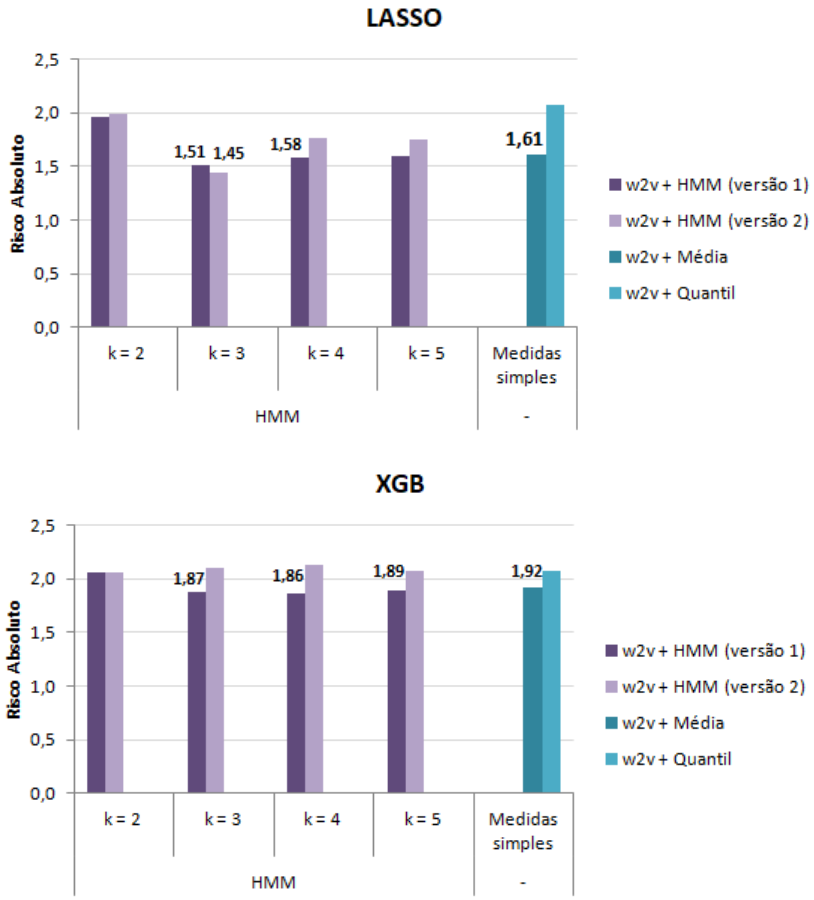

Figura 12 – Risco absoluto dos modelos de regressão para representação *word2vec* com dimensão 100 + t-SNE, considerando LASSO e XGB com diferentes abordagens para dados de tweets.

Na Figura 12 vemos o risco absoluto para diferentes abordagens de construção de covariáveis. Notamos que, para o LASSO, a abordagem w2v+quartil é a que apresenta pior desempenho. As abordagens que utilizam HMM com 3 e 4 estados ocultos tem desempenho melhor do que a média. Já com o XGB, apenas as abordagens w2v+hmm - versão 1 com 3, 4 e 5 estados se aproximam do resultado obtido apenas com a média. Diferente do que foi obtido nas outras aplicações, os modelo com LASSO apresentam melhor desempenho, sendo que o modelo w2v+HMM (versão 2) com 3 estados ocultos apresenta o menor risco absoluto (1,45).

O modelo *bag-of-words* ajustado com o XGB obteve risco absoluto de 1,59, valor maior que o risco do melhor modelo, porém menor, por exemplo, do que os riscos obtidos com as abordagens w2v+Média e w2v+Quantil.

É importante ressaltar que está base contém um excesso de zeros, ou seja, *tweets* que não tiveram RTs. Por outro lado, alguns tiveram números maiores de 400 RTs (*outliers*). Essas características fazem com que a abordagem de ajuste considerando distribuição Poisson para a contagem não seja mais adequada. Uma alternativa é testar ajustes mais próximos à essas características, ou ainda transformar a aplicação em um problema de classificação para prever se o *tweet* terá ou não RTs.

CAPÍTULO

7

CONCLUSÕES

Neste trabalho estamos interessados em verificar se a aplicação dos *Hidden Markov Models* ou de modelos de Séries Temporais na representação vetorial dada pelo *word2vec* traz maior poder preditivo nos modelos de classificação e regressão quando comparados às abordagens utilizadas usualmente. Com as aplicações realizadas até o momento, encontramos os seguintes resultados:

- Para as análises de filmes a abordagem com os quantis apresentou melhores resultados, porém as abordagens com w2v+VAR teve resultados muito próximos;
- Para os dados de processos judiciais, encontramos que a abordagem w2v+hmm - versão 2 é melhor do que o w2v+quartil para número maior de estados ocultos, mas a abordagem w2v+VAR com p = 2 se sobressai.
- Para os dados dos *tweets*, a abordagem HMM foi melhor do que as demais.

É importante ressaltar que as abordagens alternativas descritas nesta pesquisa possuem algumas desvantagens em relação às abordagens usuais, como o tempo computacional maior, dado que após o ajuste de modelo *word2vec* passa a ser necessário também o ajuste de um modelo estocástico, seja ele o HMM ou modelo de Séries Temporais. Isso implica também em um aumento no número de parâmetros que devem ser estimados no modelo, além do aumento do número de covariáveis no modelo de predição.

O ajuste de modelos preditivos que combinem o modelo *bag-of-words* e *word2vec* buscando a melhoria do poder preditivo pode ser realizado unindo as covariáveis resultantes do *bag-of-words* ajustado com bigramas e as covariáveis resultantes das abordagens que fazem

uso do *word2vec*. Para não voltar ao problema de grande número de covariáveis do *bag-of-words* colocado na introdução deste trabalho, faz-se necessário utilizar filtros de frequência de palavras para diminuir a dimensão do problema. Por exemplo, palavras que aparecem em uma alta porcentagem de textos, ou em uma baixíssima porcentagem de textos provavelmente não tera importância quanto a predição da variável resposta do problema, sendo que o que é alta ou baixíssima porcentagem de textos deve ser definido a depender do problema.

Quanto as evoluções possíveis deste trabalho, podemos citar:

- Explorar um banco de dados de menor dimensão (cerca de 1 mil observações) a fim de verificar o comportamento preditivo dos modelos se mantém;
- Nos modelos markovianos ocultos, pode-se explorar estruturas mais complexas do que modelos de ordem 1, como modelos com mais de uma camada oculta, entre outras estruturas;
- Utilizar algoritmos para estimar o número de estados ocultos do HMM ou ainda o alcance dos modelos de Series Temporais;
- Tanto nos modelos de Séries Temporais quanto no HMM consideramos apenas a dependência da palavra presente em relação a(s) palavra(s) passada(s). Pode-se investigar abordagens que considerem também a influência e correlação da(s) palavra(s) futura(s) com a palavra atual.
- Explorar as Cadeias de Markov com alcance variável (VLMC, do inglês *Variable Length Markov Chain*), pressupondo que palavras diferentes podem depender de um número diferente de vizinhos.

Desta forma, concluímos que a abordagem alternativa utilizando processos estocásticos (HMM e Séries Temporais) apresentam resultados melhores ou iguais aos resultados com abordagens usuais. Sendo assim, a pesquisa mostra que as abordagens alternativas são uma ferramenta extra para obtenção de resultados preditivos melhores no campo da análise de texto.

REFERÊNCIAS

BAUM, L. E.; PETRIE, T. Statistical inference for probabilistic functions of finite state markov chains. **The annals of mathematical statistics**, JSTOR, v. 37, n. 6, p. 1554–1563, 1966. Citado na página 33.

BOTTOU, L. Large-scale machine learning with stochastic gradient descent. In: **Proceedings of COMPSTAT'2010**. [S.l.]: Springer, 2010. p. 177–186. Citado na página 26.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: **Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining**. [S.l.: s.n.], 2016. p. 785–794. Citado na página 41.

FAN, J.; UPADHYE, S.; WORSTER, A. Understanding receiver operating characteristic (roc) curves. **Canadian Journal of Emergency Medicine**, Cambridge University Press, v. 8, n. 1, p. 19–20, 2006. Citado na página 44.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. **The elements of statistical learning**. [S.l.]: Springer series in statistics New York, 2001. 2 p. Citado na página 19.

GOLDBERG, Y.; LEVY, O. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. **arXiv preprint arXiv:1402.3722**, 2014. Citado na página 25.

HARRIS, Z. S. Distributional structure. **Word**, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954. Citado na página 20.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The elements of statistical learning: data mining, inference, and prediction**. [S.l.]: Springer Science & Business Media, 2009. Citado na página 40.

Lakshmipathi N. **Conjunto de dados IMDB de críticas de filmes de 50K**. 2019. <https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews>. Accessed: 2020-08-04. Citado na página 45.

MAATEN, L. v. d.; HINTON, G. Visualizing data using t-sne. **Journal of machine learning research**, v. 9, n. Nov, p. 2579–2605, 2008. Citado na página 28.

MACQUEEN, J. *et al.* Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. **Proceedings of the fifth Berkeley symposium on mathematical statistics and probability**. [S.l.], 1967. v. 1, n. 14, p. 281–297. Citado na página 35.

MANNING, C.; SOCHER, R.; FANG, G. G.; MUNDRA, R. Cs224n: Natural language processing with deep learning1. 2017. Citado na página 26.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013. Citado nas páginas 15, 20, 24 e 25.

MORETTIN, P. A.; TOLOI, C. Análise de séries temporais. In: **Análise de séries temporais**. [S.l.: s.n.], 2006. p. 538–538. Citado na página 29.

PINSKY, M.; KARLIN, S. **An introduction to stochastic modeling**. [S.l.]: Academic press, 2010. Citado na página 29.

RABINER, L.; JUANG, B. An introduction to hidden markov models. **ieee assp magazine**, IEEE, v. 3, n. 1, p. 4–16, 1986. Citado na página 31.

Riturpana. **FIFA World Cup 2018 Tweets**. 2018. <https://www.kaggle.com/rgupta09/world-cup-2018-tweets>. Accessed: 2020-08-24. Citado na página 53.

SPEZIA, L. Reversible jump and the label switching problem in hidden markov models. **Journal of Statistical Planning and Inference**, Elsevier, v. 139, n. 7, p. 2305–2315, 2009. Citado na página 34.

STEPHENS, M. Bayesian analysis of mixture models with an unknown number of components-an alternative to reversible jump methods. **Annals of statistics**, JSTOR, p. 40–74, 2000. Citado na página 33.

STERN, D. B. *et al.* Vector representation of texts applied to prediction models. Universidade Federal de São Carlos, 2020. Citado na página 27.

TIBSHIRANI, R. Regression shrinkage and selection via the lasso. **Journal of the Royal Statistical Society: Series B (Methodological)**, Wiley Online Library, v. 58, n. 1, p. 267–288, 1996. Citado na página 37.


ICMC
USP
SÃO CARLOS