UNIVERSIDADE DE SÃO PAULO
FACULDADE DE ECONOMIA, ADMINISTRAÇÃO, CONTABILIDADE E ATUÁRIA

DEPARTAMENTO DE ADMINISTRAÇÃO
PROGRAMA DE MESTRADO PROFISSIONAL EM EMPREENDEDORISMO

KLEBER RODRIGUES DOS SANTOS

Alertas Sanitários em Cidades Inteligentes: artefato para previsão de doenças com base em dados de redes sociais

São Paulo
2023

Prof. Dr. Carlos Gilberto Carlotti Júnior
Reitor da Universidade de São Paulo

Profa. Dra. Maria Dolores Montoya Diaz
Diretora da Faculdade de Economia, Administração, Contabilidade e Atuária

Prof. Dr. João Maurício Gama Boaventura
Chefe do Departamento de Administração

Profa. Dra. Graziella Maria Comini
Coordenadora do Programa de Mestrado Profissional em Empreendedorismo

KLEBER RODRIGUES DOS SANTOS

Alertas Sanitários em Cidades Inteligentes:
artefato para previsão de doenças com base em dados de redes sociais

Versão Corrigida

Dissertação apresentada ao Programa de Pós-Graduação em Mestrado Profissional em Empreendedorismo do Departamento de Administração da Faculdade de Economia, Administração, Contabilidade e Atuária da Universidade de São Paulo como requisito parcial para obtenção do título de Mestre em Ciências.

Orientadora: Profa. Dra. Daielly Melina Nassif Mantovani

São Paulo

2023

Ficha Catalográfica

Catalogação na Publicação (CIP)
Ficha Catalográfica com dados inseridos pelo autor

Santos, Kleber.
Alertas Sanitários em Cidades Inteligentes: artefato para previsão de doenças com base em dados de redes sociais / Kleber Santos. - São Paulo, 2023.
123 p.

Dissertação (Mestrado) - Universidade de São Paulo, 2023.
Orientador: Daielly Mantovani.

1. Vigilância sanitária. 2. Aprendizado de máquina. 3. COVID-19. 4. Redes sociais. 5. Cidades Inteligentes. I. Universidade de São Paulo. Faculdade de Economia, Administração, Contabilidade e Atuária. II. Título.

FOLHA DE AVALIAÇÃO

Nome: SANTOS, KLEBER

Título: Alertas Sanitários em Cidades Inteligentes: artefato para previsão de doenças com base em dados de redes sociais

Dissertação apresentada ao Programa de Pós-Graduação em Mestrado Profissional em Empreendedorismo do Departamento de Administração da Faculdade de Economia, Administração, Contabilidade e Atuária da Universidade de São Paulo, para obtenção do título de Mestre em Ciências.

Aprovado em:

Banca Examinadora

Prof. Dr.: _____
Instituição: _____
Julgamento: _____

Prof. Dr.: _____
Instituição: _____
Julgamento: _____

Prof. Dr.: _____
Instituição: _____
Julgamento: _____

Este trabalho é dedicado à minha esposa, filhos e meus queridos pais.

AGRADECIMENTOS

Primeiramente, aos meus pais, José Carlos e Maria das Graças que sempre me apoiaram em todos os meus sonhos e objetivos.

Aos meus filhos Gabriel e Eduardo, por despertarem em mim o interesse em cursar o Mestrado e voltar para a academia.

A minha querida e amada esposa Lígia, pelo suporte e por me incentivar a sempre ser uma versão melhor a cada dia.

A minha orientadora, Profa. Dra. Daielly Melina Nassif Mantovani por todo aprendizado durante a jornada da dissertação, pelas várias conversas pessoais a quem passei a ter como uma amiga.

Aos membros da banca de qualificação, Prof. Dr. Antonio Geraldo da Rocha Vidal, Profa. Dra. Deise Santana de Jesus Barbosa e Prof. Dr. Francisco Aparecido Rodrigues pelas críticas e sugestões que ajudaram a nortear o desenvolvimento deste trabalho.

Aos demais professores do Mestrado Profissional em Empreendedorismo da Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo (FEA-USP), pela grande contribuição para o meu aprendizado. Em especial, à Prof. Dra. Jane Aparecida Marques, pelo apoio na definição da metodologia da dissertação.

A todos os colegas da Turma 8 do Mestrado Profissional em Empreendedorismo da FEA-USP, pela inspiração e apoio durante este período. Foi um prazer fazer parte de uma turma tão especial!

Ao meu chefe direto na diretoria de planejamento da Vivo Rafael Valente Belhot que me apoiou e ajudou a conciliar as atividades profissionais e a jornada do mestrado.

Ao meu grande amigo Prof. Dr. Jaercio Alex Barbosa que tive o privilégio de ser par na Telefonica/Vivo durante 8 anos e ser sócio em uma jornada empreendedora. Obrigado Jaercio por me indicar e incentivar incontáveis vezes a cursar este mestrado na FEA.

Sem vocês, este trabalho não seria possível.

“- Gostaria que isso não tivesse acontecido na minha época – disse Frodo.

- Eu também – disse Gandalf. – Como todos os que vivem nestes tempos. Mas a decisão não é nossa. Tudo o que temos de decidir é o que fazer com o tempo que nos é dado.” (TOLKIEN, J. R. R. O Senhor dos Anéis: A Sociedade do Anel)

RESUMO

Durante a elaboração desta dissertação, o mundo enfrentou a pandemia do COVID-19. No Brasil, entre janeiro de 2020 e fevereiro de 2023, 697mil brasileiros morreram devido a COVID-19 e 36,9 milhões de casos foram registrados. Além disso, os impactos econômicos e sobre os sistemas produtivos levaram a uma crise de proporções globais, com aumentos de preços, redução dos níveis de emprego e escassez de matérias primas entre outros. Essa pandemia também deixou claro que a demanda por serviços de saúde pode se tornar imprevisível, o que fortalece a relevância de se construir modelos e métodos capazes de antecipar tendências. Dentre as diversas fontes de dados complementares, as redes sociais têm sido utilizadas por bilhões de pessoas como uma ferramenta de comunicação, gerando conteúdo sobre tópicos variados e permitindo o compartilhamento de informações. Com um bom levantamento de dados, as plataformas de redes sociais estão se tornando ferramentas essenciais. Compreender essas informações, opiniões, momento, local e sua disseminação pode fornecer recursos inestimáveis para também alimentar os sistemas de alerta precoce. Para explorar o potencial dos dados das redes sociais, este estudo realiza uma análise retrospectiva da pandemia de COVID-19 no Brasil e investiga mais de 10 milhões de tweets. Com a ajuda de técnicas de processamento de linguagem natural, aprendizado de máquina e modelagem de tópico, identifica o conteúdo textual de cada tweet refletindo assim o contexto dos posts sobre o momento da pandemia. Após analisar os posts, foram criados modelos de regressão temporal demonstrando ser possível criar alertas para o aumento de casos, combinado com informações oficiais de contagens de casos anteriores. São calibrados três modelos que foram capazes de estimar os casos semanais com 1 semana de antecedência com razoável precisão, sendo um modelo para a São Paulo, outro para Amazonas e um modelo para o Brasil. Essa abordagem tem as vantagens de rapidez, quantidade, informação temporal e espacial, além de boa previsibilidade para auxiliar no desdobramento de crises, podendo alimentar os atuais sistemas de acompanhamento ajudando as agências governamentais a melhorarem os alertas precoces em tempo hábil.

Palavras-chave: Vigilância sanitária, redes sociais, aprendizado de máquina, inteligência artificial, modelo preditivo, COVID-19, pandemia, geolocalização, análise textual

ABSTRACT

During the development of this dissertation, the world faced the COVID-19 pandemic. In Brazil, from January 2020 to February 2023, 697 thousand Brazilians died due to COVID-19 and 36.9 million cases were registered. In addition, the economic impacts and impacts on production systems led to a crisis of global proportions, with price increases, reduced levels of employment and shortages of raw materials, among others. This pandemic also made it clear that the demand for health services can become unpredictable, which strengthens the relevance of building models and methods capable of anticipating trends. Among the various sources of complementary data, social networks have been used by billions of people as a communication tool, generating content on various topics and allowing the sharing of information. With good data collection, social networks platforms are becoming essential tools. Understanding this information, opinions, time, place, and its dissemination can provide invaluable resources to also increase early warning systems. To explore the potential of social networks data, this study performs a retrospective analysis of the COVID-19 pandemic in Brazil and investigates more than 10 million tweets. Using natural language processing techniques, machine learning and topic modeling, it identifies the textual content of each tweet, thus reflecting the context of the posts about the moment of the pandemic. After analyzing the posts, temporal regression models were created, demonstrating that it is possible to create alerts for the increase in cases, combined with official information from previous case counts. Three models are calibrated that were able to estimate weekly cases 1 week in advance with reasonable accuracy, one model for São Paulo, another for Amazonas and a model for Brazil. This approach has the advantages of speed, quantity, temporal and spatial information and good predictability to support crisis unfold and can increase current warning systems by supporting government agencies to improve early warnings.

Keywords: Health surveillance, social networks, machine learning, artificial intelligence, predictive model, COVID-19, pandemic, geolocation, textual analysis

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE EQUAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

AIDS - Síndrome da Imunodeficiência Humana

AM - Amazonas

ANATEL - Agência Nacional de Telecomunicações

API - *Application Programming Interface* (Interface de Programação de Aplicação)

COVID-19 - Corona Vírus 2019

DataSUS - Dados Sistema Único de Saúde

DSR - *Design Science Research*

LDA - *Latent Dirichlet Allocation*

ML - *Machine Learning* (Aprendizado de Máquina)

PLN - Processamento de Linguagem Natural

RSL - Revisão Sistemática da Literatura

SP - São Paulo

TIC - Tecnologias da Informação e Comunicação

ODS - Objetivos de Desenvolvimento Sustentável

ONU - Organização das Nações Unidas

GPS - Global Positioning System

VIF - Variance Inflation Factor

SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTO DO PROJETO OU SITUAÇÃO PROBLEMA

Durante a elaboração desta dissertação, o mundo enfrentou a pandemia de COVID-19. Com possível origem em Wuhan na China, o vírus se espalhou por todo o planeta em poucos meses, forçando muitos países a adotarem diversas medidas para conter a disseminação do vírus (Spurlock & Elgazzar, 2020). No Brasil, segundo dados divulgados pelo Sistema Único de Saúde (*DataSus*, 2022) entre janeiro de 2020 e fevereiro de 2023, 697mil brasileiros morreram devido a COVID-19 e 36,9 milhões de casos foram registrados. Além disso, os impactos econômicos e sobre os sistemas produtivos levaram a uma crise de proporções globais, com aumentos de preços, redução dos níveis de emprego e escassez de matérias primas entre outros.

O enfrentamento da COVID-19 desencadeou uma série de ocorrências que até então estavam latentes (Lima et al., 2020). Tal situação acelerou os processos que inevitavelmente aconteceriam, antecipando-os em alguns anos. O ensino público e privado *online* em todos os níveis educacionais, com especial aplicabilidade no nível superior e formação continuada, a ampliação massiva da realização de eventos e reuniões mediadas pela tecnologia, trabalho remoto e transformação digital de organizações de diferentes portes e setores, são exemplos dessa antecipação da incorporação da tecnologia, provocada pelo surto de coronavírus.

A pandemia da COVID-19 deixou claro que a demanda por serviços de saúde pode se tornar imprevisível, o que fortalece a relevância de se construir modelos e métodos capazes de antecipar tendências com um bom levantamento de dados (Popkova & Sergi, 2021).

O grande desafio está em como obter dados diversificados e em grande escala com informações sobre a saúde da população de forma automática e com baixo custo. Ferramentas tecnológicas e de análise de dados, apesar de apresentarem vários benefícios, possuem limitações para serem utilizadas pela sociedade de forma massificada. Entre as principais barreiras estão os elevados custos de investimento, implementação, manutenção e eventual substituição de sistemas (Allam & Jones, 2020).

No caso da saúde, investimentos em tecnologia podem trazer grandes benefícios para a população, principalmente em locais nos quais a infraestrutura de saúde é mais precária. O conhecimento da situação epidemiológica pode auxiliar no desenvolvimento de novos conhecimentos, aprimorando as estratégias de intervenção (Xavier et al., 2020).

Com o desenvolvimento em ritmo exponencial, a tecnologia passou a fazer parte do dia a dia de organizações privadas e públicas mudando a estratégia de negócios e a vida das pessoas, fazendo com que as decisões passem a ser orientadas por dados (Davenport, 2006).

Essa evolução da cultura de dados, se bem enraizada em infraestrutura de tecnologia da informação com qualidade, pode ser o alicerce para o desenvolvimento de toda a sociedade baseada na coleta e uso eficiente dos dados.

No Brasil, a penetração de *smartphones* é ampla na população, sendo a principal forma de comunicação e acesso à internet atualmente. De acordo com dados divulgados pela ANATEL, em julho de 2021, o Brasil possuía 246,8 milhões de acessos de telefonia móvel, equivalente a uma densidade (acessos por habitantes) de 101,4%, ou seja, pouco mais de 1 acesso por habitante (Tabela 1-1). Observa-se ainda que a região sudeste, mais populosa, concentra a maior quantidade de acessos absolutos e também a maior densidade (número de acessos por habitantes), enquanto a região norte apresenta a menor quantidade de acessos e menor densidade do país, indicando-se oportunidades de expansão para esta região.

Tabela 1-1 - Número de acessos de telefonia móvel por região

Região	Acessos	Densidade
Brasil	246.792.264	101,4%
Centro-Oeste	18.801.560	105,6%
Nordeste	56.111.613	93,6%
Norte	**17.245.792**	**88,3%**
Sudeste	**120.449.985**	**108,5%**
Sul	34.183.314	101,0%

Fonte: Anatel (2021).
Nota: Densidade = acessos por habitantes (%)

Com relação a cobertura de internet, 92,36% dos domicílios no Brasil estavam cobertos com acesso em 2021 (Anatel, 2021), mas existindo muita limitação de acesso principalmente em zonas rurais, o que pode dificultar a transformação digital nesses locais (Pivoto, 2018). Como apresentado na Tabela 1-2, a cobertura rural variava entre estados, sendo o Distrito Federal com cobertura rural de 90,1%, enquanto em Roraima 3,1% e no Amazonas 9,3%.

Na mesma toada, os dados indicam que apenas 46% da malha rodoviária do Brasil possui cobertura de internet móvel sendo bem heterogêneo entre as regiões como apresentado na Tabela 1-2.

Mesmo com esse cenário de cobertura de internet, felizmente, o advento das redes móveis de telecomunicações, das redes sociais e dos *smartphones*, possibilitam que milhões de brasileiros estejam conectados.

Tabela 1-2 - Cobertura de internet por estado no Brasil

Estado	Área Coberta (%)	Domicílio (%)	Domicílios Rurais com Internet (%)	Cobertura internet nas rodovias (%)
DF	78,3%	99,7%	90,1%	92,0%
SE	60,1%	92,5%	70,3%	83,4%
SP	56,2%	98,8%	72,2%	83,4%
RJ	54,2%	98,7%	67,8%	81,1%
ES	51,2%	93,1%	57,9%	77,6%
AL	47,3%	89,2%	57,2%	70,5%
PB	42,5%	89,5%	56,8%	70,2%
CE	40,5%	89,4%	56,5%	67,1%
RN	39,2%	90,8%	56,2%	66,5%
PR	36,7%	91,9%	47,3%	65,9%
PE	36,6%	92,1%	57,8%	62,8%
SC	35,5%	91,5%	49,9%	62,2%
RS	30,0%	92,0%	48,3%	60,1%
MG	24,1%	90,5%	34,1%	52,5%
BA	14,1%	80,6%	31,0%	39,5%
GO	13,4%	92,3%	26,5%	38,1%
PI	11,9%	74,8%	23,9%	37,1%
MA	10,5%	75,1%	29,2%	34,7%
TO	8,5%	82,9%	17,9%	33,5%
MS	6,6%	88,4%	21,0%	30,5%
RO	4,7%	76,8%	16,8%	26,4%
MT	3,3%	83,4%	13,6%	22,6%
PA	2,8%	77,2%	24,5%	19,3%
AC	2,5%	80,6%	20,0%	16,6%
AP	1,8%	90,2%	9,1%	11,5%
RR	1,3%	79,9%	3,1%	10,7%
AM	0,9%	83,7%	9,3%	4,6%

Fonte: (Anatel, 2021)

Com a popularização dessas tecnologias, as redes sociais se tornaram uma fonte abundante, ágil, precisa e barata para obtenção de dados, uma vez que já estão pulverizados e implantados em quase toda a sociedade. (Nguyen et al., 2021).

Aliado ao potencial de análise de dados, as plataformas de redes sociais estão se tornando ferramentas essenciais para os tomadores de decisão comunicarem informações às partes interessadas e aproveitarem as opiniões públicas dos usuários on-line. As propriedades das redes sociais permitem que os indivíduos divulguem informação e conhecimento, tornando assim possível evocar a consciência pública em larga escala num curto período. Compreender essas informações, opiniões, momento, local e sua disseminação nas redes sociais pode fornecer recursos inestimáveis para alimentar os sistemas de alerta precoce (L. Li et al., 2021).

Na era da conectividade, da internet, e das redes sociais surgiu o termo *Big Data*, indicando dados produzidos em grande volume, velocidade e tempo real, que podem ser extraídos e manipulados em busca de informações relevantes que as leve a ações ágeis e precisas. O processamento de grandes volumes de informações não estruturadas, em vários idiomas e num intervalo de minutos, tem potencial de proporcionar um melhor sistema de alertas que pode ser implementado nas cidades (Bragazzi et al., 2020).

Este modelo de atuação de análise de dados permite que a equipe de saúde foque seus esforços em responder ao risco de proliferação e tratamento das doenças em vez de dedicar tempo e energia garimpando e organizando informações. Este papel cabe à equipe de cientistas de dados que tem formação específica e fica focada na captura, organização dos dados, identificação de padrões e alertas com forte base matemática, estatística e computacional.

No caso brasileiro, há um esforço do governo em ampliar a coleta e divulgação de dados públicos com a criação do Portal Dados Abertos Brasil, que traz acesso a milhares de bases de dados das esferas municipais, estaduais e federal. Em específico, a área da saúde conta com o sistema DataSUS que congrega um conjunto de sistemas de informação referentes a pacientes do sistema de saúde público e privado.

Neste contexto, a motivação desta dissertação baseou-se em como a tecnologia, as telecomunicações, os *smartphones* e as redes sociais, em complemento aos dados oficiais, podem auxiliar em soluções com grande impacto para a sociedade durante eventos como a pandemia de COVID-19.

1.2 OBJETIVOS

1.2.1 Objetivo Principal

Criar um modelo preditivo com dados das redes sociais, utilizando técnicas de estatística, *Machine Learning* e *Big Data*, para sinalizar possíveis ameaças sanitárias no Brasil, focando nos eventos recentes da COVID-19.

1.2.2 Objetivos específicos

- Classificar e analisar a evolução do comportamento e preocupações das pessoas durante a pandemia utilizando os dados da rede social Twitter
- Correlacionar o histórico de casos e o histórico de postagem na rede social Twitter
- Desenvolver modelo de projeção para antecipar variações nos números de casos utilizando tweets e histórico de casos

1.3 ESTRUTURA DO PROJETO DE DISSERTAÇÃO

Para atingir os objetivos propostos, a presente dissertação está estruturada em seis capítulos:

Capítulo 1 – Introdução: são apresentados o contexto e os objetivos da dissertação.

Capítulo 2 – Metodologia de Pesquisa: são apresentados os procedimentos adotados para a *Design Science Research (DSR)*.

Capítulo 3 – Revisão Sistemática da Literatura: seguindo o protocolo de pesquisa do DSR, é apresentado o passo a passo da Revisão de Literatura para o levantamento da literatura, artefatos e classes de problemas.

Capítulo 4 - Referencial Teórico: são apresentados os conceitos estabelecidos por diversos autores referentes aos temas a serem abordados baseados nos agrupamentos e classes de problemas obtidos na revisão sistemática conforme o DSR.

Capítulo 5 – Projeto e Desenvolvimento do Artefato: são apresentadas as fases de construção do artefato, algoritmo utilizado e análise dos resultados

Capítulo 6 – Conclusões: são apresentadas as conclusões, limitações do trabalho atual e sugestões de trabalhos futuros.

2 METODOLOGIA

Nesta dissertação foi adotada a metodologia de *Design Science Research* (DSR). Algumas caraterísticas do DSR que justificam sua escolha estão na criação de artefatos que permitam soluções satisfatórias a problemas práticos, avaliar o artefato proposto com simulações e experimentos além da solução ser generalizável a uma determinada classe de problemas.

Seguindo o protocolo de pesquisa proposto por Dresch (2015), para o levantamento de literatura e artefatos, foi realizada uma Revisão Sistemática de Literatura (RSL) detalhada no capítulo 3 desta dissertação.

2.1 PESQUISA EM CIÊNCIA DO DESIGN (DSR)

O termo *Science of Design*, que posteriormente passou a ser *Design Science* (DS), foi introduzido pela obra "As ciências do Artificial", do economista e psicólogo Simon (1996). Segundo Simon, o Artificial, é aquilo que foi produzido ou inventado pelo homem ou que sofre intervenção deste. O *Design Science* é a ciência que se ocupa do projeto, logo, não tem como objetivo descobrir leis naturais ou universais que expliquem certo comportamento dos objetos que estão sendo estudados

Segundo Dresch et al. (2015), a *Design Science* é a ciência que procura desenvolver e projetar soluções para melhorar sistemas existentes, resolver problemas ou, ainda, criar artefatos que contribuam para uma melhor atuação humana, seja na sociedade ou nas organizações.

A *Design Science Research*, por sua vez, é o método que fundamenta e operacionaliza a condução da pesquisa, buscando, a partir do entendimento do problema, construir e avaliar artefatos que permitam transformar situações, alterando suas condições para estados melhores ou desejáveis. A *Design Science Research* é um processo rigoroso de projetar artefatos para resolver problemas, avaliar o que foi projetado ou o que está funcionando, e comunicar os resultados obtidos.

A natureza deste tipo de pesquisa costuma ser pragmática e orientada à solução. Ou seja, o conhecimento deve ser construído a serviço da ação. É essencial não perder de vista que a *Design Science*, ainda que se ocupe da solução de problemas, não busca um resultado ótimo. Segundo Simon (1996), o tomador de decisão pode escolher entre decisões ótimas em um

mundo simplificado ou decisões (suficientemente boas) que o satisfazem, num mundo mais próximo da realidade.

Na busca por soluções suficientemente boas para problemas em que a solução ótima seja inacessível ou de implantação inviável, exige-se uma definição clara do que seriam os resultados satisfatórios. Isso pode ser obtido com consenso entre as partes envolvidas no problema e/ou no avanço da solução atual em comparação com as soluções geradas pelos artefatos anteriores (Dresch et al., 2015).

Os problemas existentes no mundo real costumam ser específicos, e esta especificidade poderia inviabilizar um conhecimento passível de generalização, sendo assim, o agrupamento dos artefatos em classe de problemas permite que um artefato desenvolvido para um objetivo específico pode gerar conhecimentos que agrupados com outros artefatos podem levar ao desenvolvimento ou aprimoramento de soluções geradas pelos artefatos anteriores (Dresch et al., 2015).

No Quadro 2-1, procura-se agrupar os elementos centrais do conhecimento em *Design Science.*

Quadro 2-1 - Síntese dos principais conceitos DSR

Conceito de *Design Science*	Ciência que procura consolidar conhecimentos sobre o projeto e desenvolvimento de soluções para melhorar sistemas existentes, resolver problemas e criar artefatos
Artefato	Algo que é construído pelo homem; interface entre o ambiente interno e o ambiente externo de um determinado sistema
Soluções satisfatórias	Soluções suficientemente adequadas para o contexto em questão. As soluções devem ser viáveis, não necessariamente ótimas
Classes de Problemas	Organização que orienta a trajetória e o desenvolvimento do conhecimento no âmbito da *Design Science*
Validade Pragmática	Busca assegurar a utilidade da solução proposta para o problema. Considerando o custo/benefício, particularidades do ambiente e as reais necessidades dos interessados na solução

Fonte: Adaptadp de Dresch et al. (2015, p. 59)

2.1.1 Protocolo de pesquisa

O protocolo de pesquisa visa apresentar e documentar todas as atividades que o pesquisador pretende realizar durante a sua pesquisa, bem como as percepções e *insights* que surgirem durante a realização da pesquisa.

Segundo Dresch et al. (2015) é fundamental que o documento seja atualizado constantemente, num processo de evolução contínua, para que o pesquisador possa registrar o que ocorreu conforme o esperado e o que teve que ser alterado para garantir o sucesso do trabalho. Além disso, o protocolo precisa ser robusto o suficiente para garantir que outros pesquisadores possam replicar a pesquisa com sucesso. Ou seja, outros interessados em construir ou utilizar o artefato poderão, com o acesso ao protocolo de pesquisa, obter sucesso na sua missão.

No Quadro 2-2 é apresentado as etapas de *Design Science*, suas saídas e onde estão localizadas nesta dissertação.

Quadro 2-2 - Etapas da Design Science

Estágio DSR	Saídas	Capítulo da Dissertação
Identificação do problema	Questão de pesquisa formalizada	1.1
Revisão sistemática da literatura	Revisão sistemática da literatura para busca de artefatos	3
Identificação de artefatos e configuração	Artefatos identificados, classes de problemas estruturados e soluções satisfatórias explicitas	3
Configuração de classes de problemas	Proposta de artefato	5
Projeto do artefato para solucionar o problema específico	Projeto explicitando técnicas e ferramentas para o desenvolvimento e a avaliação do artefato, e detalhamento dos requisitos do artefato	5
Desenvolvimento do artefato	Artefato em seu estado funcional	5
Avaliação do artefato	Avaliação dos resultados	5
Conclusões	Resultados da pesquisa, decisões tomadas e principais limitações	6

Fonte: Elaborado pelo autor com base em Dresch et al. (2015)

2.2 IDENTIFICAÇÃO DOS ARTEFATOS E CONFIGURAÇÃO DAS CLASSES DE PROBLEMAS

2.2.1 Identificação de artefatos utilizando revisão sistemática da literatura

A partir de um problema identificado, teórico ou prático, é necessário identificar quais objetivos ou metas seriam necessárias para que o problema seja considerado satisfatoriamente resolvido. Esse procedimento consiste na “conscientização” e em um primeiro contorno do problema (Dresch et al., 2015).

A partir dessa conscientização, fez-se necessário realizar uma revisão sistemática na literatura, com o objetivo de estabelecer o quadro de soluções empíricas. Esta revisão consiste na busca e identificação dos artefatos que procuram encaminhar soluções ao problema em tela. Esse procedimento é necessário para consolidar os artefatos em classes de problemas para que os conhecimentos anteriores sejam generalizáveis para a solução a ser desenvolvida (Dresch et al., 2015).

Figura 2-1 - Lógica para construção de classes de problemas

Fonte: Lacerda et al. (2013, p. 748).

2.2.2 Classes de problemas

Dresch et al. (2013) definem classes de problemas como a organização de um conjunto de problemas, práticos ou teóricos, que contenham artefatos avaliados ou não, úteis para a ação nas organizações.

Esta organização em classes de problemas permite que os artefatos já desenvolvidos, não sejam apenas uma resposta pontual a certo problema em determinado contexto, mas que os artefatos com soluções satisfatórias possam compartilhar características comuns que proporcionam a generalização e o avanço do conhecimento na área.

2.2.3 Artefatos

Uma vez definidas as classes de problemas, é necessário caracterizar os artefatos associados.

O artefato é a organização dos componentes do ambiente interno para atingir objetivos de um determinado ambiente externo (Simon, 1996). Os artefatos podem ser divididos em: constructos, modelos, métodos, instanciações e *design propositions* (Quadro 2-3).

Quadro 2-3 Classificação dos artefatos

Classificação	Descrição
Constructos	São os componentes mais básicos no desenvolvimento da *Design Science*. São elementos conceituais constituindo uma espécie de vocabulário sobre determinado campo em que tal problema está inserido e que o pesquisador utilizará para evoluir do puramente abstrato para o tangível.
Modelos	São descrições sobre determinado sistema que estabelecem relações entre os constructos previamente definidos. O objetivo dos modelos está na sua utilidade e não na aderência da representação da verdade. Embora um modelo possa ser impreciso sobre a realidade, ele precisa capturar a estrutura geral da realidade buscando assegurar sua utilidade.
Métodos	São conjuntos de procedimentos e ações orientados para o desempenho de determinada tarefa ou solução de dado problema.
Instanciações	São os artefatos que operacionalizam outros artefatos, visando demonstrar a viabilidade e a eficácia dos artefatos construídos. As instanciações nos permitem também avaliar algo importante dentro da proposta da *Design Science*: sua efetividade em relação ao problema proposto ou às melhorias pretendidas no sistema existente.
Design Propositoins	São contribuições teóricas que podem ser feitas a partir da aplicação dos princípios da Design Science diante de um tipo específico de problemas. Correspondem a um *template* genérico que pode ser utilizado para o desenvolvimento de soluções para uma determinada classe de problemas.

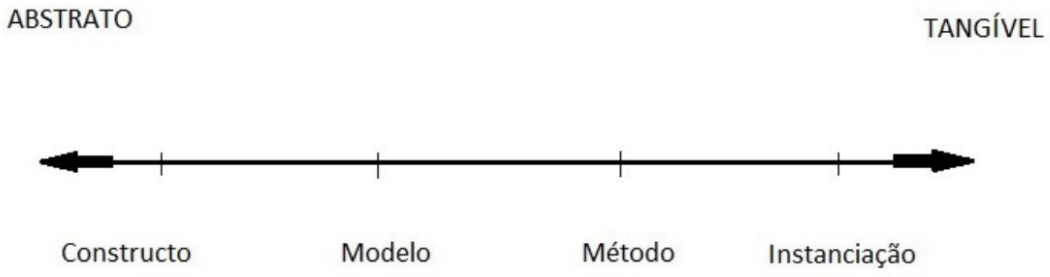
Fonte: Elaborado pelo autor com base em Dresch et al. (2015) e Santos (2018).

Para esta de dissertação a classificação do artefato em instanciação melhor representa o projeto de pesquisa. Santos (2018) define a instanciação como a criação de uma solução tangível que permite avaliar algo importante dentro da proposta de *design science* e sua efetividade em relação ao problema proposto. Este conceito de instanciar é bastante conhecido entre programadores e cientistas da computação e talvez represente o nível mais tangível de solução.

Uma vez definido que nesta dissertação será utilizado a instanciação, é importante salientar que o termo "modelo" também é amplamente utilizado para descrever modelos estatísticos. Senso assim, n

esta dissertação (excluindo o capítulo de metodologia) o termo "modelos" é utilizado referindo-se a modelos estatísticos.

Figura 2-2 Escala de tangibilidade do artefato na DS



Fonte: Santos (2018, p. 24).

2.3 VALIDAÇÃO DO ARTEFATO

Esta dissertação possui características multidisciplinares envolvendo várias áreas do conhecimento e especialização de diversos tipos de profissionais, tais como administração, ciências da computação, redes sociais, telecomunicações, vigilância sanitária, cidades inteligentes, saúde coletiva, dentre várias outras.

A participação de especialistas de domínio é fundamental para o desenvolvimento da pesquisa, não somente na definição dos objetivos e na validação dos resultados, mas em todas as fases da pesquisa para que as informações possam ser utilizadas para eventuais mudanças no artefato e no roteiro de avaliação.

Dresch et al. (2015) sugerem que uma validação rigorosa do artefato não pressupõe o uso de métodos sofisticados, entretanto o rigor implica em cuidados para evitar que algo seja

afirmado ou concluído sem que a pesquisa tenha condições de embasar. No caso da *Design Science Research* rigor implica aumentar a confiabilidade do artefato e de seus resultados.

Nesta dissertação a validação do artefato foi realizada de forma estatística comparando os resultados dos modelos com a realidade observada ao longo da janela temporal estudada.

A avaliação foi baseada em um processo iterativo conforme evolução do artefato em busca dos melhores resultados.

O projeto e construção do artefato desta dissertação são detalhados no capítulo 5.

3 REVISÃO SISTEMÁTICA DA LITERATURA (RSL)

Para responder à questão de pesquisa, foi elaborada uma revisão sistemática da literatura (RSL) baseando-se no processo descrito por Kitchenham (2004) dividido em cinco passos: Definição dos Objetivos da Pesquisa, Estratégia da Pesquisa, Triagem dos Documentos, Extração dos dados e Classificação.

A pesquisa teórica foi realizada nas bases de dados da Scopus, Periódicos Capes, Science Direct, Web of Science e Google Scholar. As palavras-chave utilizadas na busca foram: (“Health Surveillance”) AND (“Social Network” OR “Social Media”) AND (“Machine Learning” OR “Big Data” OR “Artificial Intelligence” OR “predictive model”) AND (”Twitter”). A rede social Twitter foi escolhida como recorte da pesquisa, por ser a única que disponibilizava acesso aos dados de interação entre seus usuários para fins de pesquisa, por meio de uma API (Interface de Programação de Aplicações) oferecida pela empresa. As demais empresas controladoras das redes sociais não oferecem o mesmo acesso aos dados. Foram filtrados documentos escritos nos idiomas inglês, português e espanhol, de domínio do pesquisador. A busca foi realizada em 17 de maio de 2021 resultando em 3.212 documentos (n= 3.212, Fase 1).

O objetivo desta revisão sistemática é obter um mapa com os principais problemas e conceitos abordados no mapeamento de ameaças sanitárias utilizando a rede social Twitter, bem como as técnicas de coleta e análise de resultados utilizadas. Para atingir este objetivo utilizou-se uma planilha em Excel para consolidar as informações de cada artigo da revisão. Foram tabuladas as informações contendo o nome do artigo, autores, doença estudada, data do estudo, abordagem de dados georreferenciados, país do estudo, métodos estatísticos ou de aprendizado de máquinas utilizados e o principal objetivo do estudo.

Como a pesquisa tem por objetivo abordar acontecimentos recentes incluindo a pandemia da COVID-19, foi estabelecido um filtro de data com artigos a partir de 2018 permitindo também analisar como a COVID-19 estava sendo tratado antes e durante a pandemia. Esta aplicação de filtro de datas resultou em 1.845 documentos (n= 1.845, Fase 2). A partir da aplicação dos filtros por palavras-chave e por data de publicação, realizou-se a leitura dos títulos dos documentos avaliando a aderência ao objetivo da RSL que resultou em 457 documentos (Fase 3).

Alinhado ao objetivo de saber como a rede social Twitter vem sendo utilizada para publicar informações e dados referentes à saúde coletiva, foram definidos alguns atributos que

deveriam ser abordados nos artigos selecionados (critérios de qualidade). A identificação geográfica (geolocalização) é um fator relevante para definir áreas de alto risco de contaminação, a identificação por regiões e mapas foi utilizado como critério de inclusão. Foram excluídos artigos que não incluíam uma visualização espacial e temporal das informações.

Para entender os principais modelos de análise dos dados utilizados, foram excluídos artigos que não descreviam os algoritmos de *Machine Learning* utilizados. Os estudos deveriam utilizar dados da rede social Twitter e descrever detalhadamente como os dados foram obtidos e processados.

Após a seleção pela leitura dos títulos, foi avaliada a partir da leitura do resumo (abstract) e da conclusão de cada artigo a aderência aos objetivos da RSL, restando 60 documentos (Fase 4). Realizou-se, então, nova triagem com a leitura atenta e detalhada de cada artigo, resultando em 23 artigos (Fase 5). O Quadro 3-1 apresenta as fases da revisão.

Quadro 3-1 - Fases da RSL

Fase	Critério	Número Artigos
1	Bases = Scopus, Periódicos Capes, Science Direct, Web of Science e Google Scholar Palavras chaves = (“Health Surveillance”) AND ("Social Network" OR "Social Media") AND ("Machine Learning" OR "Big Data" OR "Artificial Intelligence" OR "predictive model") AND (“Twitter”) Idiomas = inglês, português e espanhol	N = 3.212
2	Data publicação: janeiro 2018 a junho 2021	N = 1.845
3	Leitura dos títulos dos documentos relacionados ao objetivo de pesquisa. Foram excluídos artigos que não incluíam visualização espacial e temporal das informações, não descreviam os algoritmos de *Machine Learning* utilizados. Foram considerados somente os artigos que utilizaram a rede social Twitter e que descreviam como os dados foram obtidos e processados.	N = 457
4	Leitura dos resumos e conclusões dos documentos relacionados ao objetivo de pesquisa	N = 60
5	Leitura detalhada dos documentos e seleção dos artigos para referências à pesquisa	N = 23

Fonte: Elaborado pelo autor

3.1 RESULTADOS DA RSL

A triagem dos documentos resultou em 23 artigos aderentes ao objetivo desta revisão sistemática. Para análise dos artigos, primeiramente foi adotado uma abordagem com o uso de técnicas de visualização computacional, foi elaborada uma nuvem de palavras consistindo em uma representação da ocorrência das palavras nos resumos (abstract) dos artigos selecionados. Na Figura 3-1, são representadas as 50 palavras mais citadas, quanto maior o tamanho da palavra maior é a frequência de uso.

Figura 3-1 - Nuvem de palavras dos resumos dos artigos da RSL

Fonte: Elaborado pelo autor

O Quadro 3-2 traz as principais informações obtidas durante a leitura dos textos.

Nota-se que existem estudos com dados de redes sociais para várias doenças. Na maioria (n=16) é analisada uma única doença por artigo, sendo 9 artigos sobre a COVID-19, seguidos por Dengue e Influenza com 3 estudos cada. Os demais artigos (n=7) são de estudos incluindo mais de uma doença. A maior parte dos artigos analisados foi publicada a partir de

2020, o que coincide com o início da pandemia da COVID-19 que trouxe discussões sobre a necessidade de se criar mecanismos de identificação precoce de doenças.

Quadro 3-2 - Objetivo dos artigos selecionados na RSL

#Artigo	*Primeiro Autor*	*Ano*	*Objetivo*
1	Edo-Osagie et al.	2019	Propõe modelo supervisionado para classificação de *tweets* sintomáticos e filtra a relevância.
2	Pruss et al.	2019	Examina com modelo LDA as discussões no Twitter sobre a epidemia de Zikavirus em 2015.
3	Nguyen, H	2021	Desenvolve índice de saúde da população usando Twitter e modelo de redes neurais convolucionais.
4	Hassan Zadeh, Amir	2019	Apresenta modelo de acompanhamento de Influenza utilizando redes sociais e faz correlação temporal.
5	Rashid & Wang	2021	Introdução da ferramenta CovidSens que faz uso das redes sociais para alertas sobre COVID.
6	Souza et al.	2019	Analisa e detecta pontos de risco de Dengue em 2 cidades do Brasil.
7	Martínez, Pascual	2020	Apresenta forma de geolocalização dos tweets com Processamento de Linguagem Natural.
8	Heth et al.	2018	Análise correlação (Pearson) da evolução dos casos semanais de influenza com *tweets*.
9	Li, Lingyao	2021	Desenvolve criação de um sinal de alerta por estado dos EUA usando PLN e ML.
10	AlAgha	2021	Utiliza modelo não supervisionado para análise de sentimento e classificação por tema dos tweets sobre COVID-19.
11	Adriani, M	2020	Desenvolve *framework* para identificar emergências sanitárias pelo Twitter e gera informações em mapas na Indonésia.
12	Mackey T,	2020	Usa ML não supervisionado (BTM) e desenvolve sinais de casos positivos de Covid durante o início da pandemia.
13	Cuomo et al.	2021	Desenvolve modelo com SVM para mapa de risco de contaminação de COVID-19.
14	Marsri, S	2019	Desenvolve 2 modelos calibrados por auto-regressão usando Twitter para estimar casos com 1 semana de antecedência.
15	Fazeli et al., 2021	2021	Apresenta um *framework* para coleta, análise e mapeamento de conteúdos sobre a pandemia de COVID-19.
16	Euzebio, C	2020	Desenvolve algoritmo que analisa uma pequena amostra do Twitter para monitorar Dengue em Ribeirão Preto.
17	Spurlock & Elgazzar	2020	Desenvolve aplicação que agrupa usuários por propensão a contaminação de COVID-19 de acordo com sua rede de contatos nas redes sociais.
18	Jain & Cherikkallil	2018	Apresenta ferramenta Mendinsights que transforma *tweets* em informações para ações na saúde.
19	Sidana et al.	2018	Desenvolve 2 modelos para análise temporal de possíveis doenças analisando informações das redes sociais.
20	Șerban et al.	2019	Descreve *software* criado para monitorar propagação de doenças pelo Twitter.
21	Z. Li et al.	2020	Desenvolve modelo usando redes neurais com dados das redes sociais e dados socioeconômicos.
22	Tufts et al.	2018	Analise a correlação da incidência de 14 doenças na Pensilvânia com menções no Twitter no período de 2012 a 2015.
23	Xavier et al.	2020	Faz análises exploratórias de dados do Twitter e demonstra o potencial das Redes Sociais como ferramenta de vigilância sanitária.

Fonte: Elaborado pelo autor

Para a apresentação dos dados, os artigos utilizaram ferramentas capazes de apresentar os dados em mapas, com variações de cores para diferenciar e evidenciar regiões com maior risco, demonstrando as possibilidades de entender os dados de forma rápida.

Com relação aos modelos utilizados para análise dos dados obtidos do Twitter há uma divisão bem próxima entre modelos supervisionados e não supervisionados. É importante esclarecer que os dados advindos da rede social são classificados como não estruturados, ou seja, contemplam uma mensagem de texto, identificação do usuário e informações complementares (curtidas, compartilhamentos e réplicas). Pela natureza dos dados, para sua manipulação e processamento, em geral são adequadas técnicas de mineração de texto e M*achine Learning*, por isso, a presença desse tipo de algoritmo nos artigos analisados. O exemplo apresentado na Figura 3-2, do trabalho de Rashid & Wang, (2021) traz o tipo de mensagem postada pelos usuários.

Figura 3-2 - Exemplos de postagens no Twitter

Fonte: (Rashid & Wang, 2021)

Os modelos supervisionados, também conhecidos como modelos de aprendizado via exemplos, são modelos que através e uma amostra dos dados contendo os corretos dados de saída são usados como base para o treino do modelo, com estas amostras o algoritmo aprende comparando as saídas oferecidas com os dados de entrada. Uma vez que o modelo passe pela

etapa de treino, são oferecidos novos dados de entrada para que o modelo automaticamente gere a variável de saída (Alzubi et al., 2018).

Nos modelos não supervisionados, os dados de treino não possuem identificação prévia do resultado. O modelo reconhece os padrões que não são facilmente identificados nos dados e definem regras para estes padrões. Esta técnica é bastante utilizada para classificar dados por categoria quando estas classificações não são conhecidas (Alzubi et al., 2018).

Dentre os 23 artigos selecionados, os modelos supervisionados foram utilizados em 8 artigos, sendo que o algoritmo *Support Vector Machine* (SVM) foi utilizado em todos os casos, e em 2 estudos houve a utilização também de outros modelos para fazer comparações de performance. Já os modelos de classificação não supervisionados foram utilizados em 10 estudos, sendo que a maioria utilizou *Latent Dirichlet Allocation* (LDA) (n=5) seguido por *K-means* e *Biterm Topic Models for Short Text* (BTM), ambos com 2 artigos.

O Quadro 3-3 traz as informações de análise observadas nos artigos.

Quadro 3-3 - Matriz de visualização dos artigos

Artigo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Doença																							
Covid					x				x	x		x	x		x		x				x		x
Geral			x															x	x			x	
Dengue						x					x					x							
Zika		x												x									
Influenza				x				x			x												
Malária											x												
Asma	x																						
Diarreia											x												
Elefantíase											x												
Região Geográfica																							
País		x								x							x		x				
Estado			x						x					x								x	
Cidade	x			x	x	x					x	x	x		x	x		x			x		
Fonte Comparação																							
Google Trends									x														
Governo	x	x	x	x		x	x	x	x	x	x	x	x	x	x	x			x			x	x
Visualização no Mapa																							
Ponto		x	x						x			x		x							x	x	
Polígono				x	x	x					x		x		x			x	x				
Modelo Estatístico																							
TF-IDF									x											x			x
Supervisionado																							
LR	x								x														
SVM	x			x					x		x		x					x		x			x
Decision Tree	x								x														
RandomForest																		x					
NaiveBayes	x								x									x		x			
StochasticGradiente Descent																		x					
GradientBoosting																		x					
FFNN																		x					
Não Supervisionado																							
LDA		x								x					x				x			x	
K-Means											x						x						
ATAM																			x				
BTM												x	x										
Redes Neurais			x																				
CNN																				x	x		
LSTM																				x	x		
Contagem palavras							x	x								x							
LCA						x																	
K-S				x																			
Moran (Correlação)				x																			
Regressão Linear														x									
Pearson R				x				x	x			x		x		x							

Fonte: Elaborada pelo autor.

Nota: Term Frequency Inverse Document Frequency (TF-IDF); Logistic Regression (LR); Feed Forward Neural Network (FFNN); Architecture tradeoff analysis method (ATAM); Convolutional Neural Network (CNN); Long short-term memory (LSTM); Life Cycle Assessment (LCA); Kolmogorov-Smirnov (K-S)

3.2 CONCLUSÕES SOBRE A RSL

Seguindo a metodologia do DSR a revisão sistemática teve os objetivos de fazer um levantamento da literatura e estabelecer um quadro de soluções empíricas identificando artefatos que procurem encaminhar soluções ao problema em tela. Com esse procedimento foi possível consolidar os artefatos em classes de problemas para que os conhecimentos anteriores sejam generalizáveis para a solução desenvolvida nesta dissertação que é apresentada no capítulo 5.

A revisão sistemática traz uma importante contribuição na análise da utilização dos dados da rede social Twitter no contexto de monitoramento e alertas precoces de emergências sanitárias. É apresentado uma visão analítica dos principais achados e como o tema vem sendo tratado na literatura, incluindo as técnicas de análise de dados utilizadas e as principais aplicações. Também são analisados como as aplicações podem trazer informações úteis por localidade (georreferenciadas) podendo ser úteis para o desenvolvimento de ferramentas governamentais e particulares para monitoramento e prevenções alertas precoces.

Os estudos que compuseram a revisão evidenciaram diferentes algoritmos analíticos possíveis para a modelagem de dados não estruturados advindos de interações em redes sociais, baseando-se todos eles em postagens textuais. Demonstraram ser possível criar alertas para riscos em saúde, em diferentes doenças. Os algoritmos utilizados contemplam, em geral aprendizado de máquina, com processamento opaco (*black-box*), o que não permite identificar prontamente como foram criadas as regras de classificação dos dados.

O uso dos dados do Twitter em conjunto com técnicas de Processamento de Linguagem Natural (PLN) vem sendo explorados em diversos estudos e pesquisas. A pandemia do COVID-19 e o maior interesse nestas técnicas podem gerar maiores interesses dos estudiosos e ampliar o conhecimento neste campo de estudo.

Desta forma, constata-se que os estudos apresentam abordagens interessantes sobre o uso de dados de redes sociais para a criação de alertas em saúde, sendo esses métodos, passíveis de replicação em outras áreas, contudo, há ainda muitas perguntas que precisam ser respondidas, tanto no nível técnico, quanto no nível ético de sua utilização.

4 REFERENCIAL TEÓRICO

4.1 CIDADES INTELIGENTES E ODS 3

A infraestrutura das cidades é um fator de atratividade no crescente número de pessoas em busca dos benefícios da vida urbana, como resultado as cidades têm enfrentado numerosos desafios na alocação de recursos e na melhoria contínua da infraestrutura. Uma tendência emergente para minimizar os impactos destes desafios está no uso das tecnologias de informações e comunicações (TIC) para auxiliar as cidades a fazer melhor uso de seus recursos. Este conceito é conhecido como Cidades Inteligentes (Ismagilova et al., 2019).

As cidades com funcionalidades efetivas e integradas alavancam soluções como monitoria de tráfego, transporte público, cadeias logísticas, gerenciamento de serviços de saúde, turismo, lazer, resposta a emergências e comércio. Segundo Schuurman et al. (2012), o uso da tecnologia e da participação dos cidadãos num contexto de colaboração digital facilita o desenvolvimento de inovações com ênfase nas pessoas, gerando melhores soluções que a digitalização sozinha pode trazer.

Paralelamente ao avanço dos estudos das cidades inteligentes, em 2015 os membros da ONU (Organização das Nações Unidas) estabeleceram 17 objetivos de desenvolvimento sustentável com metas a serem atingidas até 2030 (Figura 4-1). Estes objetivos abrangem o desenvolvimento econômico, a erradicação da pobreza, da miséria e da fome, a inclusão social, a sustentabilidade ambiental e a boa governança em todos os níveis, incluindo paz e segurança(*ODS Brasil*, 2021).

Figura 4-1 – Objetivos de desenvolvimento sustentáveis

Fonte: *ODS Brasil* (2021).

Dentre os 17 objetivos, o objetivo número 3 refere-se a assegurar uma vida saudável e promover o bem-estar para todos. Duas de suas metas são destacadas para o propósito desta dissertação:

- Até 2030, acabar com as epidemias de AIDS, tuberculose, malária e doenças tropicais negligenciadas, combater a hepatite, doenças transmitidas pela água, e outras doenças transmissíveis.
- Reforçar a capacidade de todos os países, particularmente os países em desenvolvimento, para o alerta precoce, redução de riscos e gerenciamento de riscos nacionais e globais de saúde.

4.2 REDES SOCIAIS E SITUAÇÕES DE CALAMIDADE, PANDEMIAS E EPIDEMIAS

As informações disponíveis nas redes sociais são terreno fértil para a criação de novas ideias para gerar sistemas de alertas precoces em vários cenários. O entendimento das informações online criadas em um determinado lugar e horário são subsídios que podem ser incorporados em sistemas de alertas precoces (L. Li et al., 2021).

Na literatura é possível encontrar alguns trabalhos relacionados com a utilização de redes sociais em situações de calamidade, pandemias e epidemias, como os apresentados a seguir.

Por meio de análise espaço-temporal das informações georreferenciadas, o estudo apresentado por Hassan Zadeh et al. (2019), sobre o uso de dados georreferenciados das redes sociais, monitora as informações sobre o vírus influenza. Esse estudo mostrou uma correlação relevante entre a frequência das postagens relacionadas ao vírus e a quantidade de pacientes reportados oficialmente nos Estados Unidos.

Masri et al. (2019) analisam informações do Twitter durante o surto de Zika Vírus em 2016, apresentando dois modelos de regressão que foram calibrados para estimar a quantidade de casos na semana seguinte.

No Brasil, Euzebio et al. (2020) apresentaram um estudo sobre o monitoramento da dengue por meio de postagens no Twitter na cidade de Ribeirão Preto. Nesse estudo, observou-se que o número de comentários no Twitter sobre a dengue acompanha o crescimento de números de casos oficiais da doença indicando a existência de uma significante correlação entre os rumores sobre a dengue e o aumento do número de casos notificados.

Adriani et al. (2015) estudaram a possibilidade de detectar a ocorrência das principais doenças na Indonésia a partir da classificação de posts das redes sociais. Nesse estudo são abordadas formas de identificar o tipo da doença a partir dos tweets e como pode ser visualizada em mapas a disseminação das doenças.

Li et al. (2021) exploram o potencial do uso dos dados da rede social Twitter para melhorar os alertas sobre a pandemia do COVID-19, o trabalho conduz um estudo investigando mais de 14 milhões de tweets postados de 01 janeiro de 2020 a 10 março de 2020. Com o uso de processamento de linguagem natural e técnicas de M*achine Learning* é criado um sinal de alerta que, segundo os autores, poderia antecipar a detecção do risco em 16 dias

4.3 REDE SOCIAL TWITTER

Com a popularização do acesso à internet, as redes sociais estão entre as principais plataformas em número de usuários. Dados estimados de julho de 2021 indicam que a principal rede social, o Facebook, tinha cerca de 2,5 bilhões de usuários. O Twitter teria cerca de 400 milhões de usuários ativos, com quase 17,2 milhões de usuários no Brasil (Statista, 2021).

Comparando o Twitter com o Facebook e Instagram, outras ferramentas de rede social de grande alcance, com o Twitter existe a possibilidade de um usuário (*Follower*) seguir outros usuários (*Followed*) sem a necessidade de reciprocidade, o que permite uma divulgação mais acelerada de comentários. Além desta característica é possível responder a um comentário (*Reply*), permitir que o usuário reencaminhe (*Retweet*) um determinado comentário para os usuários seguidores, o que permite a disseminação de informação além do alcance do *tweet* original. Este mecanismo de relacionameno entre os usuários (*Follower* e *Follwed*), acrescido do encaminhamento de comentários (*Retweet*) torna o uso do Twitter uma ferramenta poderosa para a disseminação de informações (Kwak et al., 2010).

4.4 DADOS GEORREFERENCIADOS NO TWITTER

Análises automáticas dos dados gerados pelos usuários das redes sociais envolvem a extração das coordenadas geográficas dos posts para identificação de local com a possibilidade de plotar em um sistema de geolocalização em mapas (Martínez & Pascual, 2020)

Diversos estudos utilizando o Twitter têm enfatizado o uso de posts georreferenciados com a localização via GPS fornecida pelos smartphones (Dredze et al., 2013). Associar os

comentários ou tweets sobre diversos assuntos ao local de origem dos usuários do Twitter possibilita efetuar análises espaciais interessantes.

No Twitter, a informação da localidade pode ser inserida manualmente no perfil do usuário ou por meio da habilitação do dispositivo GPS, o qual fornece as coordenadas geográficas latitude e longitude. Entretanto, Dredze et al. (2013) relatam que aproximadamente 1,2% dos tweets possuem coordenadas de localização. No mesmo trabalho os autores apresentam outras formas primárias de se obter informações de localização, utilizando o campo de descrição da localidade dos usuários (cidade ou país), sendo possível incrementar a informação de localização para 22,3% dos posts.

Na literatura, diferentes abordagens vêm sendo utilizadas com o objetivo de identificar a localidade do usuário do Twitter. Martínez e Pascual (2020) fazem uso de modelos baseados no conteúdo dos comentários que permite extrair nomes de localidades ou frases associadas às localidades. No mesmo estudo é extraída a localidade a partir dos comentários e por meio da utilização de processamento de linguagem natural, em que as palavras são classificadas em sujeito, predicado e objeto. Assim, o conteúdo do objeto permite extrair nomes de localidades e a localidade também pode ser extraída do texto do comentário utilizando como fonte de referência a base de dados geográfica *GeoNames*.

4.5 PROCESSAMENTO DE LINGUAGEM NATURAL (PLN)

O advento das redes sociais proporcionou às pessoas, em tempo real, compartilhar seus próprios conteúdos, ideias, opiniões virtualmente para milhões de pessoas conectadas. Entretanto, este volume enorme e crescente de informações, em sua maioria, é não estruturado devido à especificidade de ser produzida para o consumo humano e por consequência não é diretamente processada por máquinas. O processamento de linguagem natural (PLN) é área de técnicas computacionais que estuda as análises automáticas e representação da linguagem humana pelos computadores (Cambria & White, 2014).

No contexto relacionado à COVID-19, Li et al. (2021) analisaram manualmente 5.000 comentários do Twitter classificando os tweets contendo sinais de propagação da doença. Verificou-se que 13,7% dos posts tinham informações contendo tais sinais. Utilizando a classificação desses 5.000 tweets como dados de treinamento, aliados a técnicas de *Machine Learning* e Processamento de Linguagem Natural os pesquisadores desenvolveram um algoritmo que automaticamente analisou 14,7milhões de tweets coletados de 20 janeiro a 10 de

março de 2020 demonstrando o potencial que as opiniões públicas via redes sociais têm de gerar alertas precoces.

Em outro estudo relacionado à COVID-19, Mackey et al. (2020) coletaram 4,5 milhões de tweets durante 17 dias em março de 2020. Neste estudo foi possível também identificar comentários relacionados a sintomas, resultados de testes laboratoriais ou outros temas relacionados ao contato com o vírus pelos usuários.

Analisando diversos estudos, há evidências de que existem oportunidades de pesquisa na análise de dados do Twitter durante a pandemia de COVID-19 no território brasileiro com aplicação de Processamento de Linguagem Natural. Além disso, por meio de métodos computacionais, essas informações podem ser acessadas em tempo real pelos gestores e profissionais da saúde (Xavier et al., 2020). As pesquisas com dados de redes sociais podem ir além de estudos retrospectivos e gerar produtos de apoio à tomada de decisão em tempo real. Algumas possíveis aplicações com Processamento de Linguagem aplicada a redes sociais podem incluir:

- Análise da opinião sobre medidas adotadas (análise de sentimentos);
- Avaliação do impacto das estratégias de comunicação;
- Identificação de possíveis sintomas relacionados às doenças e antecipação de surtos;
- Identificação e avaliação do impacto de *fake news*.

4.6 MODELAGEM DE TÓPICOS

Para estabelecer um conjunto de dados que possibilite a classificação e agrupamento de tweets semelhantes, a técnica escolhida baseou-se no levantamento feito na revisão sistemática de literatura que, seguindo a metodologia do DSR, consiste na busca e identificação dos artefatos que procuram encaminhar soluções ao problema em tela. Esse procedimento é necessário para consolidar os artefatos em classes de problemas para que os conhecimentos anteriores sejam generalizáveis para a solução a ser desenvolvida (Dresch et al., 2015).

Desta forma optou-se por utilizar técnicas de modelagem de tópicos que está presente em diversos artigos analisados na revisão sistemática (capítulo 3).

A modelagem de tópico é uma técnica de aprendizado de máquina não supervisionada para descobrir tópicos que ocorrem no agrupamento de documentos (Blei et al., 2003). Os tópicos gerados pela modelagem são misturas de palavras estatisticamente representativas do comportamento de ocorrências equivalentes nos textos.

Tópicos são temas que os documentos discutem ou falam sobre. Uma boa modelagem de tópico é aquela que pode produzir tópicos distintos com as mesmas palavras atribuindo pesos diferentes para cada palavra.

O treinamento do modelo é um processo iterativo que, conforme a quantidade de documentos e demais parâmetros, pode demandar grande processamento de máquina e diversas horas, dias ou até semanas.

A técnica de modelagem de tópico mais utilizada nos artigos analisados na revisão sistemática é o *Latent Dirichlet Allocation* (LDA) desenvolvido por Blei et al., (2003).

Segundo (Vayansky & Kumar, 2020) o LDA é um algoritmo muito utilizado para treinar modelos de tópicos e funciona procurando a repetição de ocorrências de palavras contidas no mesmo documento. O LDA assume que as palavras que aparecem no mesmo documento são normalmente pertencentes ao mesmo tópico, e para cada documento que contém as mesas palavras são considerados que contém os mesmos tópicos.

Com relação a modelagem de tópicos usando dados do Twitter, cada tweet é tratado como um documento, e pode ser visto como um conjunto de diferentes tópicos com diferentes ponderações dependendo da frequência que os termos aparecem (AlAgha, 2021).

Dentre os diversos parâmetros do LDA, o número de tópicos que o modelo deve gerar é um parâmetro essencial. Caso seja definido uma quantidade pequena é provável que os tópicos sejam muito genéricos, em contrapartida se o número de tópicos for muito grande irá resultar em tópicos muito individualizados e confusos (Murzintcev Nikita, 2015).

Sendo assim, nesta dissertação, foram criados vários modelos LDA variando o número de tópicos comparando a qualidade de cada um para a definição do melhor parâmetro. Este processo iterativo é adotado no artigo de AlAgha (2021) e está de acordo com a metodologia de DSR.

5 PROJETO E DESENVOLVIMENTO DO ARTEFATO

Após a conscientização do problema, revisão sistemática da literatura, criação de quadro com soluções empíricas satisfatórias com características comuns, foi desenhado uma solução baseada na organização e agrupamento dos artefatos já desenvolvidos e analisados, permitindo que os achados desses estudos pudessem ser aplicados no contexto desta dissertação.

5.1 PROJETO DO ARTEFATO

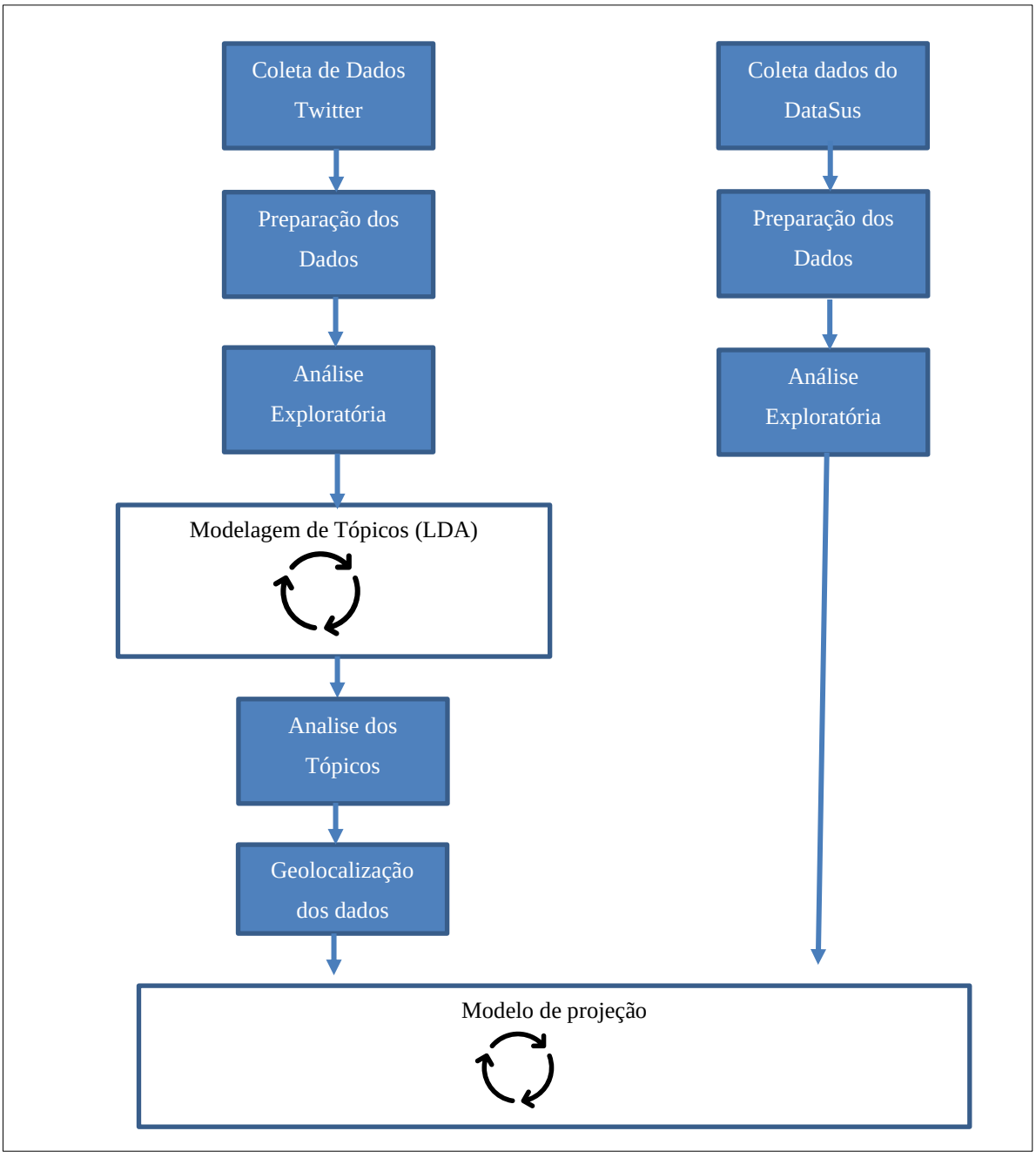
A Figura 5-1 retrata a arquitetura da solução adotada neste trabalho. Vários passos foram realizados sequencialmente, incluindo coleta de dados, preparação dos dados, análise de volumes, modelagem de tópico, análise de series temporais e análise dos resultados.

A coleta dos dados envolve a extração dos dados das plataformas Twitter e DataSus, na etapa de preparação envolve vários tratamentos nos dados para ser possível analisá-los e gerar melhores resultados nas modelagens.

Análise exploratória tem por objetivo estabelecer uma visão geral e os atributos dos dados coletados. Modelagem de tópico é utilizada para identificar os principais tópicos e agrupamentos dos textos por temas. Os tópicos também são analisados através do tempo e entre os principais estados do Brasil para fornecer *insights* envolvendo localização e datas diferentes. As implicações de cada método de análise serão abordadas e colocadas em contexto nesta dissertação

A criação do modelo de projeção avalia se os posts classificados do Twitter têm a possibilidade de ajudar no acompanhamento e na evolução dos casos.

Figura 5-1 Arquitetura da solução



Fonte: Elaborado pelo autor

5.2 DADOS DA REDE SOCIAL TWITTER

5.2.1 Coleta de dados

5.2.1.1 Forma de coleta de dados

Para a coleta dos dados, o autor deste projeto criou um *script* em linguagem Python, que coleta dados de postagens no Twitter através da *Application Programming Interface V2* (API) disponibilizada pela plataforma (Twitter, 2021). Por padrão, só é disponibilizado os *posts* do Twitter dos últimos 7 dias dificultando em muito trabalhos que precisam analisar informações com um histórico bem maior.

Entretanto, para a realização desta pesquisa, após pedidos formais e garantindo que os dados seriam utilizados para fins acadêmicos, o autor desta dissertação obteve uma conta de usuário acadêmico para a plataforma Twitter com acesso a *tweets* desde 2006, o que facilitou a análise dos efeitos a serem abordados.

Como limitação, a plataforma Twitter permitia, na data da coleta de dados, baixar até 10milhões de tweets a cada 30 dias para o usuário acadêmico.

5.2.1.2 Filtros utilizados para a busca

Para a busca dos *tweets* foram aplicados filtros por palavras-chave, listadas no Quadro 5-1, com termos relacionados a COVID-19 e principais sintomas.

Procurou-se, também, colocar diferentes variações de palavras relacionadas à COVID-19, visto que usuários se referem à doença de diferentes formas. Para a definição dos sintomas foram utilizados os termos disponíveis no *site* da Organização Mundial da Saúde (*WHO*, 2021).

Quadro 5-1 Palavras-chave utilizadas

Grupo	**Palavras**
Nome da doença	COVID-19, coronavírus, corona, covid, covid19
Sintomas	Febre, tosse, sem paladar, sem olfato, falta de ar, dor, diarreia
Outros	Pandemia, epidemia

Fonte: elaborado pelo autor.

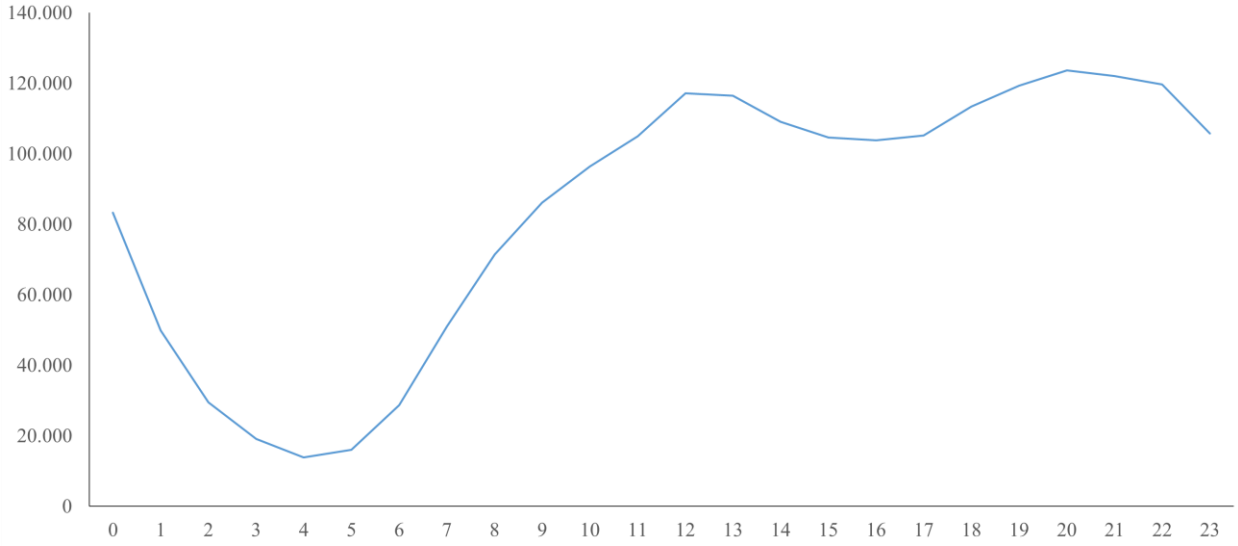
5.2.2 Volume de dados coletados

Dada a regra da licença acadêmica que permite baixar até 10milhões de tweets em 30 dias e ao grande volume de posts, foi selecionada uma faixa horária para que fosse possível fazer uma boa amostra dos dados de janeiro de 2020 até abril de 2022.

Para a definição dos horários, primeiramente foram obtidos dados de 9 dias de março de 2021 para entender os principais volumes por horários conforme a Tabela 5-1 e apresentado em forma de gráfico na Figura 5-2. Com essa informação foi possível notar que existem 2 picos diários sendo um às 12:00 e outro próximo às 20:00, esses picos podem estar relacionados ao período de maior volume de noticiários e programas de rádio e televisão e aos horários de folga dos usuários, ou seja, períodos em que não estão trabalhando.

Também é possível observar que entre os picos mencionados há um patamar um pouco mais estável de posts, sendo assim, para este estudo optou-se por capturar os dados de janeiro de 2020 a abril de 2022 compreendendo a faixa horária das 16:00 às 18:59. Esta opção foi adotada para evitar possível viés no volume de tweets causados pela divulgação de notícias nos programas de rádio e televisão.

Figura 5-2 - Número de tweets por horário durante 9 dias de março/21



Fonte: Elaborado pelo autor com dados obtidos do Twitter

Tabela 5-1 - Número de tweets por horário durante 9 dias de março/21

Horário	01/mar	02/mar	04/mar	05/mar	06/mar	07/mar	08/mar	09/mar	Total
0	9.363	8.948	12.848	11.190	10.849	9.981	9.604	10.503	83.286
1	5.236	4.702	7.599	6.827	6.548	6.905	5.571	6.438	49.826
2	3.064	2.767	4.436	4.087	4.047	4.617	3.228	3.173	29.419
3	2.093	1.884	2.808	2.707	2.622	2.966	2.046	1.924	19.050
4	1.601	1.482	2.150	1.900	1.780	2.066	1.535	1.318	13.832
5	2.163	1.622	2.548	2.775	1.832	1.827	1.666	1.589	16.022
6	4.017	3.038	4.707	5.138	2.954	2.469	3.267	3.027	28.617
7	6.942	5.278	8.936	8.704	4.795	4.064	6.401	5.848	50.968
8	9.228	7.372	12.212	11.645	7.899	5.957	8.667	8.452	71.432
9	10.759	9.037	13.592	13.335	9.832	8.651	10.894	10.026	86.126
10	11.468	9.921	15.950	13.821	11.802	11.127	11.574	10.697	96.360
11	12.316	10.623	17.545	15.031	14.033	12.327	12.494	10.582	104.951
12	12.215	13.667	17.410	16.693	18.596	13.182	14.231	11.120	117.114
13	12.206	13.842	17.330	17.028	19.099	13.031	13.669	10.214	116.419
14	11.805	12.430	17.911	14.735	16.836	13.307	13.294	8.677	108.995
15	10.828	11.413	18.850	13.170	15.779	12.739	13.565	8.276	104.620
16	**11.231**	**11.396**	**17.946**	**13.073**	**13.733**	**13.076**	**14.243**	**9.140**	**103.838**
17	**11.126**	**13.545**	**18.827**	**14.692**	**13.055**	**12.523**	**12.623**	**8.827**	**105.218**
18	**11.961**	**15.199**	**18.072**	**16.356**	**13.916**	**13.767**	**12.360**	**11.749**	**113.380**
19	12.633	15.647	19.648	16.128	15.126	14.335	12.750	13.036	119.303
20	12.787	20.052	19.999	16.776	14.889	14.429	12.409	12.287	123.628
21	13.302	19.948	20.018	16.549	13.550	13.942	12.523	12.162	121.994
22	14.514	20.088	17.804	14.876	12.704	15.537	12.491	11.608	119.622
23	11.846	19.798	15.622	12.839	11.605	12.008	11.147	10.818	105.683
Total	**224.704**	**253.699**	**324.768**	**280.075**	**257.881**	**234.833**	**232.252**	**201.491**	**2.009.703**

Fonte: Elaborado pelo autor com dados obtidos do Twitter

5.2.3 Preparação dos dados

A preparação dos dados é um passo essencial antes de iniciar o processamento. Uma boa preparação resultará na redução de dados inconsistentes e aumento na qualidade dos resultados.

Os posts no Twitter (tweets) possuem características particulares fazendo com que a preparação tenha etapas diferentes que um texto tradicional. Os tweets têm tamanho limitado (140 caracteres na sua maioria) e muitos contêm caracteres especiais como *hashtags, URLs, emoticons, @ e usernames.* Os tweets foram preparados para a modelagem de análise de tópicos aplicando os seguintes passos:

Conversão de *emojis*: *Emoticons* são bastante utilizados para expressar sentimentos, sendo assim, seus significados devem permanecer, mas para isso é necessário a conversão de símbolos em palavras, para isso foi utilizada a biblioteca EMOJI (Kim & Wurster, 2022) que contêm um dicionário com a maioria dos *emoticons* e seus significados. Por exemplo: 😊 , 😕 foram substituídos pelas palavras “feliz” e “triste” respectivamente.

Limpeza: Os usuários do Twitter comumente utilizam símbolos. Esses símbolos, no contexto desta pesquisa, possuem pouca relevância para as análises dos textos comparado com o ganho computacional de mantê-las. Sendo assim, na etapa de limpeza removeu-se as seguintes partes: Nome de usuários (*usernames*), URLs e *hashtags* (#).

Tokenização: a tokenização é o ato de simplificar os tweets e prepará-lo para os outros estágios de processamento. Durante esse processo, pontuações e caracteres especiais são completamente removidos e uma frase é dividida em palavras ou tokens individuais (Daniel Jurafsky & Martin, 2021).

Como os textos dos tweets apresentam características específicas que os diferenciam dos textos tradicionais, para este trabalho, optou-se por utilizar um *tokenizador* especialmente desenvolvido para tratar tweets, textos informais e conversas online. Neste trabalho foi utilizado o *tokenizador* presente na biblioteca NLTK (Bird et al., 2009).

Remoção de stop words e palavras-chaves: As *stopwords* são palavras que aparecem com muita frequência e que têm pouco sentido e pouco significado. A remoção das *stopwords* gera uma melhora significativa no processamento computacional com pequeno impacto na análise dos textos. Para este trabalho foi utilizada a lista de *stopwords* em português disponível na biblioteca NLTK (Bird et al., 2009).

Lematização: A *lematização* é o processo de reduzir as palavras à sua forma raiz conhecido como *lema.* Por exemplo, as palavras “correr”, “corre”, “correu” são formas da palavra “correr”, portanto, “correr” é o lema de todas as palavras anteriores. Neste trabalho foi utilizado a biblioteca *spaCy* disponível em Python que pode executar a lematização na língua portuguesa (Honnibal, 2023).

Criação de Bigramas: uma das limitações do LDA é a geração de tópicos baseado no agrupamento de palavras únicas, ou seja, palavras que normalmente são usadas na sequência conferindo um significado importante podem perder esta característica. Por exemplo, termos como "passaporte sanitario ", "efeito colateral" e ‘usar mascara’’ as palavras serão apresentadas de forma separadas e utilizadas na modelagem do LDA. Para endereçar esta limitação, foi

utilizado o processo de formação de bigramas utilizando a biblioteca *Gensim* disponível em *Python* (Řehůřek, 2022). Foram detectados os bigramas (sequência de 2 palavras) que ocorreram juntas no mínimo 100 vezes. Para colocar as palavras juntas, foi substituído o espaço pelo caractere *underscore,* ou seja, termos como "passaporte sanitário " e "efeito colateral" foram convertidos para "passaporte_sanitario " e "efeito_colateral". Dessa forma esses termos foram tratados como uma palavra única no processamento da modelagem de tópico LDA.

Após aplicar todos os passos do pré-processamentos citados o tamanho final da base de dados foi de 10.497.219 tweets.

5.3 MODELAGEM DE TÓPICO

A modelagem de tópico é uma técnica de aprendizado de máquina não supervisionada para descobrir tópicos que ocorrem no agrupamento de documentos (Blei et al., 2003). Dessa forma os tópicos gerados são agrupamento estatísticos de palavras conforme ocorrências equivalentes nos tweets.

Nesse capítulo é apresentado como o classificador de tweets foi criado. Utilizando técnica de modelagem de tópicos num processo iterativo para obtenção do melhor número de tópicos, foi possível classificar os tweets em 10 tópicos e analisar a evolução ao longo do tempo.

5.3.1 Quantidade de tópicos ideais

Seguindo a metodologia de DSR, num processo de evolução contínua, é fundamental que o documento seja atualizado constantemente registrando o que ocorreu e não ocorreu conforme o esperado para garantir o sucesso do trabalho (Dresch et al., 2015).

Sendo assim, para a definição da quantidade de agrupamentos dos tweets (tópicos), foi realizada a geração de 4 modelos de classificação (LDA) distintos. Cada um com uma quantidade de tópicos diferente para a determinação do parâmetro mais adequado.

Para a avaliação foi considerada a métrica de coerência descrita por Mimno et al., (2011) presente na biblioteca *Gensim* em *Python* (Řehůřek, 2022)

Segundo Mimno et al., 2011, um tópico com um bom indicador de coerência contém palavras que são relacionadas umas com as outras, isto é, palavras que são mais prováveis de aparecerem no mesmo tweet. Em contrapartida, um tópico com um indicador ruim contém

palavras que geralmente não aparecem juntas no mesmo tweet. Após calcular o índice para cada tópico, a performance do modelo total é calculada pela média do indicador dos tópicos.

Para todas as quantidades de tópicos, foram aplicadas as mesmas etapas de preparação dos dados conforme capítulo 5.2.3 .

Na Tabela 5-2 são apresentados os resultados de coerência para cada quantidade de tópicos, pode-se notar que o indicador de coerência modifica conforme o número de tópicos é alterado, indicando que a performance da modelagem de tópico atinge seu pico em 10 tópicos, sendo este o parâmetro escolhido.

Tabela 5-2 - Índice de coerência para escolha do número de tópicos

# Tópicos	Índice Coerência
5	0,306
10	0,337
15	0,293
20	0,259

Fonte: Elaborado pelo autor

5.3.2 Geração do Modelo de classificação

O Quadro 5-2 e o Quadro 5-3 mostram os resultados da modelagem de tópicos apresentando as 30 principais palavras por tópico com base em suas probabilidades. Para ilustrar graficamente, na Figura 5-3 é apresentado cada tópico com nuvens de palavras, sendo o tamanho de uma palavra proporcional à probabilidade da palavra dentro do tópico.

Como todas as palavras possuem pesos diferentes em cada tópico, é normal que sejam encontradas muitas palavras que apareçam em tópicos diferentes.

Nota-se que alguns tópicos incluem frases com mais de uma palavra como "passaporte_sanitario ", "efeito_colateral" e ‘usar_mascara”. Essas frases aparecem devido a etapa de pré-processamento de criação de bigramas (descrito no capítulo 5.2.4). Sem esse passo só seriam gerados tópicos de palavras únicas. A criação de bigramas proporciona um significado mais preciso e consequentemente uma melhor modelagem de tópico.

Quadro 5-2 – Principais palavras dos topicos gerados pelo LDA(1/2)

Tópico 0	Tópico 1	Tópico 2	Tópico 3	Tópico 4
Palavra	Palavra	Palavra	Palavra	Palavra
'covid'	'quarentena'	'vacinar'	'ano'	'dia'
'morte'	'primeiro'	'tomar'	'costa'	'bolsonaro'
'dizer'	'mundo'	'dose'	'morrer'	'mês'
'governo'	'fazer'	'pessoa'	'mãe'	'matar'
'causa'	'entrar'	'sim'	'sofrimento'	'vírus'
'perder'	'acontecer'	'deus'	'coração'	'presidente'
'doença'	'obrigar'	'pfizer'	'saúde'	'mulher'
'causar'	'antes'	'existir'	'cara'	'estudo'
'contar'	'anunciar'	'poder'	'único'	'aplicar'
'dever'	'gostar'	'receber'	'mostrar'	'rosto_chorando_aos_berro'
'médico'	'motivo'	'comprar'	'insuportável'	'afirmar'
'morrer'	'anvisa'	'povo'	'idade'	'internar'
'risco'	'obrigatório'	'chamar'	'população'	'rosto_de_palhaço'
'reação'	'segundo'	'rir'	'colocar'	'hoje'
'poder'	'menos'	'amanhã'	'público'	'imunizar'
'ódio'	'importante'	'Brasil'	'foto'	'sono'
'atenção'	'local'	'passaporte'	'uso'	'menino'
'impedir'	'fechar'	'hoje'	'homem'	'rosto_implorar'
'ainda'	'shakiro_desemprego'	'comprovante'	'estar'	'luto'
'criar'	'depressão'	'agora'	'tristeza'	'relação'
'pessoal'	'caraio_corno'	'pagar'	'descobrir'	'medida'
'dado'	'lidar'	'terceiro_dose'	'aprovar'	'testar'
'exigir'	'distanciamento_social'	'funcionar'	'realmente'	'passaporte_sanitário'
'informar'	'leo_recife'	'ter'	'caro'	'desgraçar'
'além'	'teclado_celular'	'reclamar'	'aumentar'	'calor'
'coronavíru'	'antir'	'defender'	'chato'	'prova'
'sirene'	'colapso'	'distanciamento'	'junto'	'rirrir'
'sus'	'total'	'paladar'	'jovem'	'mandar'
'paz'	'documento'	'precisar'	'eficácia'	'vítima'
'servir'	'maluco'	'conhecer'	'imagem'	'proibir'

Fonte: Elaborado pelo autor

Quadro 5-3 - Principais palavras dos topicos gerados pelo LDA(2/2)

Tópico 5	Tópico 6	Tópico 7	Tópico 8	Tópico 9
Palavra	Palavra	Palavra	Palavra	Palavra
'dar'	'ficar'	'dor'	'covid'	'criança'
'gente'	'poder'	'cabeça'	'caso'	'querer'
'saber'	'momento'	'sentir'	'novo'	'falar'
'hoje'	'aqui'	'achar'	'pandemia'	'deixar'
'fazer'	'fazer'	'vez'	'pegar'	'fazer'
'grande'	'dizer'	'tanto'	'registrar'	'tempo'
'bom'	'filho'	'vida'	'chegar'	'usar'
'tudo'	'pai'	'passar'	'pessoa'	'nada'
'passar'	'saber'	'acabar'	'óbito'	'ficar'
'coisa'	'isolamento'	'febre'	'variante'	'imaginar'
'tão'	'falar'	'morrer'	'testar_positivo'	'ninguém'
'ainda'	'querer'	'chorar'	'efeito'	'ver'
'bem'	'país'	'querer'	'máscara'	'vir'
'sair'	'pessoa'	'corpo'	'confirmar'	'teste'
'ter'	'amigo'	'nunca'	'início'	'tosse'
'assim'	'semana'	'sinto'	'hospital'	'forte'
'tomeir'	'casa'	'nada'	'realizar'	'esquecer'
'mal'	'brasileiro'	'barriga'	'hora'	'problema'
'febre'	'apenas'	'ver'	'ver'	'vídeo'
'agora'	'bom'	'remédio'	'infectar'	'meio'
'poder'	'fim'	'peito'	'falso'	'pouco'
'alguém'	'sofrer'	'número'	'estado'	'dentro'
'eu'	'precisar'	'chegar'	'positivo'	'escola'
'viver'	'último'	'garganto'	'usar_máscara'	'nao'
'esperar'	'ouvir'	'mano'	'médico'	'poder'
'mau'	'ajudar'	'lembrar'	'devido'	'direito'
'ontem'	'seguir'	'aguento'	'proteger'	'paciente'
'pessoa'	'família_contigo'	'cheio'	'efeito_colateral'	'aqui'
'ver'	'onde'	'pqp'	'curar'	'verdade'
'braço'	'lugar'	'preciso'	'acordo'	'ainda'

Fonte: Elaborado pelo autor

Figura 5-3 – Nuvem de palavras dos tópicos gerados

Fonte: Elaborado pelo autor

5.3.3 Classificação dos tweets

O LDA gera uma matriz de tópicos e documentos. Nesta dissertação, os documentos são os tweets, dessa forma é possível mapear para cada tweet a probabilidade de pertencer a cada tópico.

Depois de definido e gerado o modelo de classificação, para cada um dos 10 milhões de tweets foi atribuído o tópico com maior relevância, resultando na distribuição da Tabela 5-3

Tabela 5-3 - Distribuição percentual de tweets por tópicos

Tópico	**%**
Tópico 0	8,8%
Tópico 1	7,2%
Tópico 2	11,4%
Tópico 3	5,2%
Tópico 4	5,9%
Tópico 5	13,3%
Tópico 6	14,0%
Tópico 7	11,2%
Tópico 8	12,1%
Tópico 9	10,9%

Fonte: Elaborado pelo Autor

A modelagem de tópico forma os tópicos com agrupamento estatístico de palavras, podendo gerar tópicos difíceis para a interpretação e classificação humana. Para auxiliar nesta limitação, para cada tópico foi levantado os 5 tweets com a maior probabilidade de pertencer a cada tópico. Estes tweets são apresentados no Quadro 5-4

Quadro 5-4 - Principais tweets para cada tópico

Tópico	Tweets
0	Via @estadao: Jogos Olímpicos de Tóquio serão adiados por causa do coronavírus, diz jornal - https://t.co/nNgiMtqPyl
0	@derflecha Se morrer por bala perdida diminui a estatística de morte por COVID-19 https://t.co/rErFosgtjo
0	RT @UOLNoticias: Governo diz que ainda não há relato de impacto por coronavírus nas exportações https://t.co/df389zR3F7
0	RT @RevistaEpoca: Igreja evangélica que promete 'imunização' com bênção contra coronavírus pode ser enquadrada por charlatanismo, diz MP
0	OMS decreta coronavírus como emergência sanitária global. Surto é "sem precedentes" https://t.co/ZGV0zNbLeY
1	RT @tracklist: Azealia Banks, em seus stories, comentou sobre os posicionamentos anti-vacina de Nicki Minaj: \n\n“Faz cirurgias perigosas
1	RT @MarcosRogerio: Mais uma vez, Emanuela Medrades desmente os irmãos Miranda sobre a data de criação das invoices que tratam da vacina
1	RT @jairbolsonaro: - "Não desistam do que é certo e do que é necessário fazer...\n- Ou a cobrança pela fraqueza e desistência será maior
1	RT @BlogdoNoblat: Que se denuncie Bolsonaro em tribunais internacionais por crime contra a Humanidade. É isso o que ele pratica
1	calça de moletom e crocs tem sido meu look da quarentena
2	RT @pretademaiss: A Pfizer endereçou uma carta ao Brasil IMPLORANDO pela compra de vacina. A Pfizer precisou AVISAR que isso ia conter a pandemia
2	RT @tracklist: Anahi, Poncho, Maite Perroni e Christian Chávez incentivaram a campanha de vacinação contra a COVID-19.
2	RT @Anitta: Hoje vou tomar a terceira dose da vacina. Será que é agora que viro jacaré?
2	Duas pessoas que eu amo tomaram a vacina hoje #VivaOSuS
2	RT @SICNoticias: Terceira dose da vacina? Alemanha admite que pode ser necessária https://t.co/DC4xMVb8KD
3	o quão saudável vc é?\nBronquite ☑\nAsma ❌\nRinite ☑\nIntolerância a lactose ❌\nProblema de visão☑\nAlergia a poeira☑\nDor no joelho ❌\nDor nas costas ☑\nAlergia ao calor/Frio ☑\nAlergias de pele ❌\nDor no coração❌\nPressão baixa ❌\nPressão alta ❌\ngastrite ☑ (refluxo) https://t.co/6CDqD2uenT

3	RT @ludmilagrilo: O show da Claudia Leitte ontem em SP já pode ser considerado como o marco do retorno à normalidade?
3	Humorista de “A Praça é Nossa” morre de COVID-19 aos 39 anos\nhttps://t.co/iIfmPf9DpR
3	RT @jairbolsonaro: - O desempenho das contas públicas é resultado da combinação do crescimento de receitas próprias com o auxílio financeiro
3	RT @JornalOGlobo: Cobrança de 'passaporte da vacina' falha em algumas entradas do Sambódromo https://t.co/Xpp8znYCjE
4	RT @obsaludasturias: 🏔ASTURIAS 🗓 04/01 (23:59) 👇\n\n🍇 135 positivos\n👴 46 positivos >=65a\n\n💾 10 ingresos, 14 altas\n💉 2958 PCR, 164 Ag\n🪦 2 fall…
4	RT @Rconstantino: Jornalistas torcendo abertamente para que Bolsonaro e sua equipe tenham coronavírus: liberdade de expressão\nPresidente da…
4	RT @obsaludasturias: 🏔ASTURIAS 🗓 10/03 (23:59) 👇\n\n🍇 96 positivos\n👴 23 positivos >=65a\n\n💾 35 ingresos, 27 altas\n💉 3775 PCR, 216 Ag\n🪦 4 falle
4	RT @senadorhumberto: Bolsonaro é como um vírus que ataca o Brasil. E, aos poucos, ele está matando o seu hospedeiro. \nhttps://t.co/eNtBge9h
4	RT @BrazilFight: URGENTE-GUEDES ESCLARECE MARCO AURÉLIO \n"Não fomos nós que cortamos o Censo do Orçamento. Quem aprovou o corte foi o Congresso
5	gente tenho plena certeza que ja perdi uns 2kg só por diarreia pq eu to há 1 semana cagando igual uma doida
5	RT @Consterna_: você conseguiu ser tudo,\nem todas as coisas;\na minha maior vontade,\nàquela saudade.\nmeu grande amor\ne minha pior dor
5	Pelo jeito vc n tem medo desse vírus né — o coronavírus nunca me assustou, eu já peguei coisa pior e ainda chamei de amor https://t.co/EqTnPR7qnV
5	Tô passando mal é como se a dor fosse em mim na boa quem dúvida de coisas assim é monstro igual quem fez
5	RT @FatosEx: “Passei mal demais hoje a tarde, tive até febre...”\n\nMeu sintoma: https://t.co/rtS5QaQVLA
6	@loud_dudstheboy o que é ser loud? não é apenas hype, é o amor, uma família que está contigo independente do que for, tanto em momentos felizes quanto em momentos de dor, o peito arde, o sangue vibra no olhar de cada torcedor. profunda, mais do que uma org que não se afunda. #goLOUD 💚

6	RT @gigattf: Gigi era jogadora de basquete. E ia para alguns jogos com o pai - depois que ele se aposentou. Ele falou sobre ela no Kimmel
6	@TelaineFreitas @erlan_bastos @ObsFamosos @UOLtvefamosos É o certo. Teve contato com doente, deve ficar em isolamento
6	@NetflixBrasil @emicida @Anitta @bridgerton Já tô querendo saber o que tem pra janeiro em especial aos brasileiros que vão ter que ficar em casa sem vacina.
6	RT @rauqmp3: quem ta fazendo enem pelo menos vai poder ficar feliz vendo todas as notícias da vacina
7	Mano que dor do inferno na minha coluna, tô chorando de dor pqp, nunca senti tanta dor
7	RT @Brunozor: mano muito doido a ciencias e a medicina né\n\neu tava com uma dor insuportavel na garganta febre e parece que isso nunca ia acabar
7	RT @catarinajbb: toda vez q eu começar a chorar eu vou lembrar da dor de cabeça q eu fico e vou parar na hora
7	Porra eu nunca senti tanta dor na minha vida inteira pqp que inferno
7	minha cabeça só parou de doer agora, quase chorei de tanta dor pqp
8	Bahia registra 1.955 novos casos de COVID-19 e 20 óbitos nas últimas 24 horas - https://t.co/YZQ3geP3kI\n\n#PNoticias #News #Noticias #Ultimasnoticias #Jornalismo #PiatãFM #RadiodaGente #coronavirus #Sesab #Bahia
8	O boletim epidemiológico desta sexta (11) registra 30.246 casos ativos de COVID-19 na Bahia. Nas últimas 24 horas, foram registrados 8.659 casos de COVID-19 (taxa de crescimento de +0,60%), 8.382 recuperados (+0,61%) e mais 49 óbitos.\nAcesse o boletim: https://t.co/SqOT7fPYJC
8	Alagoas registra dois óbitos e 43 novos casos de COVID-19 em 24 horas. Outros 1.152 casos suspeitos estão em investigação epidemiológica\n\nLeia mais\nhttps://t.co/cDQdL8COrh
8	Uberlândia registra mais de 170 casos positivos e não tem novo óbito no boletim desta sexta https://t.co/7OPUGgNyap #G1TriânguloMG
8	COVID-19: Mais 13 novos casos são registrados em Matozinhos \nA atualização do boletim do Coronavírus (COVID-19) desta terça-feira, 29, registra mais 13 casos confirmados em Matozinhos e um óbito. Os positivos d.....\nAcesse...\nhttps://t.co/wy2GbP406u
9	RT @Gus_Moreira: Facada, Glenn, Porteiro, Queiroz, Droga no avião, óleo na praia, Fogo na Amazônia, arroz, Leite Condessado, Sérgio Morno

9	RT @MazitaChaves: Olha aí quem sabe as verdades sobre Michel Temer! \n\nMas em nome de salvar vidas e em busca da vacina, vamos esquecer
9	RT @taoquei1: Pra quem tá falando por aí que o vídeo da Rainha foi editado de forma maldosa por "Bolsonaristas" que incluíram a foto da ive
9	Segundo a minha teoria falta pouco para sermos dominados por completo, falta so vir a vacina
9	Não toque em nada, fique contra o vento, use máscara , mantenha o distanciamento. https://t.co/5kJvmqs5W8
9	@pjgotic vamos todos ficar em isolamento e fazer o teste

Fonte: Elaborado pelo autor

Devido ao grande volume de posts com opiniões políticas, vários agrupamentos apresentam comentários sobre governantes incluindo comentários de apoio e críticas dificultando ainda mais a interpretação humana de cada tópico.

Mesmo com estas limitações, analisando a nuvem de palavras (Figura 5-3) e os principais tweets (Quadro 5-4), foram inferidas as principais características dos tópicos e atribuído uma nomenclatura conforme abaixo:

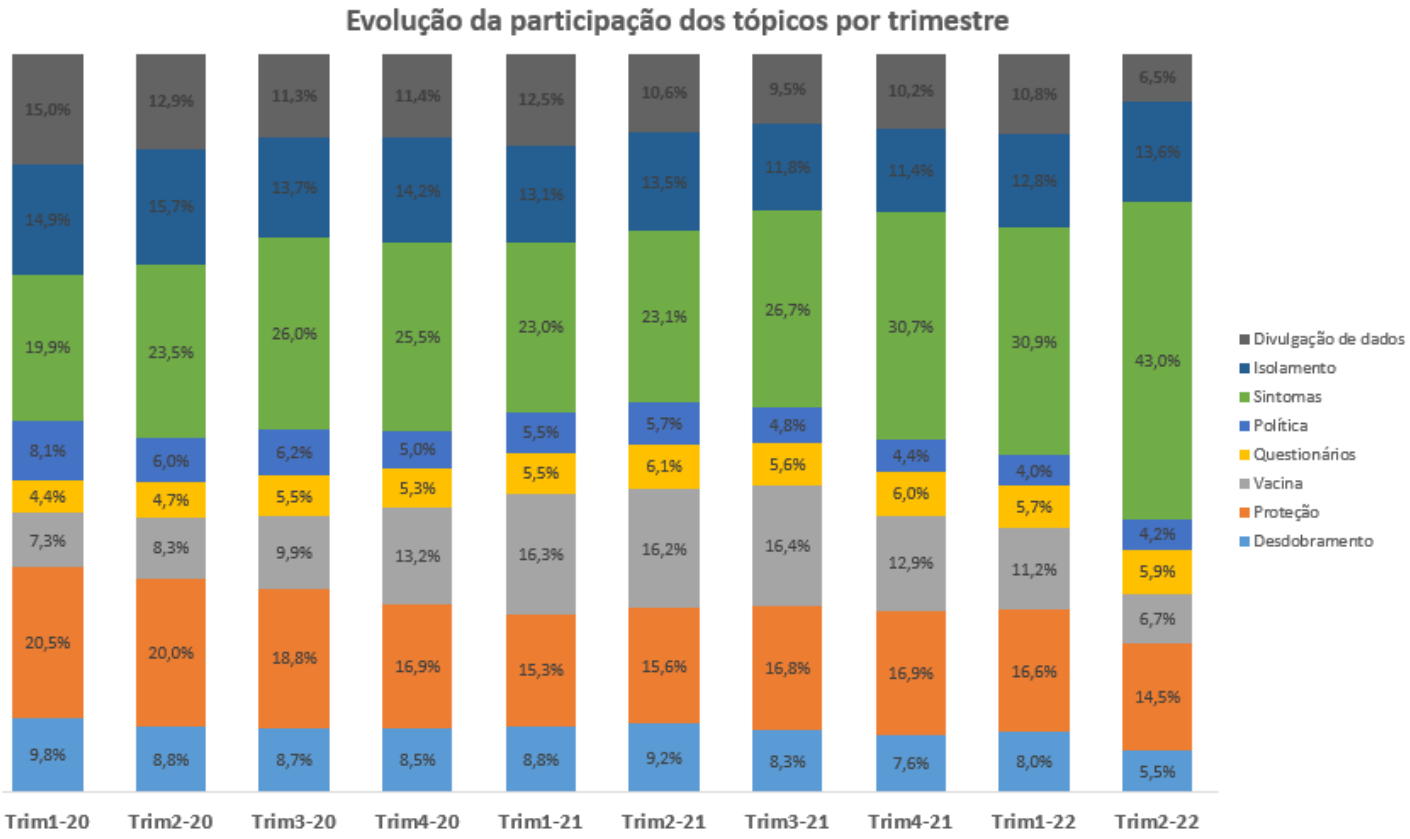
- **Desdobramentos:** O tópico 0 apresenta tweets mais relacionados aos desdobramentos e consequências da pandemia.
- **Proteção:** Nos tópicos 1 e 9 existe concentração de posts e palavras relativas a medidas de proteção a contaminação como quarentena, distanciamento social e “fazer testes”.
- **Vacina:** No tópico 2 foi notado a presença da palavra vacina com bastante relevância, assim como posts sobre negociações de doses de vacinas.
- **Questionários**: No tópico 3 estão agrupados tweets com questionários sobre a saúde e hábitos das pessoas.
- **Política:** Tópico 4 inclui palavras como “bolsonaro”, “presidente”, e posts usados principalmente no contexto de opiniões sobre os políticos e sobre as políticas governamentais relativas ao combate a COVID-19.
- **Sintomas**: Tópicos 5 e 7 estão mais relacionados a sintomas da doença como “passar mal”, “dor de cabeça” e “febre”.
- **Isolamento:** O tópico 6 apresenta palavras mais relativas ao convívio durante o período de isolamento como “ficar”, “momento”, “família_contigo” e “filho”
- **Divulgação de dados:** Tópico 8 agrupa tweets contendo a divulgação de dados como boletins informativos para atualizar a população sobre o andamento da pandemia. Entre os

principais tweets é possível observar tweets específicos de prefeituras de pequenas e medias cidades.

5.3.4Análise estatística temporal

O principal objetivo da análise temporal é explorar como o volume de posts sobre cada tópico mudou ao longo do tempo e as possíveis razões por trás da mudança. A Figura 5-4 mostra um gráfico com a evolução temporal do percentual de tweets pertencentes a cada tópico por trimestre.

Figura 5-4 - Evolução da participação dos tópicos por trimestre



Fonte: Elaborado pelo autor

Como mostrado, a distribuição dos tópicos foram mudando gradualmente com o tempo. No início da pandemia foi quando os tópicos de isolamento e divulgação de dados (tópicos 6 e 8) representaram maior quantidade, o que pode demonstrar preocupação com as estatísticas (casos e mortes) e as medidas de isolamento que entravam em vigor naquele momento.

Nota-se que no final de 2020 até meados de 2021 o tópico de vacina (tópico 2), teve um aumento de participação. Esse pode refletir que as redes sociais estavam comentando sobre o início das compras e aplicações da vacina.

No último trimestre de 2021 e nos 2 primeiros de 2022 os tópicos relacionados a sintomas da doença apresentaram um maior crescimento (tópicos 7 e 5). Esse comportamento pode evidenciar que as redes sociais estavam refletindo o momento de maior número de novos casos diários da pandemia.

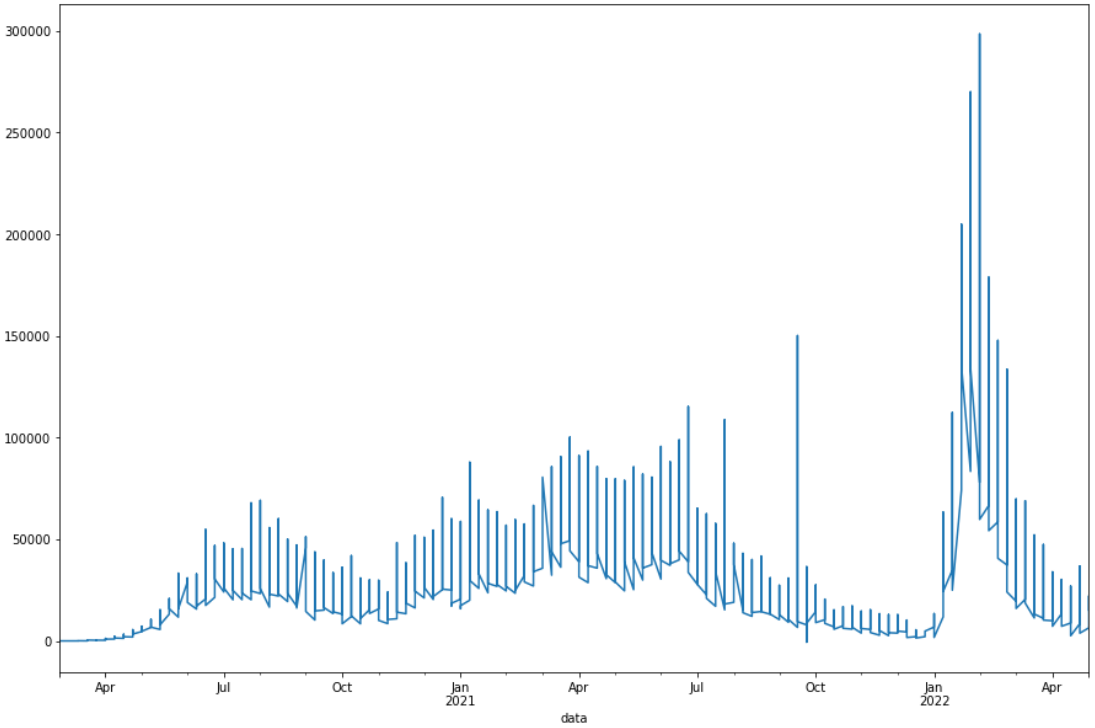
5.3.5 Levantamento dos dados oficiais de COVID-19

Os dados oficiais de casos de COVID-19, no Brasil, de janeiro de 2020 a abril de 2022 foram obtidos no painel de casos e óbitos Covid-19 existente no site covid.saude.gov.br que é disponibilizado pelo Sistema Único de Saúde brasileiro (*DataSus*, 2022) e foram baixados em formato .csv. Os dados foram utilizados para correlacionar a quantidade de casos com a quantidade de tweets por tópicos classificados via modelo LDA para validar a hipótese de pesquisa.

Analisando os casos diários (Figura 5-5) é possível notar que existem alguns dias principalmente nos finais de semana, com informações de menos casos e picos nos dias uteis seguintes refletindo um *report* acumulado dos casos. Como o objetivo deste trabalho é correlacionar com postagens nas redes sociais, que não possuem esta limitação, optou-se por consolidar os dados por semana (Figura 5-7).

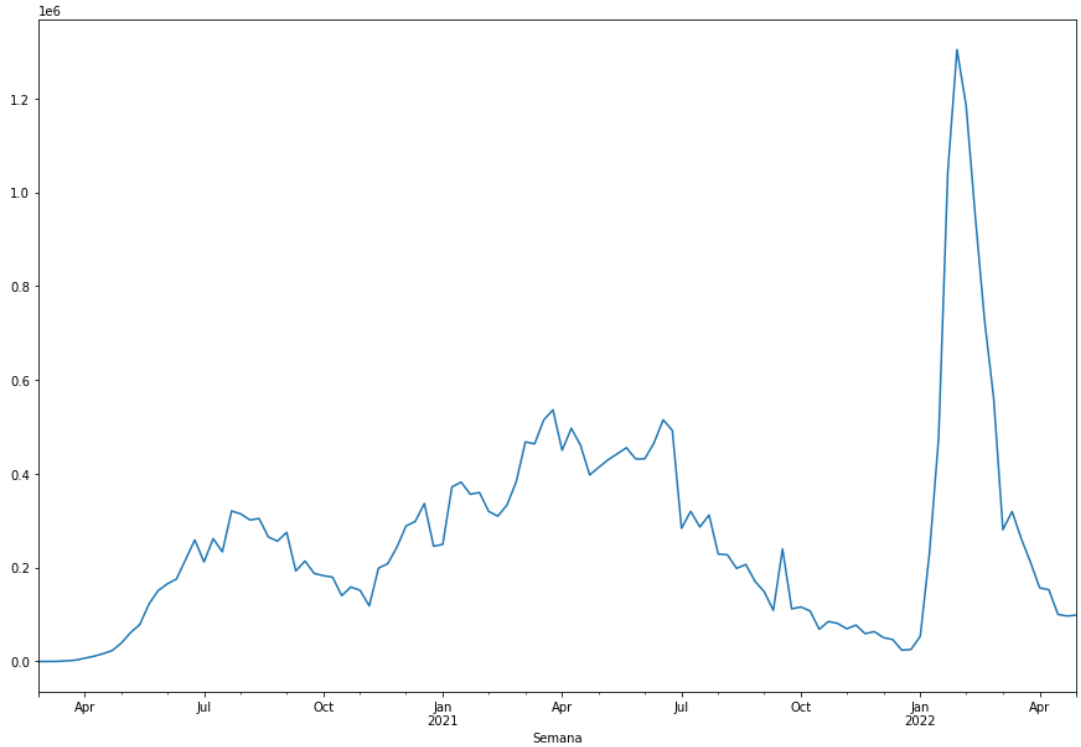
Conforme abordado no capítulo 5.3.4 quando é analisado o número de casos novos de COVID-19 nos gráficos presentes na FiguraFigura 5-7 nota-se que o pico acontece no primeiro trimestre de 2022 que é o mesmo momento que os tópicos relativos a sintomas (tópicos 5 e 7) tem maior participação, isso pode ter relação com o fato de que as redes sociais estavam com mais posts relativos a sintomas da doença.

Figura 5-5 Evolução de casos diários de COVID-19



Fonte: (*DataSus*, 2022)

Figura 5-6 Evolução de casos semanais de COVID-19 (Milhões)

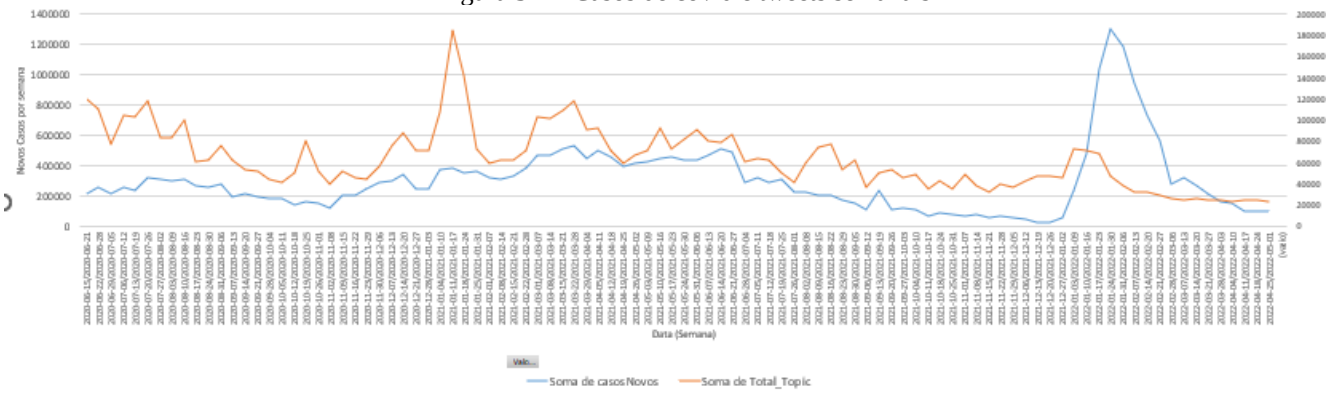


Fonte: (*DataSus*, 2022)

5.3.6 Análise dos volumes dos casos de COVID-19 e tweets

Analisar o volume de tweets é um primeiro passo essencial para explorar os dados e pode fornecer alguns resultados úteis por si só. Também pode nos ajudar a entender melhor e contextualizar os resultados da modelagem de tópicos. A Figura 5-7 mostra o número de tweets e casos por semana coletados, o gráfico apresentado indica que é possível observar que a tendência de longo prazo do volume de tweets tem comportamento similar a evolução dos casos de covid. Mas a variação semanal possui comportamentos muitas vezes distintos.

Figura 5-7 - Casos de covid e tweets semanais



Fonte: Elaborado pelo autor com dados do DataSus e Twitter

Para avaliar estatisticamente a correlação dos tweets com os casos semanais, foram produzidos coeficientes de correlação de Pearson entre o volume total de tweets e os casos de COVID-19 em diferentes momentos.

Análise de correlação Pearson apontou índice de 0,4 em 2020, em 2021 o índice foi de 0,7 e em 2022 o índice observado foi de 0,39. Demonstrando que a correlação linear direta entre volume total de tweets e quantidade total de casos pode não ser adequada para o objetivo deste trabalho.

Nos próximos capítulos serão apresentados modelos correlacionando os diferentes tópicos gerados pela modelagem de tópico (LDA) e a evolução temporal conforme a arquitetura da solução proposta no capítulo 5.1.

5.3.7 Análise Espacial dos casos de COVID-19 e dos tweets coletados

No Twitter, a informação da localidade pode ser inserida manualmente no perfil do usuário ou por meio da habilitação do dispositivo GPS que fornece as coordenadas geográficas de latitude e longitude.

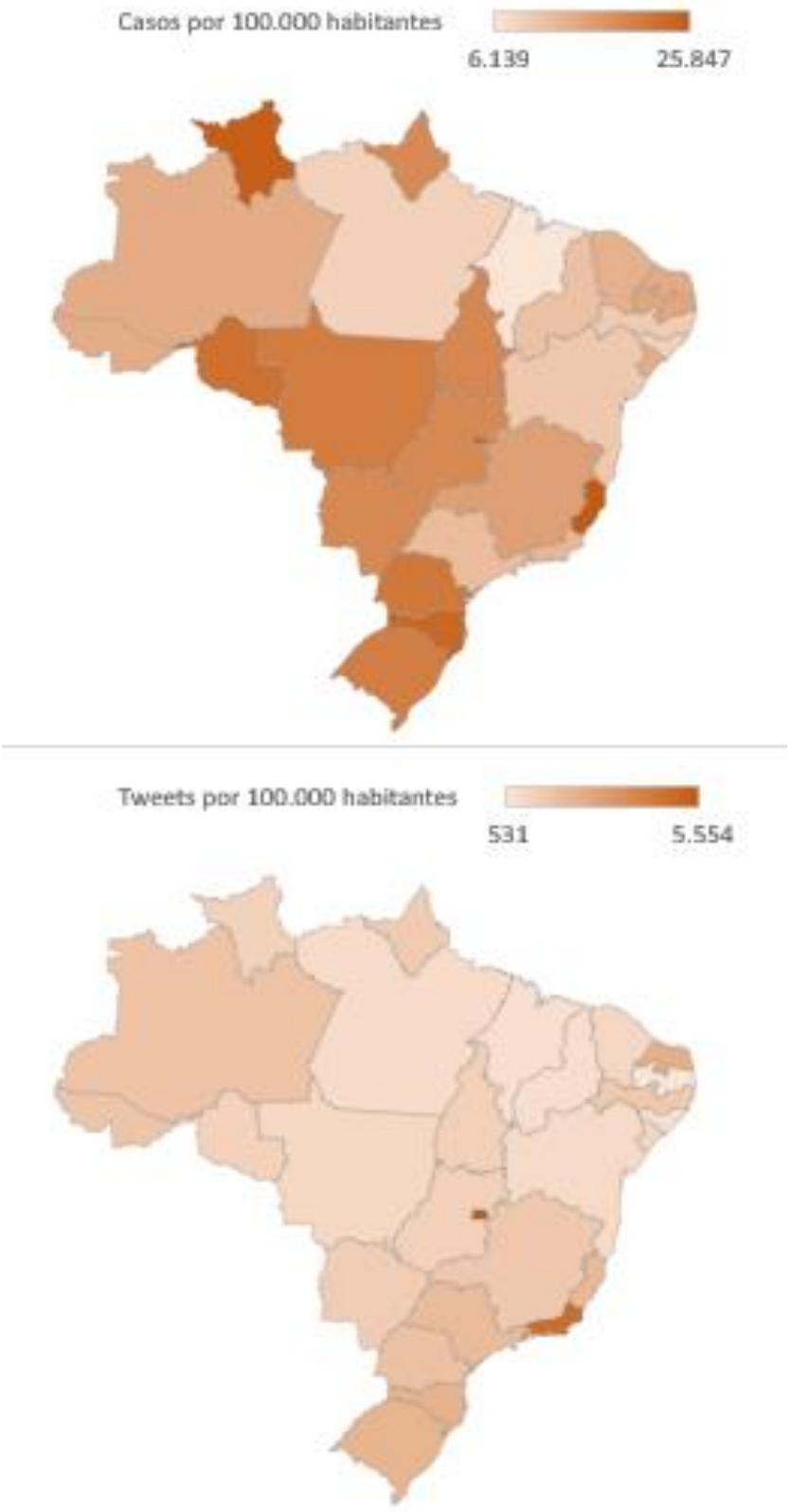
Nos tweets coletados neste trabalho, 2,21% possuem localização pela informação de coordenadas no momento que foi feito o tweet. Incluindo os tweets feitos por usuários que tinham descrição de localidade em seu perfil, a informação de localização chega a 59,4%. A limitação desta informação é que o usuário pode ter sido incorreto na descrição da localidade ou não estava no mesmo local no momento do post. Mesmo com esta limitação, devido ao ganho expressivo do volume de tweets (59,4% versus 2,2%), neste trabalho foi adotado esta informação principalmente na analise por estado, além disso o uso deste tipo de informação foi abordado nos estudos obtidos na RSL (Capítulo 3) em especial no trabalho de Dredze et al.,(2013).

Considerando os tweets e os dados de casos com identificação por estado, na Figura 5-8 existem mapas por estado com o total de casos de COVID-19 e o total de tweets para cada 100.000 habitantes.

Analisando a correlação de Pearson entre as duas variáveis (casos e tweets por 100.000 habitantes) dos estados foi obtido um o coeficiente de 0,28. Como também pode ser notado, os estados com maior prevalência de casos (cores mais escuras) não apresentam uma semelhança tão direta ao número de tweets (cores mais escuras).

Como apresentado no capítulo 4.6, a modelagem de tópicos tem o objetivo de buscar um modelo de projeção mais preciso, separando os tweets em grupos semelhantes e possibilitando a utilização dos grupos mais relevantes ao objetivo desta dissertação.

Figura 5-8 - Comparação dos casos acumulados de COVID-19 e número de tweets ajustado por cada 100.000 habitantes



Fonte: Elaborado pelo autor

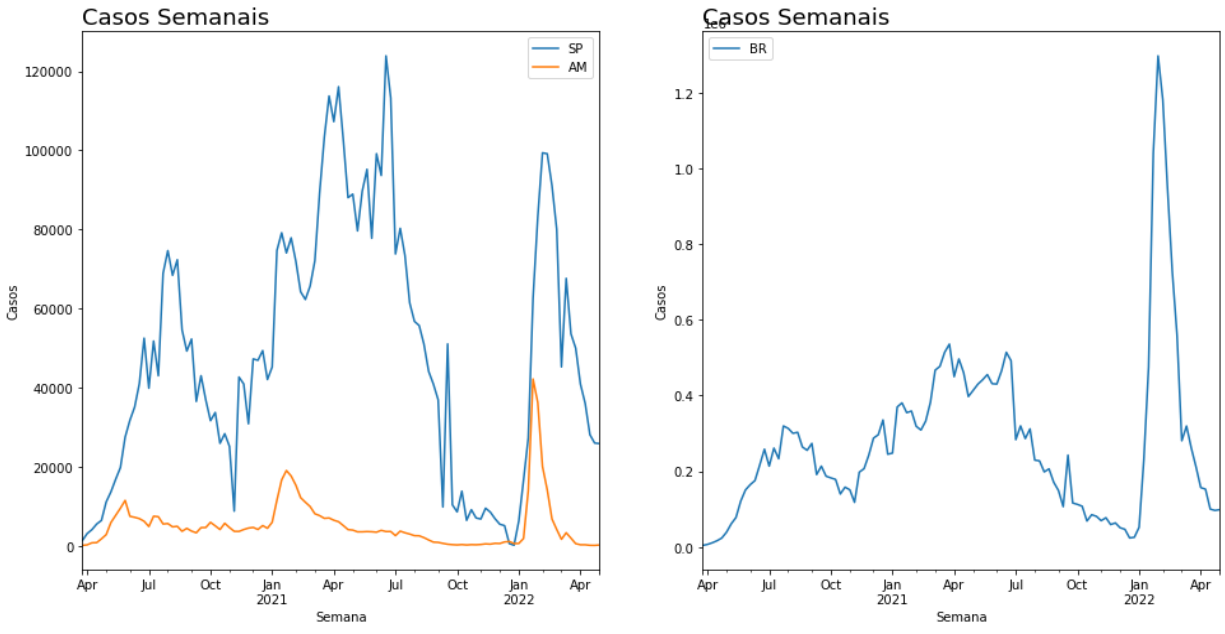
5.3.8 Desenvolvimento dos modelos de projeção

Para a análise e criação dos modelos de projeção foram utilizados os dados de janeiro de 2021 a abril de 2022, período que compreende o crescimento e queda dos casos durante a segunda e terceira onda da pandemia.

Foram desenvolvidos 3 modelos distintos, sendo um para o Brasil composto pelo total de casos semanais e tweets no país e, para a criação de modelos utilizando dados geolocalizados foram criados outros 2 modelos com os casos e tweets identificados no estado de São Paulo e por fim um outro modelo para o estado Amazonas.

A escolha destes dois estados (São Paulo e Amazonas) foi devido a São Paulo apresentar a maior população do Brasil, e por consequência o maior volume de tweets e casos de COVID-19 e por ter o maior pico de casos em meados de 2021 diferente do pico no Brasil como um todo. O estado de Amazonas foi escolhido também por ter um comportamento na curva de casos um pouco diferente do Brasil e diferente de São Paulo, principalmente por ter um pico no número de casos no início de 2021, com grandes consequências para o sistema de saúde local, fato amplamente noticiado na ocasião. Esses comportamentos podem ser observados nos gráficos da Figura 5-9.

Figura 5-9 - Evolução do número de casos BR, SP e AM



Fonte: (*DataSus*, 2022)

Para examinar a relação entre os casos semanais de COVID-19 e os tweets de semanas anteriores, foi realizada uma correlação dos dados. Na Tabela 5-4 é apresentado o valor de correlação dos casos com a defasagem (atraso) no total de tweets em todo o Brasil. Observa-se que os tweets relativos a 1 semana anterior aos casos possuem uma correlação de 0,35 sendo superior à da semana dos casos ou de semanas anteriores. Esta informação é um indício de que dados de semanas anteriores podem ser úteis para sinalizar variações nos casos da semana seguinte.

Tabela 5-4 – Correlação dos casos de COVID-19 com # tweets defasados

Posts	**Pearson R**
Semana dos casos	0,26
1 semana anterior	0,35
2 semanas anterior	0,33
3 semanas anterior	0,31
4 semanas anterior	0,28
5 semanas anterior	0,26

Fonte: Elaborado pelo autor

Sendo assim, o desenvolvimento dos modelos baseou-se na utilização combinada dos casos de COVID-19 e a quantidade de tweets por tópicos de uma e duas semanas anteriores. Especificamente foi aplicado um modelo autorregressivo usando contagens de tweets por tópicos semanais anteriores e contagens semanais anteriores de casos de COVID-19.

Antes do desenvolvimento do modelo de projeção, a diferenciação de primeira ordem foi aplicada a variáveis dependentes e independentes. Esta é a prática padrão para abordar a questão da estacionariedade que é comum aos dados de séries temporais. Após a diferenciação, foram examinados vários modelos usando defasagens de 1 e 2 semanas tanto para a variável autorregressiva (casos de COVID-19) quanto para cada tópico dos tweets, de acordo com a seguinte equação geral:

Equação 1 - Equação do modelo de séries temporais

$$C'_t = \alpha C'_{t1} + \beta C'_{t2} + \sum_{k=0}^{k=9} (\gamma_k T'_{k\,t1} + \mu_k T'_{k\,t2}) + \theta$$

Onde:

C_t' é a diferença entre a quantidade de casos na semana t e a semana t-1 (diferença de primeira ordem);

α é a estimativa do efeito do total de casos de COVID-19 1 semana (t1) antes de t após a diferenciação de primeira ordem;

β é a estimativa do efeito da contagem semanal de casos de COVID 2 semanas(s) (t2) antes de t após a diferenciação de primeira ordem;

γk é a estimativa de efeito do total de tweets do Tópico k, 1 semana antes (t1) de t após a diferenciação de primeira ordem;

μk é a estimativa do efeito do total de tweets do Tópico k, 2 semanas antes (t2) de t após a diferenciação de primeira ordem

Θ é a interceptação de regressão e o termo de erro.

Para a definição do modelo adotado foram realizados os seguintes passos com os resultados apresentados na Tabela 5-5.

- Passo 1: Análise do modelo com dados dos casos e dos tópicos defasado uma e duas semanas
- Passo 2: Eliminação da variável com menor contribuição para o modelo (p de maior valor)
- Passo 3: Caso alguma variável tenha p>0,1 retornar ao passo 1

Este processo iterativo foi realizado até obter-se o modelo com o melhor R-quadrado, e com todas as variáveis apresentando $p < 0,1$.

5.3.9 Modelos de projeção escolhidos

O modelo escolhido para prever a contagem de casos de COVID-19 no Brasil foi modelo de autorregressão incluindo dois termos para a quantidade de casos de COVID-19 (de 1 e duas semanas anteriores) e 8 outros termos selecionados aplicados em tópicos dos tweets entre 1 e 2 semanas anteriores conforme a Tabela 5-5. Nessa mesma tabela estão: o peso de cada variável, os valores de p e o indicador de multicolinearidade VIF. O valor de R-quadrado sobre as variações dos casos entre semanas de COVID-19 observadas versus as previstas pelo modelo também é apresentado.

Da mesma forma na tabela também são apresentados os modelos escolhidos para o estado de São Paulo e o estado Amazonas.

Tabela 5-5 - Resultados dos modelos preditivos de casos de COVID-19

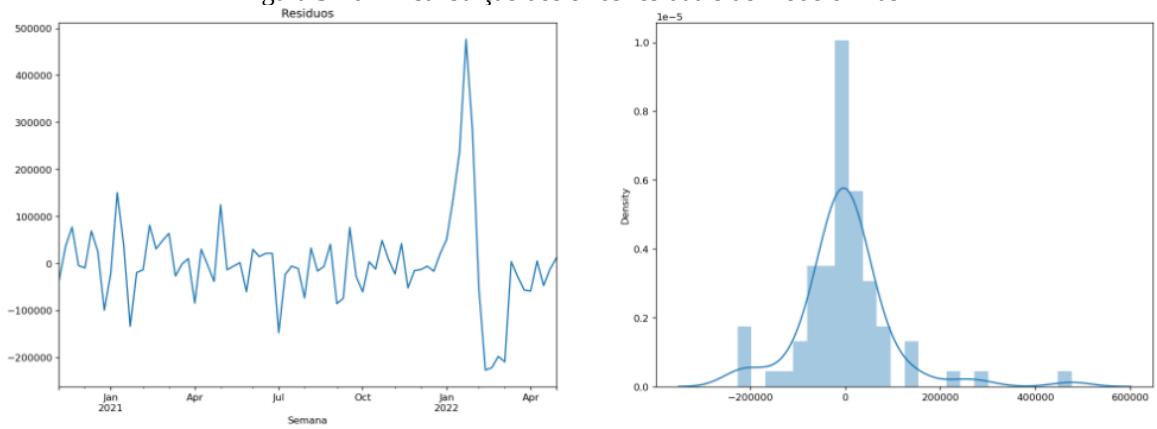
	Variável	Coeficiente	Valor p	VIF	R2 do Modelo
Modelo Brasil					0,90
	const	1232,10	0,99	1,48	
	C’t1	0,67	0,00	1,49	
	C’t2	-0,17	0,13	2,78	
	T0’t1	188,79	0,01	1,59	
	T1’t2	195,68	0,01	3,05	
	T2’t1	-52,20	0,16	3,66	
	T5’t2	-179,61	0,01	2,92	
	T6’t1	143,01	0,02	1,81	
	T7’t1	-121,86	0,13	2,90	
	T8’t1	140,52	0,06	2,29	
	T9’t1	-214,13	0,01	1,48	
Modelo SP					0,89
	const	319,91	0,98		
	C’t1	0,00	0,24	1,38	
	T1’t1	- 414,41	0,02	4,31	
	T1’t2	396,01	0,01	2,78	
	T2’t2	- 198,00	0,00	4,85	
	T3’t1	321,00	0,02	2,46	
	T3’t2	267,87	0,06	2,82	
	T4’t2	235,84	0,08	3,49	
	T5’t1	336,48	0,01	5,75	
	T7’t1	- 469,26	0,00	2,45	
	T8’t1	204,93	0,03	3,41	
	T9’t1	- 462,94	-	3,24	
Modelo AM					0,85
	const	204,33	0.952		
	C’t1	0,00	0.000	1,55	
	C’t2	-0,00	0.002	1,42	
	T0’t1	1.694,76	0.000	3,33	
	T0’t2	1.042,41	0.000	3,23	
	T5’t1	-1.334,74	0.000	5,04	
	T5’t2	-943,19	0.000	4,38	
	T7’t2	522,72	0.049	1,52	
	T9’t1	478,24	0.033	2,85	

Fonte: Elaborado pelo Autor

Const = constante do modelo,
C’t1 é a diferença entre a quantidade de casos na semana t e a semana t-1 (diferença de primeira ordem);
C’t2 é a diferença entre a quantidade de casos na semana t-1 e a semana t-2 (diferença de primeira ordem);
A mesma nomenclatura foi aplicada para os tópicos, sendo:
T1’t1 a diferença entre a quantidade de topicos1 na semana t e a semana t-1 (diferença de primeira ordem), T1t2 a diferença entre a quantidade de topicos1 na semana t-1 e a semana t-2 (diferença de primeira ordem), seguindo este critério para as demais variáveis.

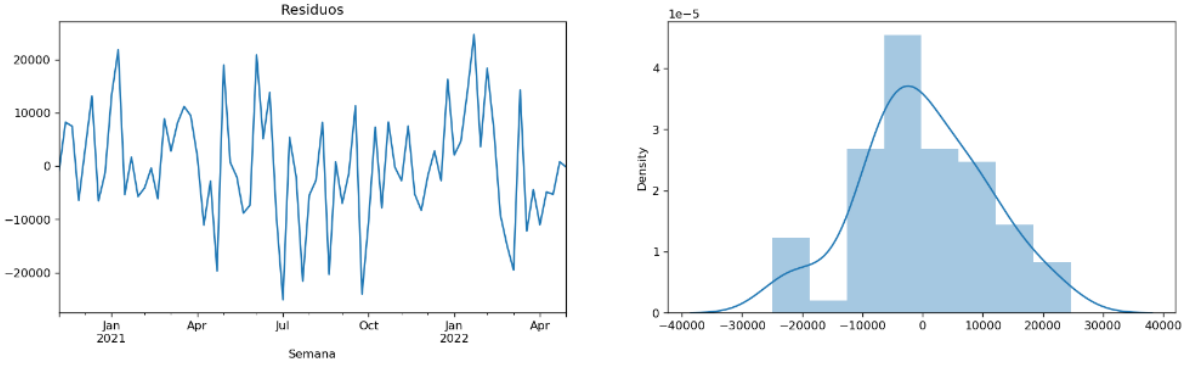
Para os três modelos foram analisados os resíduos. Os resíduos são a diferença entre o real e o projetado, esse passo é importante para um melhor diagnostico. Nas figuras abaixo são apresentados gráficos com a os resíduos por semana e em histograma (FiguraFigura 5-10, Figura 5-11, Figura 5-12).

Figura 5-10 - Distribuição dos erros residuais do modelo Brasil



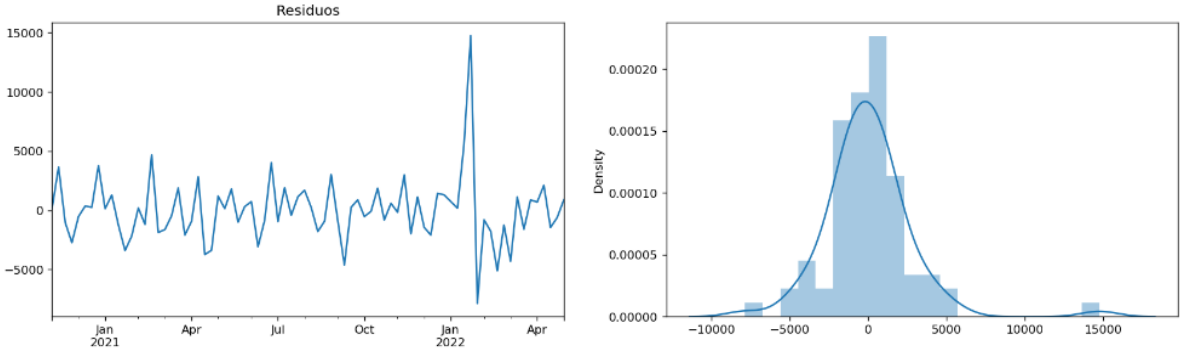
Fonte: Elaborado pelo Autor

Figura 5-11 - Distribuição dos erros residuais do modelo do estado São Paulo



Fonte: Elaborado pelo Autor

Figura 5-12 - Distribuição dos erros residuais do modelo do estado Amazonas



Fonte: Elaborado pelo Autor

Tabela 5-6 – Indicadores da distribuição dos erros dos modelos

Indicador	BR	AM	SP
Kurtosis	2,195	8,737	-0,112
Skew	0,977	1,607	-0,048

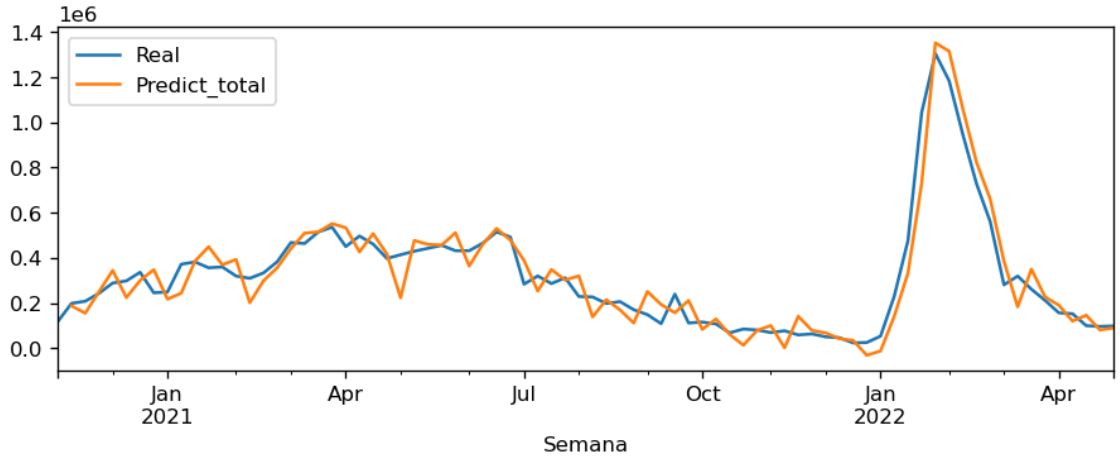
Fonte: Elaborado pelo Autor

Optou-se também por gerar indicadores de normalidade: o *Skew* que indica a assimetria de uma distribuição e o indicador *Kurtosis* que é uma medida que caracteriza o achatamento de uma distribuição de probabilidade. Quanto mais próximo de 0 para o indicador *Skew* e mais próximo de 0 para o indicador *Kurtosis* mais próximo é a curva de uma normal.

Analisando os indicadores dos modelos é observado que o modelo de São Paulo tem resíduos mais próximos de uma distribuição normal, já os demais não apresentam esta característica. Estes indicadores refletem o descolamento dos modelos nas primeiras semanas do ano de 2022, gerando um histograma com uma "cauda" maior do lado direito para o Brasil e para o estado Amazonas.

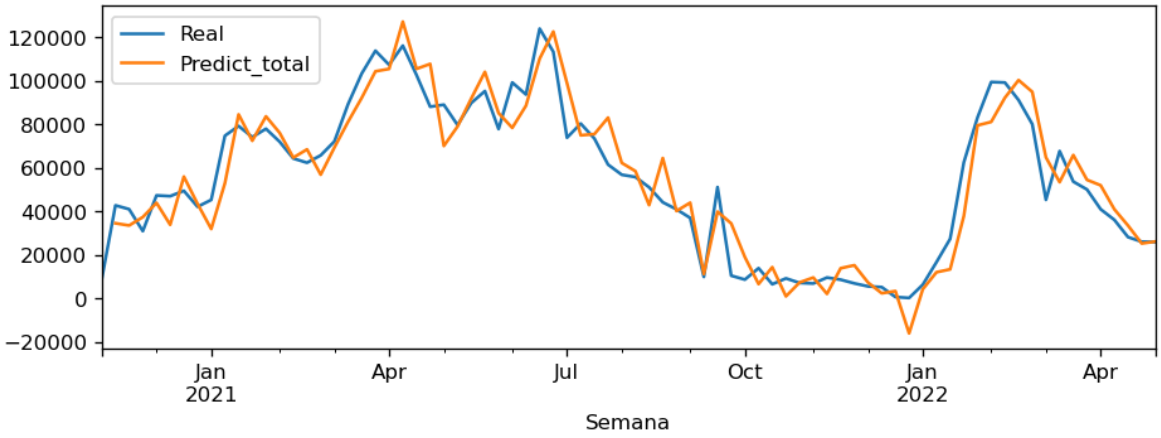
Como resultado dos modelos, a Figura 5-13 mostra um gráfico de séries temporais usando validação cruzada dos resultados das contagens semanais de casos de COVID-19 observadas e previstas durante 2021 e 2022 usando o modelo multivariado escolhido para o Brasil, o estado de São Paulo e o estado Amazonas.

Figura 5-13 - Total de casos semanais de COVID-19 no Brasil



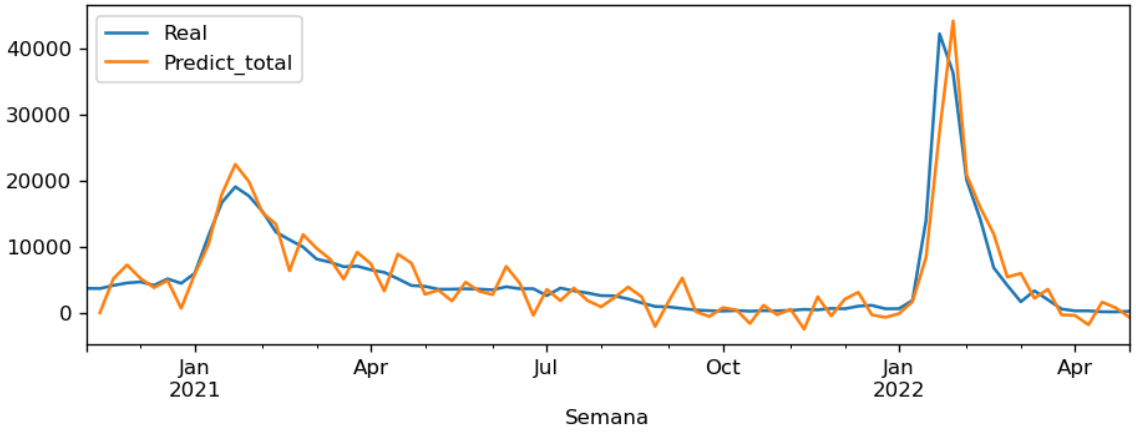
Fonte: Elaborado pelo Autor

Figura 5-14 - Total de casos semanais de COVID-19 no estado São Paulo



Fonte: Elaborado pelo Autor

Figura 5-15 - Total de casos semanais de COVID-19 no estado Amazonas



Fonte: Elaborado pelo Autor

Os modelos demonstraram que a combinação de contagens de casos anteriores e dados do Twitter resultam que a correlação entre a quantidade de casos semanais previstas e medidas foi alta, com um coeficiente de correlação de 0,85 para o estado de Amazonas, 0,89 para o estado de São Paulo e 0,90 para o Brasil, sendo bons indicadores de previsibilidade da quantidade de casos de COVID-19 com uma semana de antecedência no período de estudo.

No entanto, houve variações entres semanas apontadas pelo modelo que divergem do realizado principalmente no período com menor variação semanal de casos (abril a dezembro de 2021). Mas nos principais momentos de grande variação o modelo aparenta ser bem adequado.

6 CONCLUSÕES

6.1 DISCUSSÕES E CONTRIBUIÇÕES

Um surto de doença pode afetar uma grande população e se espalhar por vários países ou continentes em um curto período, representando ameaças significativas à saúde da humanidade. Um sistema de alerta precoce é fundamental para que as agências governamentais contenham o surto da doença.

A extensa aplicação das redes sociais abre novos *insights* para promover o alerta precoce em muitos cenários, e a compreensão das informações das redes sociais criadas a partir de determinados momentos e lugares pode fornecer recursos valiosos que podem ser incorporados aos sistemas de alerta.

A revisão sistemática apresentada nesta dissertação evidenciou diferentes algoritmos analíticos possíveis para a modelagem de dados não estruturados advindos de interações em redes sociais, baseando-se todos eles em postagens textuais.

Para explorar o potencial dos dados de mídia social, este estudo realizou uma análise retrospectiva da pandemia de COVID-19 no Brasil e investigou mais de 10 milhões de tweets relacionados de janeiro de 2020 a abril de 2022. Com a ajuda de técnicas de processamento de linguagem natural, aprendizado de máquina e modelagem de tópico, este estudo classificou cada tweet e criou modelos de regressão temporal demonstrando ser possível criar alertas para o aumento de casos de COVID-19.

Combinado com informações de contagens de casos anteriores do COVID-19, foram calibrados três modelos capazes de estimar os casos semanais com 1 semana de antecedência com razoável precisão; um modelo para a São Paulo, outro para Amazonas e um modelo para o Brasil.

A contribuição deste estudo reside em dois aspectos: do ponto de vista metodológico, este estudo apresenta um processo de quantificação do tratamento de dados de redes sociais e oferece uma abordagem para estimar quantitativamente e com antecedência a quantidade de casos de COVID-19 com base nas classificações de tweets. Ao contrário dos métodos baseados na contagem de frequência (por exemplo, rastreamento de consultas de pesquisa, contagem de padrões de palavras), este estudo aplicou métodos de aprendizado de máquina para identificar o conteúdo textual nas redes sociais, refletindo assim o contexto dos posts das pessoas sobre o momento da pandemia. Do ponto de vista prático, este estudo demonstra o potencial de agregar

opiniões públicas através das redes sociais para sinalizar o alerta precoce. Algumas informações úteis nas redes sociais podem servir como alertas para o surto da doença em um sentido mais amplo.

Nesta dissertação foram calibrados três modelos que foram capazes de estimar os casos semanais com 1 semana de antecedência com razoável precisão, sendo um modelo para a São Paulo, outro para Amazonas e um modelo para o Brasil.

Quando se é detectado postagens com conteúdo especifico e classificados em topicos definidos, é um indicativo que um evento de saude pode ocorrer dentro do prazo de 1 semana.

Analisando o modelo calibrado para o estado de Amazonas, quando é detectado postagens com conteúdo sobre desdobramento da doença (tópico 0), sintomas (topicos 5 e 7) e medidas de proteção (topico 9) é indicativo que um evento em saúde pode ocorrer dentro do prazo de uma semana.

Desta forma, o monitoramento desse tipo de semântica ajuda a identificar com uma semana de antecedência eventos em saúde.

A dissertação estuda a COVID 19, porém os aprendizados obtidos pela pesquisa podem ser extrapolados para novas ondas de doenças e permitindo à gestão pública se preparar com alguma antecedência para as demandas dessas ondas em termos de alocação de recursos físicos e humanos, bem como enviar avisos e orientações à população.

No geral, essa abordagem tem as vantagens de rapidez, quantidade, informação temporal e espacial e boa previsibilidade para auxiliar no desdobramento da crise, e pode alimentar os atuais sistemas de alerta ajudando as agências governamentais a melhorar o alerta precoce em tempo hábil.

6.2 LIMITAÇÕES E TRABALHOS FUTUROS

Algumas lacunas puderam ser identificadas neste trabalho. O estudo foi conduzido utilizando tweets em português (brasileiro) e com dados da pandemia no Brasil, portanto, os resultados não podem ser generalizados para outros países, embora o sucesso na captação do alerta sugira que o modelo possa ser replicado a outras realidades e a outros tipos de problemas, por exemplo enchentes, poluição do ar, crises do sistema de mobilidade entre outros.

Muitos tweets são difíceis de entender devido ao contexto e a forma como são escritos, isso pode afetar a qualidade da modelagem de tópico e na execução do processamento dos dados. Ainda é uma dificuldade os modelos de processamento de linguagem natural

compreender o uso de ironia, sarcasmo e especificidades culturais, por exemplo, o *emoji high five* é utilizado no Brasil para expressar gratidão, enquanto em língua inglesa pode ser usado para expressar concordância ou mesmo uma comemoração. Outro ponto importante observado é que vários tópicos identificados pela LDA têm conteúdo opinativo ou de caráter informacional, porém poucos trazem reporte de sintomas o que permitiria o estabelecimento de uma relação mais clara entre postagens e incidência da doença, mesmo tendo sido possível estabelecer uma relação satisfatória entre movimento nas redes sociais e o fato em saúde.

Adicionalmente, notícias falsas e conteúdo equivocado podem ter sido processados no conjunto de documentos avaliados na pesquisa, pois foge ao escopo do estudo a identificação e tratamento desse material.

Mesmo os modelos demonstrando a capacidade dos dados do Twitter em servir como um indicador da evolução da pandemia, esses dados não devem ser vistos como um substituto para a coleta e análise de dados tradicionais, mas sim como um suplemento para ajudar nas tomadas de decisões.

Temas que carecem de investigação:

- Como atestar a veracidade do conteúdo da mensagem ou como identificar mensagens falsas ou postadas por robôs?
- Como se certificar de que os dados, não-estruturados por natureza, não carregam padrões discriminatórios que são propagados pelos modelos opacos?
- Como incluir dados de diferentes naturezas como imagens, áudio e vídeo, comuns nas redes sociais, no pool de dados a serem analisados?
- Como identificar elementos da emoção humana nas postagens, por exemplo, ironia e sarcasmo?
- Os dados gerados em redes sociais são de propriedades das organizações que as controlam, podendo permitir ou não acesso a eles por empresas e governos; como mitigar esse risco?
- Em específico, na gestão pública, quais as barreiras legais, culturais e de infraestrutura para que as redes sociais possam ser incorporadas como fonte de informação relevante à tomada de decisão?

Há muitas direções para estender este trabalho, desde a modelagem relativa a outros tipos de doenças (por exemplo: dengue, sarampo, malária), a análise de outras plataformas de mídia social, expandir o estudo para diferentes idiomas para obter uma melhor compreensão das consequências globais do COVID-19 ou mesmo outras doenças especificas de uma região.

As pesquisas com dados de redes sociais podem ir além de estudos retrospectivos e gerar produtos de apoio à tomada de decisão em tempo real.

Também é possível analisar a atividade de *retweet* para entender melhor como a informação é propagada e quais usuários dominaram as discussões da COVID-19.

A análise de rede pode ser realizada para identificar clusters de usuários com semelhança de tópicos. Pode ser interessante também analisar a atividade de grupos específicos de usuários, como as pessoas que foram testadas positivas ou as pessoas que são mais vulneráveis.

Além dos dados da rede social pode-se combinar dados como a densidade populacional, cobertura vacinal, índice de desenvolvimento humano e muitos outros dados públicos disponibilizados pelo sistema único de saúde brasileiro.

É indicado também a criação de um “framework” robusto incluindo a utilização ética das informações para possibilitar o uso de informações sensíveis no desenvolvimento de sistemas de utilização massiva tanto pelos órgãos públicos como para organizações privadas. Ainda, seria de grande contribuição prática o desenvolvimento de um dashboard contendo relatórios de monitoramento desses dados que pudessem ser utilizados pelas gestões públicas para a tomada de decisão estratégica.

REFERÊNCIAS

Adriani, M., Azzahro, F., & Hidayanto, A. (2015). Disease surveillance in Indonesia through Twitter posts. *Journal of Applied Research and Technology*, *13*, 374–381.

AlAgha, I. (2021). Topic Modeling and Sentiment Analysis of Twitter Discussions on COVID-19 from Spatial and Temporal Perspectives. *Journal of Information Science Theory and Practice*, *9*(1), 35–53. https://doi.org/10.1633/JISTaP.2021.9.1.3

Allam, Z., & Jones, D. S. (2020). Pandemic stricken cities on lockdown. Where are our planning and design professionals [now, then and into the future]? *Land Use Policy*, *97*(May), 104805. https://doi.org/10.1016/j.landusepol.2020.104805

Alzubi, J., Nayyar, A., & Kumar, A. (2018). Machine Learning from Theory to Algorithms: An Overview. *Journal of Physics: Conference Series*, *1142*(1). https://doi.org/10.1088/1742-6596/1142/1/012012

Anatel. (2021). *Acessos de telefonia Movel*. Julho. https://informacoes.anatel.gov.br/paineis/acessos/telefonia-movel

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly. https://www.nltk.org/book/

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). *Latent Dirichlet Allocation*. *3*, 993–1022.

Bragazzi, N. L., Dai, H., Damiani, G., Behzadifar, M., Martini, M., & Wu, J. (2020). How big data and artificial intelligence can help better manage the covid-19 pandemic. *International Journal of Environmental Research and Public Health*, *17*(9), 3176. https://doi.org/10.3390/ijerph17093176

Cambria, E., & White, B. (2014). Jumping NLP curves: A review of natural language processing research. *IEEE Computational Intelligence Magazine*, *9*(2), 48–57. https://doi.org/10.1109/MCI.2014.2307227

Cuomo, R. E., Purushothaman, V., Li, J., Cai, M., & Mackey, T. K. (2021). A longitudinal and geospatial analysis of COVID-19 tweets during the early outbreak period in the United States. *BMC Public Health*, *21*(1), 793. https://doi.org/10.1186/s12889-021-10827-4

Daniel Jurafsky, & Martin, J. H. (2021). Speech and Language Processing: An introduction to natural language processing. *SPEECH and LANGUAGE PROCESSING An Introduction to Natural Language Processing Computational Linguistics and Speech Recognition*. http://www.cs.colorado.edu/~martin/slp.html

DataSus. (2022). Junho. https://covid.saude.gov.br/

Davenport. (2006). Competing on Analytics. *Harvard Business Review*, 12. http://hbr.org/product/competing-on-analytics/an/R0601H-PDF-ENG

Dredze, M., Paul, M. J., Bergsma, S., & Tran, H. (2013). Carmen: A twitter geolocation system with applications to public health. *AAAI Workshop - Technical Report*, *WS-13-09*, 20–24.

Dresch, A., Lacerda, D. P., & Antunes, J. A. V. (2015). *Desing Science Research: Método de Pesquisa para Avanço da Ciência e Tecnologia* (Bookman (Ed.)).

Edo-Osagie, O., Smith, G., Lake, I., Edeghere, O., & De La Iglesia, B. (2019). Twitter mining using semi-supervised classification for relevance filtering in syndromic surveillance. *PLoS ONE*, *14*(7), 1–29. https://doi.org/10.1371/journal.pone.0210689

Euzebio, C., Agy, S., Boldorini Jr., C., Porto, L., Alcarás, J. R., Martinez, A., & Ruiz, E. (2020). *Statistical analysis of small twitter data collection to identify dengue outbreaks*. 17–24. https://doi.org/10.5753/kdmile.2020.11954

Fazeli, S., Zamanzadeh, D., Ovalle, A., Nguyen, T., Gee, G., & Sarrafzadeh, M. (2021). *COVID-19 and Big Data: Multi-faceted Analysis for Spatio-temporal Understanding of the Pandemic with Social Media Conversations*. http://arxiv.org/abs/2104.10807

Hassan Zadeh, A., Zolbanin, H. M., Sharda, R., & Delen, D. (2019). Social Media for Nowcasting Flu Activity: Spatio-Temporal Big Data Analysis. *Information Systems Frontiers*, *21*(4), 743–760. https://doi.org/10.1007/s10796-018-9893-0

Heth, Z., Bemis, K., & Christiansen, D. (2018). Correlation of Tweets Mentioning Influenza Illness and Traditional Surveillance Data. *Online Journal of Public Health Informatics*, *10*(1), 2579. https://doi.org/10.5210/ojphi.v10i1.8773

Honnibal, M. (2023). *spaCy* (3.5.0). spacy.io

Ismagilova, E., Hughes, L., Dwivedi, Y. K., & Raman, K. R. (2019). Smart cities : Advances in research — An information systems perspective. *International Journal of Information Management*, *47*(January), 88–100. https://doi.org/10.1016/j.ijinfomgt.2019.01.004

Jain, A., & Cherikkallil, S. (2018). Medinsights: Twitter Based Platform for Health Care Analytics. *Proceedings of the International Conference on Inventive Research in Computing Applications, ICIRCA 2018*, *Icirca*, 1104–1109. https://doi.org/10.1109/ICIRCA.2018.8597360

Kim, T., & Wurster, K. (2022). *Emoji 2.0*. https://pypi.org/project/emoji/

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews, Version 1.0. *Empirical Software Engineering*, *33*(2004), 1–26.

Kwak, H., Lee, C., Park, H., & Moon, S. (2010). What is Twitter, a social network or a news media? *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, 591–600. https://doi.org/10.1145/1772690.1772751

Lacerda, D. P., Dresch, A., Proença, A., & Antonio, J. (2013). *Design Science Research : método de pesquisa para a engenharia de produção*. 741–761.

Li, L., Gao, L., Zhou, J., Ma, Z., Choy, D. F., & Ha, M. A. (2021). Can Social Media Data Be Utilized to Enhance Early Warning: Retrospective Analysis of the U.S. Covid-19 Pandemic. *MedRxiv*, *XX*, 1–13. http://medrxiv.org/cgi/content/short/2021.04.11.21255285v1?rss=1&utm_source=researcher_app&utm_medium=referral&utm_campaign=RESR_MRKT_Researcher_inbound

Li, Z., Li, X., Porter, D., Zhang, J., Jiang, Y., Olatosi, B., & Weissman, S. (2020). Monitoring the spatial spread of covid-19 and effectiveness of control measures through human movement data: Proposal for a predictive model using big data analytics. *JMIR Research Protocols*, *9*(12), 1–10. https://doi.org/10.2196/24432

Lima, C. R. M. de, Röder, E. dos S. F., Carvalho, F. da S., & Günther, H. F. (2020). Tensões e conflitos na vigilância digital de pessoas para controle da pandemia de COVID-19. *P2P E INOVAÇÃO*, *7*, 241–257. https://doi.org/10.21721/p2p.2020v7n1.p241-257

Mackey, T., Purushothaman, V., Li, J., Shah, N., Nali, M., Bardier, C., Liang, B., Cai, M., & Cuomo, R. (2020). Machine Learning to Detect Self-Reporting of Symptoms, Testing Access, and Recovery Associated With COVID-19 on Twitter: Retrospective Big Data Infoveillance Study. *JMIR Public Health and Surveillance*, *6*(2), e19509. https://doi.org/10.2196/19509

Martínez, N. J. F., & Pascual, C. P. (2020). Knowledge-based rules for the extraction of complex, fine-grained locative references from tweets. *Revista Electronica de Linguistica Aplicada*, *19*(1), 136–163.

Masri, S., Jia, J., Li, C., Zhou, G., Lee, M.-C., Yan, G., & Wu, J. (2019). Use of Twitter data to improve Zika virus surveillance in the United States during the 2016 epidemic. *BMC Public Health*, *19*(1), 761. https://doi.org/10.1186/s12889-019-7103-8

Mimno, D., Wallach, H. M., Talley, E., Leenders, M., & McCallum, A. (2011). Optimizing semantic coherence in topic models. *EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, *2*, 262–272.

Murzintcev Nikita. (2015). *Select number of topics for LDA model*.

Nguyen, H., Nguyen, · Thin, Duc, ·, & Nguyen, T. (2021). A graph-based approach for population health analysis using Geo-tagged tweets. *Multimedia Tools and Applications*, *80*, 7187–7204. https://doi.org/10.1007/s11042-020-10034-0

ODS Brasil. (2021). https://odsbrasil.gov.br/home/agenda

Popkova, E. G., & Sergi, B. S. (2021). Digital public health: Automation based on new datasets and the Internet of Things. *Socio-Economic Planning Sciences*, *February*, 101039. https://doi.org/10.1016/j.seps.2021.101039

Pruss, D., Fujinuma, Y., Daughton, A. R., Paul, M. J., Arnot, B., Szafir, D. A., & Boyd-Graber, J. (2019). Zika discourse in the Americas: A multilingual topic analysis of Twitter. *PLoS ONE*, *14*(5), 1–23. https://doi.org/10.1371/journal.pone.0216922

Rashid, M. T., & Wang, D. (2021). CovidSens: a vision on reliable social sensing for COVID-19. *Artificial Intelligence Review*, *54*(1), 1–25. https://doi.org/10.1007/s10462-020-09852-3

Řehůřek, R. (2022). *Gensim*. radimrehurek.com/gensim

Santos, M. C. dos. (2018). O estranhamento da interdisciplinaridade que nos assombra. *Comunicacao e Inovação PPGCOM/USCS*, *19*, 19–33.

Schuurman, D., Baccarne, B., De Marez, L., & Mechant, P. (2012). Smart ideas for smart cities: Investigating crowdsourcing for generating and selecting ideas for ICT innovation in a city context. *Journal of Theoretical and Applied Electronic Commerce Research*, *7*(3), 49–62. https://doi.org/10.4067/S0718-18762012000300006

Șerban, O., Thapen, N., Maginnis, B., Hankin, C., & Foot, V. (2019). Real-time processing of social media with SENTINEL: A syndromic surveillance system incorporating deep learning for health classification. *Information Processing and Management*, *56*(3), 1166–1184. https://doi.org/10.1016/j.ipm.2018.04.011

Sidana, S., Amer-Yahia, S., Clausel, M., Rebai, M., Mai, S. T., & Amini, M.-R. (2018). Health Monitoring on Social Media over Time. *IEEE Transactions on Knowledge and Data Engineering*, *30*(8), 1467–1480. https://doi.org/10.1109/TKDE.2018.2795606

Simon, H. A. (1996). *The sciences of the artificial* (Third Edit).

Souza, R. C. S. N. P., Assunção, R. M., Oliveira, D. M., Neill, D. B., & Meira, W. (2019). Where did I get dengue? Detecting spatial clusters of infection risk with social network data. *Spatial and Spatio-Temporal Epidemiology*, *29*, 163–175. https://doi.org/10.1016/j.sste.2018.11.005

Spurlock, K., & Elgazzar, H. (2020). Predicting COVID-19 Infection Groups using Social Networks and Machine Learning Algorithms. *2020 11th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2020*, 0245–0251. https://doi.org/10.1109/UEMCON51285.2020.9298093

Statista. (2021). *Statista*. https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/

Tufts, Polsky, D., Volpp, K. G., Groeneveld, P. W., Ungar, L., Merchant, R. M., & Pelullo, A. P. (2018). Characterizing tweet volume and content about common health conditions across Pennsylvania: Retrospective analysis. *JMIR Public Health and Surveillance*, *4*(4), 1–9. https://doi.org/10.2196/10834

Twitter. (2021). *Twitter APIv2 - Academic Research product*. https://developer.twitter.com/en/products/twitter-api/academic-research/product-details#academic-track

Vayansky, I., & Kumar, S. A. P. (2020). A review of topic modeling methods. *Information Systems*, *94*, 101582. https://doi.org/10.1016/j.is.2020.101582

WHO. (2021). https://www.who.int/health-topics/coronavirus#tab=tab_3

Xavier, F., Olenscki, J. R. W., Acosta, A. L., Sallum, M. A. M., & Saraiva, A. M. (2020). Analise de redes sociais como estrategia de apoio a vigilancia em saude durante a Covid-19. *Estudos Avancados*, *34*(99), 261–282. https://doi.org/10.1590/S0103-4014.2020.3499.016

APÊNDICES

CODIGOS EM PYTHON

COLETAR POSTS DO TWITTER

Bibliotecas com funções para busca

```
"""
Biblioteca: coleta_tw.py

Created on Tue Oct  5 15:00:18 2021
@author: Kleber Santos email:ksantos@usp.br
"""
"""
"""

# Para enviar requisições gets a API do twitter
import requests
# importar o MongoClient do pymongo
from pymongo import MongoClient

###### Função para carregar o bearertoken em formato adequado para acessar a API

def create_headers(bearer_token):
    headers = {"Authorization": "Bearer {}".format(bearer_token)}
    return headers

###### Função para criar a URL de requisição ao endpoint escolhido

def create_url(keyword, start_date, end_date, max_results = 10): #max_result tem que ser no maximo
500

    search_url = "https://api.twitter.com/2/tweets/search/all" # endpoint escolhido

    #Parametros escolhidos no endpoint. Na documentação do endpoint tem detalhes dos parametros
possiveis https://developer.twitter.com/en/docs/twitter-api/tweets/search/api-reference/get-tweets-search-all
    query_params = {'query': keyword,
                    'start_time': start_date,
                    'end_time': end_date,
                    'max_results': max_results,
                    'expansions': 'author_id,in_reply_to_user_id,geo.place_id',
                    'tweet.fields':
'id,text,author_id,in_reply_to_user_id,geo,conversation_id,created_at,lang,public_metrics,referenced_tweets,repl
y_settings,source',
                    'user.fields':
'id,name,username,location,created_at,description,public_metrics,verified,profile_image_url',
                    'place.fields': 'full_name,id,country,country_code,geo,name,place_type',
                    'next_token': {}}
    return (search_url, query_params)

###### Função para conectar ao endpoint e retornar os tweets em formato JSON

def connect_to_endpoint(url, headers, params, next_token = None):
    params['next_token'] = next_token   #params object recebido da função create_url
    response = requests.request("GET", url, headers = headers, params = params)
    print("Endpoint Response Code: " + str(response.status_code))
    if response.status_code != 200:
        raise Exception(response.status_code, response.text)
    return response.json()
```

```
        ## cria a url para buscar os seguidos
        def create_url_following(user_id, max_results = 10): #max_result tem que ser no maximo 500
            search_url = "https://api.twitter.com/2/users/"+str(user_id)+"/following" # endpoint escolhido

            #Parametros escolhidos no endpoint. Na documentação do endpoint tem detalhes dos parametros possiveis
            query_params = {'max_results': max_results,
                        'user.fields': 'id,username,name,profile_image_url,location',
                        'pagination_token':{}}

            return (search_url, query_params)


        ## cria url para buscar seguidores
        def create_url_follower(user_id, max_results = 10): #max_result tem que ser no maximo 500
            search_url = "https://api.twitter.com/2/users/"+str(user_id)+"/followers" # endpoint escolhido
            #https://api.twitter.com/2/users/:id/followers

            #Parametros escolhidos no endpoint. Na documentação do endpoint tem detalhes dos parametros possiveis
            query_params = {'max_results': max_results,
                        'user.fields': 'id,username,name,profile_image_url,location',
                        'pagination_token':{}}

            return (search_url, query_params)

        # função para conectar ao endpoint de seguidores ou seguidos

        def connect_to_endpoint_f(url, headers, params, next_token = None):
            params['pagination_token'] = next_token   #params object recebido da função create_url
            response = requests.request("GET", url, headers = headers, params = params)
            print("Endpoint Response Code: " + str(response.status_code))
            if response.status_code != 200:
                raise Exception(response.status_code, response.text)
            return response.json()


        ## cria url para buscar tweets de um usuario
        def create_url_user(user_id, start_time, end_date, max_results = 10): #max_result tem que ser no maximo 500
            search_url = "https://api.twitter.com/2/users/"+str(user_id)+"/tweets" # endpoint escolhido

            query_params = {'max_results': max_results,
                        'start_time':start_time,
                        'end_time': end_date,
                        'expansions': 'author_id,in_reply_to_user_id,geo.place_id',
                        'tweet.fields':
'id,text,author_id,in_reply_to_user_id,geo,conversation_id,created_at,lang,public_metrics,referenced_tweets,reply_settings,source',
                        'user.fields':
'id,name,username,location,created_at,description,public_metrics,verified,profile_image_url',
                        'place.fields': 'full_name,id,country,country_code,geo,name,place_type',
                        'pagination_token': {}}
            return (search_url, query_params)

        def connect_to_endpoint_user(url, headers, params, next_token = None):
            params['pagination_token'] = next_token   #params object recebido da função create_url
            response = requests.request("GET", url, headers = headers, params = params)
```

```
        print("Endpoint Response Code: " + str(response.status_code))
        if response.status_code != 200:
            raise Exception(response.status_code, response.text)
        return response.json()


    def create_url_user_mentions(user_id, start_time, max_results = 10): #max_result tem que ser no maximo 500
        search_url = "https://api.twitter.com/2/users/"+str(user_id)+"/mentions" # endpoint escolhido

        query_params = {'max_results': max_results,
                        'start_time':start_time,
                        'expansions': 'author_id,in_reply_to_user_id,geo.place_id',
                        'tweet.fields':
'id,text,author_id,in_reply_to_user_id,geo,conversation_id,created_at,lang,public_metrics,referenced_tweets,reply_settings,source',
                        'user.fields':
'id,name,username,location,created_at,description,public_metrics,verified,profile_image_url',
                        'place.fields': 'full_name,id,country,country_code,geo,name,place_type',
                        'pagination_token': {}}
        return (search_url, query_params)


    # função para buscar informações de um determinado usuário
    def search_user(user,headers, flag = 0):
        if (flag == 0):
            search_url = "https://api.twitter.com/2/users/by/username/"+str(user)
        if (flag == 1):
            search_url = "https://api.twitter.com/2/users/"+str(user)
        query_params = {'user.fields':
'id,name,username,location,created_at,description,public_metrics,verified,profile_image_url'}
        response = requests.request("GET", search_url,headers=headers, params = query_params)
        print("Endpoint Response Code: " + str(response.status_code))
        if response.status_code != 200:
            raise Exception(response.status_code, response.text)
        return response.json()


    # buscar de tweet pelo id
    def search_tw(tw_id, headers):
        search_url = "https://api.twitter.com/2/tweets/"+str(tw_id)
        query_params = {'expansions': 'author_id,in_reply_to_user_id,geo.place_id',
                        'tweet.fields':
'id,text,author_id,in_reply_to_user_id,geo,conversation_id,created_at,lang,public_metrics,referenced_tweets,reply_settings,source',
                        'user.fields':
'id,name,username,location,created_at,description,public_metrics,verified,profile_image_url',
                        'place.fields': 'full_name,id,country,country_code,geo,name,place_type',
                        'next_token': {}}
        response = requests.request("GET", search_url, headers=headers, params = query_params)
        print("Endpoint Response Code: " + str(response.status_code))
        if response.status_code != 200:
            raise Exception(response.status_code, response.text)
        return response.json()


    ### funções para adicionar os dados em banco de dados
    def append_to_mdb(json_response,tipo=0):
        # tipo =0 tweet, 1 = seguidores , 2 = seguindo
```

```
#cria base mongodb
con_mdb = MongoClient('localhost', 27017)
db = con_mdb.tw
if tipo ==0:
   collection = db.posts
if tipo == 1:
   collection = db.seguidores
if tipo ==2:
   collection = db.seguindo
#Loop through each tweet e armazena no arquivo de mongodb
meta = json_response['meta']
count = meta['result_count']    # Print the number of tweets for this iteration
post_id = collection.insert_one(json_response).inserted_id
post_id
print("# of Tweets added from this response: ", count)
```

Script principal para coleta de tweets

Processo para coletar tweets diretamente da APIv2¶
versao 2.0 data 19/12/2021¶
Elaborada passo a passo com funçoes separadas
Foi desenvolvido por Kleber Santos pois nao encontrei um boa biblioteca que esteja adaptada para a api V2 do twitter e usando licença academica

```
###### PARTE I: Importação das bibliotecas necessárias:
# Para enviar requisições gets a API do twitter
import requests
# Para salvar tokens e variáveis de sistema de forma mais rápida
import os
# Para tratar objetos em formato JSON que são recebidos pelo Twitter
import json
# Para trabalhar com tabelas
import pandas as pd
# Para trabalhar com marcação de datas
import datetime
import dateutil.parser
import unicodedata
#Para adicionar delay entre iterações e requests ao twitter e nao ser bloqueado
import time

import sqlite3

#Importar as funções criadas pela biblioteca coleta_tw desenvolvida por KS
from coleta_tw import *

#### PARTE II: inserir as credenciais e parametros de busca
# Script principal para obter os dados do Twitter conforme a necessidade

#Inputs for tweets
bearer_token = '?????????' #inserir o token neste campo
keyword = '(covid-19 OR coronavirus OR corona OR Febre OR tosse OR dor OR diarreia OR pandemina OR epidemia OR paladar OR olfato OR isolamento OR distanciamento OR quarentena OR vacina) lang:pt'
start_list =    ['2020-03-21T00:00:00.000Z']
end_list =      ['2020-03-24T00:00:00.000Z']

max_results = 450 # maximo permitido pelo Twitter
max_count =10000000 # maximo de tweets por período
```

```
#Total number of tweets we collected from the loop
total_tweets = 0

# Cria a base de dados e os campos que serão coletados dos tweets

# Abre a base de dados tw e cria o cursor para armazenar os tweets
conn = sqlite3.connect('tw.db')
cursor = conn.cursor()

# criando a tabela (schema)
cursor.execute("""
CREATE TABLE tw (
        id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
        author_id TEXT,
        created_at DATE,
        geo TEXT,
        tweet_id TEXT,
        lang TEXT,
        like_count TEXT,
        quote_count TEXT,
        reply_count TEXT,
        retweet_count TEXT,
        source TEXT,
        text TEXT
        );
""")

print('Tabela "tw" criada com sucesso.')
conn.close()
# Cria a base de dados de localidades

# Abre a base de dados tw e cria o cursor para armazenar os tweets
conn = sqlite3.connect('tw.db')
cursor = conn.cursor()

# criando a tabela (schema)
cursor.execute("""
CREATE TABLE places (
        id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
        id_place TEXT,
        full_name TEXT,
        country TEXT,
        country_code TEXT,
        geo_a TEXT,
        geo_b TEXT,
        geo_c TEXT,
        geo_d TEXT,
        name TEXT,
        place_type TEXT
        );
""")
print('Tabela "places" criada com sucesso.')
conn.close()
# Cria a base de dados de localidades
# Abre a base de dados tw e cria o cursor para armazenar os tweets
conn = sqlite3.connect('tw.db')
cursor = conn.cursor()

# criando a tabela (schema)
```

```
cursor.execute("""
CREATE TABLE users (
    id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    id_user TEXT,
    profile_image_url TEXT,
    name TEXT,
    username TEXT,
    location TEXT
    );
""")
print('Tabela "users" criada com sucesso.')
conn.close()

# Função que coloca cada objeto JSON obtido do Twitter em cada coluna da base de dados SQLite, é importante
# pois nos campos nulos é colocado "" para não dar erro no script
import sqlite3
def append_to_sqlite(json_response):

   # Abrir e criar o cursor da base de dados SQLite
   conn = sqlite3.connect('tw.db')
   cursor = conn.cursor()
   # counter variable
   counter = 0
   #Loop through each tweet
   for tweet in json_response['data']:

      # We will create a variable for each since some of the keys might not exist for some tweets
      # So we will account for that
      # 1. Author ID
      t_author_id = tweet['author_id']
      # 2. Time created
      t_created_at = dateutil.parser.parse(tweet['created_at'])
      # 3. Geolocation
      if ('geo' in tweet):
         t_geo  = tweet['geo']['place_id']
      else:
         t_geo = ""
      # 4. Tweet ID
      t_tweet_id = tweet['id']
      # 5. Language
      t_lang = tweet['lang']
      # 6. Tweet metrics
      t_retweet_count = tweet['public_metrics']['retweet_count']
      t_reply_count = tweet['public_metrics']['reply_count']
      t_like_count = tweet['public_metrics']['like_count']
      t_quote_count = tweet['public_metrics']['quote_count']
      # 7. source
      t_source = tweet['source']
      # 8. Tweet text
      t_text = tweet['text']

      # Assemble all data in a list
      res = [t_author_id, t_created_at, t_geo, t_tweet_id, t_lang, t_like_count, t_quote_count, t_reply_count, t_retweet_count, t_source, t_text]
      # Adiciona os campos a base de dados SQLite
      cursor.execute("""
        INSERT INTO tw (author_id, created_at, geo, tweet_id, lang, like_count, quote_count, reply_count, retweet_count, source, text)
```

```
            VALUES (?,?,?,?,?,?,?,?,?,?)
            """, res)

        #adiciona no banco de dados
        conn.commit()
        counter += 1

    # Fecha banco dedados SQLite
    conn.close()

    # Print the number of tweets for this iteration
    print("# Tweets incluidos: ", counter)
    print("Data taweet:", t_created_at)

##### incluir localidades

    # Abrir e criar o cursor da base de dados SQLite
    conn = sqlite3.connect('tw.db')
    cursor = conn.cursor()

    # counter variable
    counter = 0


#comando para testar (04/01 não efetivado)
# if 'places' in json_response['includes']:

    #Loop through each tweet
    for tweet in json_response['includes']['places']:

        p_id = tweet['id']
        p_full_name = tweet['full_name']
        p_country = tweet['country']
        p_country_code = tweet['country_code']

        p_geo_a = tweet['geo']['bbox'][0]
        p_geo_b = tweet['geo']['bbox'][1]
        p_geo_c = tweet['geo']['bbox'][2]
        p_geo_d = tweet['geo']['bbox'][3]


        p_name = tweet['name']
        p_place_type = tweet['place_type']


        # Assemble all data in a list
        res = [p_id, p_full_name, p_country, p_country_code, p_geo_a, p_geo_b, p_geo_c, p_geo_d,
p_name, p_place_type]
        # Adiciona os campos a base de dados SQLite
        cursor.execute("""
            INSERT INTO places (id_place, full_name, country, country_code, geo_a, geo_b, geo_c,
geo_d, name, place_type)
            VALUES (?,?,?,?,?,?,?,?,?,?)
            """, res)

        #adiciona no banco de dados
        conn.commit()
        counter += 1
```

```
    # Fecha banco dedados SQLite
    conn.close()

    # Print the number of tweets for this iteration
    print("# locais incluidos: ", counter)

##### incluir usuarios

    # Abrir e criar o cursor da base de dados SQLite
    conn = sqlite3.connect('tw.db')
    cursor = conn.cursor()

    # counter variable
    counter = 0

    #Loop through each tweet
    for tweet in json_response['includes']['users']:

        u_id = tweet['id']
        u_image = tweet['profile_image_url']
        u_name = tweet['name']
        u_username = tweet['username']

        if ('location' in tweet):
            u_location  = tweet['location']
        else:
            u_location = ""

        # Assemble all data in a list
        res = [u_id, u_image, u_name, u_username, u_location]
        # Adiciona os campos a base de dados SQLite
        cursor.execute("""
            INSERT INTO users (id_user, profile_image_url, name, username, location)
            VALUES (?,?,?,?,?)
            """, res)

        #adiciona no banco de dados
        conn.commit()
        counter += 1

    # Fecha banco dedados SQLite
    conn.close()

    # Print the number of tweets for this iteration
    print("# usuarios incluidos: ", counter)




#### PARTE III: processo abaixo para obter os twittes conforme parametros definidos:

#cria headers com o token do usuário
headers = create_headers(bearer_token)

#iterar e buscar vários tweets

#iterando para cada periodo da lista
```

```
for i in range(0,len(start_list)):
    # Inputs
    count = 0 # Counting tweets per time period
    flag = True
    next_token = None

    # Check if flag is true
    while flag:
        # Check if max_count reached
        if count >= max_count:
            break
        print("-------------------")
        print("Token: ", next_token)
        url = create_url(keyword, start_list[i],end_list[i], max_results)
        json_response = connect_to_endpoint(url[0], headers, url[1], next_token)
        result_count = json_response['meta']['result_count']

        if 'next_token' in json_response['meta']:
            # Save the token to use for next call
            next_token = json_response['meta']['next_token']
            print("Next Token: ", next_token)
            if result_count is not None and result_count > 0 and next_token is not None:
                print("Start Date: ", start_list[i])
                append_to_sqlite(json_response) #inclui os tweets na base de dados
                count += result_count
                total_tweets += result_count
                print("Total # of Tweets added: ", total_tweets)
                print("-------------------")
                time.sleep(2)
        # If no next token exists
        else:
            if result_count is not None and result_count > 0:
                print("-------------------")
                print("Start Date: ", start_list[i])
                append_to_sqlite(json_response) #inclui os tweets na base de dados
                count += result_count
                total_tweets += result_count
                print("Total # of Tweets added: ", total_tweets)
                print("-------------------")
                time.sleep(2)

            #Since this is the final request, turn flag to false to move to the next time period.
            flag = False
            next_token = None
        time.sleep(2)
print("Quantidade total
```

SCRIPT DE PROCESSAMENTO DOS TWEETS, DADOS DATASUS E CRIAÇÃO DE CORRELAÇÕES E ANÁLISES

Bibliotecas com funções de limpeza dos dados:

```
# -*- coding: utf-8 -*-.

"""
```

```
Created on Sun Apr  3 18:26:55 2022.

@author: k_rsa.
"""

# %% Importa Bibliotecas necessárias


import re
import pandas as pd
import warnings
import emoji
import spacy
from gensim.models import Phrases

from nltk.tokenize import TweetTokenizer
from tqdm import tqdm

# Desabilita avisos que podem travar o codigo
warnings.filterwarnings("ignore", category=DeprecationWarning)

# %% Função para separar horario (15 as 19/ 18 as 22 em +GMT) \
#    e criar colunas com dados importantes para criação de chunks


def filtra_hora(df_inicio, hora_min, hora_max, minute_max):
    """Coloca campos horarios, cria coluna chunk e escolhe os campos."""
    print('Separação tw por hora iniciado:')
    df_inicio['data'] = pd.to_datetime(df_inicio['created_at'],
                                       format='%Y-%m-%d')
    df_inicio['ano'] = df_inicio['data'].dt.year
    df_inicio['mes'] = df_inicio['data'].dt.month
    df_inicio['dia'] = df_inicio['data'].dt.day
    df_inicio['hora'] = df_inicio['data'].dt.hour
    df_inicio['min'] = df_inicio['data'].dt.minute
    df_inicio['data']= df_inicio['data'].dt.strftime('%Y-%m-%d')
    df_mask = ((df_inicio['hora'] < hora_max) & (df_inicio['hora'] > hora_min)
            & (df_inicio['min'] < minute_max))
    df_out = df_inicio[df_mask]
    df_out['chunk'] = df_out['hora'] * 10 + round(df_out['min']/5, 0)
    df_out = df_out[['tweet_id', 'author_id', 'created_at', 'geo',
                     'like_count', 'quote_count', 'reply_count',
                     'retweet_count', 'text', 'media_keys',
                     'ano', 'mes','dia','chunk','data','hora','min']]
    df_out.reset_index(inplace=True)  # reindexa a base de dados
    print('Terminado separação por hora')
    return df_out

# %% Funções abaixo para converter emojis


def emoji_transf(text):
    """Função para converter emoji de um unico texto."""
    saida = []

    for i in text.split():
        if emoji.is_emoji(i):
            for e in emoji.demojize(i,
                                    language='pt').replace(':', "").split('-'):
```

```
            saida.append(e)
        else:
            saida.append(i)

    return ' '.join(map(str, saida))


# função para percorrer lista e para cada texto converter emoji

def emoji_transf_list(data):
    """Função para coverter listas de emojis."""
    out = []

    for i in tqdm(data):
        out_parcial = emoji_transf(i)
        out.append(out_parcial)

    print("Emoji convertido!")

    return out


# %% tokenização e pre_processamento com a remoção das stopwords
# para tokenizar sera utilizada uma biblioteca especial para textos do twitter

# Função para rodar o tokenizador e limpar o texto


def clean_tokenize(data, stop_words):
    """Usa tokenizador especifico para tweets,\
    letras minusculas, retira usernames."""
    tweet_tokenizer = TweetTokenizer(preserve_case=False,
                                     strip_handles=True, reduce_len=True)

    print(" Clean iniciado:")



    saida = []

    for i in tqdm(data):

        data_1 = i
#       #remove urls
        data_1 = re.sub(r'http\S+|www\S+|https\S+', '',
                        data_1, flags=re.MULTILINE)

        data_1 = re.sub(r'coranavíru|covid|Covid|coronavírus', 'covid',
                        data_1, flags=re.MULTILINE)
        data_1 = re.sub(r'mds|Deus', 'deus',
                        data_1, flags=re.MULTILINE)
        data_1 = re.sub(r'vacino|vacina|vacinação', 'vacinar',
                        data_1, flags=re.MULTILINE)
        data_1 = re.sub(r'anvisar', 'anvisa',
                        data_1, flags=re.MULTILINE)
        data_1 = re.sub(r'kkk', 'rir',
                        data_1, flags=re.MULTILINE)

#       #remove RT
```

```
        data_1 = re.sub(r'^RT\s+', '', data_1)
#       #remove the # symbol
        data_1 = re.sub('#', '', data_1)
#       #tokenize
        data_tokens = tweet_tokenizer.tokenize(data_1)
        del data_1
#       # remove stopwords and punctuation
        data_tokens = [tk for tk in data_tokens if tk not in stop_words]
#       # remove tokens that are only 1 char in length
        data_tokens = [tk for tk in data_tokens if len(tk) > 2]

        saida.append(data_tokens)


    print(" Fim do clean_tokenize! ")

    return saida

# %% Função para lemmatizar


def lem(data, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV', 'PRON']):
    """Lematiza as palavras."""
    nlp = spacy.load('pt_core_news_sm')
    print("lem iniciado")
    texts_out = []

    for da in tqdm(data):
        doc = nlp(" ".join(da))
        texts_out.append([token.lemma_ for token in doc
                          if token.pos_ in allowed_postags])


    print(" Fim do lem ! ")
    return texts_out


# %% Funções para criar Bigram e Trigram


def make_gram(c):
    """Gera bigramas."""
    bigram_mod = Phrases(c, min_count=100, threshold=10)
    # iteração para criação de bigrams
    out = []
    for x in tqdm(c):
        out.append(bigram_mod[x])
    return out


# %% Funções para ajuste fino nas palavras
def sub_palavra_fino(data):

    saida=[]
    for j in tqdm(data):
        saida_01 =[]
        for i in j:
            data_1 = i
```

```
                data_1    =    re.sub(r'coranavíru$|covid$|Covid$|coronavírus$|corona$|coronavirus$|coronar$',
'covid',
                        data_1)
                data_1 = re.sub(r'mds$|Deus$', 'deus',
                        data_1)
                data_1 = re.sub(r'^vacina$|vacinar$|vacino$|vacinação$|vacinarção$|vacinarr', 'vacinar',
                        data_1)
                data_1 = re.sub(r'anvisar$', 'anvisa',
                        data_1)
                data_1 = re.sub(r'kkk$', 'rir',
                        data_1)
                data_1 = re.sub(r'govern$', 'rir',
                        data_1)
                data_1 = re.sub(r'quarenteno$', 'quarentena',
                        data_1)
                data_1 = re.sub(r'actualização$', 'atualizar',
                        data_1)
                data_1 = re.sub(r'viru$', 'vírus',
                        data_1)
                data_1 = re.sub(r'vacinarr$|vacinarrr$|vacinarrrr$|vacinarrrrr$', 'vacinar',
                   data_1)


                   saida_01.append(data_1)
               saida.append(saida_01)

           return saida
```

Script principal de agrupamento

```
#!pip install spacy
#!pip install SciPy  --upgrade --user


# pip install emoji
#rodar comando abaixo no prompt do anaconca
#python -m spacy download pt
01. importa bibliotecas que serão necessárias

# garantir limpeza da memoria eliminando qualquer variável.

from IPython import get_ipython
get_ipython().magic('reset -sf')

# Set up log to external log file
# import logging
#    logging.basicConfig(filename='topic_modeling.log',    format='%(asctime)s    :    %(levelname)s    :
%(message)s', level=logging.INFO)

# Set up log to terminal
import logging
logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)

# pacotes

# sistemas
import os
import time
```

```
#basicos
import re
import numpy as np
import pandas as pd


import warnings
warnings.filterwarnings("ignore",category=DeprecationWarning)


#importar biblioteca desenvolvida por KS
from data_clean import *
2. Importando arquivo com textos a serem utilizados e criação de arquivos "chunks"

# abrir diretorio de repositorio de dados
os.chdir('Dados09')
# Importar base com os textos para fazer a modelagem de topico

print('Arquivos disponíveis na pasta de dados :'+str(os.listdir()))
datafile_2020 = 'tw_2020.csv'
datafile_2021 = 'tw_2021.csv'
datafile_2022 = 'tw_2022.csv'

df_20 = pd.read_csv(datafile_2020, encoding='utf-8')
print("Number of tweets 2020: ",len(df_20['id']))
df_21 = pd.read_csv(datafile_2021, encoding='utf-8')
print("Number of tweets 2021: ",len(df_21['id']))
df_22 = pd.read_csv(datafile_2022, encoding='utf-8')
print("Number of tweets 2022: ",len(df_22['id']))

del datafile_2020, datafile_2021, datafile_2022

# visao de todos os tweets

df_inicio = df_20.append(df_21, ignore_index=True)
# analisando arquivo carregado
#print("Number of tweets: ",len(df_inicio['id']))
df_inicio = df_inicio.append(df_22, ignore_index=True)
# analisando arquivo carregado

#df_inicio = df_21.append(df_22, ignore_index=True)
print("Number of tweets: ",len(df_inicio['id']))

# eliminar tweets duplicados

df_inicio = df_inicio.drop_duplicates(subset = ['tweet_id'])
print("Number of tweets sem duplicação  : ",len(df_inicio['id']))

#limpar horarios e incluir campos para ajudar a definir chunks
df = filtra_hora(df_inicio, 17, 24, 70)


del df_20, df_21, df_22, df_inicio

df.shape

#Seleciona os principais campos e formata o campo data
```

```
# converter campo de data para ficar equivalente ao campo do banco de dados sus
df['date'] = pd.to_datetime(df.created_at)
df['date'] = df['date'].dt.strftime('%Y-%m-%d')

df.tail(10)

start_date = '2020-01-01'
end_date = '2022-12-31'
hora = 20

df_recorte = df[df.date < end_date]
df_recorte = df_recorte[df_recorte.date > start_date]
df_recorte = df_recorte[df_recorte.hora == hora]

df_recorte.shape

df_recorte.tail(10)

#salvar arquivo com os tw com as datas selecionadas
os.chdir('clean_data/chunks')

filename = 'recorte_tw_.csv'
df_recorte.to_csv(filename)



consolidado_dia_tw = df.groupby(['data','dia',
                          'mes', 'ano'])['tweet_id'].count()

consolidado_dia_tw.head(5)

consolidado_dia_tw.to_csv('consolidado_dia_tw.csv')

consolidado_hora_tw = df.groupby(['hora','data','dia',
                          'mes', 'ano'])['tweet_id'].count()
consolidado_hora_tw.to_csv('consolidado_hora_tw.csv')
```

2.1 Caso queira selecionar periodo (somente dados selecionados de 2021 e 2022) e depois salvar o arquivo com os dados utilizados

```
#df_inicio = df_20.append(df_21, ignore_index=True) # analisando arquivo carregado #print("Number of tweets: ",len(df_inicio['id'])) #df_inicio = df_inicio.append(df_22, ignore_index=True) # analisando arquivo carregado df_inicio = df_21.append(df_22, ignore_index=True) print("Number of tweets: ",len(df_inicio['id']))

# eliminar tweets duplicados
df_mask = df['hora'] == 20
df_out = df[df_mask]

df_out.shape

#Salavar dados em arquivo
os.chdir('clean_data\chunks')
df_out.to_csv('dados_usados_h20.csv',encoding='utf-8')
```

3. Criar arquivos diferentes para ajudar no processamento

```
# criar lista com id dos chunks
q1 = df_out.groupby(['chunk'])['tweet_id'].count()
q1 = q1.reset_index()
grupo = q1.chunk.values.tolist()
```

```
i =1
for g in grupo:
   d_chunk = []
   df_mask=((df_out['chunk'] == g))
   d_chunk = df_out[df_mask]
   filename = 'chunk'+str(i)+'.csv'
   d_chunk.to_csv(filename)
   i +=1
   print(filename)

quadro = df_out.groupby(['chunk'])['tweet_id'].count()

quadro

tw_geo = df_out.dropna(subset = ['geo'])

tw_geo.shape

tw_geo.to_csv('tw_gwo.csv',encoding='utf-8')
```

Preparação dos dados

```
01. importa bibliotecas que serão necessárias

# garantir limpeza da memoria eliminando qualquer variável.

from IPython import get_ipython
get_ipython().run_line_magic('reset', '-sf')
#get_ipython().run_line_magic(DeprecationWarning)

# Set up log to external log file
# import logging
#  logging.basicConfig(filename='topic_modeling.log',  format='%(asctime)s  :  %(levelname)s  :
%(message)s', level=logging.INFO)

# Set up log to terminal
import logging
logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)

# sistemas
import os
import time
from tqdm import tqdm

#basicos
import re
import numpy as np
import pandas as pd
pd.set_option('display.max_colwidth',1000)
from pprint import pprint

# lista de pontuação
from string import punctuation

#nltk
```

```
from nltk.corpus import stopwords

import warnings
warnings.filterwarnings("ignore",category=DeprecationWarning)

#importar biblioteca desenvolvida por KS
from data_clean import *

# Criar lista de palavras a serem retiradas
# Stop words
# Criar lista com as stop_words em portuges , pontuação e lista complementar caso necessario
list_comp =['q','º','°','pro','de']
stop_words = stopwords.words('portuguese') + list(punctuation)+list_comp

# abrir diretorio de repositorio de dados
os.chdir('Dados09/clean_data/chunks')
```

2. Função para charmar funções de pre-processamento dos textos (utilizando funções da biblioteca data_clean)

```
def pre_proc(din, stw):
   # conversao dos emojis
   d = emoji_transf_list(din)

   # limpando os textos, tokenizando e Lematizando
   d = clean_tokenize(d, stw)

   #Lematizar os tokens
   d = lem(d)

   #ajuste fino de palavras
   d = sub_palavra_fino(d)

   #Bigramas e Trigramas
   d = make_gram(d)

   return d
```

3.2 Importar base com os textos para pre-processamento

```
# arquivos disponiveis
print('Arquivos disponíveis na pasta de dados :'+str(os.listdir()))

import re
datafile = 'recorte_tw.csv'
dout_in = pd.read_csv(datafile, encoding='utf-8')

###### ESCOLHE periodo para teste ou pode ser o dout inteiro. dout=dout_in.head(100)

dout = dout_in
dout.shape

#2. faz pre-proc do chunk primeira fase

d = dout.text.values.tolist()
d = pre_proc(d, stop_words)

len(d)

d
```

```
#3. verifica quantos elementos do chunk pre_proces são nulos
a = 0
for j in tqdm(range(len(d))): #bow_corpus is the corpus
    if len(d[j])==0: #check for empty document
            a = a+1
print("# de linhas com zero tokens", a)

# 4. transformar chunk pre_proces em pandas

dl = pd.Series(d)
dl = pd.DataFrame(dl)
dl.columns = ['tokens_gram']
dl.reset_index(drop=True, inplace=True)



#5. Agregar as linhas do chunk original com o chunk pre_proces com clean_data
dc = pd.concat([dout, dl], axis=1, join = 'inner')

#6. Cria coluna num_tokens_gram
dc['num_tokens_gram'] = 1


len(dc)

#7. Conta quantos tokens existem em cada coluna
#usando progress_map a velocidade muda de 10horas para 3 segundos e apresenta barra de evolução!!!

tqdm.pandas()  # <- incluido para apresentar barra de evolução
dc.loc[:,'num_tokens_gram'] = dc.loc[:,'tokens_gram'].progress_map(lambda calc: len(calc))

dc.head(5)

#8. Refazer o dout sem as coluna que tem zero tokens
dc = dc.drop(dc[(dc.num_tokens_gram)==0].index)

#9. Confirmar que quantidade de tokens vazios seja igual a zero
a = 0
for y in dc['tokens_gram']:
    if len(y) ==0:
        a = a+1
print("# de linhas com zero tokens após limpeza:", a)

#10. Salvar arquivo final
datafile_out = 'recorte_tw_pre_proc.csv'
dc.to_csv(datafile_out, encoding='utf-8')
print("Arquivo salvo:", datafile_out)

dc.head(5)

len(dc)
```

Script para criar dicionário com os dados preparados

```
01. importa bibliotecas que serão necessárias

# garantir limpeza da memoria eliminando qualquer variável.
```

```
from IPython import get_ipython
get_ipython().run_line_magic('reset', '-sf')

# Set up log to external log file
# import logging
#    logging.basicConfig(filename='topic_modeling.log',    format='%(asctime)s    :    %(levelname)s    :
%(message)s', level=logging.INFO)

# Set up log to terminal
import logging
logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)

# pacotes

# sistemas
import os
import time
from tqdm import tqdm

#basicos
import re
import numpy as np
import pandas as pd
pd.set_option('display.max_colwidth',1000)
from pprint import pprint


#Gensim
import gensim
import gensim.corpora as corpora
from gensim.utils import simple_preprocess
from gensim.models import CoherenceModel
from gensim.models import Phrases
from gensim.models import  LdaMulticore

# spacy for Lemmatization
import spacy

import warnings
warnings.filterwarnings("ignore",category=DeprecationWarning)

#importar biblioteca desenvolvida por KS
from data_clean import *
from modelagem import *
from topicos_textos import *

# abrir diretorio de repositorio de dados
os.chdir('Dados09/clean_data/chunks')
3.2 Importar base com os textos

dout = pd.DataFrame()
#1. carregar arquivo i do chunk
datafile = 'recorte_tw_pre_proc.csv'

dread = pd.read_csv(datafile, encoding='utf-8')

print("Iniciando o pre_processamento do arquivo:", datafile)
dout = dout.append(dread)
```

```
dout.reset_index(drop=True, inplace=True)

dout['tokens_gram'][0:30]

dout.shape
4. Criando dicionario

from tqdm import tqdm
dout_list = []

for i in tqdm(range(len(dout))):
   dout_list.append(eval(dout['tokens_gram'][i]))

os.chdir('lda_model')
# Criando dicionário necessário para a modelagem de topico
# as duas principais entradas para a modelagem de topico com LDA é o dicionário de palavras (id2word)
# que atribui um id para cada palavra e o corpus que cria uma lista com o id da palavra e quantidade de
vezes que ela repete no doc

# criando dicionário
id2word = corpora.Dictionary(dout_list)
# Save the Dict
id2word.save('mydict_recorte_tw.dict')  # save dict to disk
```

Script para modelagem de topico com LDA

Consolidar tweets com tópicos e base casos covid

```
# garantir limpeza da memoria eliminando qualquer variável.

from IPython import get_ipython
get_ipython().run_line_magic('reset', '-sf')
#get_ipython().run_line_magic(DeprecationWarning)

# os.chdir('Dados09/clean_data/chunks_teste/lda_teste')
# 1. importar bibliotecas


import os
import glob


import numpy as np   #https://numpy.org/
import matplotlib.pyplot as plt #https://matplotlib.org/
import pandas as pd #https://pandas.pydata.org/
import seaborn as sns # https://seaborn.pydata.org/

# 2. Coletar base de dados

## base de casos fonte SUS https://covid.saude.gov.br/
# agrupar varios arquivos csv em um unico
os.chdir('Sus')

csv_files = glob.glob('*.csv')
```

```
for a in csv_files:
   print(a)

data_sus = []
data_sus = pd.DataFrame(data_sus)
for f in csv_files:
   a = pd.read_csv(f, encoding = 'utf-8', sep=';')
   print(a.shape)
   data_sus = data_sus.append(a)

data_sus.shape

# base de casos por estado e total Brasil

data_sus_sumario = data_sus.groupby(['data','estado','municipio'], as_index = False)['casosNovos','obitosNovos','populacaoTCU2019'].sum()#.pivot('data','estado').fillna(0)
data_sus_municipio = data_sus_sumario.groupby(['data','estado'], as_index = False)['casosNovos','obitosNovos'].sum().pivot('data','estado').fillna(0)
data_sus_br = data_sus_municipio['casosNovos']
data_sus_br['BR']=data_sus_br.sum(axis = 1)

del data_sus_sumario,data_sus_municipio

#base com população
#selecionar uma data
data_sus_pop = data_sus.loc[(data_sus['data']=='2022-03-01')].copy()
data_sus_pop_sumario = data_sus_pop.groupby(['estado','municipio'], as_index = False)['populacaoTCU2019'].sum()#.pivot('data','estado').fillna(0)
data_sus_pop_br = data_sus_pop_sumario.groupby(['estado'], as_index = False)['populacaoTCU2019'].sum()#.pivot('estado').fillna(0)

#inserir linha com dados total do Brasil
s = data_sus_pop_br['populacaoTCU2019'].sum()
data_sus_pop_br = data_sus_pop_br.append({'estado':'BR', 'populacaoTCU2019': s}, ignore_index = True)

del data_sus_pop, data_sus_pop_sumario, s

plt.figure(figsize=(16,8))
ax = plt.subplot(1,2,1)
ax.set_title('Casos Diários', fontsize=18, loc='left')
plt.plot(data_sus_br['SP'], label='São Paulo')
plt.plot(data_sus_br['RJ'], label='Rio de Janeiro')
plt.xlabel("Dia")
plt.ylabel("Casos")
plt.legend();

ax = plt.subplot(1,2,2)
ax.set_title('Casos Diários', fontsize=18, loc='left')
plt.plot(data_sus_br['BR'], label='Brasil')
plt.xlabel("Dia")
plt.ylabel("Casos")

plt.legend();

data_sus_br

# alguns estados por habitantes
```

```
data_sus_br['BR_hab'] = data_sus_pop_br[data_sus_pop_br['estado'] == 'BR']['populacaoTCU2019'].values[0]
data_sus_br['SP_hab'] = data_sus_pop_br[data_sus_pop_br['estado'] == 'SP']['populacaoTCU2019'].values[0]
data_sus_br['AM_hab'] = data_sus_pop_br[data_sus_pop_br['estado'] == 'AM']['populacaoTCU2019'].values[0]

data_sus_br['BR_phab'] = data_sus_br['BR']/data_sus_br['BR_hab']
data_sus_br['SP_phab'] = data_sus_br['SP']/data_sus_br['SP_hab']
data_sus_br['AM_phab'] = data_sus_br['AM']/data_sus_br['AM_hab']

data_sus_br
# selecionar somente linhas do Total do pais (coluna regiao = Brasil) data_sus_br = data_sus.loc[(data_sus['regiao']=='Brasil') & (data_sus['casosNovos'] != 0)].copy() data_sus_br = data_sus_br.reset_index() data_sus_br.shape#deixando somente as colunas que serão utilizadas data_sus_br = data_sus_br[['data','casosNovos','obitosNovos']]data_sus_br#https://seaborn.pydata.org/generated/seaborn.lineplot.html plt.figure(figsize=(15,15)) plt.annotate('Ominicrom', xy=('2022-01-11', 300000),arrowprops=dict(arrowstyle='->'), xytext=('2021-05-11', 300000)) plt.title('Casos diarios de covid no Brasil') sns.lineplot(data=data_sus_br, x='data', y='casosNovos')

os.chdir('..')
2. Criando bases com tweets
2.1 Carregando base de dados de tw

os.chdir('Dados09/clean_data/chunks')

os.chdir('..')
os.chdir('..')

print('Arquivos disponíveis na pasta de dados :'+str(os.listdir()))

datafile_1 = 'text_topic_estado.csv' #input('Digite o nome do arquivo:')
data_tw = pd.DataFrame
data_tw = pd.read_csv(datafile_1, encoding='utf-8')

data_tw.head(5)

print(data_tw.tweet_id.count())

print('GPS')
print(data_tw.geo.count())
print(data_tw.geo.count()/data_tw.tweet_id.count()*100)


print('Descrição')
print(data_tw.Sigla.count())
print(data_tw.Sigla.count()/data_tw.tweet_id.count()*100)

#Seleciona os principais campos e formata o campo data

data_tw_out = data_tw[['tokens_gram','Perc_Contribution','created_at','Dominant_Topic','text','tweet_id','Sigla','pais']]

# converter campo de data para ficar equivalente ao campo do banco de dados sus
data_tw_out['created_at'] = pd.to_datetime(data_tw_out.created_at)
data_tw_out['created_at'] = data_tw_out['created_at'].dt.strftime('%Y-%m-%d')
data_tw_out.rename(columns={'created_at': 'data'}, inplace = True)

data_tw_out
```

2.2. Selecionando os top 50 de cada topico

b = pd.DataFrame() for i in range(0,10): best_tw_topic = data_tw.loc[(data_tw['Dominant_Topic']==i)].copy() a = best_tw_topic.sort_values(by=['Perc_Contribution'], ascending=False) b = b.append(a[['text','tokens_gram','Dominant_Topic','Perc_Contribution','created_at','tweet_id']].head(10000)) filename = 'best_tw_topic_recorte.csv' b.to_csv(filename)c = b.loc[(b['Dominant_Topic']==9)].copy() #pd.options.display.width pd.set_option('max_colwidth', 500) c.iloc[1500:1550] #c.iloc[100:150]

2.3. Gera arquivo de saida

Dados SP e AM

```
data_tw_exp_2 = data_tw_out[['data','Dominant_Topic','Sigla','pais']]

data_tw_exp_2.head(5)

data_tw_sp = data_tw_exp_2[data_tw_exp_2['Sigla']=='BR-SP']
data_tw_am = data_tw_exp_2[data_tw_exp_2['Sigla']=='BR-AM']

data_tw_sp_exp_t = data_tw_sp[['data','Dominant_Topic']]
data_tw_sp_exp_t.rename(columns={'Dominant_Topic': 'Dominant_Topic_SP'}, inplace = True)
data_tw_sp_exp = pd.get_dummies(data_tw_sp_exp_t, columns = ['Dominant_Topic_SP'])
data_tw_sp_gr = data_tw_sp_exp.groupby(['data'], as_index = False).sum()

data_tw_am_exp_t = data_tw_am[['data','Dominant_Topic']]
data_tw_am_exp_t.rename(columns={'Dominant_Topic': 'Dominant_Topic_AM'}, inplace = True)
data_tw_am_exp = pd.get_dummies(data_tw_am_exp_t, columns = ['Dominant_Topic_AM'])
data_tw_am_gr = data_tw_am_exp.groupby(['data'], as_index = False).sum()

data_tw_am_gr.head(5)

data_tw_sp_gr.head(5)
```

Arquivo de Saída BR

```
data_tw_exp_1 = data_tw_out[['data','Dominant_Topic']]
data_tw_exp = pd.get_dummies(data_tw_exp_1, columns = ['Dominant_Topic'])

data_tw_exp.head(10)

data_tw_gr = data_tw_exp.groupby(['data'], as_index = False).sum()
data_tw_gr

#https://pandas.pydata.org/docs/user_guide/merging.html
#inner em ambos somente, preenche todos da primeira
result = pd.merge(data_sus_br, data_tw_gr, on="data", how="left") #inner em ambos somente, preenche todos da primeira
result = result.dropna(subset = ['Dominant_Topic_0'])

result

result = pd.merge(result, data_tw_sp_gr, on="data", how="left") #inner em ambos somente, preenche todos da primeira
result = pd.merge(result, data_tw_am_gr, on="data", how="left") #inner em ambos somente, preenche todos da primeira

result.info()

filename = 'result_full_recorte_local.csv'
result.to_csv(filename)
```

Script para a projeção

https://acervolima.com/python-modelo-arima-para-previsao-de-serie-temporal/!pip install pmdarima
1. Importar bibliotecas necessárias

```
# os.chdir('Dados09/clean_data/chunks_teste/lda_teste')
# 1. importar bibliotecas

import os
import glob
from datetime import datetime

# eliminando warnings das bibliotecas
import warnings
warnings.filterwarnings('ignore')

# pandas
import pandas as pd

#bibliotecas para autocrrelação
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tsa.seasonal import seasonal_decompose as ets
from statsmodels.graphics.gofplots import qqplot
from statsmodels.graphics.tsaplots import plot_acf


# bibliotecas para graficos
import matplotlib.pyplot as plt
import seaborn as sns # https://seaborn.pydata.org/


#https://scikit-learn.org/stable/
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import PolynomialFeatures
2. Criação de funções utilizadas

# função para treinar o algortimo:
def modelo(d, X_in=1, y_in=1):

    #definindo parametro do modelo polynomial grau 2
    poli_reg = PolynomialFeatures(degree = d)

    #convertendo variável X para polinomial
    X_m = pd.DataFrame(poli_reg.fit_transform(X_in))
    y_m = y_in

    #Treinando o Modelo
    X_m = sm.add_constant(X_in) # insere contante B0
    model_sm = sm.OLS(y_m, X_m).fit()

    return(model_sm)


# analise gráfica
def graficos(model):
```

```
    residuos = model.resid

    fig, ax = plt.subplots(2,2, figsize=(17,10))
    residuos.plot(title = 'Residuos', ax = ax[0][0])

    sns.distplot(residuos, ax=ax[0][1])

    #plot_acf(residuos, lags =30, ax = ax[1][0])
    qqplot(residuos, line='s', ax=ax[1][1])
    plt.show()


# fazer previsões

def predict(model, X= 1, y=1):


    # Cria variável constante
    X = sm.add_constant(X)

    # Faz previsões com o modelo e dados de entrada
    casos_predict = model.predict(X)
    casos_predict = pd.DataFrame(casos_predict, columns = ['predict'])

    # Criar pandas
    dados_comp= pd.DataFrame()
    dados_comp.index = y.index

    #inserir dados originais e dados previstos
    dados_comp['diff_cN'] =y['diff_cN']
    dados_comp['predict'] =casos_predict['predict']

    #gerar grafico
    plt.rcParams.update({'figure.figsize':(9,3), 'figure.dpi':120})
    dados_comp.plot()

os.chdir('Dados09')
#print('Arquivos disponíveis na pasta de dados :'+str(os.listdir()))
3 Tratamento dos dados e criação dos campos necessários

#arquivo = pd.read_csv('result_full.csv')
arquivo = pd.read_csv('result_full_recorte_local.csv')

arquivo.data = pd.to_datetime(arquivo.data, format="%Y-%m-%d")
arquivo.set_index("data", drop = False, inplace = True)
arquivo.index = pd.DatetimeIndex(arquivo.index).to_period('W')

arquivo['Total_Topic'] = (arquivo['Dominant_Topic_0'] +
                          arquivo['Dominant_Topic_1'] +
                          arquivo['Dominant_Topic_2'] +
                          arquivo['Dominant_Topic_3'] +
                          arquivo['Dominant_Topic_4'] +
                          arquivo['Dominant_Topic_5'] +
                          arquivo['Dominant_Topic_6'] +
                          arquivo['Dominant_Topic_7'] +
                          arquivo['Dominant_Topic_8'] +
                          arquivo['Dominant_Topic_9'])
```

```
arquivo['Sinal'] = (arquivo['Dominant_Topic_5'] +
                    arquivo['Dominant_Topic_7'])

arquivo

arquivo['BR'].plot(figsize=(15, 10))

arquivo['mes_ano']= arquivo['data']
arquivo['Semana'] = arquivo['mes_ano'].dt.to_period('W')

#agrupamento por semana
arquivo_w = arquivo.groupby(by=['Semana']).sum()

arquivo_w

arquivo_w['BR'].plot(figsize=(15, 10))
arquivo_w['SP'].plot(figsize=(15, 10))
arquivo_w['AM'].plot(figsize=(15, 10))

plt.figure(figsize=(16,8))
ax = plt.subplot(1,2,1)
ax.set_title('Casos Semanais', fontsize=18, loc='left')
arquivo_w['SP'].plot()
arquivo_w['AM'].plot()
plt.xlabel("Semana")
plt.ylabel("Casos")
plt.legend();

ax = plt.subplot(1,2,2)
ax.set_title('Casos Semanais', fontsize=18, loc='left')
arquivo_w['BR'].plot()
plt.xlabel("Semana")
plt.ylabel("Casos")

plt.legend();

arquivo_w['Dominant_Topic_3'].plot(figsize=(15, 10))

#Escolhe periodo
#start_date = '2021-01-01'


start_date = '2020-10-26'
end_date = '2022-12-01'

temporaria = arquivo_w[['BR','Dominant_Topic_0',
                  'Dominant_Topic_1','Dominant_Topic_2',
                  'Dominant_Topic_3','Dominant_Topic_4',
                  'Dominant_Topic_5','Dominant_Topic_6',
                  'Dominant_Topic_7','Dominant_Topic_8',
                  'Dominant_Topic_9','Total_Topic']]

temporaria = temporaria[temporaria.index < end_date]
temporaria = temporaria[temporaria.index > start_date]

# Tabela de correlação
corr = temporaria.corr()
import seaborn as sns # https://seaborn.pydata.org/
#Vamos ver a correlação graficamente
```

```
f, ax = plt.subplots(figsize=(15, 10))
sns.heatmap(corr, cmap=sns.color_palette("Blues"), linewidths=.5, annot=True);

#del temporaria
lag=pd.DataFrame()
lag['casosNovos'] = arquivo_w['BR']
lag['Total_Topic'] = arquivo_w['Total_Topic']
lag['Total_Topic1'] = arquivo_w['Total_Topic'].shift(1)
lag['Total_Topic2'] = arquivo_w['Total_Topic'].shift(2)
lag['Total_Topic3'] = arquivo_w['Total_Topic'].shift(3)
lag['Total_Topic4'] = arquivo_w['Total_Topic'].shift(4)
lag['Total_Topic5'] = arquivo_w['Total_Topic'].shift(5)
lag['Total_Topic6'] = arquivo_w['Total_Topic'].shift(6)
lag['Total_Topic7'] = arquivo_w['Total_Topic'].shift(7)
lag['Total_Topic8'] = arquivo_w['Total_Topic'].shift(8)



lag = lag[lag.index < end_date]
lag = lag[lag.index > start_date]

# Tabela de correlação
corr = lag.corr()
import seaborn as sns # https://seaborn.pydata.org/
#Vamos ver a correlação graficamente
f, ax = plt.subplots(figsize=(15, 10))
sns.heatmap(corr, cmap=sns.color_palette("Blues"), linewidths=.5, annot=True);

dados = arquivo_w
#dados = arquivo_p


dados['diff_cN'] = dados['BR'].diff()
dados['diff_cN_lag1'] = dados['diff_cN'].shift(1)
dados['diff_cN_lag2'] = dados['diff_cN'].shift(2)
dados['diff_cN_lag3'] = dados['diff_cN'].shift(3)
dados['diff_cN_lag4'] = dados['diff_cN'].shift(4)
dados['diff_cN_lag5'] = dados['diff_cN'].shift(5)
dados['diff_cN_lag6'] = dados['diff_cN'].shift(6)


dados['diff_Total'] = dados['Total_Topic'].diff()
dados['diff_Total_lag1'] = dados['diff_Total'].shift(1)
dados['diff_Total_lag2'] = dados['diff_Total'].shift(2)


dados['diff_T0'] = dados['Dominant_Topic_0'].diff()
dados['diff_T0_lag1'] = dados['diff_T0'].shift(1)
dados['diff_T0_lag2'] = dados['diff_T0'].shift(2)



dados['diff_T1'] = dados['Dominant_Topic_1'].diff()
dados['diff_T1_lag1'] = dados['diff_T1'].shift(1)
dados['diff_T1_lag2'] = dados['diff_T1'].shift(2)


dados['diff_T2'] = dados['Dominant_Topic_2'].diff()
dados['diff_T2_lag1'] = dados['diff_T2'].shift(1)
```

```
dados['diff_T2_lag2'] = dados['diff_T2'].shift(2)


dados['diff_T3'] = dados['Dominant_Topic_3'].diff()
dados['diff_T3_lag1'] = dados['diff_T3'].shift(1)
dados['diff_T3_lag2'] = dados['diff_T3'].shift(2)


dados['diff_T4'] = dados['Dominant_Topic_4'].diff()
dados['diff_T4_lag1'] = dados['diff_T4'].shift(1)
dados['diff_T4_lag2'] = dados['diff_T4'].shift(2)


dados['diff_T5'] = dados['Dominant_Topic_5'].diff()
dados['diff_T5_lag1'] = dados['diff_T5'].shift(1)
dados['diff_T5_lag2'] = dados['diff_T5'].shift(2)


dados['diff_T6'] = dados['Dominant_Topic_6'].diff()
dados['diff_T6_lag1'] = dados['diff_T6'].shift(1)
dados['diff_T6_lag2'] = dados['diff_T6'].shift(2)


dados['diff_T7'] = dados['Dominant_Topic_7'].diff()
dados['diff_T7_lag1'] = dados['diff_T7'].shift(1)
dados['diff_T7_lag2'] = dados['diff_T7'].shift(2)


dados['diff_T8'] = dados['Dominant_Topic_8'].diff()
dados['diff_T8_lag1'] = dados['diff_T8'].shift(1)
dados['diff_T8_lag2'] = dados['diff_T8'].shift(2)


dados['diff_T9'] = dados['Dominant_Topic_9'].diff()
dados['diff_T9_lag1'] = dados['diff_T9'].shift(1)
dados['diff_T9_lag2'] = dados['diff_T9'].shift(2)
3.1 Seleciona periodo a ser utilizado

result = dados[dados.index < end_date]
result = result[result.index > start_date]
result

# Tabela de correlação
corr = result.corr()
import seaborn as sns # https://seaborn.pydata.org/
#Vamos ver a correlação graficamente
f, ax = plt.subplots(figsize=(40, 20))
sns.heatmap(corr, cmap=sns.color_palette("Reds"), linewidths=.5, annot=True);
4. Cria modelo com dados somente de casos

X_result = result[['diff_cN_lag1','diff_cN_lag2',

                   'diff_T0_lag1','diff_T0_lag2',
                   'diff_T1_lag1','diff_T1_lag2',
                   'diff_T2_lag1','diff_T2_lag2',
                   'diff_T3_lag1','diff_T3_lag2',
                   'diff_T4_lag1','diff_T4_lag2',
                   'diff_T5_lag1','diff_T5_lag2',
                   'diff_T6_lag1','diff_T6_lag2',
```

```
                'diff_T7_lag1','diff_T7_lag2',
                'diff_T8_lag1','diff_T8_lag2',
                'diff_T9_lag1','diff_T9_lag2',



               #'diff_T3_lag1','diff_T3_lag2'
                #'diff_Sinal_lag1','diff_Sinal_lag2'
               ]]

y_result = result[['diff_cN']]

# divide entre treino e teste
#X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
#
#

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)

model = modelo(1, X_train, y_train)
model.summary()
5. Cria Modelo e analisa os resultados

#somente semanas anteriores
X_result = result[['diff_cN_lag1','diff_cN_lag2','diff_cN_lag3'
                ]]

y_result = result[['diff_cN']]

# divide entre treino e teste
#X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
#
#

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)

model = modelo(1, X_train, y_train)
model.summary()

X_result = result[['diff_cN_lag1','diff_cN_lag2',
                #'diff_cN_lag3',#'diff_cN_lag4',#'diff_cN_lag5','diff_cN_lag6',
                #'diff_T0_lag1',
```

```
                'diff_T0_lag2',
                #'diff_T1_lag1',
                'diff_T1_lag2',
                'diff_T2_lag1',
                #'diff_T2_lag2',
                #'diff_T3_lag1',
                #'diff_T3_lag2',
                #'diff_T4_lag1',
                #'diff_T4_lag2',
                #'diff_T5_lag1',
                'diff_T5_lag2',
                'diff_T6_lag1',
                #'diff_T6_lag2',
                'diff_T7_lag1',
                #'diff_T7_lag2',
                'diff_T8_lag1',
                #'diff_T8_lag2',
                'diff_T9_lag1',
                #'diff_T9_lag2',
               ]]

y_result = result[['diff_cN']]

# divide entre treino e teste
#X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
#
#

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)

model = modelo(1, X_train, y_train)
model.summary()

graficos(model)

# resultados de previsao recompostos

X = X_train
y = y_train

# Cria variável constante
X = sm.add_constant(X)

# Faz previsões com o modelo e dados de entrada
casos_predict = model.predict(X)
casos_predict = pd.DataFrame(casos_predict, columns = ['predict'])

# Criar pandas
dados_comp= pd.DataFrame()
dados_comp.index = y.index
```

```
#inserir dados originais e dados previstos
dados_comp['diff_cN'] =y['diff_cN']
dados_comp['predict'] =casos_predict['predict']

#gerar grafico
plt.rcParams.update({'figure.figsize':(9,3), 'figure.dpi':120})
dados_comp.plot()
grafico recompondo casos

#base com casos semanais + variações reais + projeção

dados_comp['Real'] = arquivo_w['BR']
dados_comp['Predict_total'] = 0
dados_comp['Predict_total'] = dados_comp['Real'].shift()+dados_comp['predict']
dados_comp['Erro'] = dados_comp['diff_cN']- dados_comp['predict']
dados_comp[['Real','diff_cN','predict','Erro','Predict_total']]
dados_comp_fim = dados_comp[['Real','Predict_total']]
dados_comp_fim.head(5)

#gerar grafico
plt.rcParams.update({'figure.figsize':(9,3)})
dados_comp_fim.plot()

#https://tedboy.github.io/statsmodels_doc/generated/generated/statsmodels.regression.linear_model.R
egressionResults.html

model.mse_model
model.mse_resid

from statsmodels.stats.outliers_influence import variance_inflation_factor

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["variavel"] = X_train.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(X_train.values, i)
                   for i in range(len(X_train.columns))]

print(vif_data)
6. Analise do modelo e ruidos

# analise gráfica
residuos = model.resid

fig, ax = plt.subplots(2,2, figsize=(20,15))
residuos.plot(title = 'Residuos', ax = ax[0][0])

sns.distplot(residuos, ax=ax[0][1])

plt.show()

round(residuos.mean(),5)

round(residuos.std(),5)

from sklearn.metrics import r2_score

a = dados_comp_fim.dropna()
```

```
a

r2_score(a['Real'],a['Predict_total'])
APLICADA EM AM E SP
São paulo

del dados
dados = pd.DataFrame()
dados[['CN', 'Dominant_Topic_0','Dominant_Topic_1','Dominant_Topic_2',
        'Dominant_Topic_3','Dominant_Topic_4','Dominant_Topic_5','Dominant_Topic_6',
        'Dominant_Topic_7','Dominant_Topic_8','Dominant_Topic_9']] = arquivo_w[['SP',
'Dominant_Topic_SP_0','Dominant_Topic_SP_1','Dominant_Topic_SP_2',

'Dominant_Topic_SP_3','Dominant_Topic_SP_4','Dominant_Topic_SP_5','Dominant_Topic_SP_6',

'Dominant_Topic_SP_7','Dominant_Topic_SP_8','Dominant_Topic_SP_9']]



dados.info()

def create_diff(dados):

    dados['diff_cN'] = dados['CN'].diff()
    dados['diff_cN_lag1'] = dados['diff_cN'].shift(1)
    dados['diff_cN_lag2'] = dados['diff_cN'].shift(2)
    dados['diff_cN_lag3'] = dados['diff_cN'].shift(3)
    dados['diff_cN_lag4'] = dados['diff_cN'].shift(4)
    dados['diff_cN_lag5'] = dados['diff_cN'].shift(5)
    dados['diff_cN_lag6'] = dados['diff_cN'].shift(6)


    dados['diff_T0'] = dados['Dominant_Topic_0'].diff()
    dados['diff_T0_lag1'] = dados['diff_T0'].shift(1)
    dados['diff_T0_lag2'] = dados['diff_T0'].shift(2)



    dados['diff_T1'] = dados['Dominant_Topic_1'].diff()
    dados['diff_T1_lag1'] = dados['diff_T1'].shift(1)
    dados['diff_T1_lag2'] = dados['diff_T1'].shift(2)


    dados['diff_T2'] = dados['Dominant_Topic_2'].diff()
    dados['diff_T2_lag1'] = dados['diff_T2'].shift(1)
    dados['diff_T2_lag2'] = dados['diff_T2'].shift(2)


    dados['diff_T3'] = dados['Dominant_Topic_3'].diff()
    dados['diff_T3_lag1'] = dados['diff_T3'].shift(1)
    dados['diff_T3_lag2'] = dados['diff_T3'].shift(2)


    dados['diff_T4'] = dados['Dominant_Topic_4'].diff()
    dados['diff_T4_lag1'] = dados['diff_T4'].shift(1)
    dados['diff_T4_lag2'] = dados['diff_T4'].shift(2)


    dados['diff_T5'] = dados['Dominant_Topic_5'].diff()
```

```
    dados['diff_T5_lag1'] = dados['diff_T5'].shift(1)
    dados['diff_T5_lag2'] = dados['diff_T5'].shift(2)


    dados['diff_T6'] = dados['Dominant_Topic_6'].diff()
    dados['diff_T6_lag1'] = dados['diff_T6'].shift(1)
    dados['diff_T6_lag2'] = dados['diff_T6'].shift(2)


    dados['diff_T7'] = dados['Dominant_Topic_7'].diff()
    dados['diff_T7_lag1'] = dados['diff_T7'].shift(1)
    dados['diff_T7_lag2'] = dados['diff_T7'].shift(2)


    dados['diff_T8'] = dados['Dominant_Topic_8'].diff()
    dados['diff_T8_lag1'] = dados['diff_T8'].shift(1)
    dados['diff_T8_lag2'] = dados['diff_T8'].shift(2)


    dados['diff_T9'] = dados['Dominant_Topic_9'].diff()
    dados['diff_T9_lag1'] = dados['diff_T9'].shift(1)
    dados['diff_T9_lag2'] = dados['diff_T9'].shift(2)

    return dados

def create_result(dados, start_date, end_date):

    result = dados[dados.index < end_date]
    result = result[result.index > start_date]
    return result

dados = create_diff(dados)
result = create_result(dados, start_date, end_date)

X_result = result[['diff_cN_lag1','diff_cN_lag2',

                   'diff_T0_lag1','diff_T0_lag2',
                   'diff_T1_lag1','diff_T1_lag2',
                   'diff_T2_lag1','diff_T2_lag2',
                   'diff_T3_lag1','diff_T3_lag2',
                   'diff_T4_lag1','diff_T4_lag2',
                   'diff_T5_lag1','diff_T5_lag2',
                   'diff_T6_lag1','diff_T6_lag2',
                   'diff_T7_lag1','diff_T7_lag2',
                   'diff_T8_lag1','diff_T8_lag2',
                   'diff_T9_lag1','diff_T9_lag2',




                  #'diff_T3_lag1','diff_T3_lag2'
                   #'diff_Sinal_lag1','diff_Sinal_lag2'
                  ]]

y_result = result[['diff_cN']]

    # divide entre treino e teste
    #X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
    #
```

```
  #

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)

model = modelo(1, X_train, y_train)

model.summary()

X_result = result[['diff_cN_lag1',#'diff_cN_lag2',
                   #'diff_cN_lag3','diff_cN_lag4',#'diff_cN_lag5','diff_cN_lag6',
                   #'diff_T0_lag1',
                   #'diff_T0_lag2',
                   'diff_T1_lag1',
                   'diff_T1_lag2',
                   #'diff_T2_lag1',
                   'diff_T2_lag2',
                   'diff_T3_lag1',
                   'diff_T3_lag2',
                   #'diff_T4_lag1',
                   'diff_T4_lag2',
                   'diff_T5_lag1',
                   #'diff_T5_lag2',
                   #'diff_T6_lag1',
                   #'diff_T6_lag2',
                   'diff_T7_lag1',
                   #'diff_T7_lag2',
                   'diff_T8_lag1',
                   #'diff_T8_lag2',
                   'diff_T9_lag1',
                   #'diff_T9_lag2',
                  ]]

y_result = result[['diff_cN']]

# divide entre treino e teste
#X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
#
#

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)

model = modelo(1, X_train, y_train)
model.summary()
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["variavel"] = X_train.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(X_train.values, i)
                   for i in range(len(X_train.columns))]

print(vif_data)

graficos(model)

# resultados de previsao recompostos

X = X_train
y = y_train

# Cria variável constante
X = sm.add_constant(X)

# Faz previsões com o modelo e dados de entrada
casos_predict = model.predict(X)
casos_predict = pd.DataFrame(casos_predict, columns = ['predict'])

# Criar pandas
dados_comp= pd.DataFrame()
dados_comp.index = y.index

#inserir dados originais e dados previstos
dados_comp['diff_cN'] =y['diff_cN']
dados_comp['predict'] =casos_predict['predict']

#gerar grafico
plt.rcParams.update({'figure.figsize':(9,3), 'figure.dpi':120})
dados_comp.plot()

#base com casos semanais + variações reais + projeção

dados_comp['Real'] = arquivo_w['SP']
dados_comp['Predict_total'] = 0
dados_comp['Predict_total'] = dados_comp['Real'].shift()+dados_comp['predict']
dados_comp['Erro'] = dados_comp['diff_cN']- dados_comp['predict']
dados_comp[['Real','diff_cN','predict','Erro','Predict_total']]
dados_comp_fim = dados_comp[['Real','Predict_total']]
#dados_comp_fim.head(5)

#gerar grafico
plt.rcParams.update({'figure.figsize':(9,3)})
dados_comp_fim.plot()

#https://tedboy.github.io/statsmodels_doc/generated/generated/statsmodels.regression.linear_model.RegressionResults.html

print(model.mse_model)
print(model.mse_resid)
print(round(residuos.mean(),5))
```

```
print(round(residuos.std(),5))
a = dados_comp_fim.dropna()
print(r2_score(a['Real'],a['Predict_total']))
Amazonas

del dados
dados = pd.DataFrame()
dados[['CN', 'Dominant_Topic_0','Dominant_Topic_1','Dominant_Topic_2',
        'Dominant_Topic_3','Dominant_Topic_4','Dominant_Topic_5','Dominant_Topic_6',
        'Dominant_Topic_7','Dominant_Topic_8','Dominant_Topic_9']] = arquivo_w[['AM',
'Dominant_Topic_AM_0','Dominant_Topic_AM_1','Dominant_Topic_AM_2',

'Dominant_Topic_AM_3','Dominant_Topic_AM_4','Dominant_Topic_AM_5','Dominant_Topic_AM_6',

'Dominant_Topic_AM_7','Dominant_Topic_AM_8','Dominant_Topic_AM_9']]



dados.info()

dados = create_diff(dados)
result = create_result(dados, start_date, end_date)

X_result = result[['diff_cN_lag1','diff_cN_lag2',

         'diff_T0_lag1','diff_T0_lag2',
         'diff_T1_lag1','diff_T1_lag2',
         'diff_T2_lag1','diff_T2_lag2',
         'diff_T3_lag1','diff_T3_lag2',
         'diff_T4_lag1','diff_T4_lag2',
         'diff_T5_lag1','diff_T5_lag2',
         'diff_T6_lag1','diff_T6_lag2',
         'diff_T7_lag1','diff_T7_lag2',
         'diff_T8_lag1','diff_T8_lag2',
         'diff_T9_lag1','diff_T9_lag2',




        #'diff_T3_lag1','diff_T3_lag2'
         #'diff_Sinal_lag1','diff_Sinal_lag2'
        ]]

y_result = result[['diff_cN']]

   # divide entre treino e teste
   #X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
   #
   #

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)
```

```
model = modelo(1, X_train, y_train)

model.summary()

X_result = result[['diff_cN_lag1','diff_cN_lag2',
                #'diff_cN_lag3',#'diff_cN_lag4',#'diff_cN_lag5','diff_cN_lag6',
                'diff_T0_lag1',
                'diff_T0_lag2',
                #'diff_T1_lag1',
                #'diff_T1_lag2',
                #'diff_T2_lag1',
                #'diff_T2_lag2',
                #'diff_T3_lag1',
                #'diff_T3_lag2',
                #'diff_T4_lag1',
                #'diff_T4_lag2',
                'diff_T5_lag1',
                'diff_T5_lag2',
                #'diff_T6_lag1',
                #'diff_T6_lag2',
                #'diff_T7_lag1',
                'diff_T7_lag2',
                #'diff_T8_lag1',
                #'diff_T8_lag2',
                'diff_T9_lag1',
                #'diff_T9_lag2',
               ]]

y_result = result[['diff_cN']]

# divide entre treino e teste
#X_train, X_test, y_train, y_test = train_test_split(X_result, y_result, test_size = 0.1)
#
#

X_train = X_result
y_train = y_result
X_test = X_result
y_test = y_result

X_train.sort_index(inplace=True)
y_train.sort_index(inplace=True)
X_test.sort_index(inplace=True)
y_test.sort_index(inplace=True)

model = modelo(1, X_train, y_train)
model.summary()

from statsmodels.stats.outliers_influence import variance_inflation_factor

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["variavel"] = X_train.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(X_train.values, i)
                   for i in range(len(X_train.columns))]
```

```
print(vif_data)

graficos(model)

# resultados de previsao recompostos

X = X_train
y = y_train

# Cria variável constante
X = sm.add_constant(X)

# Faz previsões com o modelo e dados de entrada
casos_predict = model.predict(X)
casos_predict = pd.DataFrame(casos_predict, columns = ['predict'])

# Criar pandas
dados_comp= pd.DataFrame()
dados_comp.index = y.index

#inserir dados originais e dados previstos
dados_comp['diff_cN'] =y['diff_cN']
dados_comp['predict'] =casos_predict['predict']

#gerar grafico
plt.rcParams.update({'figure.figsize':(9,3), 'figure.dpi':120})
dados_comp.plot()

#base com casos semanais + variações reais + projeção

dados_comp['Real'] = arquivo_w['AM']
dados_comp['Predict_total'] = 0
dados_comp['Predict_total'] = dados_comp['Real'].shift()+dados_comp['predict']
dados_comp['Erro'] = dados_comp['diff_cN']- dados_comp['predict']
dados_comp[['Real','diff_cN','predict','Erro','Predict_total']]
dados_comp_fim = dados_comp[['Real','Predict_total']]
#dados_comp_fim.head(5)

#gerar grafico
plt.rcParams.update({'figure.figsize':(9,3)})
dados_comp_fim.plot()

#https://tedboy.github.io/statsmodels_doc/generated/generated/statsmodels.regression.linear_model.R
egressionResults.html

print(model.mse_model)
print(model.mse_resid)
print(round(residuos.mean(),5))
print(round(residuos.std(),5))
a = dados_comp_fim.dropna()
print(r2_score(a['Real'],a['Predict_total']))
```