PROJETO CANGAÇO: UM ENSAIO SOBRE RECONHECIMENTO DE MISOGINIA ONLINE

Wilson Ronaldo de Souza Assis[1], Lucas Baggio Figueira[1]

[1]Faculdade de Tecnologia de FATEC Ribeirão Preto (FATEC)

Ribeirão Preto, SP – Brasil

wilson.assis@fatec.sp.gov.br,
lucas.figueira@fatec.sp.gov.br

Resumo. *O presente estudo realizou a exploração de técnicas de aprendizado de máquina supervisionado e de processamento de linguagem natural, aplicadas para realizar o reconhecimento de padrões de escrita que correspondam à comportamentos misóginos, praticados nas interações realizadas através da rede social Twitter, nos quais possam incorrer contravenções e crimes. Foram percorridas todas as etapas para criação de um modelo de aprendizado de máquina, com a obtenção de bons resultados e a clara inferência dos próximos passos para o aprimorar.*

Abstract. *The present study explored supervised machine learning and natural language processing techniques, applied to recognize the writing patterns that correspond to misogynistic behaviors, practiced in interactions through the social network Twitter, in which misdemeanors and crimes may be incurred. All steps were taken to create a machine learning model, with good results and a clear inference of the next steps to improve it.*

1. Introdução

A proposta do Projeto Cangaço, mais do que a criação propriamente dita de um modelo de aprendizado de máquina supervisionado, é a de realizar uma exploração das diferentes técnicas que envolvem a sua construção, com aplicação de alguns dos métodos clássicos de processamento de linguagem natural, usados no reconhecimento de padrões de escrita.

1.1. Contexto social

O escritor e jornalista Giovani Da Empoli, em seu último livro publicado, "Os Engenheiros do Caos", utilizando como plano de fundo a Itália, seu país natal, analisou a forma como movimentos nacional-populistas operam atualmente e os impactos por eles causados na sociedade, notadamente, nas interações através das redes sociais. Escreve o autor "a vulgaridade e os insultos não são mais tabus. Os preconceitos, o racismo e a discriminação de gênero saem do buraco. As mentiras e o conspiracionismo se tornam chaves de interpretação da realidade. E tudo isso apresentado como uma guerra sacrossanta para a libertação da voz do povo" (DA EMPOLI, p. 992, 2019).

Assim como o movimento nacional conhecido como Cangaço, entretanto de forma pacífica e norteado pelo Estado democrático de direito, o modelo desenvolvido por este trabalho poderá criar meios para colaborar com a busca pela equidade social neste contexto online, reconhecendo possíveis condutas contraventoras e criminosas, visando também facilitar a sua judicialização, uma vez que possibilita a captura dos dados

necessários para identificar potenciais agressores.

E, como escopo inicial, foi definido o reconhecimento de comportamentos misóginos, uma vez que a desigualdade entre gêneros remonta às sociedades antigas e ainda perdura. Então, permanecendo neste sentido de brasilidades, a "inteligência artificial" do Projeto Cangaço foi batizada como Maria Bonita, a MB.

"A coragem de desfazer um casamento falido para acompanhar o homem que desejava e a disposição para enfrentar fome, sede e perseguição policial em nome de um grande amor inspiraram gerações de mulheres por décadas." (NEGREIROS, p. 104, 2018)

1.2. Contexto técnico

A criação de modelos de aprendizados de máquina supervisionados (CRIAR, 2021), sejam quais forem seus objetivos e técnicas utilizadas, de maneira geral, passam por etapas semelhantes. Primeiramente, os dados são capturados na fonte definida para o estudo, e então é realizada uma exploração nos mesmos, avaliando sua viabilidade e quais são as necessidades de aprimoramento.

Após formar uma base de dados consistente, é necessário então efetuar seu pré-processamento, que corresponde a utilização de diversas estratégias de limpeza, buscando, posteriormente, aumentar a assertividade do modelo, assim como mitigar a incorrência de possíveis erros, por exemplo, causados pela ausência de determinada informação em alguns registros.

Na sequência, o pesquisador ou responsável designado para tal atribuição, realiza a supervisão dos dados. Manualmente, classifica as informações obtidas, definindo a qual das classes dentre aquelas previamente elencadas cada um dos registros pertence.

Por fim, é efetivamente realizado o processamento, isto é, são executados o treinamento e os testes do modelo, utilizando a base de dados refinada e supervisionada e, no caso do Projeto Cangaço, foram aplicadas as técnicas de processamento de linguagem natural (JURAFSKY, MARTIN, 2020). Este momento é crucial, pois nele é treinado o classificador do modelo. E, do confronto entre dados supervisionados e classificador criado, é calculada a acurácia dos testes. Assim, se os resultados obtidos forem considerados satisfatórios, o modelo poderá ser utilizado para reconhecer comportamentos semelhantes àquele objeto do estudo em desenvolvimento.

2. Materiais e Métodos

O Projeto Cangaço foi inteiramente desenvolvido utilizando a linguagem de programação Python, muito aplicada no contexto da ciência de dados por dispor de excelentes bibliotecas para auxiliar na construção dos modelos.

Como ambientes de desenvolvimento integrados, foram empregadas algumas ferramentas, buscando otimizar o desempenho e generalizar a execução do modelo em diferentes infraestruturas, sendo elas:

- Docker Desktop, versão 20.10.5 (WHAT, 2021): foi utilizado para criação de um container com uma instalação do Anaconda 3 (ANACONDA, 2021), o continuumio/anaconda3, baseado no Python versão 3.5 (PYTHON, 2020). Essa é uma distribuição com bibliotecas científicas já embarcadas.

- PyCharm Professional, versão 2021.1 x64 (PYCHARM, 2021): foram desenvolvidos módulos customizados para as necessidades do projeto, nas etapas de captura, pré-processamento e processamento, utilizando como interpretador aquele mesmo do container criado via Docker;
- Jupyter Notebook (THE JUPYTER, 2015): também instalado no mesmo container mencionado, o Jupyter foi utilizado para fazer a captura dos dados de interesse via *API* do Twitter (DOCS, 2021) e no pré-processamento, por serem procedimentos extensos, que demandam muito tempo, mas com baixo consumo da capacidade de computacional do equipamento;
- Google Colab: quanto maior a base de dados utilizada, maior será a exigência de poder de processamento para a criação do modelo de aprendizado de máquina, portanto a ferramenta do Google foi considerada mais apropriada, por disponibilizar gratuitamente uma infraestrutura robusta aos seus usuários;
- Git (BOOK, 2014): foi utilizado no versionamento do Projeto Cangaço, que se encontra disponível no endereço: https://github.com/WillAssisDev/Projeto_Cangaco.

3. Fluxo de trabalho

No Projeto Cangaço, as fases que compuseram seu desenvolvimento, evidentemente, são aquelas mesmas introduzidas no presente artigo, agora detalhadas.

3.1. Captura

O processo de captura dos dados de interesse é um dos mais importantes, uma vez que dele irá decorrer o modelo como um todo e a sua respectiva precisão. Desta forma, é fundamental obter informações de qualidade, tendo em vista os objetivos traçados para o trabalho, e em grande quantidade.

Neste estudo, como já mencionado, foi selecionada a rede social Twitter como a fonte de dados. Foi pleiteado junto a mesma o acesso de desenvolvedor a sua *API*, que foi aprovado após a participação em processo no qual foram expostos os objetivos acadêmicos do projeto.

De posse das credenciais de acesso desta *API*, foi possível autenticar a conexão do *notebook* maria_bonita/conjunto_dados/misoginia.ipynb com o servidor do Twitter, no qual foram realizadas as capturas de publicações, os *tweets*.

O método de captação, disponibilizado pela biblioteca *Tweepy* (TWEEPY, 2021), recebe uma lista de palavras alvo a serem monitoradas e, uma vez que um usuário qualquer utilizar uma ou mais delas em uma publicação, a captura acontece. As palavras definidas para este monitoramento foram os nomes de usuário de mulheres renomadas, sendo elas jornalistas, políticas, atrizes, cantoras e outras celebridades, por serem, lamentavelmente, as vítimas mais recorrentes de misoginia nas redes. Assim foi produzida a lista *screen_names*.

```
screen_names = [
    "@ademaravilha", "@amorimvivian_", "@anaclaraac", "@ANAMARIABRAGA", "@ana_furtado", "@AndreiaSadi", "@angelicaksy",
    "@Anitta", "@ahickmann", "@bernardipaloma", "@bollemdb", "@BruMarquezine", "@brumelianebrum", "@CamilaPitanga",
    "@camilladelucas", "@camposmello", "@Carladiaz", "@CarolCastroReal", "@ClaudiaLeitte", "@cleo","@cissa_guimaraes",
    "@cynaramenezes", "@danisuzuki", "@dasilvabenedita", "@debranascimento", "@dedesecco",  "@dilmabr", "@DiraPaesCom",
    "@ErikakHilton", "@erikakokay", "@EuFernandaSouza", "@eususanavieira", "@FaReipert", "@FePaesLeme", "@fernandapsol",
    "@FioMattheis", "@flaayoficial", "@FlaviaAleReal", "@FreireJuliette", "@GabrielaPrioli", "@GalisteuOficial", "@gioewbank",
    "@gio_antonelli", "@gizellybicalho", "@gleicidamasceno", "@gleisi", "@gloriafperez", "@gloriapires", "@GretaThunberg",
    "@iglesbiteriana", "@IngridGuimaraes", "@IsaPennaPsol", "@ivetesangalo", "@ivymoraesb", "@IzaReal", "@jacindaardern",
    "@JanainaDoBrasil",  "@jandira_feghali", "@joicehasselmann", "@Jujuca1987", "@Julianaalvesiam", "@julianapaes",
    "@Karolconka", "@KamalaHarris", "@kerlinecardoso", "@lauraabcarvalho", "@leandraleal", "@LexaOficial", "@LoloPerisse",
```

Figura 1. Lista *screen_names*
Fonte: Autoria própria

Entretanto, somente o monitoramento do nome de usuário desta lista de mulheres não se mostrou suficiente, pois, felizmente, a maioria dos *tweets* não contém misoginia. Então mantive à *API* a atribuição do monitoramento da lista de usuárias do Twitter e, internamente, foi filtrada a captura com uma outra lista de palavras.

Para tanto, foi criada a classe Chave_Busca, uma generalização da estrutura na qual são armazenadas as palavras alvo da captura de informação.

```
class Chave_Busca:
    """Classe para rotular e instanciar chaves de busca, que serão utilizadas na captura de Tweets."""

    def __init__(self, rotulo:str, adjetivos_singular:list=[], adjetivos_singular_erro:list=[], adjetivos_plural:list=[],
                 adjetivos_plural_erro:list=[], substantivos_singular:list=[], substantivos_singular_erro:list=[],
                 substantivos_plural:list=[], substantivos_plural_erro:list=[], verbos_infinitivo:list=[],
                 verbos_infinitivo_erro:list=[], verbos_presente_singular:list=[], verbos_presente_singular_erro:list=[],
                 verbos_presente_plural:list=[], verbos_presente_plural_erro:list=[], verbos_passado_singular:list=[],
                 verbos_passado_singular_erro:list=[], verbos_passado_plural:list=[], verbos_passado_plural_erro:list=[],
                 lista_normalizacao:list=[]):
```

Figura 2. Classe Chave_Busca
Fonte: Autoria própria

Este foi um ponto que tornou o trabalho realmente desafiador, considerando que a entrada de dados nos *tweets* é realizada livremente pelos usuários da rede social. Desta forma, além da norma culta da língua portuguesa e as variações de palavras nela contempladas, como aumentativos, diminutivos, tempos verbais distintos, entre outras, também foi necessário prever possíveis estilizações oriundas do dialeto utilizado em interações online e, até mesmo, erros ortográficos.

Considerando o escopo da pesquisa, criei a constante MISOGINIA, do tipo Chave_Busca.

```
MISOGINIA = Chave_Busca(
    rotulo='misoginia',
    adjetivos_singular=['puta', 'putinha', 'prostituta', 'piranha', 'biscate', 'vaca', 'vaquinha', 'pistoleira',
                        'gostosa', 'gostosinha', 'gostosona', 'delícia', 'delicinha',
                        'feminista', 'feministinhas', 'feminazi', 'mocréia', 'baranga', 'gorda', 'ridícula',
                        'mulherzinha', 'menininha', 'empregada', 'empregadinha', 'burra',
                        'louca', 'louquinha', 'loucona', 'maluca', 'maluquinha', 'malucona', 'desequilibrada', 'p
                        'mal comida', 'mal amada'],
    adjetivos_singular_erro=['biscati', 'biskate', 'biskati', 'vaka', 'vakinha', 'pistolera',
```

Figura 3. Chave_Busca MISOGINIA
Fonte: Autoria própria

As informações contidas em um *tweet* são armazenadas em um dicionário de dados, uma estrutura de dados nativa de várias linguagens de programação, a Python inclusive.

Figura 4. A estrutura de um *tweet*
Fonte: Autoria própria

Desta forma, havendo a publicação de um *tweet* direcionado a uma das usuárias relacionadas na lista *screen_names*, contendo uma das palavras da chave de busca MISOGINIA, o mesmo é armazenado em uma lista de dicionários que, ao fim da captura, são convertidos em um *DataFrame* da biblioteca *Pandas* (PANDAS, 2021).

Considerando que um dos resultados do estudo é criar um modelo que permita indentificar os responsáveis por *tweets* em que incorrerem possíveis contravenções ou crimes, facilitando sua respectiva judicialização, foi necessário armazenar diversas informações que não são pertinentes ao modelo propriamente dito, mas que podem ser demandadas em juízo, como a localização, data e hora da publicação entre outras.

3.2. Exploração dos dados

Na construção do modelo, a duração desta etapa varia conforme a complexidade do estudo e a experiência dos envolvidos em lidar com a gama de situações que podem se apresentar.

No caso do Projeto Cangaço, inicialmente foram observados, no próprio *notebook*, os primeiros registros do *DataFrame* conjunto_dados.

```
1 conjunto_dados = pd.DataFrame(lista_tweets)
2 conjunto_dados.head()
```

	criado_em	retweet	tweet_id	tweet_url	tweet_texto	tweet_hash
0	2021-02-20 19:06:28	False	1363203384958980097	https://twitter.com/LauraSodr5/status/13632033...	@Karolconka se manca sua malucaꓼ Tirando a @ca...	
1	2021-02-20 19:06:49	False	1363203474121494528	https://twitter.com/mai_dias/status/1363203474...	"Eu não tô nervosa não, sou cria de nova Iguaç...	
2	2021-02-20 19:07:10	False	1363203561719554053	https://twitter.com/Marrento_SP/status/1363203...	@MariliaMReal Mulher faz alguma coisa, o Gil s...	
3	2021-02-20 19:07:34	False	1363203659962732549	https://twitter.com/birablue2/status/136320365...	@josi_eu_mesma @FreireJuliette Exaltar, brigar...	
4	2021-02-20	False	1363203668611375104	https://twitter.com/mervinha84/status/13632036...	@plantaumBBB21 @FreireJuliette	

Figura 5. Recorte dos primeiros registros do DataFrame conjunto_dados
Fonte: Autoria própria

Levando em consideração o tema do trabalho, o atributo *tweet_texto* foi escolhido como aquele a ser atacado. Então, foi realizada uma análise mais minuciosa no conjunto de dados exportado, da qual foram deduzidos os pré-processamentos necessários.

Neste momento de análise sobre o conjunto de dados, também se concluiu que o relacionamento entre os usuários vinculados ao *tweet* – autor, mencionado(s) e respondido(s) – é relevante para a inferência de possíveis publicações misóginas. Por exemplo, uma determinada publicação contendo exatamente o mesmo texto, pode ser uma brincadeira entre amigos ou uma ofensa. Sendo assim, fornecer este contexto ajuda em uma melhor tomada de decisão.

3.3. Pré-processamento

Com a finalidade de executar o pré-processamento, foram desenvolvidas várias funções, divididas no *notebook* entre as que necessitam estabelecer conexão com a *API* do Twitter e aquelas que rodam de maneira local.

Nesta etapa com *notebook* conectado ao Twitter, foram estabelecidos os relacionamentos entre os usuários pertencentes ao *tweet*, podendo ser:

- AMIZADE: usuário autor e usuário mencionado e/ou respondido se seguem mutuamente;
- SEGUINDO: usuário autor segue o usuário mencionado e/ou respondido;
- SEGUIDO: usuário mencionado e/ou respondido segue o usuário autor;
- INDEFINIDO: indica que não foi possível estabelecer o relacionamento entre os usuários. Essa situação ocorre quando, no intervalo de tempo entre o momento de captura e este do pré-processamento, um dos usuários excluiu sua conta. Somente modificar o nome de usuário não impediria a classificação do relacionamento, uma vez que são utilizados os identificadores numéricos dos usuários, atributo invisível e imutável.

Em seguida, foi iniciada a fase de pré-processamento offline, com o armazenamento do texto das publicações em estruturas de dados chamadas *doc*, recurso da biblioteca *spaCy* (GET, 2021), que possui uma série de mecanismos para facilitar a manipulação dos dados sendo trabalhados.

Posteriormente, foi realizada a tokenização dos textos contidos nos *docs*, através do método *tokenize*, pertencente ao *tokenizer* da biblioteca *NLTK* (BIRD, KLEIN, LOPER, 2019).

Na sequência, foi praticada a limpeza nos registros, na qual a informação é normalizada, assim minimizando os ruídos na base de dados. Por exemplo, a capitalização – letras maiúsculas / minúsculas – das palavras é desprezada.

Em modelos mais clássicos de aprendizado de máquina, no processo de limpeza, também são removidas do texto em análise uma lista de palavras chamada *stopwords*, obtida através do módulo *Corpus*, da biblioteca *NLTK* (BIRD, KLEIN, LOPER, 2019). Nela estão contidas uma relação de palavras com baixa carga semântica para a compreensão do texto, como artigos, pronomes, alguns verbos. Entretanto, considerando o estudo aqui objeto, ao proceder com a remoção das *stopwords*, seria descaracterizada a possível incorrência de uma infração penal no texto resultante. Por exemplo, nas frases “Nossa, como você é boba” e “Eu sou uma boba mesmo”, o resultado seria “boba”. Assim sendo, as *stopwords* não foram totalmente retiradas do modelo desenvolvido neste trabalho, de maneira que, depois de formado o corpus, aquelas *stopwords* nele contidas, não foram retiradas.

No entanto, foram criados outros tratamentos, que, assim como a remoção de *stopwords*, buscam aumentar a precisão do classificador. São funções reunidas no módulo maria_bonita.conjunto_dados.utilidades.pre_processamento.normalizacao, abaixo detalhadas.

```
def tokenizar(tweet_texto:str, chaves_busca:list, mencionados:list, vocabulario:list):
  """Função que recebe um tweet cru e lhe aplica os tratamentos necessários para o posterior processamento...."""
  try:
    tweet_texto = tweet_texto.lower()
    tweet_texto = normalizacao.chaves_busca(tweet_texto, chaves_busca)
    tweet_texto = normalizacao.usuarios_mencionados(tweet_texto, mencionados)
    tweet_texto = normalizacao.emoji(tweet_texto)
    tweet_texto = normalizacao.vogais_soltas(tweet_texto)
    tweet_texto = normalizacao.internetes(tweet_texto)
    tweet_texto = normalizacao.risadas(tweet_texto)
    tweet_texto = normalizacao.simbolos_com_alfanumericos(tweet_texto)

    tokens = tokenize.word_tokenize(tweet_texto, 'portuguese')
    tokens = __tratamento_stopwords(tokens, vocabulario)

    tweet_tokens_texto = []
    for token in tokens:
      token = normalizacao.numeros(token)
      token = __tratamento_remover_caractere(token, '_')
```

Figura 6. Tokenização
Fonte: Autoria própria

- chaves_busca: esta função ajusta as chaves de busca. Por exemplo, foram capturados *tweets* com as palavras “louca”, “loca”, “louka” e “loka”, que, por serem grafadas de formas diferentes, seriam consideradas distintas pelo classificador. Com o tratamento, as ocorrências de “loca”, “louka” e “loka” são substituídas por “louca”. Desenvolver este tratamento foi possível através do armazenamento das grafias corretas e incorretas, devidamente identificadas, na Chave_Busca MISOGINIA;

```
verbos_passado_plural_errado=['estroparam', 'sortaram'],
lista_normalizacao=[('biscate', ['biscati', 'biskate', 'biskati']), ('biscates', ['biscatis', 'biska
                    ('vaca', ['vaka'],), ('vacas', ['vakas'],),
                    ('vaquinha', ['vakinha']), ('vaquinhas', ['vakinhas']),
                    ('pistoleira', ['pistolera']), ('pistoleiras', ['pistoleras']),
                    ('gostosa', ['gostoza']), ('gostosas', ['gostozas']),
                    ('gostosinha', ['gostozinha']), ('gostosinhas', ['gostozinhas']),
                    ('gostosona', ['gostozona']), ('gostosonas', ['gostozonas']),
                    ('delícia', ['delicia']), ('delícias', ['delicias']),
                    ('mocréia', ['mocreia']), ('mocréias', ['mocreias']),
                    ('ridícula', ['ridicula']), ('ridículas', ['ridiculas']),
                    ('menininha', ['minininha']), ('menininhas', ['mininhas']),
                    ('louca', ['loca', 'louka', 'loka']), ('loucas', ['locas', 'loukas', 'lokas']),
                    ('louquinha', ['loquinha', 'loukinha', 'lokinha']), ('louquinhas', ['loquinhas'
```

Figura 7. Chave_Busca MISOGINIA.lista_normalizacao
Fonte: Autoria própria

- usuarios_mencionados: os nomes de usuário do Twitter são únicos, mas pouco importam para o classificador. Para fazer com que o mesmo não considere cada um deles como uma palavra distinta, são substituídos por “MENÇÃO”. Por exemplo, a frase “O @wrsAssis é um desenvolvedor de software” resulta “O MENÇÃO é um desenvolvedor de software”;
- emoji: na captura, os *emojis* já recebem um tratamento preliminar. É utilizada a biblioteca Demoji (DEMOJI, 2020), para converter seus respectivos símbolos em palavras, que estão em inglês e separadas do restante do texto pelo caractere “§”. Entretanto, este tratamento preliminar causava problemas na sequência do processo de tokenização, portanto são realizados os ajustes necessários e as representações são destacadas, tornando as letras maiúsculas. Por exemplo: “SMILLING FACE”;
- vogais_soltas: para demonstrar melhor seus sentimentos, é comum que as pessoas escrevam “ah”, ou “aahhh” ou ainda “aaaahhhhhh”. No contexto deste estudo, este sentimento não é pertinente, então usando técnicas de *RegEx*, os três exemplos

mencionados resultam em “a”.

- internetes: utilizando lógica semelhante à da função de normalização chave_busca, estão armazenadas na aplicação diversas grafias de palavras e expressões comumente utilizadas em interações online, de forma que suas ocorrências são centralizadas em somente uma única grafia. Por exemplo, “pq” e “vc” se tornam “porque” e “você”;
- risadas: de forma parecida com a função vogais_soltas, as pessoas utilizam diversas formas para representar estarem considerando algo engraçado, como “hahaha”, “kkkk” e “rsrsrs”. E, da mesma forma, o formato da risada não faz diferença para o modelo, então, utilizando *RegEx*, são todas convertidas em “RISADA”;
- simbolos_com_alfanumericos: utilizando a mesma lógica das funções chaves_busca e internetes, mas neste caso, somente implementada de modo básico (para que fique registrada a necessidade de a desenvolver efetivamente), de forma que somente reconhece a representação “S2” e a substitui por “formato de coração”;
- numeros: função que reconhece números e os normaliza, de maneira que seus dígitos são substituídos por “0”. Assim, “1234” e “98” ficariam “0000” e “00”, reduzindo infinitamente as possibilidades, mas ainda permite ao classificador mensurar superficialmente suas respectivas grandezas.

3.4. Supervisão dos dados

Na etapa de supervisão da base de dados, foi necessário imputar a possibilidade da prática de infrações penais aos usuários autores dos *tweets* que foram capturados e utilizados no desenvolvimento do Projeto Cangaço. Assim sendo, foi um processo delicado e que necessitou de respaldo jurídico especializado.

Neste sentido, é importante ressaltar a existência de uma grande variedade dos tipos penais constantes do ordenamento jurídico brasileiro relacionados ao objeto de estudo pelo Projeto Cangaço, com aqueles considerados fundamentais devidamente contemplados, para consolidar um entendimento consistente, que forneceu os subsídios para a efetiva realização da supervisão. Deste modo, do Código Penal Brasileiro (CÓDIGO PENAL, 2021), foram abarcados crimes contra a honra e contra a liberdade pessoal, sendo eles:

- Art. 138, Calúnia: é a imputação falsa da prática de crime ou da divulgação deste, sabendo ser falsa;
- Art. 139, Difamação: é a atribuição de fato ofensivo à reputação;
- Art. 140, Injúria: ofensa a dignidade ou decoro.
- Art. 147, Ameaça: é a ameaça de causar mal, injusto e grave, por palavra, escrito, gesto ou qualquer outro meio.

Entretanto, no intuito de simplificar a classificação das informações coletadas e, de acordo com o entendimento formado, os *tweets* serão considerados como “possível infração” ou não, respectivamente, “1” ou “0”. Assim, o escopo de uma “possível infração” é muito amplo, então sua tipificação pormenorizada deverá ocorrer em momento oportuno precedente a eventual judicialização, nos termos da legislação vigente.

Por fim, é interessante ressaltar que a Lei nº 13.964, de 2019, incluiu, no art. 141 do Código Penal, seu § 2º, que versa "se o crime é cometido ou divulgado em quaisquer modalidades das redes sociais da rede mundial de computadores, aplica-se em triplo a pena". É uma atualização consideravelmente recente na letra da lei, mas que demonstra um avanço do Estado em responsabilizar adequadamente estes tipos de infrações, dado seu grande potencial de dano.

3.5. Processamento: treinamento e testes

Para o Projeto Cangaço, foi adotada como técnica de processamento dos dados uma representação *word embedding* conhecida como *word2vec* (JURAFSKY, MARTIN, 2020), disponível na biblioteca *Gensim* (DOCUMENTATION, 2021), que consiste em representar palavras de forma vetorial, ou seja, para cada uma delas é calculada uma posição em um plano cartesiano e, conforme a distância entre si, é atribuído maior ou menor grau de semelhança.

No exemplo a seguir, é possível notar que as palavras "fogo", "terra" e "água", por serem elementos da natureza, foram posicionados de maneira adjacente, assim como as palavras "cachorro" e "peixes", que são animais. De forma semelhante, é factível inferir que as palavras "terra" e "cachorro", assim como "água" e "peixe", também foram posicionadas de forma mais próxima, por também ter sido identificada uma grande recorrência delas juntas em distintas composições textuais.

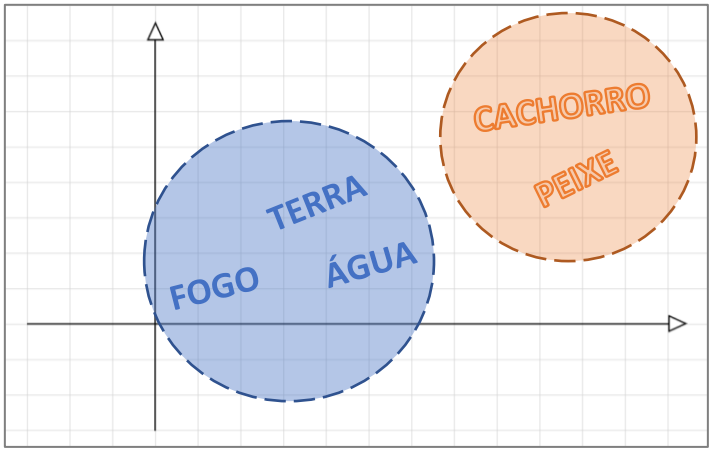

Figura 8. Exemplo de representação *word2vec*
Fonte: Autoria própria

Ao empregar o método *word2vec* no corpus textual, para cada palavra são construídos *keyed vectors* (DOCUMENTATION, 2021), os vetores densos, que são aqueles cujo todas as posições estão ocupadas, sendo n posições, em quantidade parametrizada previamente. Este processo de construção dos vetores, possui três camadas sendo elas a de entrada, a oculta – aquela que faz o processamento dos pesos e seus ajustes, ou seja, é a camada matemática, onde os vetores são calculados – e a de saída.

Neste método, são usados dois tipos de arquitetura, ambas contempladas neste trabalho, sendo que cada uma delas trabalha com uma camada de entrada diferente e, igualmente, resulta em uma camada de saída distinta.

- *Continous bag of words (MIKOLOV, CHEN, CORRADO, DEAN, 2013)*: utiliza o contexto para prever a palavra que nele melhor se encaixa. Por exemplo, da frase "Eu amo meu cachorro lindo", a camada de entrada será "meu _____ lindo" e a de

saída, possivelmente, “cachorro”;

- *Skip-gram (MIKOLOV, CHEN, CORRADO, DEAN, 2013)*: utiliza uma palavra como camada de entrada e prevê o melhor contexto em que ela se encaixa. Recorrendo ao exemplo anterior, a camada de entrada seria “cachorro” e a de saída, provavelmente, “meu _____ lindo”

Ao finalizar a representação *word2vec* do vocabulário do corpus textual, para cada uma das arquiteturas mencionadas, foi exportado um arquivo de texto, utilizado na criação dos classificadores, contendo a vetorização das palavras.

Para o *notebook* maria_bonita/modelos/Word2Vec_misoginia.ipynb do Projeto Cangaço, através dos recursos da biblioteca *Scikit-learn* (DOCUMENTATION…, 2019), como classificadores, foram criados regressores logísticos (JURAFSKY, MARTIN, 2020). Um regressor logístico busca classificar os dados, os separando através da melhor curva de reta que conseguir encontrar, formada pela equação sigmóide. Para cada entrada de dados, esta equação a encaixa em um intervalo entre 0 e 1, e, tendo em vista a posição obtida, a classifica efetivamente como 0 ou 1. No seu treinamento, cada regressor recebeu como parâmetros umas das representações *word2vec* e os *docs* em que estão armazenados os textos pré-processados dos *tweets*.

Após concluir o treinamento, como uma das formas de demonstração prática da técnica *word2vec*, os regressores logísticos resultantes disponibilizam a função *most_similar*, que recebe uma palavra como entrada e lista as outras 10 mais similares contidas no corpus textual. Abaixo, ao regressor logístico de arquitetura *continous bag of words*, foi fornecida a palavra “mulherzinha” e então retornada a lista a seguir, em ordem decrescente de semelhança:

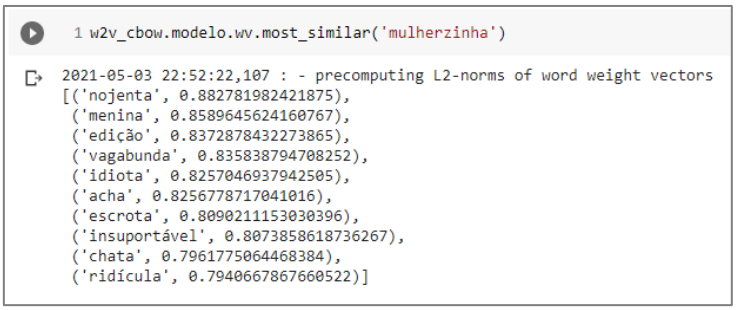

Figura 9. Regressor logístico, com arquitetura *continous bag of words*: palavras mais similares
Fonte: Autoria própria

Por fim, a etapa final foi a avaliação dos resultados obtidos pela realização dos testes do modelo.

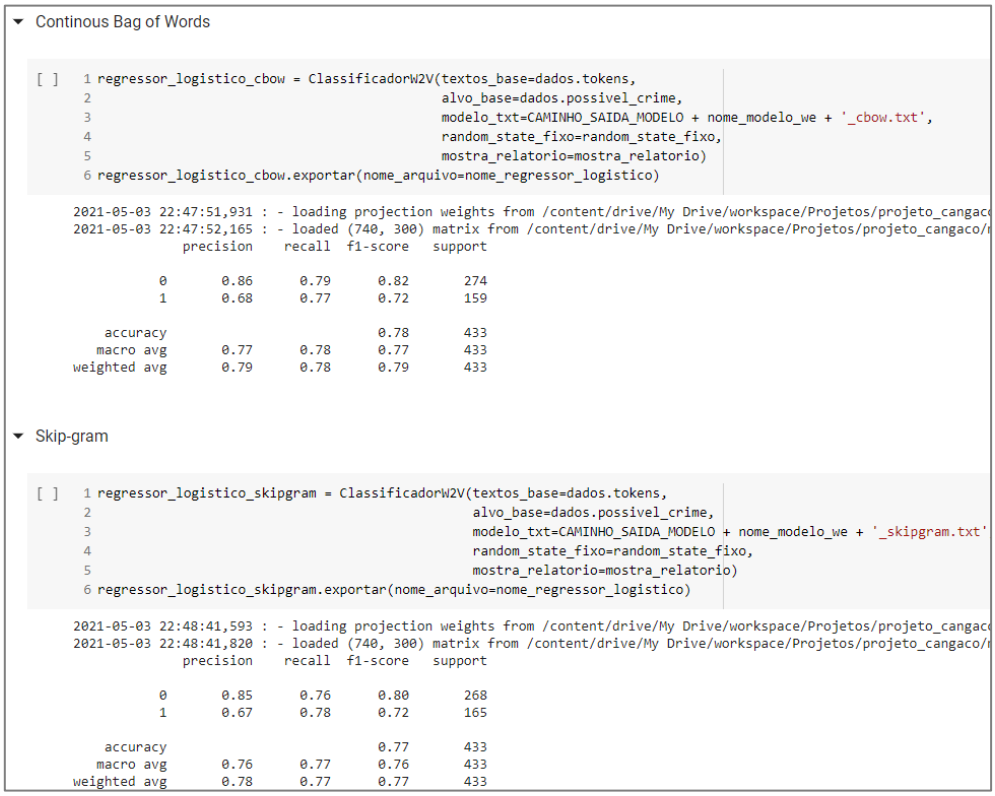

Figura 11. Resultados de acurácia dos classificadores com arquiteturas, respectivamente, *continous bag of words* e *skip-gram*
Fonte: Autoria própria

4. Conclusão

Objeto deste trabalho, a exploração dos métodos clássicos de aprendizado de máquina supervisionados e do processamento de linguagem natural foi realizada com êxito.

Tendo em vistas as etapas percorridas, se tornou notório que a acurácia do modelo decorre diretamente da fase inicial de composição de uma base de dados, que deve ser, obrigatoriamente, volumosa. Quanto maior for esta base, maior o leque de possibilidades às quais o classificador do modelo será exposto durante seu treinamento e, portanto, mais preparado estará para realizar novas classificações com um melhor percentual de precisão.

Por outro lado, não será somente a etapa de captura que irá demandar mais tempo; todas elas serão mais extensas. Para realizar a supervisão dos dados, será necessária uma equipe especializada com um número maior de integrantes. E, no caso do pré-processamento e processamento efetivo, será exigida uma maior capacidade computacional do equipamento.

Desta forma, para ser possível desenvolver o Projeto Cangaço, considerando a escassez de recursos imposta pelas dificuldades anteriormente mencionadas, foi necessário trabalhar com uma base de dados reduzida. Então, por razão desta limitação, foram criadas as diversas técnicas de pré-processamento previamente detalhadas, que ainda possibilitaram a obtenção dos bons resultados demonstrados.

4.1. Ações futuras

Ao planejar como o Projeto Cangaço poderá evoluir, a primeira e mais importante maneira, como já destacado, é a ampliação da base de dados. É fundamental contemplar no modelo a maior variedade possível de registros de interações em redes sociais, pois assim o respectivo classificador irá obter melhores resultados. E este é um processo cíclico, havendo sempre que ampliar e atualizar a base de dados, em razão de que, a todo momento, as relações humanas se modernizarem nos universos analógicos e digitais.

Aspirando ainda a ampliação da base de dados, mas tendo em vista aquelas limitações de equipe e infraestrutura referidas, foi iniciado o desenvolvimento de um recurso para criar interações misóginas artificiais. Foi solicitado a um grupo de mulheres que simulassem, de forma anônima, o texto de *tweets* misóginos de situações que elas vivenciaram ou presenciaram. E, para multiplicar as simulações, foi iniciado o desenvolvimento de um algoritmo que, no texto dos *tweets* artificiais fornecidos, identifica as palavras chaves de busca contidas na constante MISOGINIA, e então cria outras simulações com as demais chaves. Assim, em uma simulação de *tweet* "Você é uma mulher louca", rotacionando com as demais chaves, são produzidas, por exemplo, "Você é uma mulher maluca", "Você é uma mulher burra", entre outras. Ao fim do processamento deste algoritmo, os tweets simulados obtidos serão incorporados à base de dados original.

Na versão atual do Projeto Cangaço, na customização criada para o recurso *word2vec* da biblioteca *Gensim*, também foi implementada a exportação do modelo, através da biblioteca *pickle* (PYTHON, 2020), em que a informação vetorizada é serializada através de protocolos binários. Este arquivo gerado já pode ser utilizado em outros contextos, por exemplo, em chats de aplicações web, identificando prontamente a ocorrência de misoginia.

Entretanto, como destacado inicialmente no trabalho, a misoginia não é o único dos problemas enfrentados pela sociedade para um convívio pacífico nos diferentes contextos de mundo, incluído o digital. Então, guiado pela constante busca pela igualdade social, o Projeto Cangaço, em suas versões futuras, irá buscar identificar outros tipos de infrações penais, como racismo, homofobia e intolerância religiosa.

Para proceder com a efetiva responsabilização de pessoas que praticarem crimes de ódio em redes sociais, é necessário consolidar as informações capturadas de forma duradoura. Para tanto, já foi iniciada a implantação, via container do Docker, do banco de dados Maria DB.

Por fim, a ocorrência de publicações potencialmente criminosas não é uma exclusividade da rede social Twitter. Portanto, é necessário estender o monitoramento do Projeto Cangaço a outras redes, como Facebook e Instagram.

Referências

ANACONDA Documentation. ANACONDA.DOCUMENTATION, 2021. Disponível em: < https://docs.anaconda.com/ >. Acesso em: 09 mai. 2021.

BIRD, Steven, KLEIN, Ewan e LOPER, Edward. Analyzing Text with the Natural Language Toolkit. Natural Language Processing with Python, 2019. Disponível em: < http://www.nltk.org/book/ >. Acesso em: 09 mai. 2021.

BOOK. git --distributed-is-the-new-centralized, 2014. Disponível em: < https://git-scm.com/docs >. Acesso em: 09 mai. 2021.

CÓDIGO PENAL. Planalto, 2021. Disponível em: < http://www.planalto.gov.br/ccivil_03/decreto-lei/del2848compilado.htm >. Acesso em: 09 mai. 2021.

CRIAR modelos de machine learning. Microsoft Docs, 2021. Disponível em: < https://docs.microsoft.com/pt-br/learn/paths/create-machine-learn-models/ >. Acesso em: 09 mai. 2021.

DA EMPOLI, Giuliano. (2019) Engenheiros do Caos. Editora Vestígio.

DEMOJI 0.4.0. PyPi, 2020. Disponível em: < https://pypi.org/project/demoji/ >. Acesso em: 09 mai. 2021.

DOCUMENTATION. Gensim, 2021. Disponível em: < https://radimrehurek.com/gensim/auto_examples/index.html >. Acesso em: 09 mai. 2021.

DOCUMENTATION of scikit-learn 0.21.3. scikit learn, 2019. Disponível em: < https://scikit-learn.org/0.21/documentation.html >. Acesso em: 09 mai. 2021.

DOCS. Developer Twitter, 2021. Disponível em: < https://developer.twitter.com/en/docs >. Acesso em: 09 mai. 2021.

GET Started. spaCy, 2021. Disponível em: < https://spacy.io/usage >. Acesso em: 09 mai. 2021.

JURAFSKY, Dan, MARTIN, James H. Speech and Language Processing. Stanford University, 2020. Disponível em: < https://web.stanford.edu/~jurafsky/slp3/ >. Acesso em: 09 mai. 2021.

MIKOLOV, Tomas, CHEN, Kai, CORRADO, Greg, DEAN, Jeffrey. (2013) Efficient Estimation of Word Representations in Vector Space.

NEGREIROS, Adriana. (2018) Maria Bonita: Sexo, violência e mulheres no cangaço. Editora Objetiva.

PANDAS documentation. pandas, 2021. Disponível em: < https://pandas.pydata.org/docs/ >. Acesso em: 09 mai. 2021.

PYCHARM 2021.1. Jetbrains, 2021. Disponível em: < https://www.jetbrains.com/help/pycharm/quick-start-guide.html >. Acesso em: 09 mai. 2021.

PYTHON 3.5.9 documentation. python™, 2020. Disponível em: < https://docs.python.org/3.5/ >. Acesso em: 09 mai. 2021.

THE JUPYTER Notebook. Jupyter Notebook, 2015. Disponível em: < https://jupyter-notebook.readthedocs.io/en/stable/ >. Acesso em: 09 mai. 2021.

TWEEPY Documentation. tweepy, 2021. Disponível em: < https://docs.tweepy.org/en/latest/ >. Acesso em:09 mai. 2021.

WHAT can we help you find? docker docs, 2021. Disponível em: < https://docs.docker.com/ >. Acesso em: 09 mai. 2021.