PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

Pedro Ribeiro Baptista

Análise preditiva classificatória de notícias falsas

Belo Horizonte
2021

Pedro Ribeiro Baptista

Análise preditiva classificatória de notícias falsas

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Belo Horizonte
2021

SUMÁRIO

1. Introdução

1.1. Contextualização

Com a difusão da utilização das redes sociais, os meios de comunicação tradicionais, aonde a veracidade de uma determinada notícia é extensivamente checada, têm sido menos utilizados em detrimento do uso de meios de comunicação aonde qualquer um pode ser o autor de uma notícia. Isto nos levou a um cenário aonde o número de notícias falsas divulgadas nos meios de comunicação não tradicionais cresce a cada dia e preocupa diversas esferas da sociedade.

De acordo com NOTÍCIA FALSA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Not%C3%ADcia_falsa&oldid=62249891>. Acesso em: 20 nov. 2021:

> Notícias falsas (sendo também muito comum o uso do termo em inglês fake news) são uma forma de imprensa marrom que consiste na distribuição deliberada de desinformação ou boatos via jornal impresso, televisão, rádio, ou ainda online, como nas mídias sociais. Este tipo de notícia é escrito e publicado com a intenção de enganar, a fim de se obter ganhos financeiros ou políticos, muitas vezes com manchetes sensacionalistas, exageradas ou evidentemente falsas para chamar a atenção.

Ainda, de acordo com O PERIGO DAS FAKE NEWS. **TJPR - Tribunal de Justiça do Paraná**, 2021. Disponível em: <https://www.tjpr.jus.br/noticias-2-vice/-/asset_publisher/sTrhoYRKnlQe/content/o-perigo-das-fake-news/14797?inheritRedirect=false>. Acesso em: 20 nov. 2021:

> O compartilhamento de informações fraudulentas tem grande consequências, apesar de parecer inofensivo. No Brasil, em 2014, a disseminação de uma fake news provou uma verdadeira tragédia. Na ocasião, uma mulher foi linchada até a morte por moradores da cidade de Guarujá, em São Paulo. Fabiane Maria de Jesus tinha 33 anos, era dona de casa, casada, mãe de duas crianças, e foi confundida com uma suposta sequestradora de crianças, cujo retrato falado, que havia sido feito dois anos antes, estava circulando nas redes sociais.

2. O problema proposto

A necessidade de criação de formas mais ágeis para a detecção de notícias falsas que venham a impactar a sociedade é evidente dado o dano que uma notícia falsa tem o poder de causar, como por exemplo pode-se citar a divulgação de notícias falsas quanto ao uso de medicamentos sem eficácia científica comprovada para o tratamento de COVID-19.

De acordo com MONTEIRO, Ester. Liberdade de Imprensa: o Senado no combate às fake news. **Senado Federal**, 2021. Disponível em: <https://www12.senado.leg.br/noticias/materias/2021/06/07/liberdade-de-imprensa-o-senado-no-combate-as-fake-news>. Acesso em: 20 nov. 2021:

> Em 2018, quando os estudos sobre o impacto das chamadas fake news nas eleições presidenciais norte-americanas aumentaram o tom do alerta para democracias em todo o mundo, o cenário político brasileiro já era alvo de desinformação, ataques de robôs e perfis falsos." e ainda "No ano passado, uma nova pesquisa com 1.200 brasileiros de mais de 18 anos, numa amostra representativa da população brasileira, apontou o apoio da maioria dos entrevistados (84%) para a criação de uma lei contra fake news. Os usuários de redes sociais manifestam preocupação (87%) com a quantidade de notícias falsas divulgadas nessas plataformas. Três em cada quatro (76%) usuários de redes sociais concordam que, nesses ambientes virtuais, notícias falsas ganham mais visibilidade que notícias verdadeiras. Essa percepção já tinha sido registrada em estudo do Instituto de Tecnologia de Massachusetts (MIT, na sigla em inglês), dos Estados Unidos. De acordo com o trabalho, divulgado em 2018 na revista Science, as mensagens falsas se espalham 70% mais rápido que as verdadeiras e alcançam muito mais gente.

1.3. Objetivo

Este trabalho tem como objetivo a comparação de três modelos de análise preditiva para a classificação de notícias falsas.

Serão analisadas as notícias falsas, disponíveis no sítio https://piaui.folha.uol.com.br/lupa, verificadas pela Agência Lupa "plataforma de combate à desinformação através do fact-checking e da educação midiática", de acordo com O QUE É A AGÊNCIA LUPA?. **Agência Lupa**, 2015. Disponível em: <https://piaui.folha.uol.com.br/lupa/2015/10/15/como-selecionamos-as-frases-que-serao-checadas/>. Acesso em: 20 nov. 2021.

Também serão analisadas as notícias verdadeiras, disponíveis no sítio https://g1.globo.com/, do portal G1, "O g1 é um portal de notícias brasileiro mantido pelo Grupo Globo e sob orientação da Central Globo de Jornalismo", de acordo com G1. **WIKIPÉDIA,** 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=G1&oldid=62146785>. Acesso em: 20 nov. 2021.

As notícias serão classificadas quanto a sua veracidade de acordo com a origem das mesmas. As verificadas pela Agência Lupa irão compor o conjunto de notícias falsas e as publicadas no portal G1 irão compor o conjunto de notícias verdadeiras. As notícias também terão seus dados tratados para uma melhor performance dos algoritmos de aprendizado de máquina de classificação utilizados neste trabalho.

Após isto será feita a junção dos conjuntos de notícias falsas e verdadeiras e através deste novo conjunto de dados iremos realizar o treinamento e teste de previsão de classificação de notícias como falsas ou verdadeiras utilizando os modelos de classificação Naive Bayes, Regressão Logística e Árvore de Decisão .

Foi utilizada a linguagem de programação Python na versão 3.6.9. Também se fez uso de notebooks Jupyter, "aplicativo da web de código aberto que permite criar e compartilhar documentos que contêm código ao vivo, equações, visualizações e texto narrativo" segundo PROJECT JUPYTER. **Project Jupyter**, 2021. Disponível em: <https://jupyter.org/about>. Acesso em: 20 nov. 2021.

Todos os scripts, jupyter notebooks, imagens e arquivos dispostos e referenciados neste trabalho estão disponíveis no repositório github https://github.com/prbpedro/tcc_puc. Neste repositório também estão as informações necessárias para a execução dos scripts e notebooks.

2. Coleta e Tratamento dos Dados

As seguintes bibliotecas utilizadas na implementação dos scripts para a coleta e tratamento dos dados serão destacadas:

- pandas: Ferramenta de análise e manipulação de dados de código aberto
- ntkl: Plataforma para trabalhar com dados de linguagem humana
- sklearn: Ferramentas para análise preditiva de dados
- numpy: Biblioteca que suporta arrays e matrizes multidimensionais, possuindo uma larga coleção de funções matemáticas para trabalhar com estas estruturas
- requests: Biblioteca que permite a realização de chamadas HTTP
- matplotlib: Biblioteca para criar visualizações estáticas, animadas e interativas
- bs4: Biblioteca para extrair dados de arquivos HTML e XML

Os seguintes tratamentos de dados foram aplicados nos dados do tipo texto coletados:

- Remoção de caracteres de pontuação
- Remoção de caracteres de quebra de linha
- Remoção de caracteres de tabulação
- Remoção de múltiplos espaços em branco
- Transformação dos caracteres em minúsculo
- Remoção de notícias duplicadas
- Substituição dos caracteres especiais do vocabulário português por caracteres análogos não especiais
- Stemização das palavras contidas nas notícias. A stemização se consiste em: “processo de reduzir palavras flexionadas (ou às vezes derivadas) ao seu tronco (stem), base ou raiz, geralmente uma forma da palavra escrita” de acordo com STEMIZAÇÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Stemiza%C3%A7%C3%A3o&oldid=58876025>. Acesso em: 20 nov. 2021.

2.1. Notícias falsas

Para a coleta das notícias falsas foram utilizadas notícias verificadas pela Agência Lupa utilizando a técnica de raspagem de dados no portal de divulgação dos resultados das verificações realizadas, https://piaui.folha.uol.com.br/lupa/. De acordo com COLETA DE DADOS WEB. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Coleta_de_dados_web&oldid=59978509>. Acesso em: 20 nov. 2021: “A coleta de dados web, ou raspagem web, é uma forma de mineração que permite a extração de dados de sites da web convertendo-os em informação estruturada para posterior análise”.

O Notebook Jupyter pre_processing_fake_news.ipynb é responsável por obter as notícias falsas, fazer a preparação das mesmas para a utilização destas com algoritmos de machine learning e armazená-las de forma a possibilitar o processamento posterior destas.

Somente foram consideradas as notícias falsas obtidas com sucesso (status de retorno HTTP igual a 200) aonde foi encontrada a mensagem falsa compartilhada entre aspas duplas e esta possuísse mais de dez palavras, nas páginas existentes até a execução do Notebook Jupyter (366 no dia 20/10/2021).

O trecho de código a seguir foi utilizado para obter as notícias da Agência Lupa:

```
from bs4 import BeautifulSoup
import requests
import multiprocessing as mp
import pandas as pd
import re
from nltk.stem import RSLPStemmer
from wordcloud import WordCloud
from nltk.corpus import stopwords
import matplotlib.pyplot as plt

def getPage(url):
   html = requests.get(url).content
   soup = BeautifulSoup(html, 'html.parser')
   links = soup.findAll('a')
   return list(dict.fromkeys([link.get('href') for link in links if
link.get('title') != None and link.get('href') != None and
str(link.get('title')).startswith('#Verificamos:')]))
```

```
def get_is_fake_and_text(url):
   try:
       resp = requests.get(url)
       if(resp.status_code != 200):
           raise Exception('Error executing http call')

       html = resp.content
       soup = BeautifulSoup(html, 'html.parser')
       div_post_inner = soup.find('div', {'class': 'post-inner'})
       div_b_text = div_post_inner.findChildren('div', {'class': 'etiqueta
etiqueta-7'})[0].text

       is_fake = True
       if div_b_text == 'VERDADEIRO':
           raise Exception('Not fake')
       elif div_b_text == 'FALSO':
           is_fake = False
       else:
           raise Exception('Not parseable')

       ps = div_post_inner.findAll('p', recursive=False)
       p_tag = ps[2].text
       return (is_fake, p_tag)
   except Exception as e:
       print('##### ERROR - ' + str(e))
       return (None, None)

v_links = []
pages = []
for i in range(1, 366):
   pages.append(base_url + str(i) + "/")

results = [pool.apply(getPage, args=((p,))) for p in pages]
pool.close()
[v_links.extend(r) for r in results]
v_links = list(dict.fromkeys(v_links))

df = pd.DataFrame(v_links)
df.to_csv('generated/piaui_verified_links.csv', index=False)
```

Amostra de cinco registros extraídos do arquivo gerado piaui_verified_links.csv:

False	“Câncer no seio não precisa mais fazer cirurgia. Presidente Bolsonaro vai autorizar a importação dessa tecnologia de Israel. Embora chamado de machista, foi o único presidente que se lembrou em primeiro lugar das mulheres… Vamos

	divulgar!!”\nLegenda de post publicado no Facebook que, até as 18h de 18 de outubro de 2021, tinha 24 compartilhamentos\n
False	“A filha de uma amiga minha ganhou essa boneca linda de aniversário da tia. Ela veio com essa roupinha. A boneca é uma menina, né, veio com um vestido. Nada na caixa identificava coisa diferente. A criança tem 6 anos de idade, quando ela foi tirar a roupa da boneca, que ela tem costume tirar da roupa e dar banho, brincar… (…) a boneca tá sem a calcinha. Olha o que que a boneca tem. Ela tem um piu-piu e tem o ânus (…) Olha o que é a ideologia de gênero”\nVídeo que circula pelo WhatsApp\n
False	“Você sabe o que é isso? Aqui em Pernambuco, Santa Cruz do Capibaribe, o MST foi destruir as casas que seriam entregues para as pessoas mais pobres, famílias carentes (…)”\n\nConteúdo de vídeo que circula em correntes de WhatsApp\n
False	“UNIÃO EUROPEIA SUBSTITUIRÁ VACINAS POR IVERMECTINA\nBoas notícias para quem não gosta da vacina covid-19: […] Todas as vacinas não serão mais aprovadas a partir de 20 de outubro de 2021. A União Europeia aprovou cinco terapias que estarão disponíveis em todos os hospitais dos Estados-Membros para o tratamento de covid-19.”\nTrecho de texto que, até 18h do dia 15 de outubro de 2021, havia sido compartilhado por 200 usuários no Facebook\n
False	Título de texto publicado no site Stylo Urbano e que, até às 19h do dia 15 de outubro de 2021, tinha sido compartilhado por 84 pessoas no Facebook\n

O trecho de código a seguir foi o utilizado para aplicar os tratamentos elencados e gerar o arquivo results_get_fake_built.csv com as notícias falsas tratadas:

```
df = pd.read_csv('generated/results_get_fake.csv')
df['0'] = df['0'].astype(bool)
df['1'] = df['1'].astype(str)
df.info()
print(df['0'].value_counts())
df.head(10)

remove_pontuacao = lambda x:  re.sub(r'[^\w\s]', '', x)
remove_nao_citacao = lambda x: x if x.startswith('"') else ''
remove_quebra_linha = lambda x:  re.sub(r'\\n', ' ', x)
remove_tabulacao = lambda x:  re.sub(r'\\t', ' ', x)
remove_multiplos_espacos = lambda x:  re.sub(' +', ' ', x)
get_between_markers = lambda x: '' if x.find('"') == -1 else
x[x.find('"')+1: x.find('"')]
remove_little = lambda x: '' if len(x.split(' ')) < 10 else x

stemmer = RSLPStemmer()
stopws = stopwords.words('portuguese')

df['1'] =
df['1'].map(remove_quebra_linha).map(remove_tabulacao).map(remove_multiplos_
```

```
espacos).map(remove_nao_citacao).map(get_between_markers).map(remove_little)
.map(remove_pontuacao)

df.replace('FALSO', '', inplace=True)
df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')
df['1'] = df['1'].apply(lambda x: ' '.join([word for word in x.split() if
word not in (stopws)]))
df['1'] = df['1'].str.strip()

df.replace('', None, inplace=True)
df.drop_duplicates(subset='1', keep='last', inplace=True)
df.dropna(inplace=True)
df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')

words = " ".join(df['1'])
word_cloud = WordCloud(width=800, height=400, collocations = False,
background_color = 'white').generate(words)
plt.figure( figsize=(20,10) )
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()

df['1'] = df['1'].apply(lambda x: ' '.join([stemmer.stem(y) for y in
x.split(' ')]))
df.to_csv('generated/results_get_fake_built.csv', index=False)
df.info()
print(df['0'].value_counts())
df.head(10)
```

O arquivo CSV gerado possui 1340 registros e o seguinte schema:

Nome da coluna/campo	Descrição	Tipo
0	Indicador de notícia falsa	Booleano
1	Notícia falsa tratada	Texto

Amostra de cinco registros extraídos do arquivo gerado results_get_fake_built.csv:

False	canc sei nao precis faz cirurg presid bolsonar vai autor importaca dess tecnolog israel emb cham mach unic presid lembr prim lug mulh vam divulg

False	filh amig ganh bonec lind aniversari tia vei roup bonec menin ne vei vest nad caix identific cois difer crianc 6 ano idad tir roup bonec costum tir roup dar banh brinc bonec ta calc olh bonec piup anu olh ideolog gener
False	voc sab aqu pernambuc sant cruz capibarib mst destru cas entreg pesso pobr famil carent
False	fmi inflaca glob dev ating pic ult mes 2021 brasil destac control
False	infeliz acab cheg trist notic sobr at muril ros 51 ano descans paz

2.2. Notícias verdadeiras

Para a coleta das notícias verdadeiras foram utilizadas notícias publicadas no portal G1 utilizando a técnica de raspagem de dados no sítio https://g1.globo.com.

O Notebook Jupyter pre_processing_real_news.ipynb é responsável por obter as notícias verdadeiras, fazer a preparação das mesmas para a utilização destas com algoritmos de machine learning e armazená-las de forma a possibilitar o processamento posterior destas.

Somente foram consideradas as notícias obtidas com sucesso (status de retorno HTTP igual a 200) aonde foi encontrado o sumário do texto completo das notícias coletadas até a página 150 para que o número de notícias verdadeiras coletadas seja parecido com o de notícias falsas.

O trecho de código a seguir foi utilizado para obter as notícias do portal G1:

```
from bs4 import BeautifulSoup
import requests
import multiprocessing as mp
import pandas as pd
import re
from nltk.stem import RSLPStemmer
from wordcloud import WordCloud
from nltk.corpus import stopwords
import matplotlib.pyplot as plt

def getPage(url):
   html = requests.get(url).content
   soup = BeautifulSoup(html, 'html.parser')
   links = soup.findAll('a', {'class': 'feed-post-link gui-color-primary
gui-color-hover'})
```

```
    print( 'processed page'  + url)
    return list(dict.fromkeys([link.get('href') for link in links if
link.text != None and link.get('href') != None]))

def get_true_and_text(url):
    try:
        resp = requests.get(url)
        if(resp.status_code != 200):
            raise Exception('Error executing http call')

        html = resp.content
        soup = BeautifulSoup(html, 'html.parser')
        text = soup.find('h2', {'class': 'content-head__subtitle'}).text

        return (True, text)
    except Exception as e:
        print('##### ERROR - ' + str(e))
        return (None, None)

base_url = "https://g1.globo.com/index/feed/pagina-{0}.ghtml"
pool = mp.Pool(mp.cpu_count())

v_links = []
pages = []
for i in range(1, 150):
    pages.append(base_url.format(i))

results = [pool.apply(getPage, args=((p,))) for p in pages]
pool.close()
[v_links.extend(r) for r in results]
v_links = list(dict.fromkeys(v_links))

df = pd.DataFrame(v_links)
df.to_csv('generated/g1_verified_links.csv', index=False)
```

Amostra de cinco registros extraídos do arquivo g1_verified_links.csv:

True	Os que tomaram as duas doses ou vacina de dose única chegaram a 106.874.272. Levantamento é do consórcio de veículos de imprensa com base em informações das secretarias estaduais de saúde.
True	Lista completa dos pedidos de indiciamento tem três filhos do presidente da República, ministros e deputados. Relator Renan Calheiros vai ler seu parecer final em sessão da CPI.
True	Relatório final da comissão deve apontar que Bolsonaro responda por crimes como charlatanismo, epidemia com morte e fraude de documentos.

True	Bolsonaro, Pazuello, donos da Prevent Senior, Carlos Wizard e Luciano Hang estão entre os citados. Saiba quais são os crimes atribuídos a cada um.
True	Relator pede indiciamento do presidente Jair Bolsonaro por 9 crimes. Outras 65 pessoas e 2 empresas também foram indiciadas.

O trecho de código a seguir foi o utilizado para aplicar os tratamentos elencados e gerar o arquivo results_get_real_built.csv com as notícias falsas tratadas:

```
df = pd.read_csv('generated/results_get_real.csv')
df['0'] = df['0'].astype(bool)
df['1'] = df['1'].astype(str)
df = df[df['0'] == True]

df.info()
print(df['0'].value_counts())
df.head(10)

remove_pontuacao = lambda x:  re.sub(r'[^\w\s]', '', x)
remove_quebra_linha = lambda x:  re.sub(r'\\n', ' ', x)
remove_tabulacao = lambda x:  re.sub(r'\\t', ' ', x)
remove_multiplos_espacos = lambda x:  re.sub(' +', ' ', x)

stemmer = RSLPStemmer()
stopws = stopwords.words('portuguese')

df['1'] =
df['1'].map(remove_quebra_linha).map(remove_tabulacao).map(remove_multiplos_
espacos).map(remove_pontuacao)

df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')
df['1'] = df['1'].apply(lambda x: ' '.join([word for word in x.split() if
word not in (stopws)]))
df['1'] = df['1'].str.strip()

df.replace('', None, inplace=True)
df.drop_duplicates(subset='1', keep='last', inplace=True)
df.dropna(inplace=True)
df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')
```

```
words = " ".join(df['1'])
word_cloud = WordCloud(width=800, height=400, collocations = False,
background_color = 'white').generate(words)
plt.figure( figsize=(20,10) )
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()

df['1'] = df['1'].apply(lambda x: ' '.join([stemmer.stem(y) for y in
x.split(' ')]))
df.to_csv('generated/results_get_real_built.csv', index=False)
```

O arquivo CSV gerado possui 1350 registros e o seguinte schema:

Nome da coluna/campo	Descrição	Tipo
0	Indicador de notícia verdadeira	Booleano
1	Notícia verdadeira tratada	Texto

Amostra de cinco registros extraídos do arquivo results_get_real_built.csv:

True	tom dua dos vacin dos unic cheg 106874272 levant consorci veicul imprens bas informaco secret estad saud
True	list complet ped indic tre filh presid republ ministr deput rela renan calh vai ler parec final sessa cpi
True	relatori final comissa dev apont bolsonar respond crim charlatan epidem mort fraud document
True	bolsonar pazuell don prevent seni carl wizard luci hang esta cit saib qual sao crim atribu cad
True	text entreg sen nest quart ant rela apresent dua minut vej conteud tod text

2.3 Preparação dos dados

Todas as etapas descritas aqui são implementadas no Notebook Jupyter running_machine_learning_models.ipynb.

Para realizarmos os treinamentos e predições com os modelos de Machine Learning selecionados que serão descritos posteriormente foi necessário realizar a junção dos dados contidos nos arquivos results_get_fake_built.csv e results_get_real_built.csv já mencionados, resultando em um novo conjunto de dados com um total de 2690 registros de notícias falsas e verdadeiras.

A partir deste novo conjunto de dados foi gerado um outro que tem como atributos a contagem de um número determinado de palavras (as N mais frequentes) de todo o conjunto de palavras de todas as notícias, de uma determinada notícia e a classificação da notícia como falsa ou verdadeira.

Esta preparação dos dados foi feita com a utilização da classe CountVectorizer da biblioteca sklearn. Esta converte uma coleção de documentos de texto em uma matriz de contagens de tokens e caso o número de features não seja informado o mesmo será igual ao tamanho do vocabulário encontrado ao analisar os dados, de acordo com SKLEARN.FEATURE_EXTRACTION.TEXT.COUNTVECTORIZER. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html#sklearn.feature_extraction.text.CountVectorizer>. Acesso em: 20 nov. 2021.

Foram feitos testes com os seguintes números de atributos relativos à contagem das palavras mais frequentes: 10, 100, 1000, 2000, 3000, 4000 e 7552 (todas as palavras distintas do vocabulário).

Também foi utilizada a biblioteca numpy, “suporta o processamento de grandes, multi-dimensionais arranjos e matrizes, juntamente com uma grande coleção de funções matemáticas de alto nível para operar sobre estas matrizes.” segundo NUMPY. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=NumPy&oldid=61751145>. Acesso em: 20 nov. 2021.

Os seguintes trechos de código foram utilizados para realizar a junção dos dados coletados, a preparação para a utilização com os modelos de aprendizado de máquina selecionados e dividir os dados em um conjunto de treinamento e de testes através da função train_test_split da biblioteca sklearn, utilizando a técnica da validação cruzada. De acordo com VALIDAÇÃO CRUZADA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Valida%C3%A7%C3%A3o_cruzada&oldid=58989863>. Acesso em: 20 nov. 2021:

> […] técnica para avaliar a capacidade de generalização de um modelo, a partir de um conjunto de dados. Esta técnica é amplamente empregada em problemas onde o objetivo da modelagem é a predição. Busca-se então estimar o quão preciso é este modelo na prática, ou seja, o seu desempenho para um novo conjunto de dados.

```
df_fake = pd.read_csv('generated/results_get_fake_built.csv')
df_real = pd.read_csv('generated/results_get_real_built.csv')
```

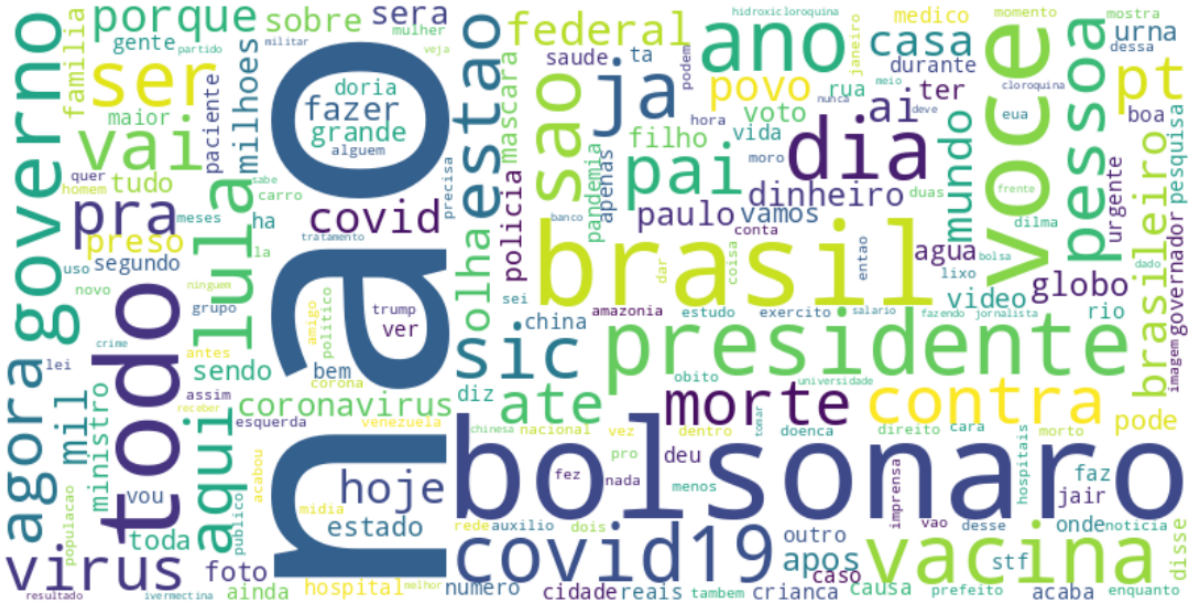
```
   cv = CountVectorizer(max_features=max_features)
   X = cv.fit_transform(np.append(df_fake['1'], df_real['1'],
axis=0)).toarray()
   y = np.append(df_fake['0'], df_real['0'], axis=0)

   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state = 42)

   feature_number = X_train.shape[1]
```

3. Análise e Exploração dos Dados

De acordo com NUVEM DE ETIQUETAS. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Nuvem_de_etiquetas&oldid=55503544>. Acesso em: 20 nov. 2021, a nuvem de etiquetas consiste em uma "lista hierarquizada visualmente, uma forma de apresentar os itens de conteúdo de um website".

Durante o pré-processamento das notícias extraídas do portal G1 e da Agência Lupa, foi feita a plotagem da nuvem de palavras do vocabulário das palavras das notícias falsas e verdadeiras de forma separada. As imagens geradas cloud_word_fake.png e cloud_word_real.png estão disponíveis da pasta generated do repositório github já citado.

Nuvem de palavras das notícias falsas:



Nuvem de palavras das notícias verdadeiras:

Apesar da variação entre o tamanho das palavras iguais entre as imagens, que representa o número de aparições destas no conjunto total das palavras das notícias falsas e verdadeiras separadamente, muitas palavras coincidem como não, ano, governo, presidente, ministro, pai, morte, etc.

A nuvem de palavras de notícias verdadeiras exibe em proporções menores os nomes próprios de pessoas, entidades, países, etc. em comparação com a nuvem de palavras falsas. Alguns exemplos são: Bolsonaro, Paulo, Doria, EUA, PT, Globo, Lula.

Palavras ligadas a pandemia do vírus COVID-19, realidade do momento da escrita deste trabalho, também aparecem em destaque na nuvem de palavras falsas. Exemplos: covid19, covid, vacina, vírus e morte.

4. Criação de Modelos de Machine Learning

Foram selecionados os modelos de aprendizado de máquina Regressão Logística, Máquina de Vetores de Suporte e Árvore de Decisão. Somente foram selecionados modelos supervisionados de aprendizado uma vez que as notícias já estão rotuladas como falsas ou verdadeiras.

Estes modelos foram selecionados por serem diferentes formas de tentar solucionar o problema exposto.

Segundo REGRESSÃO LOGÍSTICA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Regress%C3%A3o_log%C3%ADstica&oldid=62341316>. Acesso em: 20 nov. 2021, a regressão logística “É um modelo linear generalizado que usa como função de ligação a função logit.”.

O modelo Máquinas de Vetores de Suporte pode ou não ser usado de forma linear, no caso deste estudo foi usado o Kernel RBF, de acordo com SKLEARN.GAUSSIAN_PROCESS.KERNELS.RBF. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.kernels.RBF.html>. Acesso em: 20 nov. 2021, “O kernel RBF é um kernel estacionário. É também conhecido como kernel ‘exponencial ao quadrado’.”, sendo assim não linear se diferenciando do modelo de regressão logística utilizado.

O modelo Árvore de Decisão “é uma representação de uma tabela de decisão sob a forma de árvore. Trata-se de uma forma alternativa de expressar as mesmas regras que são obtidas quando se constrói a tabela” segundo ÁRVORE DE DECISÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=%C3%81rvore_de_decis%C3%A3o&oldid=61769702>. Acesso em: 20 nov. 2021.

4.1. Regressão logística

A regressão logística pode ser definida como como uma técnica estatística que permite a predição de valores categóricos a partir do treinamento do modelo um conjunto de dados pré-classificados.

Segundo REGRESSÃO LOGÍSTICA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Regress%C3%A3o_log%C3%ADstica&oldid=62341316>. Acesso em: 20 nov. 2021:

> A regressão logística é uma técnica estatística que tem como objetivo produzir, a partir de um conjunto de observações, um modelo que permita a predição de valores tomados por uma variável categórica, frequentemente binária, a partir de uma série de variáveis explicativas contínuas e/ou binárias. [...] Trata-se de um modelo de regressão para variáveis dependentes ou de resposta binomialmente distribuídas. É útil para modelar a probabilidade de um evento ocorrer em função de outros fatores. É um modelo linear generalizado que usa como função de ligação a função logit.

4.2. Máquinas de Vetores de Suporte (SVM)

Considerando o conjunto de dados, notícias falsas e verdadeiras, utilizados neste trabalho, o objetivo do modelo SVM é predizer a classificação da notícia quanto a sua veracidade. O mesmo faz isto através do mapeamento das instâncias de treinamento no espaço multidimensional e a utilização de regressão linear para encontrar um hiperplano que melhor separa as classes de entradas.

Segundo MÁQUINA DE VETORES DE SUPORTE. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=M%C3%A1quina_de_vetores_de_suporte&oldid=59794403>. Acesso em: 20 nov. 2021:

> Um modelo SVM é uma representação de exemplos como pontos no espaço, mapeados de maneira que os exemplos de cada categoria sejam divididos por um espaço claro que seja tão amplo quanto possível. Os novos exemplos são então mapeados no mesmo espaço e preditos como pertencentes a uma categoria baseados em qual o lado do espaço eles são colocados. Em outras palavras, o que uma SVM faz é encontrar uma linha de separação, mais comumente chamada de hiperplano, entre dados de

> duas classes. Essa linha busca maximizar a distância entre os pontos mais próximos em relação a cada uma das classes.

4.3. Árvore de decisão

Uma árvore de decisão classificatória tem uma estrutura em que cada nó interno (não folha) é vinculado a um atributo do conjunto de dados de treinamento, e quando aplicado determina qual o nó subsequente a ser apliacado de acordo com o valor de seu próprio nó. Cada folha da árvore é vinculada a uma classe ou à probabilidade de pertencimento sobre estas. O objetivo da árvore de decisões classificatória é, a partir da subdivisão de um problema em várias partes menores, conseguir conseguir classificar uma entrada em um determinado conjunto de classificações oriunda dos dados de treinamento do modelo de predição.

Segundo ÁRVORE DE DECISÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=%C3%81rvore_de_decis%C3%A3o&oldid=61769702>. Acesso em: 20 nov. 2021:

> O aprendizado de árvore de decisão ou indução de árvores de decisão é uma das abordagens de modelagem preditiva usadas em estatística, mineração de dados e aprendizado de máquina. Ele usa uma árvore de decisão (como um modelo preditivo) para ir de observações sobre um item (representado nos ramos) para conclusões sobre o valor alvo do item (representado nas folhas). Os modelos de árvore em que a variável de destino pode assumir um conjunto discreto de valores são chamados de árvores de classificação; nessas estruturas de árvore, as folhas representam rótulos de classe e os ramos representam conjunções de características que levam a esses rótulos de classe.

4.4. Implementação

O seguinte trecho de código foi utilizado para realizar o treinamento testes dos modelos já mencionados. Importante ressaltar a não utilização de parâmetros na construção do modelo SVM, garantindo assim a utilização do kernel RBF de acordo com a documentação disponível no sítio https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

```
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
from sklearn.metrics import f1_score,
recall_score,precision_score,cohen_kappa_score
from matplotlib.cm import ScalarMappable
from matplotlib.colors import Normalize
from xgboost import XGBClassifier
from sklearn.metrics import plot_confusion_matrix
import pprint

df_fake = pd.read_csv('generated/results_get_fake_built.csv')
df_real = pd.read_csv('generated/results_get_real_built.csv')

classifiers = {
    'Logistic Regression': LogisticRegression(random_state = 42),
    'Support Vector Machines': SVC(random_state = 42),
    'Decision Tree': DecisionTreeClassifier(random_state = 42)
}

accuracy_list = {}
f1_score_list = {}
recall_list = {}
precision_list = {}
cohen_kappa_list = {}
max_features_tested = [10, 100, 1000, 2000, 3000, 4000, None]
results = {}

for max_features in max_features_tested:
    cv = CountVectorizer(max_features=max_features)
    X = cv.fit_transform(np.append(df_fake['1'], df_real['1'],
axis=0)).toarray()
    y = np.append(df_fake['0'], df_real['0'], axis=0)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state = 42)

    feature_number = X_train.shape[1]
```

```
    results[feature_number] = {}
    results[feature_number]['accuracy'] = {}
    results[feature_number]['f1 score'] = {}
    results[feature_number]['recall'] = {}
    results[feature_number]['precision'] = {}
    results[feature_number]['cohen kappa'] = {}

    for key, model in classifiers.items():
        model.fit(X_train,y_train)
        y_pred = model.predict(X_test)

        results[feature_number]['recall'][key] = recall_score(y_test,y_pred)
        results[feature_number]['precision'][key] =
precision_score(y_test,y_pred)
        results[feature_number]['accuracy'][key] = accuracy_score(y_test,y_pred)
        results[feature_number]['f1 score'][key] = f1_score(y_test,y_pred)
        results[feature_number]['cohen kappa'][key] =
cohen_kappa_score(y_test,y_pred)

        plot_confusion_matrix(model, X_test, y_test)
        plt.title(key + ' - ' + str(feature_number))

        plt.savefig('generated/confusion_matrix_' + key.replace(' ',
'_').lower() + '_' + str(feature_number) + '.png')
        plt.show()

reformed_dict = {}
for outerKey, innerDict in results.items():
    for innerKey, values in innerDict.items():
        reformed_dict[(outerKey, innerKey)] = values

multiIndex_df = pd.DataFrame(reformed_dict)
fig, ax = plt.subplots(figsize=(20,10))
sns.heatmap(multiIndex_df, annot=True, linewidths=.5, ax=ax)
plt.xlabel("Number of features X Metric")

plt.savefig("generated/prediction_models_metrics.png")
plt.show()
```

5. Interpretação dos Resultado

5.1. Métricas utilizadas para analisar a performance dos modelos

Foram usadas as seguintes métricas para avaliar os modelos de aprendizado de máquinas treinados:

1. Pontuação de precisão:

A precisão é a razão tp / (tp + fp), onde tp é o número de verdadeiros positivos e fp o número de falsos positivos. A precisão é intuitivamente a capacidade do classificador de não rotular como positiva uma amostra negativa. O melhor valor é 1 e o pior valor é 0, segundo SKLEARN.METRICS.PRECISION_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html>. Acesso em: 20 nov. 2021.

2. Pontuação de Recall

O Recall é a razão tp / (tp + fn) em que tp é o número de verdadeiros positivos e fn o número de falsos negativos. O recall é intuitivamente a capacidade do classificador de encontrar todas as amostras positivas. O melhor valor é 1 e o pior valor é 0, segundo SKLEARN.METRICS.RECALL_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html>. Acesso em: 20 nov. 2021.

3. Pontuação de acurácia

Na classificação multilabel, a função retorna a precisão do subconjunto. Se todo o conjunto de rótulos previstos para uma amostra corresponder estritamente ao conjunto verdadeiro de rótulos, a precisão do subconjunto será 1,0; caso contrário, é 0,0, segundo SKLEARN.METRICS.ACCURACY_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html>. Acesso em: 20 nov. 2021.

4. Pontuação F1

A pontuação F1 pode ser interpretada como uma média harmônica da precisão e recall, onde uma pontuação F1 atinge seu melhor valor em 1 e a pior pontuação em 0. A contribuição relativa de precisão e recall para a pontuação F1 são iguais. A fórmula para a pontuação F1 pode ser descrita como F1 = 2 * (precisão * recall) / (precisão + recall), segundo SKLEARN.METRICS.F1_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html>. Acesso em: 20 nov. 2021.

5. Pontuação Cohen kappa

Esta medida tem como objetivo comparar as classificações de diferentes anotadores humanos, não um classificador versus uma verdade fundamental. A pontuação kappa é um número entre -1 e 1. Pontuações acima de 0,8 são geralmente consideradas de boa concordância; zero ou menos significa nenhum acordo (rótulos praticamente aleatórios), segundo COHEN'S KAPP. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/model_evaluation.html#cohen-s-kappa>. Acesso em: 20 nov. 2021.

Foram feitos testes com 10, 100, 1000, 2000, 3000, 4000 e 7552 atributos no conjunto de dados de treinamento e de testes. Será feita a análise dos resultados para cada número de atributos utilizados e a análise comparativa entre estes.

5.2. Modelos treinados com 10 atributos

Com somente 10 atributos (10 palavras mais usadas em todo o vocabulário de notícias), todos os modelos tiveram valores de métricas muito similares com variações de no máximo 0,2.

A métrica de recall obteve um valor variando entre 0,81 e 0,83, indicando que todos os modelos fazem a identificação correta de amostras positivas com assertividade entre 81% e 83% das vezes.

A métrica de precisão obteve um valor variando entre 0,58 e 0,6, indicando que todos os modelos não rotulam falsos positivos na faixa entre 58% e 60% das predições.

A métrica de F1 obteve um valor variando entre 0,68 e 0,69, indicando que todos os modelos tiveram médias harmônicas entre a precisão e o Recall parecidos, indicando performances parecidas.

A métrica de acurácia obteve um valor variando entre 0,63 e 0,64, indicando que todos os modelos fazem a identificação correta de amostras positivas e falsas com assertividade entre 63% e 64% das vezes.

A métrica de Cohen Kappa obteve um valor variando entre 0,27 e 0,3, indicando que todos os modelos fazem a identificação correta de amostras positivas e falsas com concordância justa, um pouco melhor do que tentar a predição de forma aleatória.

5.3. Modelos treinados com 100 atributos

Com 100 atributos, os modelos Regressão Logística e SVM tiveram valores de métricas muito similares com variações de no máximo 0,1.

A métrica de recall obteve um valor de 0,83 para os modelos Regressão Logística e SVM, indicando que estes fazem a identificação correta de amostras positivas com assertividade de 83% e a estabilidade destes modelos com 10 e 100 atributos, para esta métrica. Para o modelo de Árvores de Decisão a métrica de recall obteve um valor de 0,67 indicando uma piora de 14% nesta métrica em comparação com o modelo treinado com 10 atributos.

A métrica de precisão obteve um valor de 0,7 para os modelos Regressão Logística e SVM, indicando que estes modelos não rotulam falsos positivos em 70% das predições e uma melhora de 12% nesta métrica em comparação com o modelo treinado com 10 atributos. Para o modelo de Árvores de Decisão a métrica de precisão obteve um valor de 0,67 indicando uma melhora de 6% nesta métrica em comparação com o modelo treinado com 10 atributos.

A métrica de F1 obteve um valor de 0,76 para os modelos Regressão Logística e SVM, indicando performances parecidas para estes modelos e uma melhora de 0,8 nesta métrica em comparação com o modelo treinado com 10 atributos. Para o modelo de Árvores de Decisão a métrica de precisão obteve um valor de 0,66 indicando uma melhora de 0,02 nesta métrica em comparação com o modelo treinado com 10 atributos.

A métrica de acurácia obteve um valor variando entre 0,74 e 0,75 para os modelos Regressão Logística e SVM, indicando que todos os modelos fazem a identificação correta de amostras positivas e falsas com assertividade entre 74% e 75 % das vezes e uma melhora de 5% a 7% em comparação com o modelo treinado com 10 atributos. Para o modelo de Árvores de Decisão a métrica de acurácia obteve um valor de 0,67 indicando uma melhora de 0,03 nesta métrica em comparação com o modelo treinado com 10 atributos.

A métrica de Cohen Kappa obteve um valor variando de 0,49 para os modelos Regressão Logística e SVM, indicando que estes modelos fazem a identificação correta de amostras positivas e falsas com concordância moderada e uma melhora entre 0,22 e 0,25 nesta métrica em comparação com o modelo treinado com 10 atributos. Para o modelo de Árvores de Decisão a métrica de Cohen Kappa obteve um valor de 0,34 indicando uma melhora de 0,06 nesta métrica em comparação com o modelo treinado com 10 atributos e que este modelo faz e a identificação correta de amostras positivas e falsas com concordância justa.

5.4. Modelos treinados com 1000 atributos

Com 1000 atributos, o modelo SVM teve a melhor performance.

A métrica de recall obteve um valor de 0,88 para o modelo Regressão Logística, 0,91 para o modelo SVM e 0,73 para o modelo de Árvores de Decisão, indicando que estes fazem a identificação correta de amostras positivas com assertividade de 88%, 91% e 73% respectivamente e a melhora desta métrica de 6% para o modelo de Regressão logística, 9% para o modelo SVM e 6% para o modelo de Árvores de Decisão em comparação com o modelo treinado com 100 atributos.

A métrica de precisão obteve um valor de 0,82 para o modelo Regressão Logística, 0,83 para o modelo SVM e 0,77 para o modelo de Árvores de Decisão, indicando que estes modelos não rotulam falsos positivos em 82%, 83% e 77% das predições respectivamente e a melhora desta métrica de 12% para o modelo de Regressão logística, 13% para o modelo SVM e 10% para o modelo de Árvores de Decisão em comparação com o modelo treinado com 100 atributos.

A métrica F1 obteve um valor de 0,85 para o modelo Regressão Logística, 0,87 para o modelo SVM e 0,75 para o modelo de Árvores de Decisão, indicando performances parecidas para os modelos SVM e Regressão Logística, melhor performance destes em comparação com o modelo de Árvores de Decisão e a melhora desta métrica de 0,9 para esta métrica para todos os modelos em comparação com o modelo treinado com 100 atributos.

A métrica de acurácia obteve um valor de 0,85 para o modelo Regressão Logística, 0,87 para o modelo SVM e 0,77 para o modelo de Árvores de Decisão, indicando que estes modelos fazem a identificação correta de amostras positivas e falsas com assertividade de 85%, 87% e 77% das predições respectivamente e a melhora desta métrica de 12% para o modelo de Regressão logística, 13% para o modelo SVM e 12% para o modelo de Árvores de Decisão em comparação com o modelo treinado com 100 atributos.

A métrica de Cohen Kappa obteve um valor de 0,7 para o modelo Regressão Logística, 0,74 para o modelo SVM e 0,53 para o modelo de Árvores de Decisão, indicando que os modelos de Regressão Logística e SVM fazem a identificação correta de amostras positivas e falsas com concordância substancial e o modelo de Árvores de Decisão com concordância moderada, e uma melhora de 0,31, 0,35 e 0,29 respectivamente nesta métrica em comparação com os modelos treinados com 100 atributos.

5.5. Modelos treinados com 2000 atributos

Com 2000 atributos, o modelo SVM teve a melhor performance, este também foi o modelo com melhor performance dentre todos treinados com todos os diferentes números de atributos.

A métrica de recall obteve um valor de 0,89 para o modelo Regressão Logística, 0,92 para o modelo SVM e 0,8 para o modelo de Árvores de Decisão, indicando que estes fazem a identificação correta de amostras positivas com assertividade de 89%, 92% e 80% respectivamente e a melhora desta métrica de 1% para os modelos de Regressão logística e SVM e de 7% para o modelo de Árvores de Decisão em comparação com o modelo treinado com 1000 atributos.

A métrica de precisão obteve um valor de 0,86 para o modelo Regressão Logística, 0,86 para o modelo SVM e 0,76 para o modelo de Árvores de Decisão, indicando que estes modelos não rotulam falsos positivos em 86%, 86% e 76% das predições respectivamente e a melhora desta métrica de 4% para o modelo de Regressão logística, 3% para o modelo SVM e a piora de 1% para o modelo de Árvores de Decisão em comparação com o modelo treinado com 1000 atributos.

A métrica F1 obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,78 para o modelo de Árvores de Decisão, indicando performances parecidas para os modelos SVM e Regressão Logística, melhor performance destes em comparação com o modelo de Árvores de Decisão e a melhora desta métrica de 0,3, 0,1 e 0,3 respectivamente para os modelos em comparação com os treinados com 1000 atributos.

A métrica de acurácia obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,78 para o modelo de Árvores de Decisão, indicando que estes modelos fazem a identificação correta de amostras positivas e falsas com assertividade de 88%, 88% e 78% das predições respectivamente e a melhora desta métrica de 3% para o modelo de Regressão logística, 1% para os modelos SVM e Árvores de Decisão em comparação com o modelo treinado com 1000 atributos.

A métrica de Cohen Kappa obteve um valor de 0,75 para o modelo Regressão Logística, 0,77 para o modelo SVM e 0,56 para o modelo de Árvores de Decisão, indicando que os modelos de Regressão Logística e SVM fazem a identificação correta de amostras positivas e falsas com concordância substancial e o modelo de Árvores de Decisão com concordância moderada, e uma melhora de 0,05, 0,03 e 0,03 respectivamente nesta métrica em comparação com os modelos treinados com 1000 atributos.

5.6. Modelos treinados com 3000 atributos

Com 3000 atributos, o modelo SVM teve a melhor performance.

A métrica de recall obteve um valor de 0,9 para o modelo Regressão Logística, 0,92 para o modelo SVM e 0,81 para o modelo de Árvores de Decisão, indicando que estes fazem a identificação correta de amostras positivas com assertividade de 90%, 92% e 81% respectivamente e a melhora desta métrica de 1% para os modelos de Regressão logística e Árvores de Decisão e estabilidade para o modelo SVM em comparação com os modelos treinados com 2000 atributos.

A métrica de precisão obteve um valor de 0,85 para o modelo Regressão Logística, 0,85 para o modelo SVM e 0,77 para o modelo de Árvores de Decisão, indicando que estes modelos não rotulam falsos positivos em 85%, 85% e 77% das predições respectivamente e a piora desta métrica de 1% para o modelo de Regressão logística e SVM e a melhora de 1% para o modelo SVM e a piora de 1% para o modelo de Árvores de Decisão em comparação com os modelo treinados com 2000 atributos.

A métrica F1 obteve um valor de 0,87 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,79 para o modelo de Árvores de Decisão, indicando performances parecidas para os modelos SVM e Regressão Logística, melhor performance destes em comparação com o modelo de Árvores de Decisão e a melhora desta métrica de 0,1 para os modelos SVM e Árvores de Decisão e a piora de 0,1 para o modelo Regressão Logística em comparação com os modelos treinados com 2000 atributos.

A métrica de acurácia obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,79 para o modelo de Árvores de Decisão, indicando que estes modelos fazem a identificação correta de amostras positivas e falsas com assertividade de 88%, 88% e 79% das predições respectivamente e a melhora desta métrica de 1% para o modelo de Árvores de Decisão e a estabilidade desta para os modelos SVM e Regressão Logística, em comparação com os modelos treinados com 2000 atributos.

A métrica de Cohen Kappa obteve um valor de 0,75 para o modelo Regressão Logística, 0,77 para o modelo SVM e 0,59 para o modelo de Árvores de

Decisão, indicando que os modelos de Regressão Logística e SVM fazem a identificação correta de amostras positivas e falsas com concordância substancial e o modelo de Árvores de Decisão com concordância moderada, e uma melhora de 0,03 para o modelo de Árvores de Decisão e a estabilidade dos demais modelos em comparação com os modelos treinados com 2000 atributos.

5.7. Modelos treinados com 4000 atributos

Com 4000 atributos, o modelo SVM teve a melhor performance.

A métrica de recall obteve um valor de 0,89 para o modelo Regressão Logística, 0,92 para o modelo SVM e 0,81 para o modelo de Árvores de Decisão, indicando que estes fazem a identificação correta de amostras positivas com assertividade de 89%, 92% e 81% respectivamente e a piora desta métrica de 1% para o modelo de Regressão logística e estabilidade para os modelos SVM e Árvores de Decisão em comparação com os modelos treinados com 3000 atributos.

A métrica de precisão obteve um valor de 0,86 para o modelo Regressão Logística, 0,85 para o modelo SVM e 0,77 para o modelo de Árvores de Decisão, indicando que estes modelos não rotulam falsos positivos em 86%, 85% e 77% das predições respectivamente e a melhora desta métrica de 1% para o modelo de Regressão logística e a estabilidade desta para os modelos SVM e Árvores de Decisão em comparação com os modelo treinados com 3000 atributos.

A métrica F1 obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,79 para o modelo de Árvores de Decisão, indicando performances parecidas para os modelos SVM e Regressão Logística, melhor performance destes em comparação com o modelo de Árvores de Decisão e a melhora desta métrica de 0,1 para o modelo Regressão Logística e a estabilidade da mesma para os modelos SVM e Árvores de Decisão em comparação com os modelos treinados com 3000 atributos.

A métrica de acurácia obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,79 para o modelo de Árvores de Decisão, indicando que estes modelos fazem a identificação correta de amostras positivas e falsas com assertividade de 88%, 88% e 79% das predições respectivamente e a

estabilidade desta métrica para todos os modelos em comparação com os modelos treinados com 3000 atributos.

A métrica de Cohen Kappa obteve um valor de 0,75 para o modelo Regressão Logística, 0,76 para o modelo SVM e 0,58 para o modelo de Árvores de Decisão, indicando que os modelos de Regressão Logística e SVM fazem a identificação correta de amostras positivas e falsas com concordância substancial e o modelo de Árvores de Decisão com concordância moderada, e uma piora de 0,01 para os modelos SVM e Árvores de Decisão e a estabilidade do modelo Regressão Logística em comparação com os modelos treinados com 3000 atributos.

5.8. Modelos treinados com 7552 atributos

Com 7552 atributos (todas as palavras distintas do conjunto) , os modelos SVM e Regressão Logística tiveram performances similares e superiores ao modelo de Árvore de Decisão.

A métrica de recall obteve um valor de 0,9 para o modelo Regressão Logística, 0,91 para o modelo SVM e 0,8 para o modelo de Árvores de Decisão, indicando que estes fazem a identificação correta de amostras positivas com assertividade de 90%, 91% e 80% respectivamente e a piora desta métrica de 1% para os modelos Árvores de Decisão e SVM e a melhora de 1% para o modelo Regressão Logística em comparação com os modelos treinados com 4000 atributos.

A métrica de precisão obteve um valor de 0,86 para o modelo Regressão Logística, 0,85 para o modelo SVM e 0,77 para o modelo de Árvores de Decisão, indicando que estes modelos não rotulam falsos positivos em 86%, 85% e 77% das predições respectivamente e a estabilidade desta métrica para todos os modelos em comparação com os modelo treinados com 4000 atributos.

A métrica F1 obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,79 para o modelo de Árvores de Decisão, indicando performances parecidas para os modelos SVM e Regressão Logística, melhor performance destes em comparação com o modelo de Árvores de Decisão e a estabilidade desta métrica para todos os modelos em comparação com os modelos treinados com 4000 atributos.

A métrica de acurácia obteve um valor de 0,88 para o modelo Regressão Logística, 0,88 para o modelo SVM e 0,79 para o modelo de Árvores de Decisão, indicando que estes modelos fazem a identificação correta de amostras positivas e falsas com assertividade de 88%, 88% e 79% das predições respectivamente e a estabilidade desta métrica para todos os modelos em comparação com os modelos treinados com 4000 atributos.

A métrica de Cohen Kappa obteve um valor de 0,76 para o modelo Regressão Logística, 0,76 para o modelo SVM e 0,58 para o modelo de Árvores de Decisão, indicando que os modelos de Regressão Logística e SVM fazem a identificação correta de amostras positivas e falsas com concordância substancial e o modelo de Árvores de Decisão com concordância moderada, e uma melhora de 0,01 para o modelos Regressão Logística e estabilidade desta métrica para os modelos SVM e Árvores de Decisão em comparação com os modelos treinados com 4000 atributos.

5.9. Comparação dos resultados

De acordo com os resultados relatados anteriormente foi identificada a similaridade de performance geral dos modelos SVM e Regressão Logística. Estes modelos registraram diferenças nas métricas de desempenho de no máximo 0,04, e na maioria dos casos de 0,01 ou inexistente, entre métricas e número de atributos iguais para os modelos treinados.

Os modelos SVM e Regressão Logística têm desempenho melhor em todos os cenários testados em comparação com o modelo Árvore de Decisões.

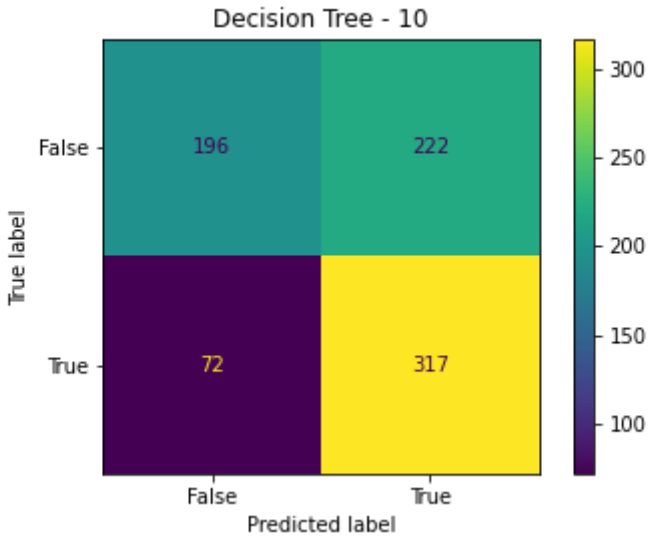
Os modelos SVM e Regressão Logística treinados com número de atributos igual ou superior a 1000 registraram diferenças nas métricas de desempenho de no máximo 0,04.

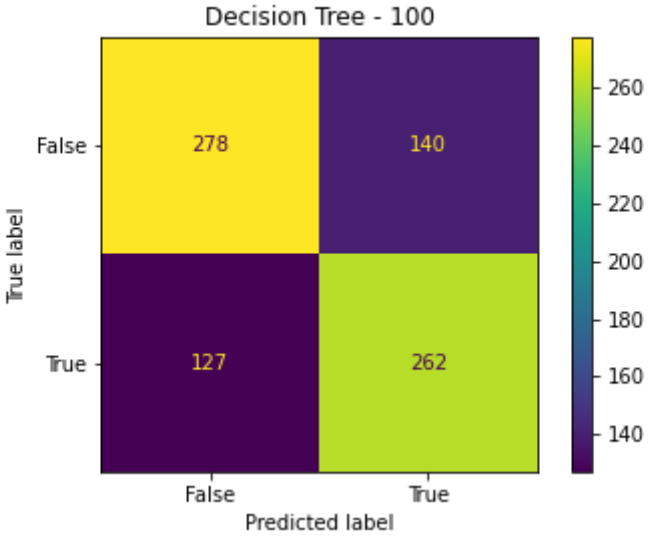
O modelo com melhor performance foi o SVM treinado com 2000 atributos.

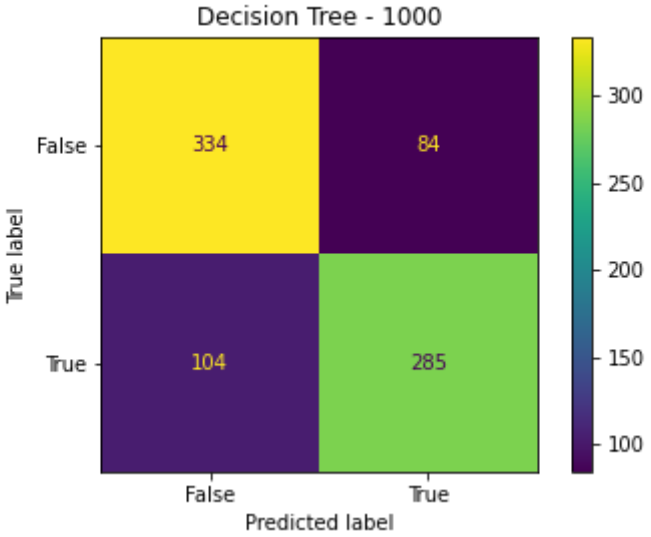
6. Apresentação dos Resultados

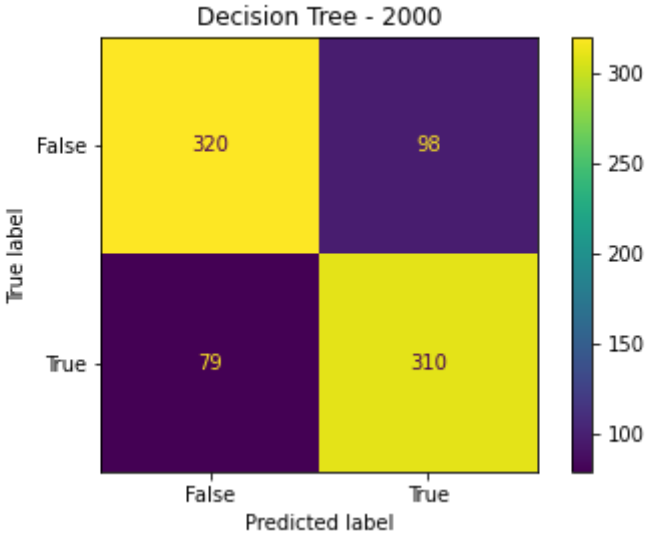
Para uma melhor análise dos resultados foi gerada uma matriz de confusão para cada modelo treinado com cada número de atributos distintos no conjunto de dados de treinamento. Segundo MATRIZ DE CONFUSÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Matriz_de_confus%C3%A3o&oldid=59389253>. Acesso em: 20 nov. 2021, “Essa tabela de contingência 2x2 especial é também chamada de matriz de erro. Cada linha da matriz representa instâncias de uma classe prevista enquanto cada coluna representa instâncias da classe atual (ou vice versa).” .

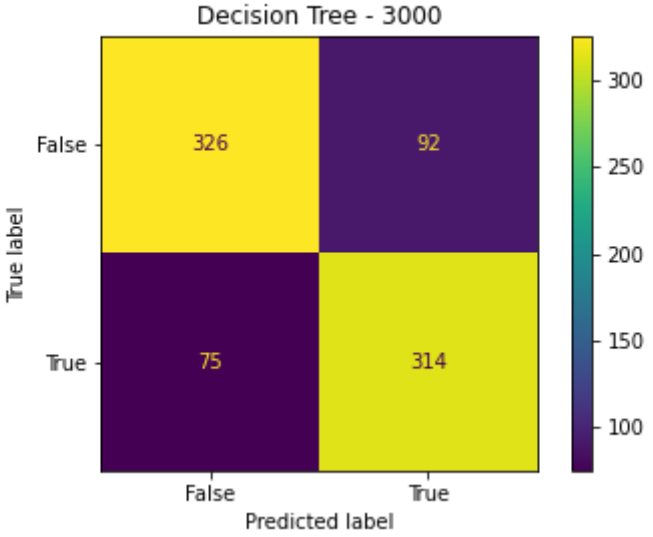
6.1. Árvore de Decisão

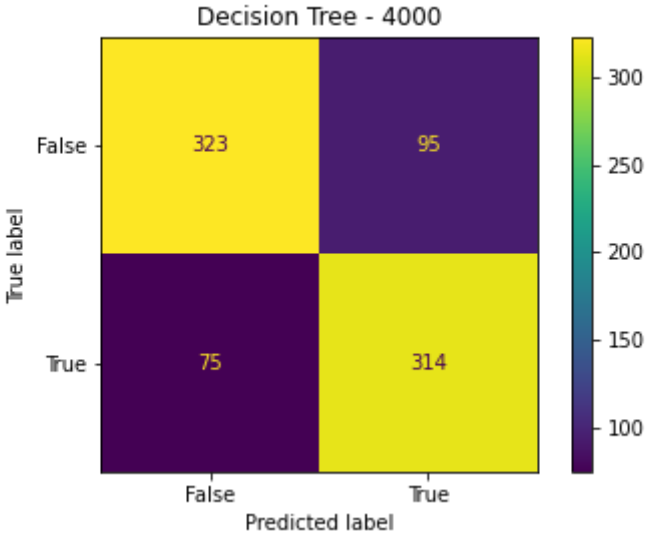


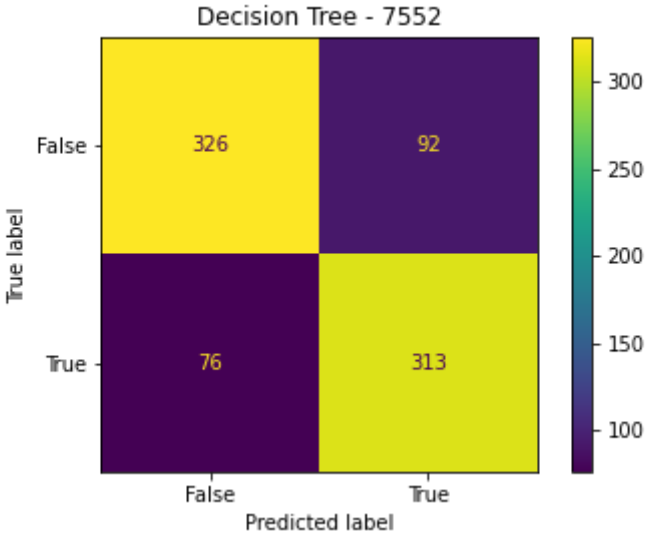

Decision Tree - 100
False
True
True label
278
140
127
262
False
True
Predicted label
260
240
220
200
180
160
140


Decision Tree - 1000
False
True
True label
334
84
104
285
False
True
Predicted label
300
250
200
150
100

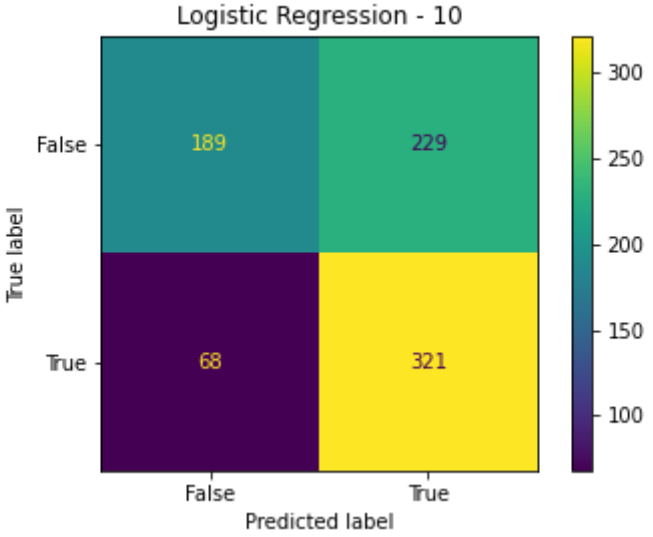

Decision Tree - 2000
False
True
True label
320
98
79
310
False
True
Predicted label
300
250
200
150
100

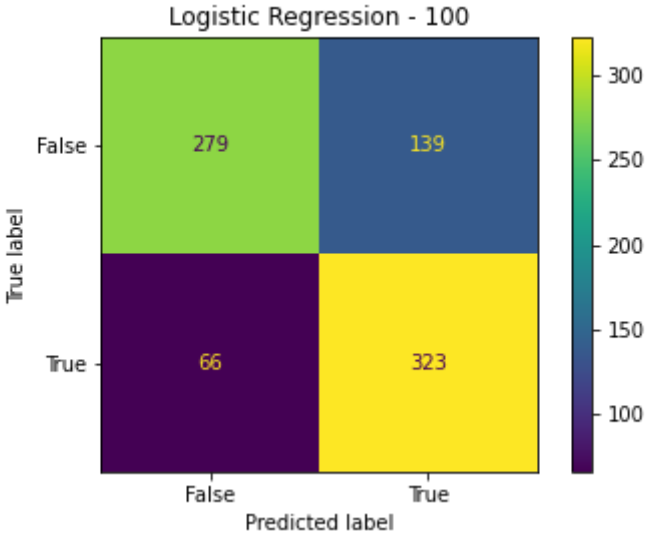

Decision Tree - 3000
False
True
True label
326
92
75
314
False
True
Predicted label
300
250
200
150
100

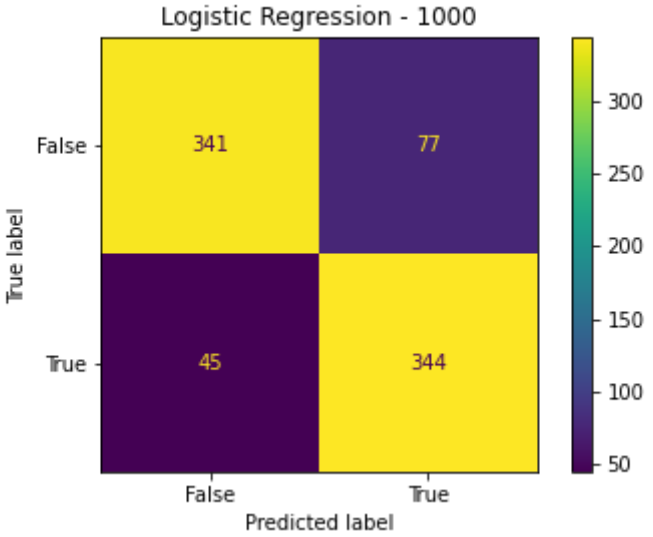

Decision Tree - 4000
False
True
True label
323
95
75
314
False
True
Predicted label
300
250
200
150
100

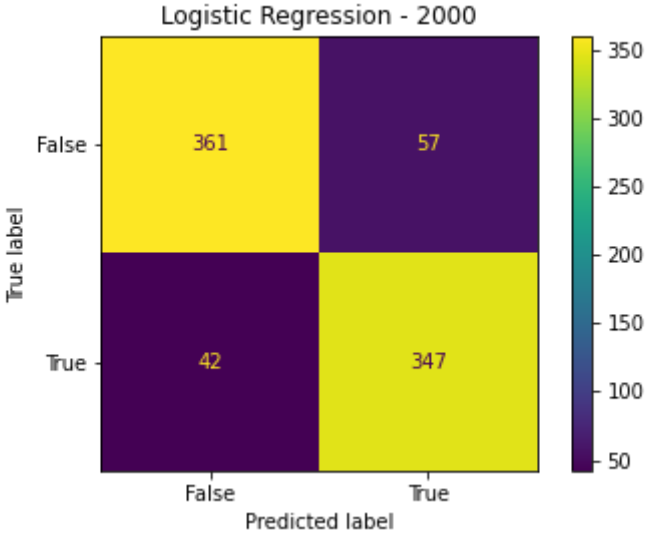

Decision Tree - 7552
326
92
76
313
False
True
True label
False
True
Predicted label
300
250
200
150
100

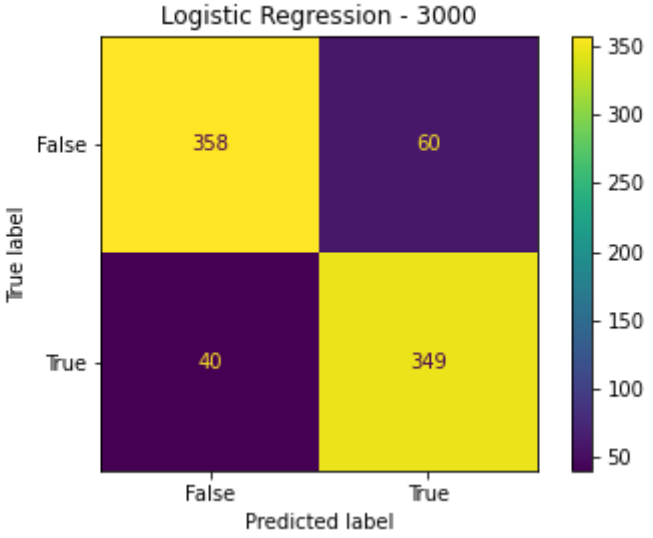
6.2. Regressão logística

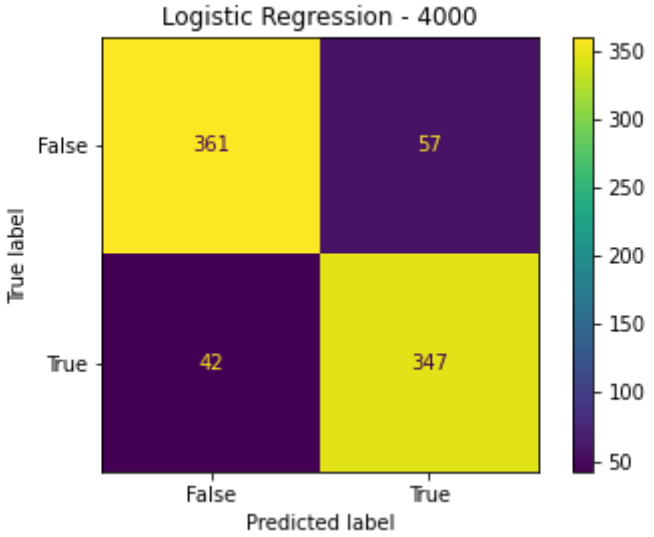


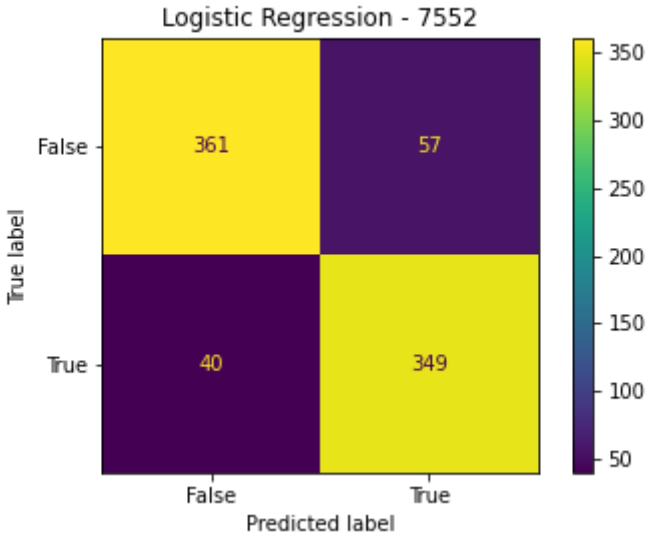



Logistic Regression - 1000
341
77
45
344
False
True
True label
Predicted label
300
250
200
150
100
50

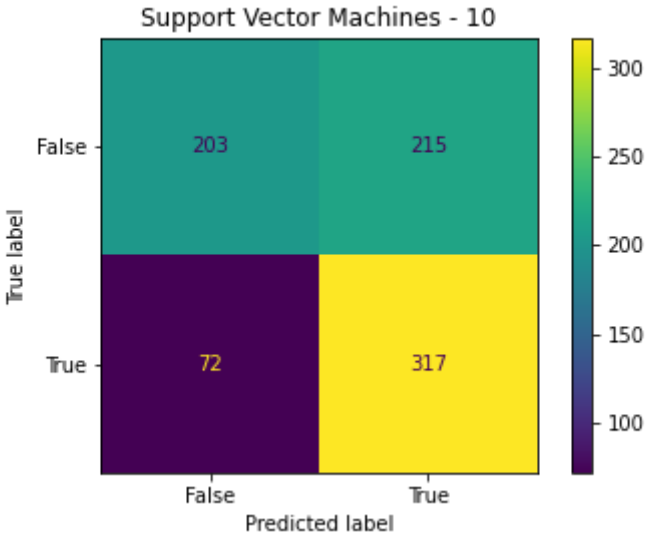

Logistic Regression - 2000
361
57
42
347
False
True
True label
Predicted label
350
300
250
200
150
100
50

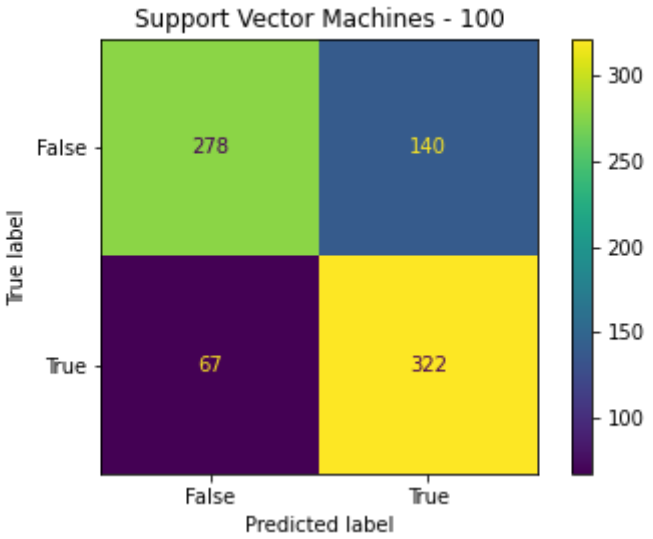

Logistic Regression - 3000
358
60
40
349
False
True
True label
Predicted label
350
300
250
200
150
100
50

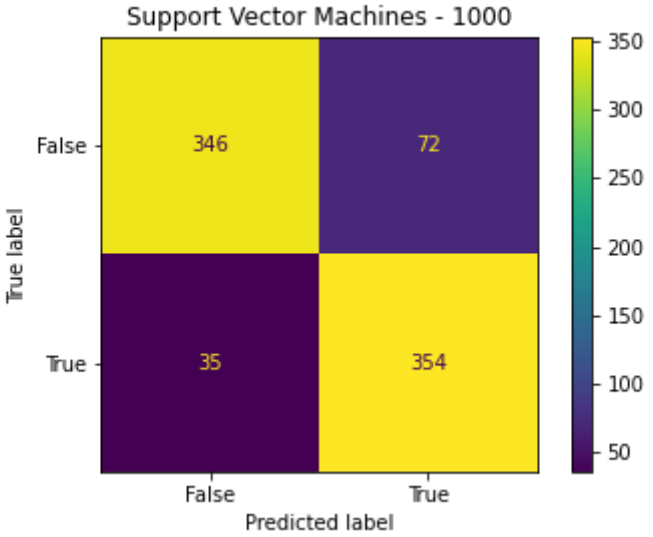

Logistic Regression - 4000
361
57
42
347
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50

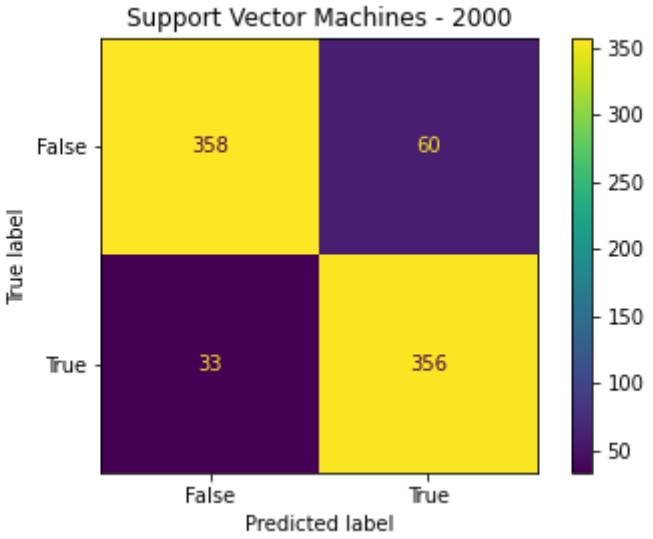

Logistic Regression - 7552
361
57
40
349
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50

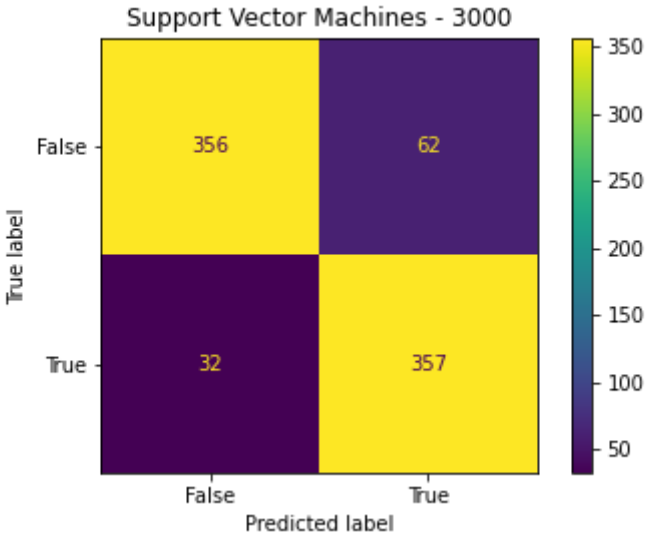
6.3. Support Vector Machines

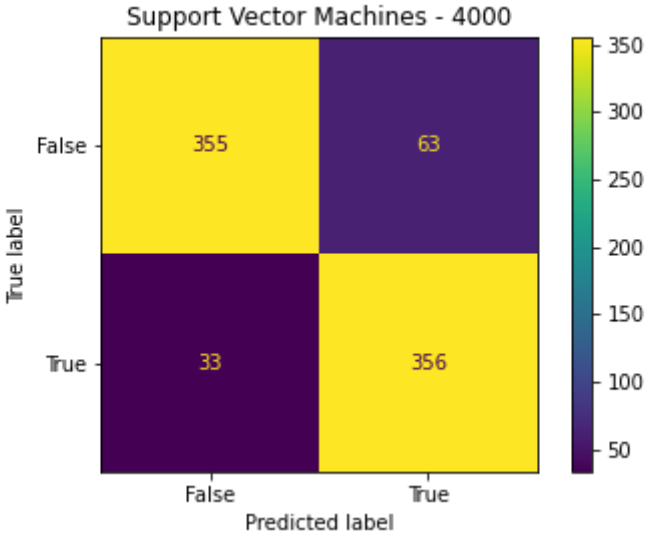


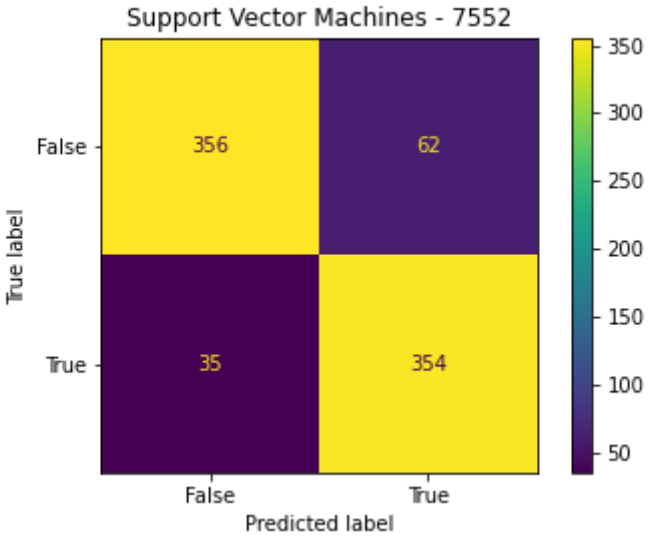



Support Vector Machines - 1000
346
72
35
354
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50


Support Vector Machines - 2000
358
60
33
356
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50


Support Vector Machines - 3000
356
62
32
357
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50

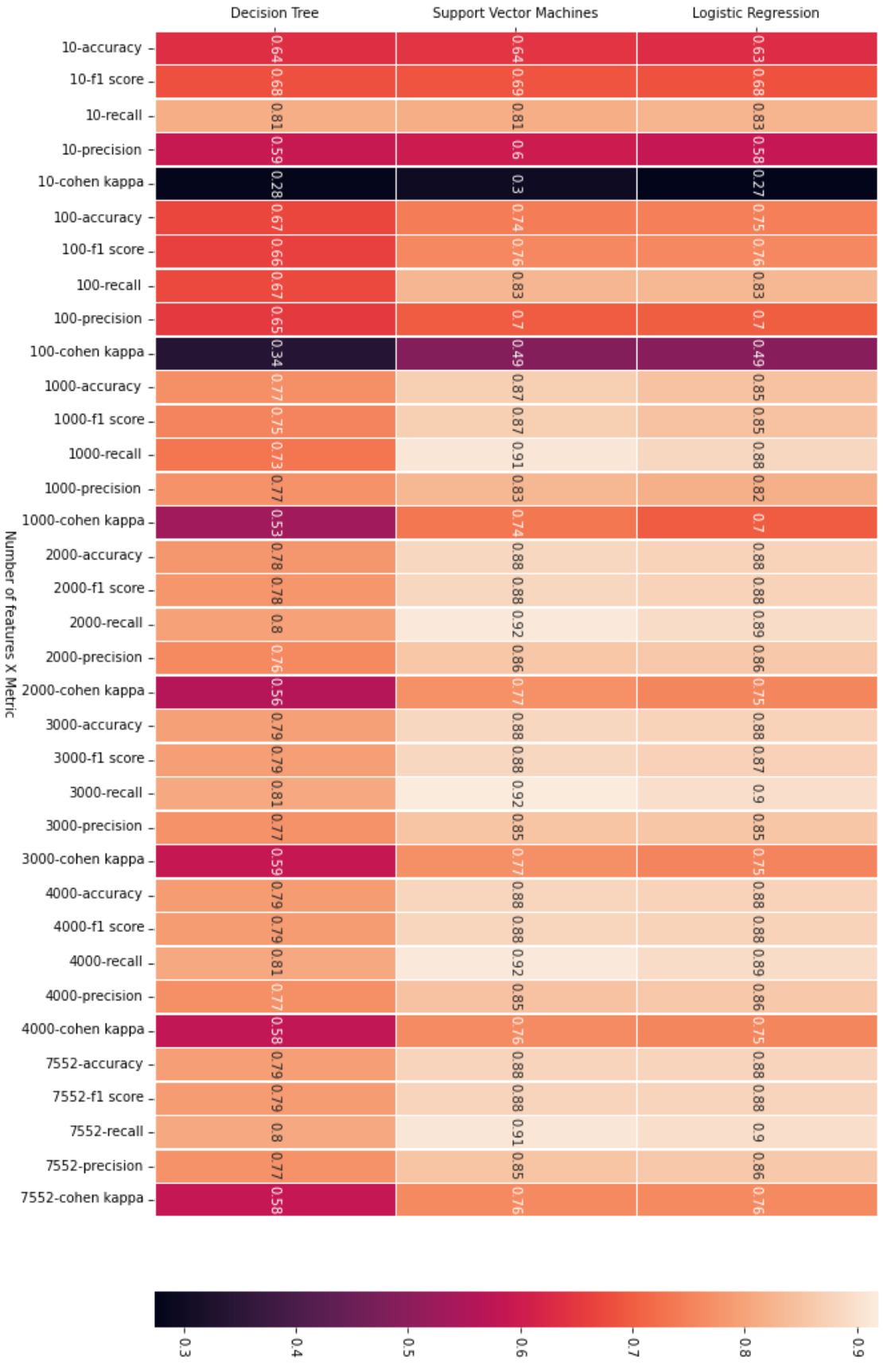

Support Vector Machines - 4000
355
63
33
356
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50


Support Vector Machines - 7552
356
62
35
354
False
True
True label
False
True
Predicted label
350
300
250
200
150
100
50

6.4. Resultado comparativo das métricas de desempenho

De acordo com **HEAT MAP.** In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2021. Disponível em: <https://en.wikipedia.org/w/index.php?title=Heat_map&oldid=1041704610>. Acesso em: 20 nov. 2021, um mapa de calor é uma técnica de visualização de dados que mostra a magnitude de um fenômeno como cor em duas dimensões.

Foi gerado um mapa de calor com os valores das métricas de desempenho para cada modelo treinado com cada número de atributos distintos no conjunto de dados de treinamento.


Decision Tree
Support Vector Machines
Logistic Regression
Number of features X Metric
10-accuracy 0.64 0.64 0.63
10-f1 score 0.68 0.69 0.68
10-recall 0.81 0.81 0.83
10-precision 0.59 0.6 0.58
10-cohen kappa 0.28 0.3 0.27
100-accuracy 0.67 0.74 0.75
100-f1 score 0.66 0.76 0.76
100-recall 0.67 0.83 0.83
100-precision 0.65 0.7 0.7
100-cohen kappa 0.34 0.49 0.49
1000-accuracy 0.77 0.87 0.85
1000-f1 score 0.75 0.87 0.85
1000-recall 0.73 0.91 0.88
1000-precision 0.77 0.83 0.82
1000-cohen kappa 0.53 0.74 0.7
2000-accuracy 0.78 0.88 0.88
2000-f1 score 0.78 0.88 0.88
2000-recall 0.8 0.92 0.89
2000-precision 0.76 0.86 0.86
2000-cohen kappa 0.56 0.77 0.75
3000-accuracy 0.79 0.88 0.88
3000-f1 score 0.79 0.88 0.87
3000-recall 0.81 0.92 0.9
3000-precision 0.77 0.85 0.85
3000-cohen kappa 0.59 0.77 0.75
4000-accuracy 0.79 0.88 0.88
4000-f1 score 0.79 0.88 0.88
4000-recall 0.81 0.92 0.89
4000-precision 0.77 0.85 0.86
4000-cohen kappa 0.58 0.76 0.75
7552-accuracy 0.79 0.88 0.88
7552-f1 score 0.79 0.88 0.88
7552-recall 0.8 0.91 0.9
7552-precision 0.77 0.85 0.86
7552-cohen kappa 0.58 0.76 0.76
0.3
0.4
0.5
0.6
0.7
0.8
0.9

7. Links

Link para o vídeo: https://www.youtube.com/watch?v=qcxIQkC_--M

Link para o repositório: https://github.com/prbpedro/tcc_puc

REFERÊNCIAS

ÁRVORE DE DECISÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=%C3%81rvore_de_decis%C3%A3o&oldid=61769702>. Acesso em: 20 nov. 2021.

COHEN'S KAPP. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/model_evaluation.html#cohen-s-kappa>. Acesso em: 20 nov. 2021.

COLETA DE DADOS WEB. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Coleta_de_dados_web&oldid=59978509>. Acesso em: 20 nov. 2021.

G1. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=G1&oldid=62146785>. Acesso em: 20 nov. 2021.

HEAT MAP. **WIKIPÉDIA**, 2020. Disponível em: <https://en.wikipedia.org/w/index.php?title=Heat_map&oldid=1041704610>. Acesso em: 20 nov. 2021.

MÁQUINA DE VETORES DE SUPORTE. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=M%C3%A1quina_de_vetores_de_suporte&oldid=59794403>. Acesso em: 20 nov. 2021.

MATRIZ DE CONFUSÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Matriz_de_confus%C3%A3o&oldid=59389253>. Acesso em: 20 nov. 2021.

MONTEIRO, Ester. Liberdade de Imprensa: o Senado no combate às fake news. **Senado Federal**, 2021. Disponível em: <https://www12.senado.leg.br/noticias/materias/2021/06/07/liberdade-de-imprensa-o-senado-no-combate-as-fake-news>. Acesso em: 20 nov. 2021.

NOTÍCIA FALSA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Not%C3%ADcia_falsa&oldid=62249891>. Acesso em: 20 nov. 2021.

NUMPY. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=NumPy&oldid=61751145>. Acesso em: 20 nov. 2021.

NUVEM DE ETIQUETAS. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Nuvem_de_etiquetas&oldid=55503544>. Acesso em: 20 nov. 2021.

O PERIGO DAS FAKE NEWS. **TJPR - Tribunal de Justiça do Paraná**, 2021. Disponível em: <https://www.tjpr.jus.br/noticias-2-vice/-/asset_publisher/sTrhoYRKnlQe/content/o-perigo-das-fake-news/14797?inheritRedirect=false>. Acesso em: 20 nov. 2021.

O QUE É A AGÊNCIA LUPA?. **Agência Lupa**, 2015. Disponível em: <https://piaui.folha.uol.com.br/lupa/2015/10/15/como-selecionamos-as-frases-que-serao-checadas/>. Acesso em: 20 nov. 2021.

PROJECT JUPYTER. **Project Jupyter**, 2021. Disponível em: <https://jupyter.org/about>. Acesso em: 20 nov. 2021.

REGRESSÃO LOGÍSTICA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Regress%C3%A3o_log%C3%ADstica&oldid=62341316>. Acesso em: 20 nov. 2021.

SKLEARN.FEATURE_EXTRACTION.TEXT.COUNTVECTORIZER. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html#sklearn.feature_extraction.text.CountVectorizer>. Acesso em: 20 nov. 2021.

SKLEARN.GAUSSIAN_PROCESS.KERNELS.RBF. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.kernels.RBF.html>. Acesso em: 20 nov. 2021.

SKLEARN.METRICS.ACCURACY_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html>. Acesso em: 20 nov. 2021.

SKLEARN.METRICS.F1_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html>. Acesso em: 20 nov. 2021.

SKLEARN.METRICS.PRECISION_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html>. Acesso em: 20 nov. 2021.

SKLEARN.METRICS.RECALL_SCORE. **SKLEARN**, 2007. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html>. Acesso em: 20 nov. 2021.

STEMIZAÇÃO. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Stemiza%C3%A7%C3%A3o&oldid=58876025>. Acesso em: 20 nov. 2021.

VALIDAÇÃO CRUZADA. **WIKIPÉDIA**, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Valida%C3%A7%C3%A3o_cruzada&oldid=58989863>. Acesso em: 20 nov. 2021.

APÊNDICE

Programação/Scripts

- pre_processing_fake_news.ipynb

```
from bs4 import BeautifulSoup
import requests
import multiprocessing as mp
import pandas as pd
import re
from nltk.stem import RSLPStemmer
from wordcloud import WordCloud
from nltk.corpus import stopwords
import matplotlib.pyplot as plt

pd.set_option('display.max_colwidth', None)

def getPage(url):
    html = requests.get(url).content
    soup = BeautifulSoup(html, 'html.parser')
    links = soup.findAll('a')
    return list(dict.fromkeys([link.get('href') for link in links if
link.get('title') != None and link.get('href') != None and
str(link.get('title')).startswith('#Verificamos:')]))

def get_is_fake_and_text(url):
    try:
        resp = requests.get(url)
        if(resp.status_code != 200):
            raise Exception('Error executing http call')

        html = resp.content
        soup = BeautifulSoup(html, 'html.parser')
        div_post_inner = soup.find('div', {'class': 'post-inner'})
        div_b_text = div_post_inner.findChildren('div', {'class': 'etiqueta
etiqueta-7'})[0].text

        is_fake = True
        if div_b_text == 'VERDADEIRO':
            raise Exception('Not fake')
        elif div_b_text == 'FALSO':
            is_fake = False
        else:
            raise Exception('Not parseable')

        ps = div_post_inner.findAll('p', recursive=False)
        p_tag = ps[2].text
        return (is_fake, p_tag)
```

```
    except Exception as e:
        print('##### ERROR - ' + str(e))
        return (None, None)

base_url = "https://piaui.folha.uol.com.br/lupa/page/"
pool = mp.Pool(mp.cpu_count())

v_links = []
pages = []
for i in range(1, 366):
    pages.append(base_url + str(i) + "/")

results = [pool.apply(getPage, args=((p,))) for p in pages]
pool.close()
[v_links.extend(r) for r in results]
v_links = list(dict.fromkeys(v_links))

df = pd.DataFrame(v_links)
df.to_csv('generated/piaui_verified_links.csv', index=False)

pool = mp.Pool(mp.cpu_count())
results_get_fake = [pool.apply(get_is_fake_and_text, args=((p,))) for p in
v_links]
pool.close()

df = pd.DataFrame(results_get_fake)
df.to_csv('generated/results_get_fake.csv', index=False)

df = pd.read_csv('generated/results_get_fake.csv')

df['0'] = df['0'].astype(bool)
df['1'] = df['1'].astype(str)
df = df[df['0'] == False]

df.info()
print(df['0'].value_counts())
df.head(10)

remove_pontuacao = lambda x:  re.sub(r'[^\w\s]', '', x)
remove_nao_citacao = lambda x: x if x.startswith('"') else ''
remove_quebra_linha = lambda x:  re.sub(r'\\n', ' ', x)
remove_tabulacao = lambda x:  re.sub(r'\\t', ' ', x)
remove_multiplos_espacos = lambda x:  re.sub(' +', ' ', x)
get_between_markers = lambda x: '' if x.find('"') == -1 else x[x.find('"')+1:
x.find('"')]
remove_little = lambda x: '' if len(x.split(' ')) < 10 else x

stemmer = RSLPStemmer()
stopws = stopwords.words('portuguese')

df['1'] =
```

```
df['1'].map(remove_quebra_linha).map(remove_tabulacao).map(remove_multiplos_espa
cos).map(remove_nao_citacao).map(get_between_markers).map(remove_little).map(rem
ove_pontuacao)

df.replace('FALSO', '', inplace=True)
df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')
df['1'] = df['1'].apply(lambda x: ' '.join([word for word in x.split() if word
not in (stopws)]))
df['1'] = df['1'].str.strip()

df.replace('', None, inplace=True)
df.drop_duplicates(subset='1', keep='last', inplace=True)
df.dropna(inplace=True)
df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')

words = " ".join(df['1'])
word_cloud = WordCloud(width=800, height=400, collocations = False,
background_color = 'white').generate(words)
plt.figure( figsize=(20,10) )
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()

df['1'] = df['1'].apply(lambda x: ' '.join([stemmer.stem(y) for y in x.split('
')]))
df.to_csv('generated/results_get_fake_built.csv', index=False)
df.info()
print(df['0'].value_counts())
df.head(10)
```

- pre_processing_real_news.ipynb

```
from bs4 import BeautifulSoup
import requests
import multiprocessing as mp
import pandas as pd
import re
from nltk.stem import RSLPStemmer
from wordcloud import WordCloud
from nltk.corpus import stopwords
import matplotlib.pyplot as plt

pd.set_option('display.max_colwidth', None)

def getPage(url):
    html = requests.get(url).content
```

```
    soup = BeautifulSoup(html, 'html.parser')
    links = soup.findAll('a', {'class': 'feed-post-link gui-color-primary
gui-color-hover'})
    print( 'processed page'  + url)
    return list(dict.fromkeys([link.get('href') for link in links if link.text
!= None and link.get('href') != None]))

def get_true_and_text(url):
    try:
        resp = requests.get(url)
        if(resp.status_code != 200):
            raise Exception('Error executing http call')

        html = resp.content
        soup = BeautifulSoup(html, 'html.parser')
        text = soup.find('h2', {'class': 'content-head__subtitle'}).text

        return (True, text)
    except Exception as e:
        print('##### ERROR - ' + str(e))
        return (None, None)

base_url = "https://g1.globo.com/index/feed/pagina-{0}.ghtml"
pool = mp.Pool(mp.cpu_count())

v_links = []
pages = []
for i in range(1, 150):
    pages.append(base_url.format(i))

results = [pool.apply(getPage, args=((p,))) for p in pages]
pool.close()
[v_links.extend(r) for r in results]
v_links = list(dict.fromkeys(v_links))

df = pd.DataFrame(v_links)
df.to_csv('generated/g1_verified_links.csv', index=False)

pool = mp.Pool(mp.cpu_count())
results_get_fake = [pool.apply(get_true_and_text, args=((p,))) for p in v_links]
pool.close()

df = pd.DataFrame(results_get_fake)
df.to_csv('generated/results_get_real.csv', index=False)

df = pd.read_csv('generated/results_get_real.csv')
df['0'] = df['0'].astype(bool)
df['1'] = df['1'].astype(str)
df = df[df['0'] == True]

df.info()
```

```
print(df['0'].value_counts())
df.head(10)

remove_pontuacao = lambda x:  re.sub(r'[^\w\s]', '', x)
remove_quebra_linha = lambda x:  re.sub(r'\\n', ' ', x)
remove_tabulacao = lambda x:  re.sub(r'\\t', ' ', x)
remove_multiplos_espacos = lambda x:  re.sub(' +', ' ', x)

stemmer = RSLPStemmer()
stopws = stopwords.words('portuguese')

df['1'] =
df['1'].map(remove_quebra_linha).map(remove_tabulacao).map(remove_multiplos_espa
cos).map(remove_pontuacao)

df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')
df['1'] = df['1'].apply(lambda x: ' '.join([word for word in x.split() if word
not in (stopws)]))
df['1'] = df['1'].str.strip()

df.replace('', None, inplace=True)
df.drop_duplicates(subset='1', keep='last', inplace=True)
df.dropna(inplace=True)
df['1'] = df['1'].str.lower()
df['1'] = df['1'].str.normalize('NFKD').str.encode('ascii',
errors='ignore').str.decode('utf-8')

words = " ".join(df['1'])
word_cloud = WordCloud(width=800, height=400, collocations = False,
background_color = 'white').generate(words)
plt.figure( figsize=(20,10) )
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()

df['1'] = df['1'].apply(lambda x: ' '.join([stemmer.stem(y) for y in x.split('
')]))
df = df[df['1']!='nan']
df.to_csv('generated/results_get_real_built.csv', index=False)
df.info()
print(df['0'].value_counts())
df.head(10)
```

- running_machine_learning_models.ipynb

```
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
```

```
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
from sklearn.metrics import f1_score,
recall_score,precision_score,cohen_kappa_score
from matplotlib.cm import ScalarMappable
from matplotlib.colors import Normalize
from xgboost import XGBClassifier
from sklearn.metrics import plot_confusion_matrix
import pprint

df_fake = pd.read_csv('generated/results_get_fake_built.csv')
df_real = pd.read_csv('generated/results_get_real_built.csv')

classifiers = {
    'Logistic Regression': LogisticRegression(random_state = 42),
    'Support Vector Machines': SVC(random_state = 42),
    'Decision Tree': DecisionTreeClassifier(random_state = 42)
}

accuracy_list = {}
f1_score_list = {}
recall_list = {}
precision_list = {}
cohen_kappa_list = {}
max_features_tested = [10, 100, 1000, 2000, 3000, 4000, None]
results = {}

for max_features in max_features_tested:
    cv = CountVectorizer(max_features=max_features)
    X = cv.fit_transform(np.append(df_fake['1'], df_real['1'],
axis=0)).toarray()
    y = np.append(df_fake['0'], df_real['0'], axis=0)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state = 42)

    feature_number = X_train.shape[1]

    results[feature_number] = {}
    results[feature_number]['accuracy'] = {}
```

```
    results[feature_number]['f1 score'] = {}
    results[feature_number]['recall'] = {}
    results[feature_number]['precision'] = {}
    results[feature_number]['cohen kappa'] = {}

    for key, model in classifiers.items():
        model.fit(X_train,y_train)
        y_pred = model.predict(X_test)

        results[feature_number]['recall'][key] = recall_score(y_test,y_pred)
        results[feature_number]['precision'][key] =
precision_score(y_test,y_pred)
        results[feature_number]['accuracy'][key] = accuracy_score(y_test,y_pred)
        results[feature_number]['f1 score'][key] = f1_score(y_test,y_pred)
        results[feature_number]['cohen kappa'][key] =
cohen_kappa_score(y_test,y_pred)

        plot_confusion_matrix(model, X_test, y_test)
        plt.title(key + ' - ' + str(feature_number))

        plt.savefig('generated/confusion_matrix_' + key.replace(' ',
'_').lower() + '_' + str(feature_number) + '.png')
        plt.show()

reformed_dict = {}
for outerKey, innerDict in results.items():
    for innerKey, values in innerDict.items():
        reformed_dict[(outerKey, innerKey)] = values

multiIndex_df = pd.DataFrame(reformed_dict)
fig, ax = plt.subplots(figsize=(20,10))
sns.heatmap(multiIndex_df, annot=True, linewidths=.5, ax=ax)
plt.xlabel("Number of features X Metric")

plt.savefig("generated/prediction_models_metrics.png")
plt.show()
```