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

Thalita Sylvia Marques Higino

IMPACTO DE FATORES DE RISCO EM CASOS CONFIRMADOS DE COVID-19 NO ESTADO DE SÃO PAULO:
Um Estudo com Regras de Associação e Algoritmos para Classificação de Óbitos

Belo Horizonte
2022

Thalita Sylvia Marques Higino

IMPACTO DE FATORES DE RISCO EM CASOS CONFIRMADOS DE COVID-19 NO ESTADO DE SÃO PAULO:
Um Estudo com Regras de Associação e Algoritmos para Classificação de Óbitos

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
2022

SUMÁRIO

1. Introdução

1.1. Contextualização

Como é de conhecimento geral, no final de 2019 surgiu na China o novo Coronavírus, um vírus altamente contagioso, que rapidamente originou uma pandemia, levando a óbito milhões de pessoas ao redor do mundo. Até o momento em que este trabalho está sendo escrito, já foram confirmados mais de 350 milhões de casos de Covid-19, doença causada pelo Coronavírus, de acordo com dados divulgados pela Microsoft, pelo link https://www.bing.com/covid. Contudo, acredita-se que o número de casos seja muito maior, já que as instruções dadas à população foram de buscar ajuda médica em casos de sinais de severidade nos sintomas. (OLIVEIRA et al., 2021).

De forma bastante comum a vários tipos de vírus, o Coronavírus, que também é conhecido como SARS-CoV-2, sofre mutações ao longo do tempo. A Organização Mundial da Saúde juntamente a outros parceiros vem monitorando a evolução do vírus, classificando e acompanhando as variantes. As classificações para as variantes são variantes sobre monitoramento, variantes de interesse e variantes de preocupação. No momento de escrita desse trabalho, as variáveis de preocupação são Alpha, Beta, Gamma, Delta e Omicron (OMS).

Covid-19 é o nome dado à doença causada pelo SARS-CoV-2, e os sintomas mais comuns são febre, tosse, cansaço e perda de paladar ou olfato. Porém também é possível apresentar dor de garganta, dor de cabeça, diarreia, uma erupção cutânea na pele ou descoloração dos dedos das mãos ou dos pés, olhos vermelhos ou irritados, ou até mesmo sintomas mais graves como dificuldade em respirar ou falta de ar, perda de fala ou mobilidade, confusão ou dor no peito. A maioria das pessoas que contraem o vírus apresentarão sintomas leves. É verdade que pessoas mais velhas ou com doenças pré-existentes terão risco maior de ter sintomas mais graves ou chegar a óbito. Porém, ao contrário do que muitos pensam, qualquer pessoa, de qualquer idade, pode ficar gravemente doente com Covid-19 e morrer. A melhor forma de se prevenir contra o vírus e evitar a transmissão é se informar bem sobre a doença. (OMS)

O grande obstáculo de se informar bem sobre algum tema que estiver em alta atualmente são as notícias falsas ou “*fake news”*. Ainda que alguns estudos

mostram um interesse das pessoas em não propagar *fake news*, como o trabalho de Tan e Chua (2021), que mostrou mais de 90% das pessoas de uma amostra afirmando não espalhar rumores e o trabalho de Baker et al (2021), onde 93% das pessoas na amostra afirmaram confirmar as informações antes de compartilhá-las, alguns disseram inclusive usar a página da Organização Mundial de Saúde para confirmar a veracidade dos fatos, ainda assim *fake news* são um assunto bastante preocupante no mundo atual, especialmente neste contexto pandêmico.

No trabalho de Tan e Chua (2021) é visto que apesar de ter leis para tentar impedir a divulgação de *fake news* em Singapura, a pesquisa realizada com estudantes mostrou de forma geral que a motivação de não compartilhar *fake news* vem mais do pensamento da notícia não ser útil e pela consideração pelos outros do que por outros motivos. Segundo o estudo, os participantes entendem que compartilhar notícias que podem ser falsas pode ser muito prejudicial, principalmente durante um surto como uma pandemia.

Já o trabalho de Apuke e Omar (2021), também em um contexto de pandemia Covid-19, mostrou o altruísmo como o mais significante dos fatores relacionados ao compartilhamento de *fake news*. O estudo mostrou que as pessoas compartilham *fake news* por acreditarem que a informação é verídica e precisa ser vista por mais pessoas.

Se tanto no meio dos que compartilham quanto no meio dos que não compartilham *fake news* são encontradas boas intensões nas ações, para combater *fake news* em qualquer contexto é preciso que seja gerado mais conhecimento científico sobre o tema em questão. Segundo Viola et al (2021), a educação está fortemente relacionada ao comportamento em comunidade, contribuindo para melhores práticas e responsabilidade tanto de indivíduos quanto de organizações.

No contexto da pandemia Covid-19, a superabundância de informação e também o problema de desinformação (*fake news*) foram reconhecidos pela Organização Mundial de Saúde como fenômeno “*infodemic*” (ou infodemia em tradução livre), por trazer ameaças extras à saúde. O fenômeno contribui com o aumento de problemas como depressão, ansiedade, medo e pânico (YING; CHENG, 2021).

1.2. O problema proposto

No atual contexto de pandemia de Covid-19 e de *infodemic* de sobrecarga de informações e *fake news*, torna-se muito importante produzir conteúdo científico e gerar informações confiáveis através de estudos.

O presente estudo mostrará uma análise exploratória das características dos dados e os resultados de modelos de Machine Learning. Sendo os dados sobre doenças pré-existentes e outros fatores de risco nos casos confirmados de Covid-19 no estado de São Paulo. Os modelos de Machine Learning serão criados com algoritmos de classificação que predizem óbito dos indivíduos com Covid-19 utilizando as informações de comorbidades.

Os dados analisados foram obtidos pelo portal de dados abertos do governo do estado de São Paulo, sendo que o estudo utiliza dados coletados de todos os municípios do estado. São obtidos os dados do início da pandemia até o momento de download dos mesmos, no dia 20 de agosto de 2021. Porém o trabalho registra tentativas de gerar o modelo com período mais específico também, sendo o segundo semestre de 2020. Todos os resultados serão exibidos.

1.3. Objetivos

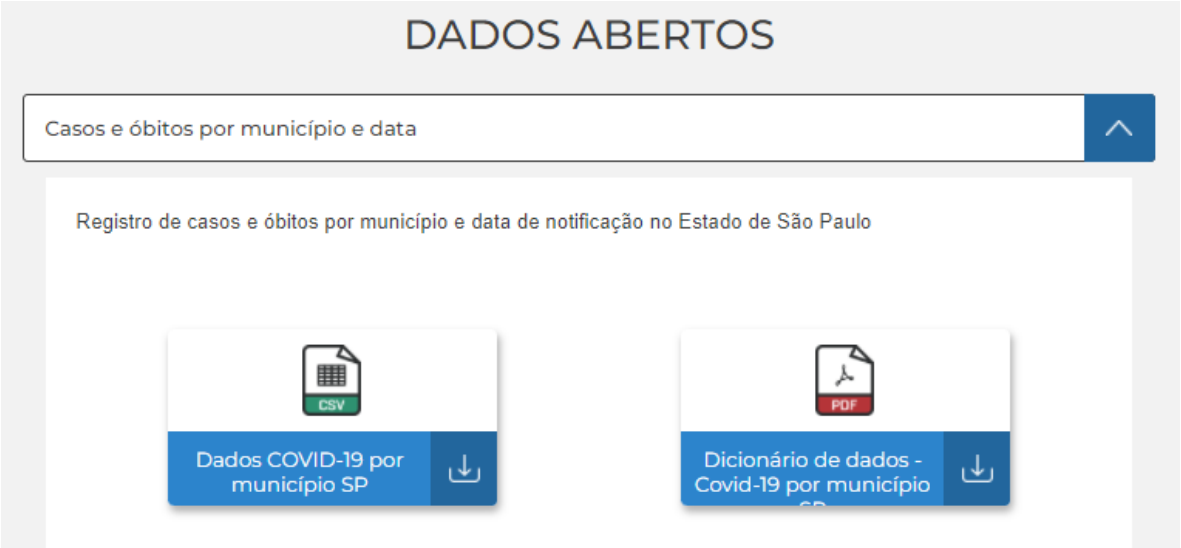
Com este trabalho, tem-se o objetivo de gerar informações e conhecimento relacionados a pandemia Covid19. As informações serão geradas sobre comorbidades (doenças pré existentes e outros fatores de risco) como causas de óbito para casos confirmados de Covid-19 no estado de São Paulo.

Para gerar as informações serão utilizadas análises exploratórias dos dados e criação de modelos preditivos de óbito, com algoritmos de Machine Learning para problemas de classificação.

.

2. Coleta de Dados

Os dados de óbitos e recuperações para os pacientes com Covid-19 no estado de São Paulo foram disponibilizadas pelo próprio governo do estado e extraídos no dia 20 de Agosto de 2021. O arquivo tem sido atualizado diariamente desde o início da pandemia e encontra-se disponível para download, juntamente com seu dicionário, no seguinte endereço: https://www.saopaulo.sp.gov.br/planosp/simi/dados-abertos/.



As informações passadas no dicionário de dados são as seguintes:

Nome da coluna/campo	Conteúdo	Descrição	Observações Relevantes
Gênero	Feminino /Masculino	Gênero do Paciente	
Município	Nome do município	Nome do município do Estado de São Paulo que ocorreu o registro	
Asma	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença respiratória

			crônica
Diabetes	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença causada pela produção insuficiente ou má absorção de insulina, hormônio que regula a glicose no sangue e garante energia para o organismo.
Cardiopatia	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença cardíaca que afeta o coração e os vasos sanguíneos, incluindo problemas estruturais e coágulos.
Doença Hematológica	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença que compromete a produção dos componentes do sangue, como as hemácias (glóbulos vermelhos), leucócitos (glóbulos brancos) e plaquetas, geradas na medula óssea.
Doença	Sim / Não	Indica se tem	Qualquer

Hepática	/ Ignorado	este fator de risco	condição que danifica o fígado e impede seu bom funcionamento.
Doença Neurológica	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença que afetam o cérebro, a medula espinhal e os nervos
Doença Renal	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença que faz com que rins percam a capacidade de filtrarem resíduos, sais e líquidos do sangue.
Imunodepressão	Sim / Não / Ignorado	Indica se tem este fator de risco	Enfraquecimento ou supressão do sistema imunológico ou das .reações imunológicas.
Obesidade	Sim / Não / Ignorado	Indica se apresenta	É uma condição médica em que se verifica acumulação excessiva de tecido adiposo ao ponto de poder ter impacto negativo na saúde.
Pneumopatia	Sim / Não / Ignorado	Indica se tem este fator de risco	É um termo que se refere em

			geral às doenças que afetam os pulmões
Puérpera	Sim / Não / Ignorado	Indica se está neste estágio	É o nome dado à fase pós-parto em que a mulher experimenta modificações físicas e psíquicas
Síndrome de Down	Sim / Não / Ignorado	Indica se tem este fator de risco	Doença genética do cromossomo 21
Outros	Sim / Não / Ignorado	Indica se tem outros fatores de risco	
Diagnóstico Covid19	Confirmado	Confirma se está com Covid	
Data Início Sintomas	Data / Nulo	Data de início dos sintomas	
Idade	Número de anos	Idade do paciente	
Obito	0 / 1	Indica se o paciente veio a óbito	

As informações de longitude e latitude para os limites dos municípios no mapa foram adquiridas de um repositório no GitHub no mesmo dia. O arquivo se encontra disponível através do link https://github.com/tbrugz/geodata-br, onde também pode-se encontrar arquivos geojson dos demais estados do Brasil.

A estrutura do geojson não será mostrada aqui por não ser relevante. Acredita-se ser suficiente informar que o arquivo traz apenas os nomes dos

municípios e uma lista de latitudes e longitudes que formam o desenho desses municípios, funcionando na prática como divisas.

3. Processamento/Tratamento de Dados

É bastante comum, enquanto se explora os dados voltar nas etapas de processamento e tratamento de dados e gerar mais modificações no conjunto de dados. Com este trabalho não foi diferente. Durante a análise exploratória, as modificações nos dados geravam inclusive novos conjuntos de dados, para buscar resultados diferentes na geração dos modelos de Machine Learning.

Portanto, na tentativa de desenvolver um relatório detalhado, em uma ordem clara para o entendimento das etapas do trabalho, optou-se por descrever parte da análise exploratória dos dados junto ao processamento e tratamento de dados. Os gráficos estão presentes durante esta etapa, principalmente no que direciona os processamentos feitos.

O códigos para o trabalho foram desenvolvidos em Python, na versão 3.9. As versões das bibliotecas podem ser consultadas no arquivo requirements.txt, gerado pelo comando pip freeze > requirements.txt. O arquivo encontra-se disponível no repositório do GitHub https://github.com/thalitasylvia/TCC.

Primeiramente, foi feita a importação das bibliotecas python.

```
import numpy as np
import pandas as pd
```

```
import matplotlib.pyplot as plt
from matplotlib import rc
import seaborn as sns
%matplotlib inline
```

```
import datetime
from datetime import timedelta
```

Foi feita então a definição da idade que será considerada idade avançada durante este trabalho. A escolha da idade 60 foi feita de acordo com informações divulgadas pelo Ministério da Saúde.

Em seguida, a leitura do arquivo de dados fornecido pelo portal de dados abertos do estado de São Paulo. Note que originalmente o arquivo possui mais de 4 milhões de linhas.

```
arquivo = '20210819_Casos-e-obitos-ESP.csv' #baixado em 20/08
```

```
idade_referencia = 60 # Segundo o Ministério da Saúde
# https://www.gov.br/saude/pt-br/coronavirus/atendimento-tratamento-e-fatores-de-risco
```

```
df = pd.read_csv(arquivo, sep = ';')
```

```
tam = len(df)
tam
```

```
4195466
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4195466 entries, 0 to 4195465
Data columns (total 19 columns):
 #   Column                   Dtype
---  ------                   -----
 0   Genero                   object
 1   Municipio                object
 2   Asma                     object
 3   Diabetes                 object
 4   Cardiopatia              object
 5   Doenca Hematologica      object
 6   Doenca Hepatica          object
 7   Doenca Neurologica       object
 8   Doenca Renal             object
 9   Imunodepressao           object
 10  Obesidade                object
 11  Pneumopatia              object
 12  Puérpera                 object
 13  Síndrome De Down         object
 14  Outros Fatores De Risco  object
 15  Diagnostico Covid19      object
 16  Data Inicio Sintomas     object
 17  Idade                    float64
 18  Obito                    int64
dtypes: float64(1), int64(1), object(17)
memory usage: 608.2+ MB
```

Sobre os dados faltantes, observa-se que apenas um registro não apresentou a informação de gênero. Parte dos registros não informa a data de início dos sintomas e parte não informa a idade do indivíduo.

```
nulos = df.isnull().sum()
nulos
```

```
Genero                      1
Municipio                   0
Asma                        0
Diabetes                    0
Cardiopatia                 0
Doenca Hematologica         0
Doenca Hepatica             0
Doenca Neurologica          0
Doenca Renal                0
Imunodepressao              0
Obesidade                   0
Pneumopatia                 0
Puérpera                    0
Síndrome De Down            0
Outros Fatores De Risco     0
Diagnostico Covid19         0
Data Inicio Sintomas    37252
Idade                   11757
Obito                       0
dtype: int64
```

Note que cada parte faltante separadamente não chega a 1% do conjunto de dados.

```
print('Porcentagem de dados nulos na coluna Genero: {0:.2f}%'.format(nulos[0]/tam*100))
```

```
Porcentagem de dados nulos na coluna Genero: 0.00%
```

```
print('Porcentagem de dados nulos na coluna Genero: {0:.2f}%'.format(nulos[16]/tam*100))
```

```
Porcentagem de dados nulos na coluna Genero: 0.89%
```

```
print('Porcentagem de dados nulos na coluna Genero: {0:.2f}%'.format(nulos[17]/tam*100))
```

```
Porcentagem de dados nulos na coluna Genero: 0.28%
```

Dada a baixíssima porcentagem de dados nulos, optou-se então por excluílos do dataframe.

```
#Exclusão dos valores nulos
df.dropna(inplace=True)
```

Evidenciando a exclusão dos nulos, a imagem a seguir mostra que não há mais dados nulos no dataframe.

```
nulos = df.isnull().sum()
nulos
```

```
Genero                    0
Municipio                 0
Asma                      0
Diabetes                  0
Cardiopatia               0
Doenca Hematologica       0
Doenca Hepatica           0
Doenca Neurologica        0
Doenca Renal              0
Imunodepressao            0
Obesidade                 0
Pneumopatia               0
Puérpera                  0
Síndrome De Down          0
Outros Fatores De Risco   0
Diagnostico Covid19       0
Data Inicio Sintomas      0
Idade                     0
Obito                     0
dtype: int64
```

A quantidade de registros no dataframe antes desta exclusão era de 4195466. Aós a exclusão possaou a ser de 4153269, ou seja, o dataframe teve 1.01% do seu conteúdo em registros excluído, como mostra a evidência a seguir.

Quantidade de dados antes da exclusão:

```
tam
```

```
4195466
```

Quantidade de dados após a exclusão dos nulos:

```
len(df)
```

```
4153269
```

Porcentagem dos dados originais que foi excluída devido aos nulos:

```
'{:.2f}%'.format((1-(len(df)/tam)) * 100)
```

```
'1.01%'
```

Sobre os dados disponibilizados pelo estado de São Paulo, não se pode dizer se os últimos registros são sobre pessoas que se recuperam ou chegam a óbito, pois esses registros não tiveram tempo o suficiente para recebe essa atualização. Sendo assim, optou-se por remover do dataframe os últimos registros informados, para que não confundam o algoritmo com informações falsas. Como converter a coluna para um tipo de data foi muito custoso para o processamento, foi escolhida a estratégia de aplicar uma função sobre a coluna que analisa a data para saber se está no período válido.

A informação da última atualização do arquivo é obtida do próprio nome do arquivo. Foi feito desta forma para caso for desejado executar os códigos com um arquivo disponibilizado em outra data, seja necessário modificar apenas o nome ou endereço do arquivo no início do código.

```
ultima_atualizacao_dados = datetime.datetime(int(arquivo[:4]), int(arquivo[4:6]), int(arquivo[6:8]))
print(ultima_atualizacao_dados)
```

```
2021-08-19 00:00:00
```

A função para verificar para cada registro se ele contém uma data considerada válida, ou seja, se tem ao menos 20 dias do início dos sintomas, encontra-se a seguir.

```
def dentro_do_periodo(data_inicio_sintomas):
    dia = data_inicio_sintomas[:2]
    mes = data_inicio_sintomas[3:5]
    ano = data_inicio_sintomas[6:]
    data_inicio_sintomas = datetime.datetime(int(ano), int(mes), int(dia))
    if data_inicio_sintomas < (ultima_atualizacao_dados - timedelta(days=20)):
        return True
    else:
        return False
```

Aplicada sobre uma coluna do dataframe, para cada registro a função leva como parâmetro o valor desta mesma coluna e retorna o resultado. Veja que uma nova coluna recebe esse resultado/retorno da função.

```
df['periodo_valido'] = df['Data Inicio Sintomas'].apply(dentro_do_periodo)
```

Observando essa validação do período, apenas uma pequena parte do conjunto de dados é considerada como fora do válido (false), como já era de se esperar.

Assim é possível ver a proporção:

```
df['periodo_valido'].value_counts(normalize=True)
```

```
True     0.987261
False    0.012739
Name: periodo_valido, dtype: float64
```

Já assim pode-se ver a contagem desses registros que serão eliminados:

```
df['periodo_valido'].value_counts()
```

```
True     4100360
False      52909
Name: periodo_valido, dtype: int64
```

A seguir, a exclusão dos registros fora desse período válido, a evidência de que foram removidos e a exclusão da própria coluna, já que esta não será mais utilizada.

```
#Exclusão dos dados fora do período válido
df.drop(df[df['periodo_valido'] == False].index, inplace=True)
```

```
df['periodo_valido'].value_counts()
```

```
True    4100360
Name: periodo_valido, dtype: int64
```

```
#Exclusão da coluna, pois não será mais útil
df.drop('periodo_valido', axis=1, inplace=True)
```

Das informações mais básicas e primordiais sobre os registros relatados no dataset sendo estudado, registra-se todos se referem a casos de Covid-19 confirmados.

```
df['Diagnostico Covid19'].value_counts()
```

```
CONFIRMADO    4100360
Name: Diagnostico Covid19, dtype: int64
```

Observa-se também a porcentagem de óbitos entre estes casos confirmados. Trata-se de uma porcentagem bem baixa, a maioria dos casos se recuperam.

```
df['Obito'].value_counts(normalize=True)
```

```
0    0.965052
1    0.034948
Name: Obito, dtype: float64
```

```
df['Obito'].value_counts()
```

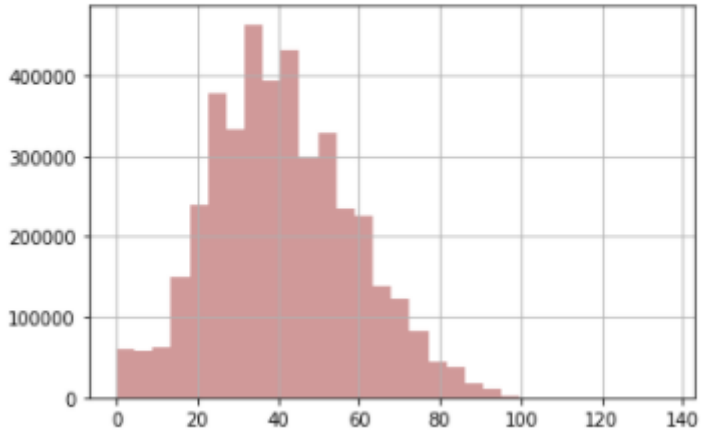
```
0    3957062
1     143298
Name: Obito, dtype: int64
```

O gráfico a seguir ilustra essa baixa porcentagem de óbitos entre os casos registrados no conjunto de dados.

Uma outra característica dos registros a ser explorada é a idade dos indivíduos. Note que a distribuição da idade entre os casos recuperados se parece muito com a distribuição geral, enquanto a distribuição das idades nos casos que chegaram a óbito já é diferente.

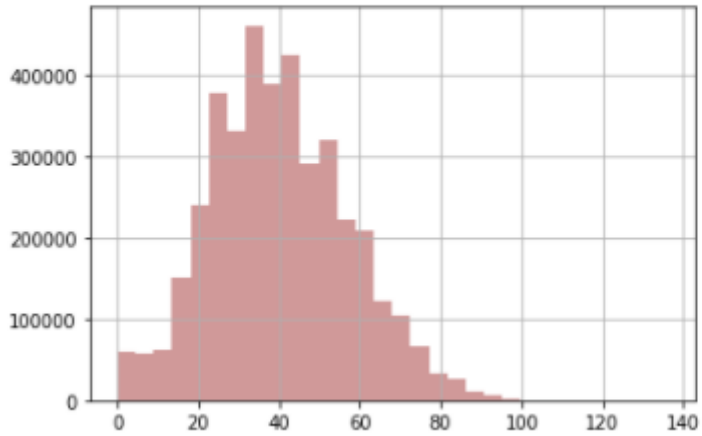
```
#Distribuição geral
df['Idade'].hist(bins=30, color='darkred', alpha=0.4)
```

```
<AxesSubplot:>
```



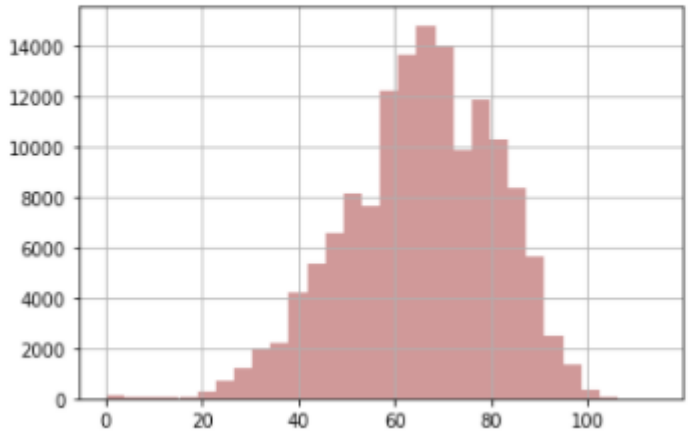

```
#Distribuição de idade entre os casos recuperados
df[df['Obito']==0]['Idade'].hist(bins=30, color='darkred', alpha=0.4)
```

```
<AxesSubplot:>
```



```
#Distribuição de idade entre os casos de óbito
df[df['Obito']==1]['Idade'].hist(bins=30, color='darkred', alpha=0.4)
```

```
<AxesSubplot:>
```



Se as idades mínimas e máximas forem observadas entre os registros, percebe-se a ocorrência de dados ruidosos. É possível que a idade 0 seja a ausência da informação, enquanto idades muito altas entre os dados é podem ser erros de cadastro. O gráfico do tipo boxplot vai evidenciar os outliers a serem removidos para que a informação seja tratada.

```
df['Idade'].mean()
```

```
41.031705021022546
```

```
df['Idade'].min()
```

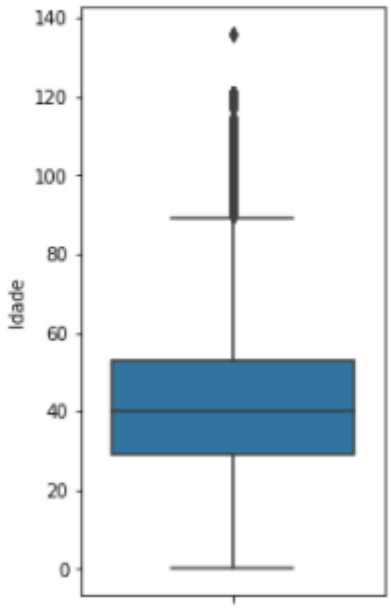
```
0.0
```

```
df['Idade'].max()
```

```
136.0
```

```
#Boxplot das idades (geral)
plt.figure(figsize=(3, 6))
sns.boxplot(y='Idade', data=df )
```

```
<AxesSubplot:ylabel='Idade'>
```

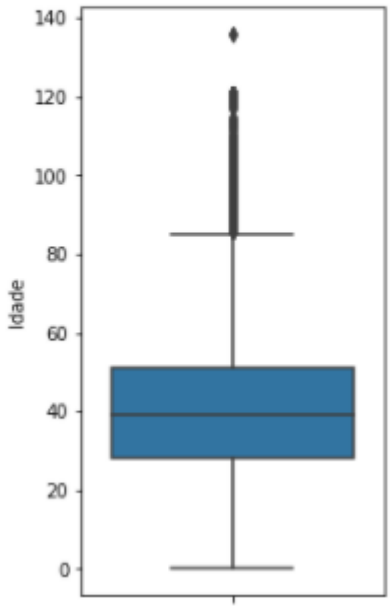


Os dados com idades registradas acima do limite superior no boxplot são considerados outliers. Este limite foi calculado e resultou em 89 anos.

A análise dos casos recuperados novamente bastante semelhante à análise dos casos em geral:

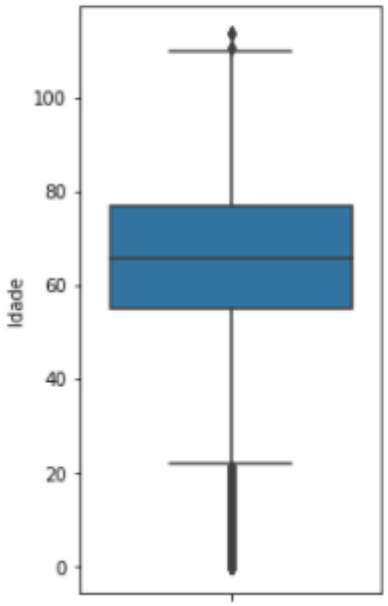
```
#Boxplot das idades (casos recuperados)
plt.figure(figsize=(3, 6))
sns.boxplot(y='Idade', data=df[df['Obito']==0] )
```

```
<AxesSubplot:ylabel='Idade'>
```




```
#Boxplot das idades (casos de óbitos)
plt.figure(figsize=(3, 6))
sns.boxplot(y='Idade', data=df[df['Obito']==1] )
```

```
<AxesSubplot:ylabel='Idade'>
```



O limite inferior deste teceiro boxplot tamém foi calculado, resultando em 22 anos. Esse gráfico indica que são tão poucos os indivíduos de menos de 22 anos chegam a óbito, que chegam a ser outliers.

Esses outliers porém não serão excluídos, já que trata-se dos dados filtrados pelo resultado de óbito.

Realmente a porcentagem de óbitos cai muito quando a idade é abaixo do limite citado. Veja o gráfico a seguir.

Observando a situação das idades informadas de forma decrescente, é possível observar uma grande quantidade de idades não válidas. Veja a seguir as 10 idades mais altas bem como a quantidade de registros informados com cada uma destas idade.

idade	quantidade
136.0	2
121.0	39
120.0	24
119.0	2
118.0	3
117.0	2
115.0	1
114.0	5
113.0	3
112.0	2

Em sequência, obeserve também a ordenação crescente das idades e suas respectivas quantidades de ocorrências nos dados.

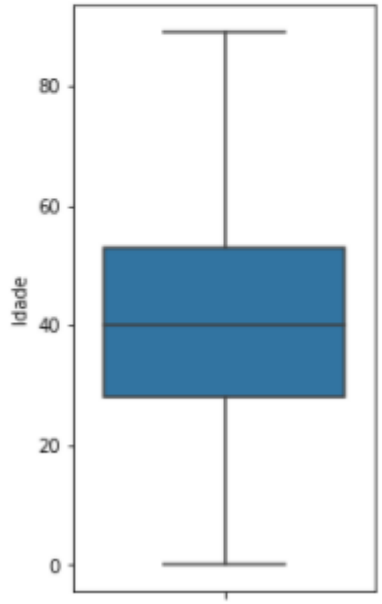
idade	quantidade
0.0	16244
1.0	13413
2.0	10770
3.0	9859
4.0	9269
5.0	10222
6.0	10621
7.0	11133
8.0	12203
9.0	12462

Levando em consideração o limite superior 89.0 para a distribuição geral de idades, foi decidido por remover do conjunto de dados os pacientes registrados com idade superior a esta. Esta medida também remove as idades inválidas mencionadas. O print a seguir mostra a exclusão dos dados e também a evidência desta exclusão.

```
#Exclusão dos dados onde a idade é considerada outlier
df.drop(df[df['Idade'] > limite_superior].index, inplace=True)
```

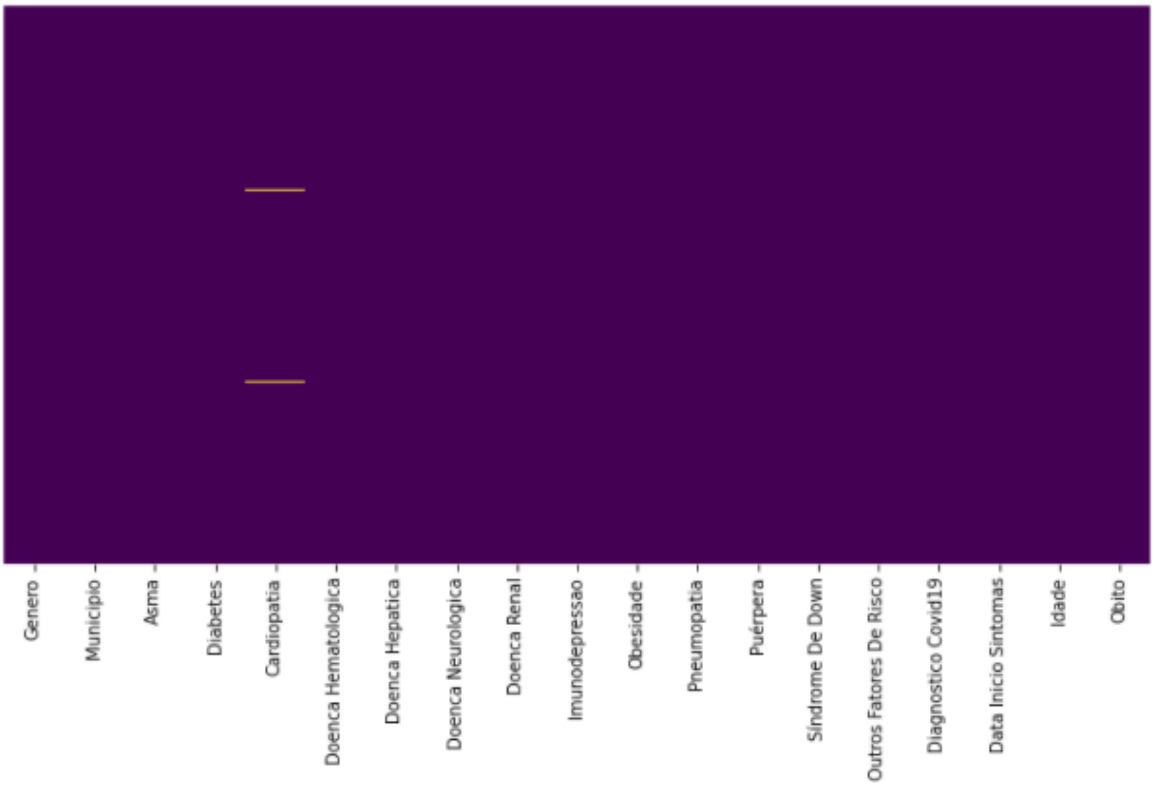
```
#Resultado da exclusão, não há mais outliers nos dados em geral
plt.figure(figsize=(3, 6))
sns.boxplot(y='Idade', data=df )
```

```
<AxesSubplot:ylabel='Idade'>
```

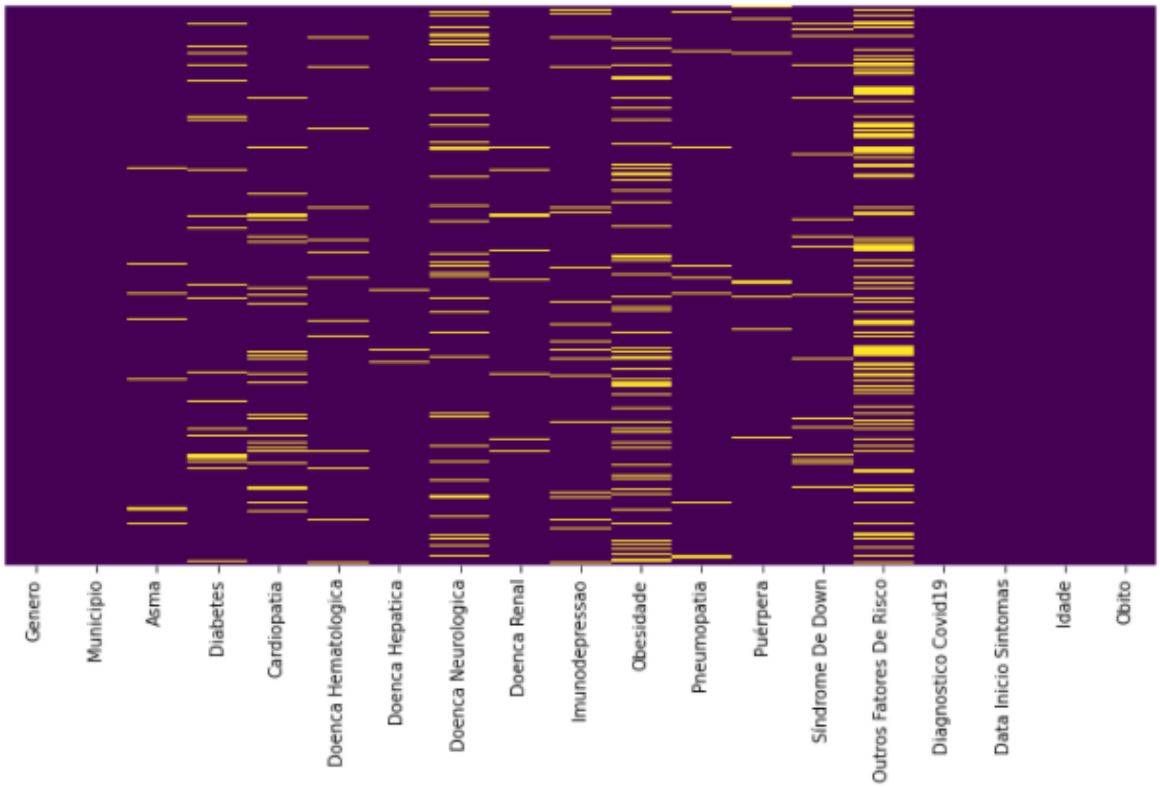


Já que o boxplot das idades para os casos de óbitos mostrou outliers antes do limite inferior, é analisada a situação dos óbitos referente à jovens e crianças com idade informada abaixo deste limite.

A imagem a seguir põe em ênfase casos de comorbidade para jovens e crianças de menos de 22 anos (limite inferior) que conseguiram se recuperar da Covid-19. Trata-se de 499066 casos na amostra.



Já esta próxima imagem mostra os casos de comorbidades para os mesmos jovens (menos de 22 anos) que infelizmente vieram a óbito. A imagem a seguir representa 490 casos de Covid na amostra, até este momento.



Esta imagem mostrou a importância destes registros na amostra, pois apesar da baixa idade, a presença de doenças pré-existentes fizeram diferença no desfecho negativo dos casos. Sendo assim, apesar de considerados outliers pelo boxplot (filtrando apenas os óbitos), não se deve remover os dados de pacientes crianças e jovens de até 22 anos.

Gênero

A respeito da característica gênero, dada a porcentagem extremamente baixa de indivíduos com gênero informado como 'indefinido' ou 'ignorado', fica decidido removê-los também.

```
df['Genero'].value_counts()
```

```
FEMININO      2165297
MASCULINO     1916280
INDEFINIDO       2149
IGNORADO           37
Name: Genero, dtype: int64
```

```
df['Genero'].value_counts(normalize=True)
```

```
FEMININO      0.530221
MASCULINO     0.469244
INDEFINIDO    0.000526
IGNORADO      0.000009
Name: Genero, dtype: float64
```

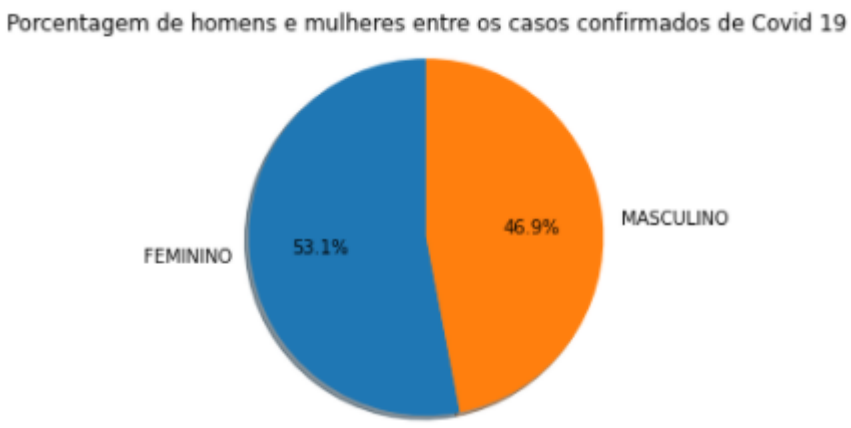
A seguir, a exclusão e a evidência da mesma.

```
#Exclusão dos registros onde o gênero é "INDEFINIDO" ou "IGNORADO"
df.drop(df[(df['Genero'] == 'INDEFINIDO') | (df['Genero'] == 'IGNORADO')].index, inplace=True)
```

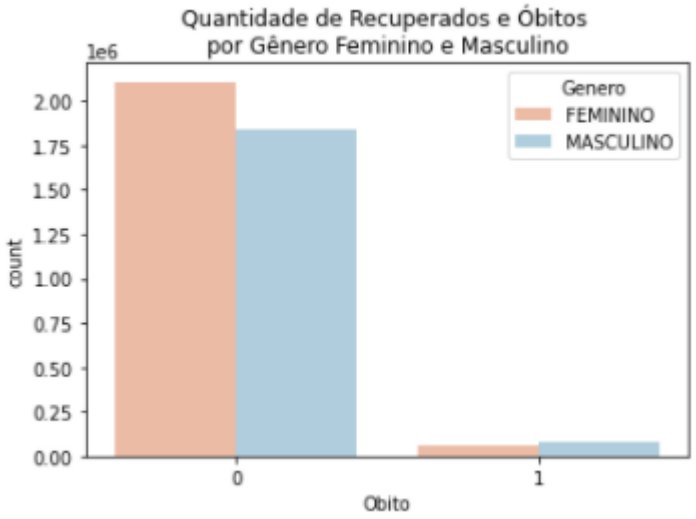
```
#Resultado da Exclusão
df['Genero'].value_counts()
```

```
FEMININO     2165297
MASCULINO    1916280
Name: Genero, dtype: int64
```

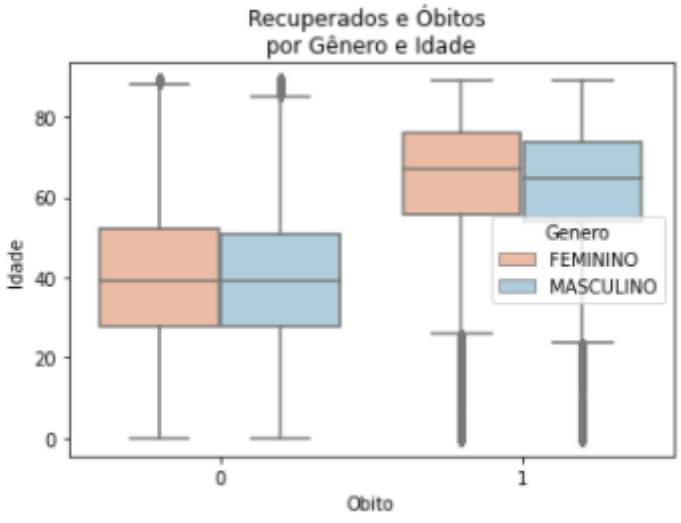
Entre os dados que permanecem no conjunto em estudo, é mostrado a seguir a porcentagem dos gêneros declarados como masculino e feminino.



O gráfico a seguir, apesar da contagem não estar na notação mais usual, é bastante útil para analisar óbitos por gênero. Mostrando que mais mulheres contraíram o vírus do que homens. Porém ao olhar para óbitos, é possível notar que são os homens que tem uma contagem maior de óbitos.



Se for observado o boxplot a seguir, pode ser visto que apesar do gênero masculino apresentar uma contagem maior de desfechos em óbito, as idades mais altas ficam com as pacientes do gênero feminino.



Análise de Comorbidades

Ao trazer um head() do dataframe, observa-se nesse ponto que grande parte dos dados está como “IGNORADO”. Primeiramente, para entender o nível de completude do dataset, foi criada uma coluna chamada “comorbidade”, que recebeu “SIM”, caso essa também fosse a resposta fornecida para todas as comorbidades, “NÃO” caso fosse não para todas, “IGNORADO” usando o mesmo critério e, por fim, “ALTERNADO” onde ainda estivesse nulo. Esta última resposta fica obviamente para os registros que receberam respostas diferentes para as diferentes comorbidades.

Analisando as quantidades e porcentagens para cada classificação nesta coluna criada, percebe-se que a maioria alarmante dos registros teve todas as informações de comorbidades ignoradas. Excluir estes dados significaria uma perda enorme de informação para o estudo.

```
df['Comorbidade'].value_counts()
```

```
IGNORADO     3526136
ALTERNADO     555158
NÃO              279
SIM                4
Name: Comorbidade, dtype: int64
```

```
df['Comorbidade'].value_counts(normalize=True).apply(lambda x: format(x, 'f'))
```

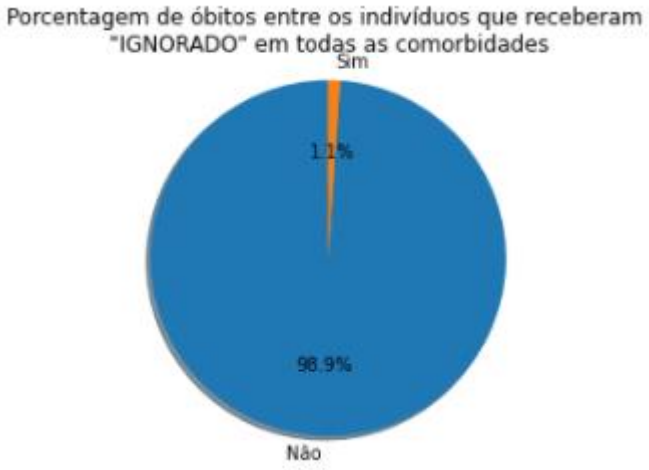
```
IGNORADO     0.863915
ALTERNADO    0.136016
NÃO          0.000068
SIM          0.000001
Name: Comorbidade, dtype: object
```

Note que 13,6% dos registros tem respostas alternadas, o que traz mais confiabilidade para as informações. Já os registros informados como “NÃO” ou como “SIM” para todas as comorbidades são tão poucos que a porcentagem se aproxima de zero, levando a acreditar que os registros “faltando” nesta parte podem estar na

classificação do “IGNORADO”, que está com uma quantidade enorme em comparação aos outros.

É mais provável que os registros que receberam “IGNORADO” em todos sejam, em sua maior parte, de pacientes sem doenças prévias, e simplesmente não tenham sido preenchidos pelos profissionais da saúde que os atenderam.

Algo possível de impactar na completude do fornecimento dessas informações é o estado do paciente de acordo com sintomas. Casos confirmados de Covid-19 com sintomas graves podem tornar mais importante que as comorbidades sejam informadas no momento do cadastro. Isto explicaria como os conjuntos de dados que não tiveram todas as comorbidades ignoradas tiveram maior desfecho em óbitos. Relembrando que a taxa de óbito no conjunto de dados geral está em 3,3%. Observe as situações pela completude de informações fornecidas:




Porcentagem de óbitos entre os indivíduos que receberam respostas diferentes entre as comorbidades
Sim
17.8%
82.2%
Não


Porcentagem de óbitos entre os indivíduos que receberam "NÃO" em todas as comorbidades
Sim
32.6%
67.4%
Não

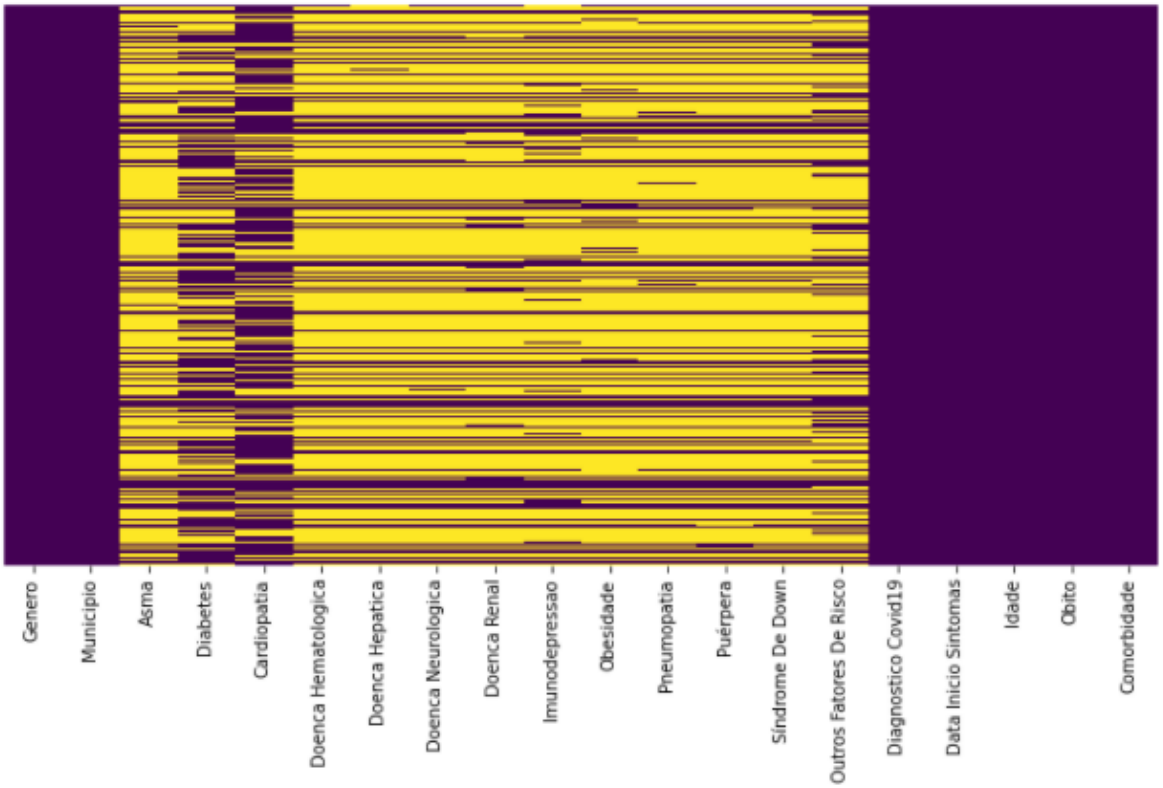

Porcentagem de óbitos entre os indivíduos que receberam "SIM" em todas as comorbidades
Sim
25.0%
75.0%
Não

Após isto, a coluna de comorbidades foi excluída e construída novamente de outra forma. Na nova definição, qualquer comorbidade informada como “SIM” já define a coluna como “SIM”. Os demais registros (que não possuem nenhum “SIM”), recebem “NÃO” na coluna comorbidade.

Sendo assim, o gráfico a seguir mostra a porcentagem de indivíduos que apresentavam ao menos uma comorbidade qualquer como “SIM” e os que não apresentavam nenhuma comorbidade como “NÃO”.

Destes 13,6% com alguma comorbidade, a porcentagem de desfechos de óbito foi de 17.6%, já entre os indivíduos sem nenhuma comorbidade informada foi de 1.1%.

Diante das características citadas, optou-se neste estudo por excluir os registros que receberam “IGNORADO” para todas as comorbidades. Desta forma, a amostra dedos para o estudo é reduzida de mais de 4 milhões de dados para menos de 600 mil, optando por permanecer informações provavelmente mais confiáveis. Ainda assim, há muitos dados marcados como “IGNORADO”, entretanto, como ao menos alguma das comorbidades para estes pacientes foi informada, acredita-se que o registro tenha uma confiabilidade maior. O gráfico a seguir mostra estas informações apresentadas como “IGNORADO”.



A palavra “IGNORADO” no conjunto de dados foi substituída por “NÃO”. Considerando que se não foi informado é devido ao paciente não apresentar a condição descrita. O mesmo gráfico executado após este tratamento ficaria apenas roxo, sem os traços amarelos, pois ele destaca a palavra “IGNORADOS” no conjunto de dados.

Neste ponto do estudo, foi gerado o primeiro arquivo de dummies para ser posteriormente usado nos modelos de Machine Learning. O arquivo de dummies é constituído de colunas binárias, contendo apenas os valores 0 e 1. A construção deste arquivo se dá da seguinte forma, cada variável categórica que será usada pelos modelos é transformada em N-1 outras colunas onde N é a quantidade de classes da variável. Subitrai-se 1 pois com as demais colunas já é possível inferir esta que foi removida, diminuindo a quantidade de colunas para serem processadas pelo algoritmo e melhorando assim sua performance. Um exemplo simples é o gênero. Não há necessidade de ter as colunas “feminino” e “masculino”, com apenas uma das duas colunas já se sabe o gênero do indivíduo, utilizando um conteúdo de “zeros e uns”.

Feito isso com todas as colunas categóricas,parte do dataframe resultante pode ser visto a seguir. Assim foi gerado também o prmeiro arquivo de dummies.

```
df.head()
```

a	Doenca Hematologica	Doenca Hepatica	Doenca Neurologica	Doenca Renal	Imunodepressao	Obesidade	Pneumopatia	Puérpera	Síndrome de Down	Outros Fatores De Risco	Comorbidade	idade_avancada	Obito
1	0	0	0	0	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	1	1	0

```
df.to_csv(r'dados_dummies_1.csv', sep = ';', index=False)
```

Tratamento das faixas etária

Criando alguns modelos de Machine Learning, foi decidido por voltar ao tratamento e processamento de dados para tentar resultados melhores criando faixas etárias para o conjunto de dados. Ao invés de usar a informação binária sobre ser ou não de idade avançada, optou-se por descrever melhor a característica de idade para o modelo, utilizando as faixas etárias.

As faixas etárias foram definidas usando décadas, exceto a primeira, que incluiu indivíduos com até 22 anos, a segunda que foi de 23 a 29 anos (apenas completando a década). A primeira faixa etária foi definida até 22 anos devido estes indivíduos terem sido outliers quando analisados apenas os casos de óbitos e também pelo fato de que ainda não há correlação para estas idades e a porcentagem de óbitos, como visto anteriormente neste trabalho.

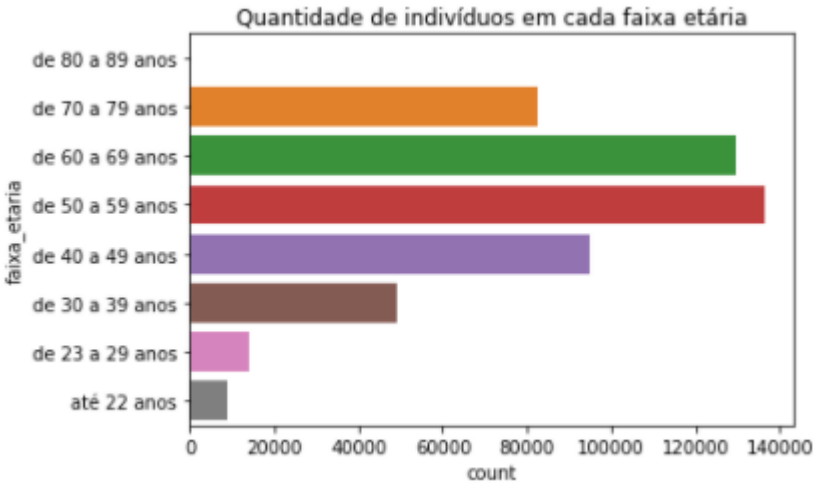
A última faixa etária incluiu indivíduos de 80 anos ou mais (ou até 89 anos, já que os que tinham mais do que isto foram excluídos como outliers), de qualquer forma, continuaria representando a década.

A seguinte função foi definida e aplicada aos dados para a criação da coluna de faixa etária:

```python
def faixa_etaria(idade):
    if idade <= 22:
        return 'até 22 anos'
    elif idade >= 23 and idade < 30:
        return 'de 23 a 29 anos'
    elif idade >= 30 and idade < 40:
        return 'de 30 a 39 anos'
    elif idade >= 40 and idade < 50:
        return 'de 40 a 49 anos'
    elif idade >= 50 and idade < 60:
        return 'de 50 a 59 anos'
    elif idade >= 60 and idade < 70:
        return 'de 60 a 69 anos'
    elif idade >= 70 and idade < 80:
        return 'de 70 a 79 anos'
    else:
        return '80 anos ou mais'
```

```python
df['faixa_etaria'] = df['Idade'].apply(faixa_etaria)
```

O gráfico a seguir mostra o resultado da definição de faixa etária, em quantidade de indivíduos.



Neste ponto do trabalho foi salvo as segunda versão do arquivo de dados tratados e logo em seguida o arquivo de dummies. Optou-se por não repetir a geração dos dummies, mas abrir o que estava salvo da primeira versão e apenas

concatenar os dummies das faixas etárias,excluindo a antiga coluna “idade_avançada”.

Tratamento de afunilamento do período

Com o intuito de tentar melhorar um pouco mais os modelos, optou-se por voltar ao processamento dos dados e selecionar o período do segundo semestre do ano de 2020. A ideia é tentar evitar o início da pandemia, onte os testes de covid talvez não fossem tão precisos e também evitar dados onde pessoas já possam ter sido vacinadas, atrapalhando a predição. Basicamente, tentar selecionar um período mais específico.

A função definida a seguir foi usada para classificar se os registros estão dentro ou fora do período desejado e foi aplicada aos dados.

```
def dentro_do_semestre(data_inicio_sintomas):
    dia = data_inicio_sintomas[:2]
    mes = data_inicio_sintomas[3:5]
    ano = data_inicio_sintomas[6:]
    data_inicio_sintomas = datetime.datetime(int(ano), int(mes), int(dia))
    if data_inicio_sintomas >= datetime.datetime(2020, 7, 1) and data_inicio_sintomas <= datetime.datetime(2020, 12, 31):
        return True
    else:
        return False
```

```
df['periodo_valido'] = df['Data Inicio Sintomas'].apply(dentro_do_semestre)
```

No código do processamento foi separado a classificação de período válido de todos os registros até então na amostra. Em seguida foram excluídos do dataframe os dados fora do período desejado, restando 163820 registros. A coluna de verificação também foi excluída, por não ser mais útil. Então foi salvo o arquivo da terceira versão de dados tratados.

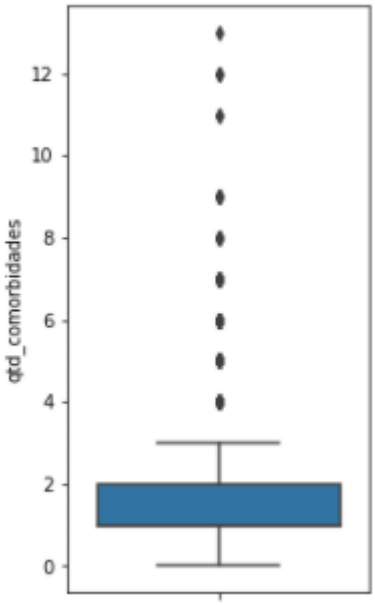
Já para o arquivo de dados dummies, optou-se por concatenar a classificação de período válido de todos os registros, separada anteriormente para isto. Os dados fora do período foram excluídos, bem como a coluna se classificação. Foi gerada então a terceira versão do arquivo de dados dummies. Exatamente com as mesmas

informações do arquivo de dados tratados, porém com as informações em zeros e uns, contendo as mesmas 163820 linhas.

Quantidade de Comorbidades

A tentativa agora é ao invés de uma coluna binária informando se o indivíduo possue alguma comorbidade, informar quantas comorbidades que ele apresenta. Essa informação precisará se tornar categórica posteriormente.

Nesta quarta tentativa de criar um conjunto de dados que melhore o resultado dos algoritmos, os arquivos de dados tratados e dummies serão gerados a partir das versões número 3. Foi criada uma coluna que contém a soma das comorbidades, representando a quantidade. Analisando com o boxplot a seguir, ficou decidido classificar a quantidade de comorbidades como "1", "2", "3" e "4 ou mais"



Foi definida a função para essa categorização e aplicada à coluna

```python
def categorizar_comorbidades(qtd_comorbidades):
    if qtd_comorbidades == 1:
        return '1'
    elif qtd_comorbidades == 2:
        return '2'
    elif qtd_comorbidades == 3:
        return '3'
    else:
        return '4 ou mais'
```

```python
df['qtd_comorbidades'] = df['qtd_comorbidades'].apply(categorizar_comorbidades)
```

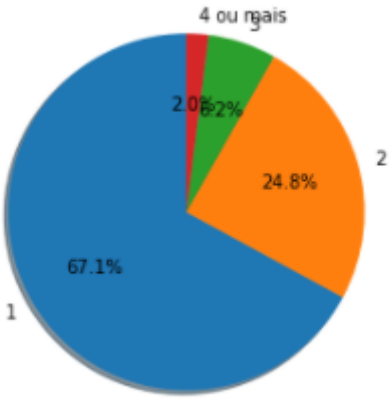
Os resultados das categorias foram os seguintes

```python
df['qtd_comorbidades'].value_counts()
```

```
1            109878
2             40551
3             10122
4 ou mais      3269
Name: qtd_comorbidades, dtype: int64
```

```python
df['qtd_comorbidades'].value_counts(normalize=True)
```

```
1            0.670724
2            0.247534
3            0.061787
4 ou mais    0.019955
Name: qtd_comorbidades, dtype: float64
```



Assim como já foi feito anteriormente, uma cópia do coluna foi guardada para ser adicionada nos dados tratados posteriormente. A coluna foi transformada em dados dummies e a coluna Comorbidade do terceiro arquivo de dummies foi substituída por esses dados dummies de quantidade de comorbidades, gerando a quarta versão do arquivo de dummies. O mesmo foi feito com o arquivo de dados tratados, porém usando a cópia da coluna que foi guardada,não os dados dummies.

Segundo conjunto de dados

O segundo dataset obtido para este trabalho foi retirado do GitHub e encontra-se no repositório disponível pelo link https://github.com/tbrugz/geodata-br. Trata-se de um geojson, um arquivo do tipo json que contém coordenadas geográficas que formar as divisas, neste caso, dos municípios do Estado de São Paulo.

A respeito da estrutura deste json, não é uma estrutura intuitiva, por isso não será detalhada muito a fundo neste trabalho. É suficiente dizer que há no json uma lista de dicionários, um para cada município do estado. Cada dicionário tem, entre outras informações, um conjunto de latitudes e longitudes que especificam os limites do município.

A função trata_nome foi definida e usada para remover os acentos dos nomes dos municípios, tanto nos dados que vêm sendo usados até o momento, quanto no geojson. Isso é importante pois o nome do município será usado como chave para o join entre esses dois conjuntos de dados, então procura-se garantir que estejam iguais. Alguns joins utilizam uma característica chamada “id” no geojson. No caso do arquivo sendo utilizado, o id precisou ser gerado copiando o nome do município, pois o json não tinha a propriedade id.

```
#Trata os municípios no dataframe:
df['Municipio'] = df['Municipio'].apply(trata_nome)
```

```
lista = []
#itera na lista de dicionarios
for municipio in SP_municipios["features"]:
    #Trata os municípios no dicionário:
    municipio["properties"]["name"] = trata_nome(municipio["properties"]["name"])
    municipio["id"] = municipio["properties"]["name"] #Cria os ids, pois não vieram no arquivo
    lista.append(municipio["id"])
```

Após este tratamento de retirada dos acentos, foi feito um left join dos dados do csv para o json e vice versa. Analisando o que não encontrou equivalência de um conjunto de dados para o outro, foram encontrados 4 nomes de municípios escritos de forma errada no geojson e "ARCO-ÍRIS" escrito sem o hífem nos dados abertos do estado de São Paulo. Também nos dados abertos, em 36 registros o município não foi informado.

Os erros foram corrigidos utilizando os códigos a seguir e após isto o join passou a ser exato entre os conjuntos de dados.

```
#Corrige o que estava errado no json
for municipio in SP_municipios["features"]:
    nome = municipio["properties"]["name"]

    if nome == 'EMBU':
        nome = 'EMBU DAS ARTES'
    elif nome == 'FLORINIA':
        nome = 'FLORINEA'
    elif nome == 'MOJI MIRIM':
        nome = 'MOGI MIRIM'
    elif nome == 'SAO LUIS DO PARAITINGA':
        nome = 'SAO LUIZ DO PARAITINGA'

    municipio["properties"]["name"] = nome
    municipio["id"] = nome
```

```
#Corrige o que estava errado no csv
df.replace('ARCO IRIS', 'ARCO-IRIS', inplace=True)
```

Os dados corrigidos foram salvos, com o seguinte código

```
df.to_csv(r'dados_tratados_1.csv', sep = ';', index=False)
```

```
with open('municipios_tratados_geo.json', 'w') as salvar:
    json.dump(SP_municipios, salvar,  indent=4)
```

Em seguida, foi gerado um arquivo para ser usado na geração do mapa do estado de São Paulo para o painel do trabalho. O mapa apresenta os municípios coloridos de vermelho, onde a intensidade da cor acompanha a proporção de óbitos entre os casos de Covid-19 na amostra em questão.

Sobre os arquivos gerados

A respeito dos arquivos gerados durante tratamentos e processamento dos dados, os arquivos dummies são de conteúdo 0 e 1 para uso pelos modelos de Machine Learning. Já os arquivos de dados tratados são apenas para acompanhar essas versões de dummies. São para facilitar, caso se deseje analisar os dados usados por algum modelo. Os arquivos são como checkpoits para este projeto, de forma geral.

Demais arquivos foram salvos para serem usados no desenvolvimento do painel, sem que fosse necessário refazer os tratamentos realizados.

Só para que não reste dúvidas, será mostrado a quantidade de registros e características de formação para os arquivos, sendo que para cada tentativa, os novos arquivos eram gerados a partida da tentativa anterior.

Os arquivos para a tentativa 1 são o dados_tratados_1 e o dados_dummies_1. Ambos possuem 555441 registros e sobre eles foram aplicados os tratamentos mais básicos.

Para a tentativa 2 foi incluído uma categorização de faixas etárias ao invés da informação binária sobre o indivíduo ter idade avançada ou não. Permanecem as

quantidades de linhas e os arquivos gerados foram o dados_tratados_2 e o dados_dummies_2.

A tentativa 3 foi gerada a partir da 2, fazendo a escolha de um período específico. Os arquivos gerados foram dados_tratados_3 e o dados_dummies_3 e a quantidade de linhas caiu para 163820.

Já na tentativa 4 (dados_tratados_3 e o dados_dummies_3), além da faixa etária e do semestre específico, a informação binária sobre o indivíduo ter comorbidades ou não foi subistituída por uma variável categórica que diz quantos são os fatores de risco que ele apresenta, sendo as classes "1", "2", "3" e "4 ou mais". Permanece a quantidade de linhas.

Feita a partir dos dados da tentativa 4, a quinta tentativa traz um balanceamento utilizando undersampling. Por se tratar apenas de já definir quais dados serão pra treino e quais pra teste, não foi criado um arquivo de dados tratados, considera-se o dados_tratados_4. São gerados 2 arquivos, dados_dummies_5_train e dados_dummies_5_test, que juntos possuem a mesma quantidade de linhas das tentativas anteriores, 163820 registros. Na mesma linha de estratégia de resampling foram gerados os arquivos

Também foram gerados os arquivos para o painel, sendo eles municipios_tratados_geo.json, df_mapa, dados_grafico_idadeXobito e df_comorbidades_graficos_barras.

O municipios_tratados_geo.json, que é o mesmo geojson baixado da internet, porém contendo os id's incluídos e alguns nomes de municípios corrigidos. Os arquivos df_mapa, dados_grafico_idadeXobito e df_comorbidades_graficos_barras foram criados a partir do dados_tratados_1 e utilizados na criação do painel.

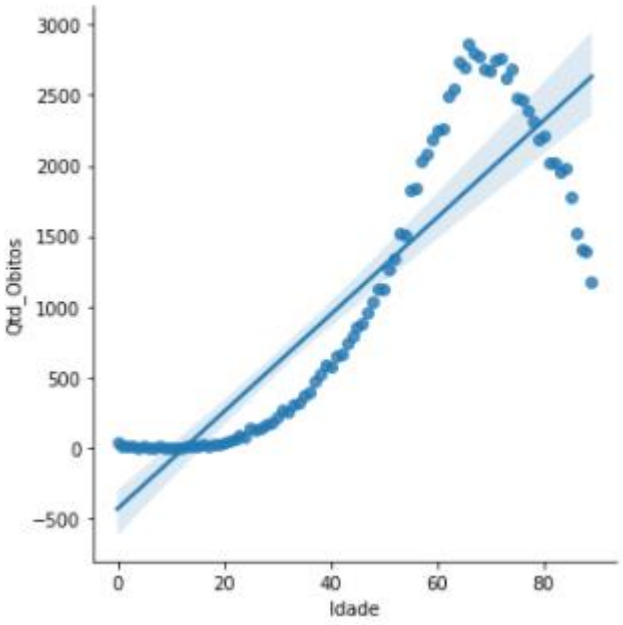
4. Análise e Exploração dos Dados

Muito já foi mostrado em relação à Análise Exploratória dos dados e documentada junto à parte de tratamento e processamento dos dados. Mais algumas análises serão feitas a seguir, gerando também alguns dados para o painel ao final do trabalho.

Idade X Obito

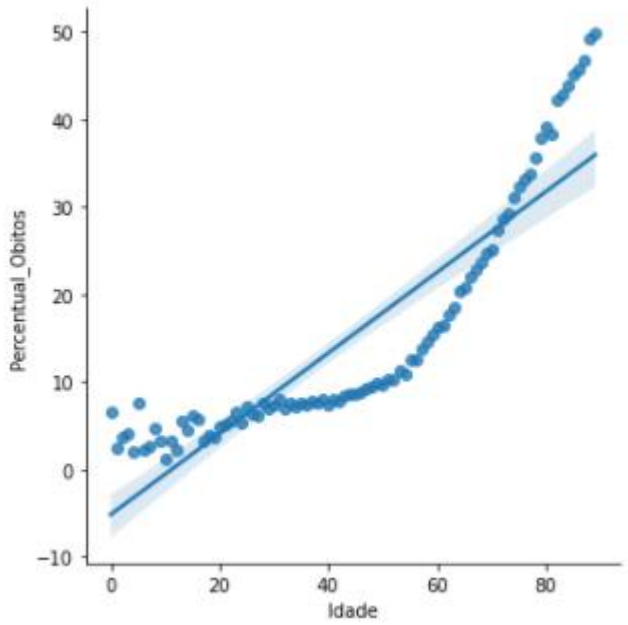
Ao analisar a característica idade dos indivíduos juntamente ao desfecho de óbito, observa-se que há uma tendência para a quantidade de óbitos aumentar quando se aumenta também o valor da idade. A correlação entre a idade e a quantidade de óbitos é positiva e bastante alta, com o valor de 0.875678.

No gráfico a seguir, os pontos mostram a distribuição de idade nos casos de óbitos, porém é possível ver também a correlação entre o aumento da idade e o aumento da quantidade de casos sendo uma correlação positiva e crescente indicada pela linha azul.



A somatória dos desfechos de óbito começam a cair a partir de aproximadamente 65 anos de idade, acredita-se que devido principalmente ao isolamento social. A própria distribuição geral das idades, apresentada anteriormente neste mesmo estudo, mostra que a quantidade de indivíduos com Covid vai caindo quando se observa idades mais avançadas. Lembrando que este trabalho já se refere apenas a casos confirmados da doença. Se os dados abrangessem também os casos com resultado negativo para covid, talvez a distribuição fosse diferente, dependendo da distribuição das idades na própria população do estado de São Paulo.

Usando agora a porcentagem de óbitos em cada idade é mais fácil entender a correlação da idade avançada com a maior probabilidade de óbitos do que se for usado apenas a contagem dos casos de óbito, como feito no gráfico anterior. Observe:



Se lembrarmos que no boxplot filtrando apenas os óbitos foi obtido um limite inferior de 22 anos, onde os casos que evoluíram a óbito eram “outliers” (ao menos dentro do contexto de óbito) fica fácil perceber o porquê das porcentagens estarem mais “desorganizadas” nestas idades mais jovens e não seguirem a correlação. São

poucos os casos nas idades mais jovens que obtiveram desfecho de óbito. Inclusive nestes casos o óbito se deve a outros fatores que não sejam a idade, por isso a falta de correlação nesta parte do gráfico.

Asma

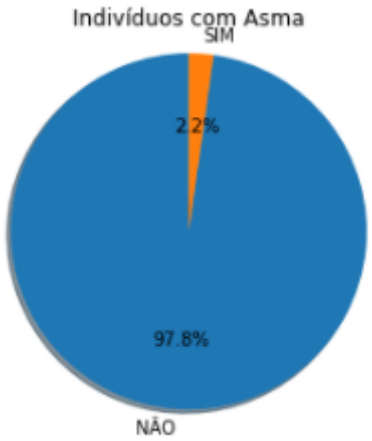
Analisando agora a comorbidade asma. Observe que 2,2% dos indivíduos na amostra apresentam a doença, sendo que destes indivíduos com asma, 25,1% chegaram a óbito.

```
df['Asma'].value_counts()
```

```
NÃO    543064
SIM     12377
Name: Asma, dtype: int64
```

```
df['Asma'].value_counts(normalize=True)
```

```
NÃO    0.977717
SIM    0.022283
Name: Asma, dtype: float64
```



Diabetes

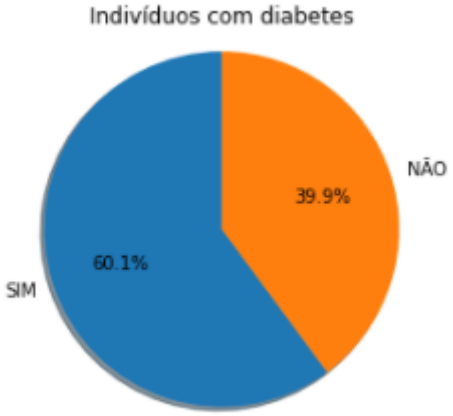
No caso da comorbidade diabetes, 39,9% dos indivíduos na amostra apresentam a doença e que 19,1% dos destes indivíduos chegaram a óbito.

```
df['Diabetes'].value_counts()
```

```
NÃO    333923
SIM    221518
Name: Diabetes, dtype: int64
```

```
df['Diabetes'].value_counts(normalize=True)
```

```
NÃO    0.601185
SIM    0.398815
Name: Diabetes, dtype: float64
```



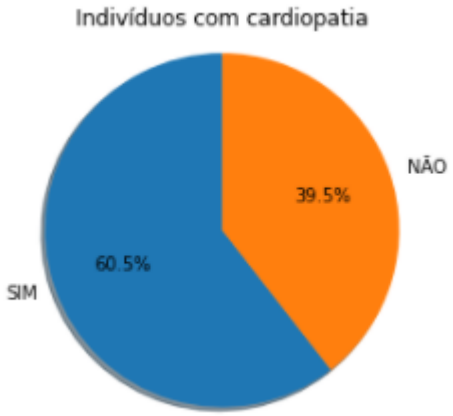
Cardiopatia

```
df['Cardiopatia'].value_counts()
```

```
SIM    336155
NÃO    219286
Name: Cardiopatia, dtype: int64
```

```
df['Cardiopatia'].value_counts(normalize=True)
```

```
SIM    0.605204
NÃO    0.394796
Name: Cardiopatia, dtype: float64
```



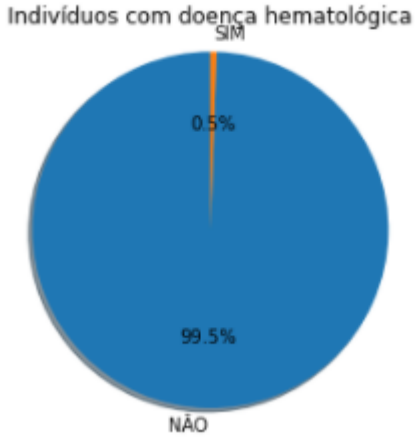
Doença Hematológica

```
df['Doenca Hematologica'].value_counts()
```

```
NÃO    552389
SIM      3052
Name: Doenca Hematologica, dtype: int64
```

```
df['Doenca Hematologica'].value_counts(normalize=True)
```

```
NÃO    0.994505
SIM    0.005495
Name: Doenca Hematologica, dtype: float64
```



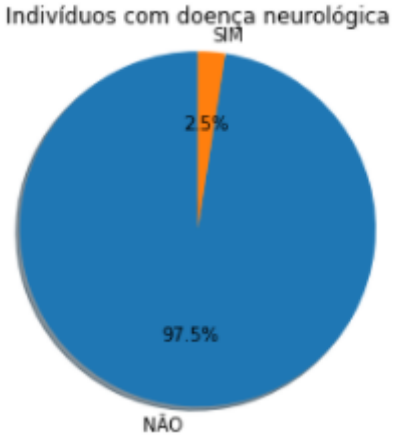
Doença Neurológica

```
df['Doenca Neurologica'].value_counts()
```

```
NÃO    541630
SIM     13811
Name: Doenca Neurologica, dtype: int64
```

```
df['Doenca Neurologica'].value_counts(normalize=True)
```

```
NÃO    0.975135
SIM    0.024865
Name: Doenca Neurologica, dtype: float64
```



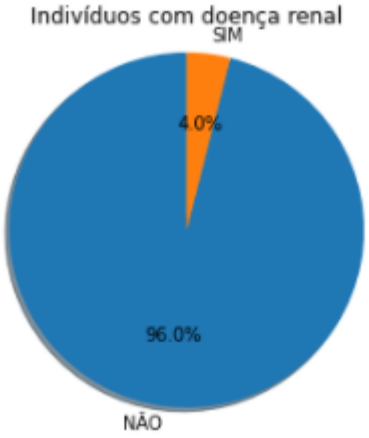
Doença Renal

```
df['Doenca Renal'].value_counts()
```

```
NÃO    533410
SIM     22031
Name: Doenca Renal, dtype: int64
```

```
df['Doenca Renal'].value_counts(normalize=True)
```

```
NÃO    0.960336
SIM    0.039664
Name: Doenca Renal, dtype: float64
```



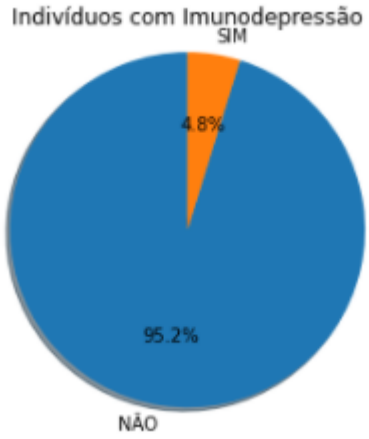
Imunodepressão

```
df['Imunodepressao'].value_counts()
```

```
NÃO    528853
SIM     26588
Name: Imunodepressao, dtype: int64
```

```
df['Imunodepressao'].value_counts(normalize=True)
```

```
NÃO    0.952132
SIM    0.047868
Name: Imunodepressao, dtype: float64
```



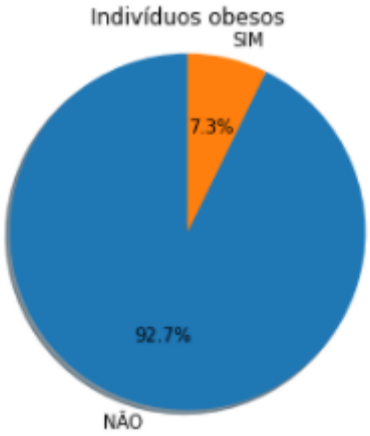
Obesidade

```
df['Obesidade'].value_counts()
```

```
NÃO    515065
SIM     40376
Name: Obesidade, dtype: int64
```

```
df['Obesidade'].value_counts(normalize=True)
```

```
NÃO    0.927308
SIM    0.072692
Name: Obesidade, dtype: float64
```



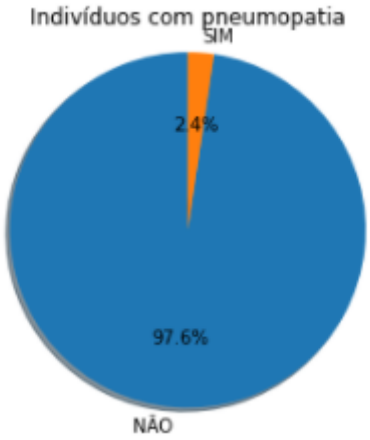
Pneumopatia

```
df['Pneumopatia'].value_counts()
```

```
NÃO    542324
SIM     13117
Name: Pneumopatia, dtype: int64
```

```
df['Pneumopatia'].value_counts(normalize=True)
```

```
NÃO    0.976385
SIM    0.023615
Name: Pneumopatia, dtype: float64
```



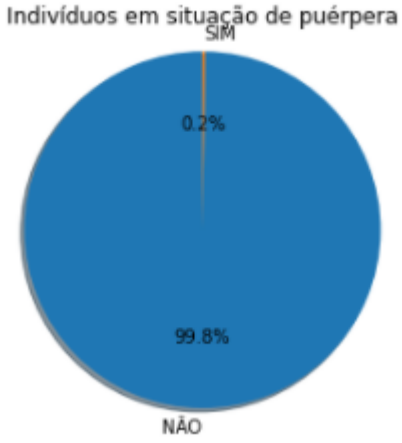
Puérpera

```
df['Puérpera'].value_counts()
```

```
NÃO    554219
SIM      1222
Name: Puérpera, dtype: int64
```

```
df['Puérpera'].value_counts(normalize=True)
```

```
NÃO    0.9978
SIM    0.0022
Name: Puérpera, dtype: float64
```



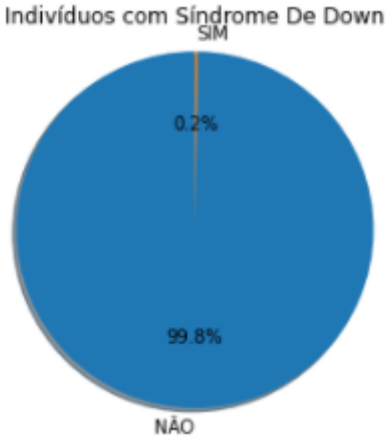
Síndrome de Down

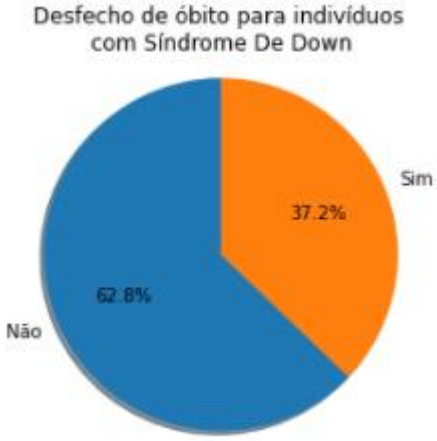
```
df['Síndrome De Down'].value_counts()
```

```
NÃO    554250
SIM      1191
Name: Síndrome De Down, dtype: int64
```

```
df['Síndrome De Down'].value_counts(normalize=True)
```

```
NÃO    0.997856
SIM    0.002144
Name: Síndrome De Down, dtype: float64
```





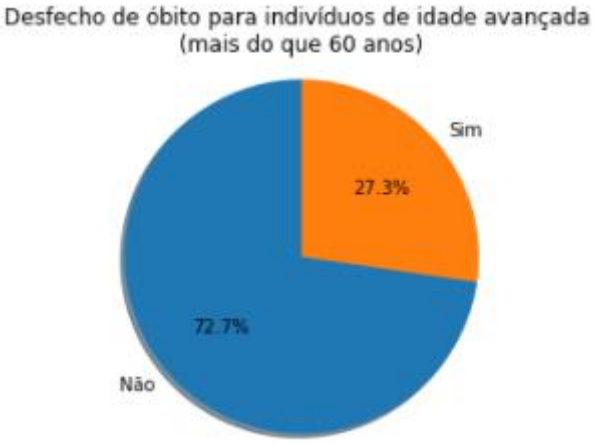
Idade Avançada

```
df['idade_avancada'].value_counts()
```

```
0    303066
1    252375
Name: idade_avancada, dtype: int64
```

```
df['idade_avancada'].value_counts(normalize=True)
```

```
0    0.545631
1    0.454369
Name: idade_avancada, dtype: float64
```



Outros Fatores de Risco

```
df['Outros Fatores De Risco'].value_counts()
```

```
NÃO    444404
SIM    111037
Name: Outros Fatores De Risco, dtype: int64
```

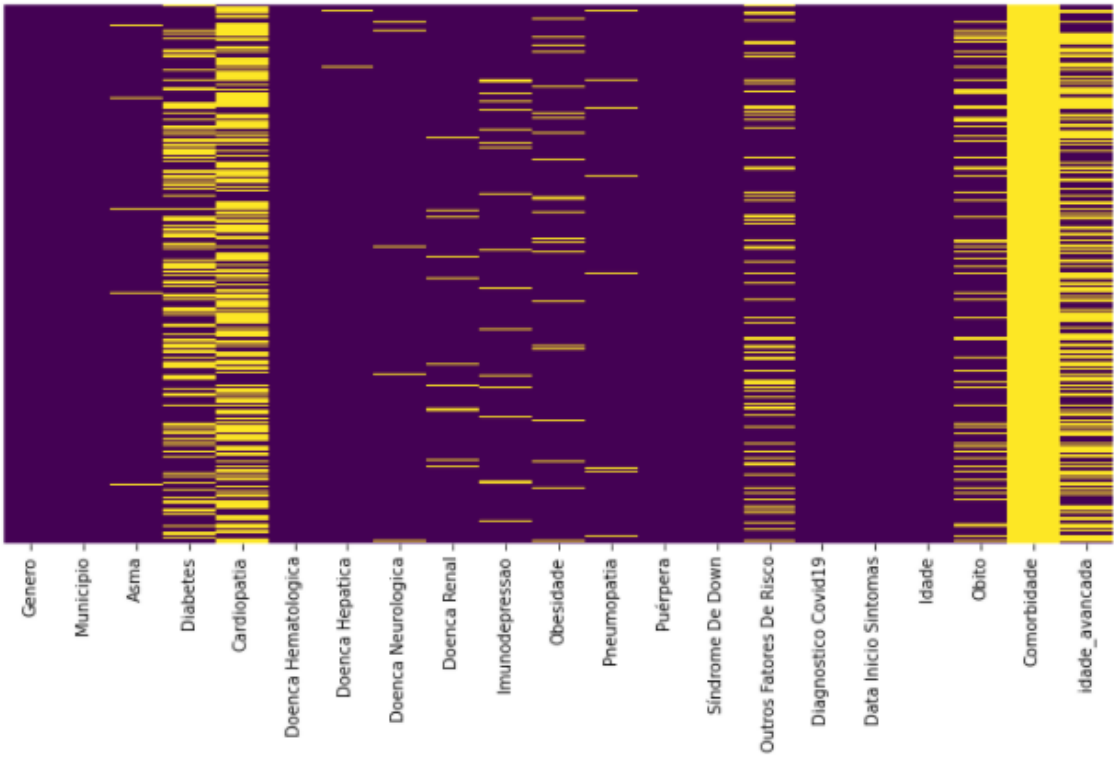
```
df['Outros Fatores De Risco'].value_counts(normalize=True)
```

```
NÃO    0.800092
SIM    0.199908
Name: Outros Fatores De Risco, dtype: float64
```

O gráfico a seguir destaca de amarelo todos as informações afirmativas no conjunto de dados.

```
plt.figure(figsize=(12, 6))
sns.heatmap((df=='SIM') | (df==1), yticklabels=False, cbar=False, cmap='viridis')
```

```
<AxesSubplot:>
```



A coluna “Comorbidade” está toda preenchida pois ela informa se a pessoa tem alguma comorbidade qualquer. Os casos em que ela informa “Não” são tão poucos que não são visíveis nesse gráfico. Estes casos são os informados como “Não” para todas as comorbidades.

Comorbidades – Análise em Conjunto

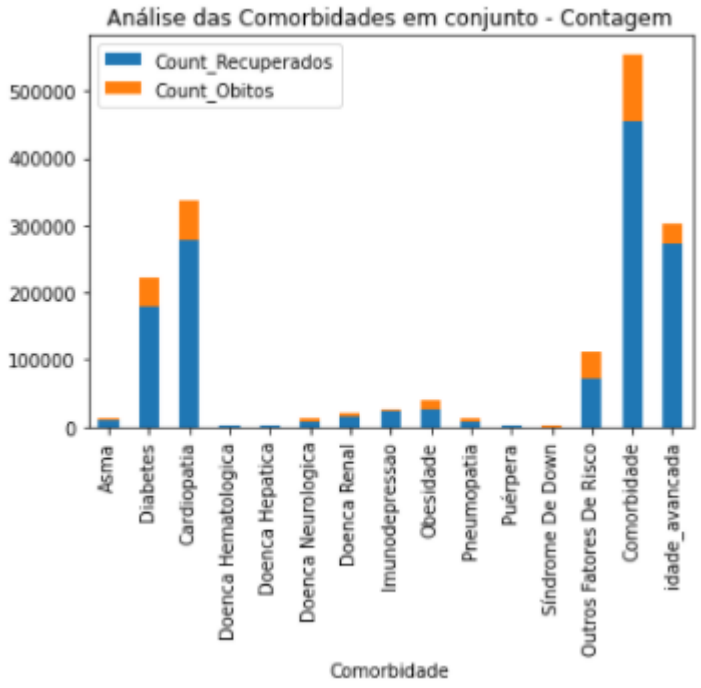
Uma análise em conjunto das comorbidades ajuda a visualizá-las em um mesmo contexto, possibilitando compará-las. Vamos à ela.

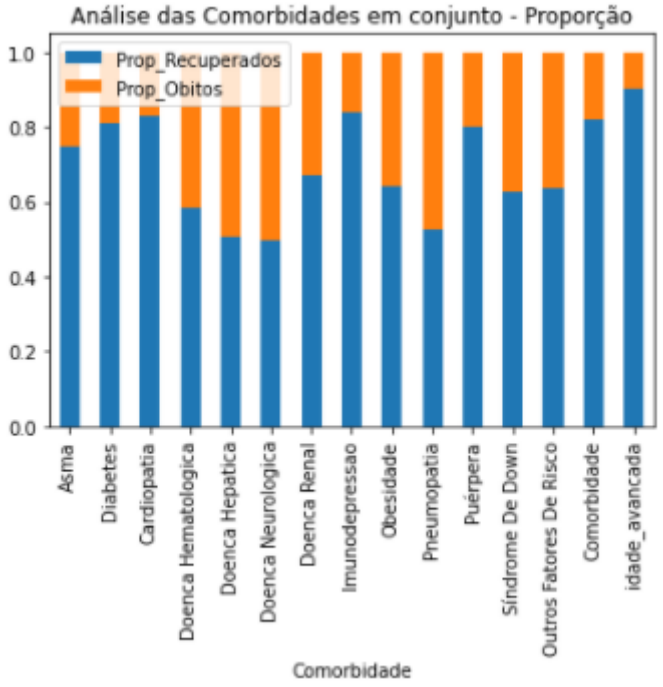
Foi gerado um dataframe que informa, para cada comorbidade, qual a sua quantidade e proporção de recuperados e óbitos.

```
df_comorbidades.head()
```

	Comorbidade	Count_Recuperados	Count_Obitos	Prop_Recuperados	Prop_Obitos
2	Asma	9267	3110	0.748727	0.251273
3	Diabetes	179266	42252	0.809262	0.190738
4	Cardiopatia	279169	56986	0.830477	0.169523
5	Doenca Hematologica	1785	1267	0.584862	0.415138
6	Doenca Hepatica	1742	1696	0.506690	0.493310

Através deste dataframe foi gerado os dois gráficos a seguir.





Comparando estes dois gráficos é possível perceber que algumas comorbidades, apesar de ter baixa incidência nos indivíduos da amostra, apresenta alta porcentagem de desfecho de óbitos para estes indivíduos. Como por exemplo a doença neurológica, que apesar de estar presente em apenas 2,5% dos indivíduos na amostra, mais de 50% destes indivíduos vieram a óbito.

Para não ser necessário, a cada parte do código que se resolver executar, ter que executar tudo novamente desde o início, no decorrer dos tratamentos alguns arquivos foram salvos. Principalmente, a cada tentativa de tratamento diferente para gerar o conjunto de dados para os modelos de Machine Learning. Os arquivos salvos, funcionaram como uma referência ou *checkpoint.*

Neste ponto foi salvo um arquivo com o dataframe de comorbidades usado nos últimos dois gráficos (para refazer estes gráficos no painel ao final do trabalho) e também um arquivo com os dados tratados como primeira versão do tratamento.

```
# exportar para colocar no gráfico
df_comorbidades.to_csv('df_comorbidades_graficos_barras.csv', sep = ';', index=False)
```

Regras de Associação

Neste ponto do trabalho foi usado regras de associação para enriquecer a análise de dados procurando por combinações de doenças pré-existentes (ou condições como gênero, obesidade ou idade avançada) nos casos de óbitos. Utilizando os dados dummies gerados na primeira versão do tratamento (dados_dummies_1) que contém 555441 registros, foi excluída a coluna Comorbidade, pois nesse caso não terá utilidade.

Para que a variável frequent_itemsets tivesse um conteúdo de quantidade ao mínimo relevante, foi necessário definir um suporte mínimo de 0.1 (10%),fazendo com que a variável tivesse um conjunto de 13 itemsets. Os itemsets apresentados nesse ponto têm poucos itens, o que faz bastante sentido, se considerar que quase 70% dos indivíduos na amostra apresentam apenas 1 comorbidade. O itemset de maior frequência foi de Cardiopatia, com um suporte de 60,5%. Logo em seguida os itemsets foram, Gênero Masculino, Idade Avançada e Diabetes. A imagem mostra os itemsets frequentes ordenados pelo suporte de forma decrescente.

support	itemsets
0.605204	(Cardiopatia)
0.479329	(Genero_Masculino)
0.454369	(idade_avancada)
0.398815	(Diabetes)
0.296575	(idade_avancada, Cardiopatia)
0.285933	(Genero_Masculino, Cardiopatia)
0.217085	(Genero_Masculino, idade_avancada)
0.210762	(Diabetes, idade_avancada)
0.199908	(Outros Fatores De Risco)
0.190317	(Diabetes, Cardiopatia)
0.188074	(Genero_Masculino, Diabetes)
0.178060	(Obito)
0.139846	(Genero_Masculino, idade_avancada, Cardiopatia)
0.123970	(Obito, idade_avancada)
0.120090	(Diabetes, idade_avancada, Cardiopatia)
0.113949	(Outros Fatores De Risco, idade_avancada)
0.103165	(Outros Fatores De Risco, Genero_Masculino)
0.102596	(Obito, Cardiopatia)

Em seguida foram geradas regras de associação definidas com a métrica confiança e um limite mínimo de 40%. Porém isto trouxe muitas regras com lift baixo (menor ou igual a 1), o que significa que a regra não é muito relevante. Então ao invés de confiança, foi usado lift como métrica para gerar as regras da associação, sendo o limite mínimo definido como 1.1. O que trouxe um número par de regras onde a recíproca era verdadeira. A única dessas regras que tinha óbito como consequente era a mostrada a seguir, com antecedente “idade avançada” e apesar de um suporte baixo, tinha confiança de quase 30% e um lift de mais do que 1.5.

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
5	(idade_avancada)	(Obito)	0.454369	0.17806	0.12397	0.27284	1.53229	0.043065	1.130342

Com a ideia de flexibilizar, para visualizar mais regras, mesmo que com um suporte mais baixo, um novo conjunto de itemsets frequents foi gerado com suporte mínimo de 0.01 (1%). Foram gerados assim 70 itemsets frequentes. As regras foram geradas em seguida, com um lift de no mínimo 1.1 e selecionadas as 36 regras que tinham óbito como consequência.

Todas as 36 regras contam com um Suporte de pelo menos 1% para os itemsets. Os valores para Confiança podem até não ser altos, mas é preciso levar em consideração que a maior parte das pessoas que contraem corona vírus conseguem se recuperar. Selecionando as que têm confiança acima da média (0.3845370589786582) restam 18 regras.

	antecedents	consequents	support	confidence
0	(Doenca Neurologica, idade_avancada)	(Obito)	0.010532	0.561906
1	(Outros Fatores De Risco, Genero_Masculino, idade_avancada, Cardiopatia)	(Obito)	0.012642	0.515982
2	(Outros Fatores De Risco, Diabetes, Genero_Masculino, idade_avancada)	(Obito)	0.010743	0.506666
3	(Doenca Neurologica)	(Obito)	0.012518	0.503439
4	(Outros Fatores De Risco, Diabetes, idade_avancada, Cardiopatia)	(Obito)	0.010835	0.497602
5	(Obesidade, idade_avancada)	(Obito)	0.011267	0.493923
6	(Outros Fatores De Risco, Genero_Masculino, idade_avancada)	(Obito)	0.028214	0.487950
7	(Outros Fatores De Risco, idade_avancada, Cardiopatia)	(Obito)	0.023232	0.481457
8	(Pneumopatia)	(Obito)	0.011193	0.473965
9	(Outros Fatores De Risco, Diabetes, idade_avancada)	(Obito)	0.020270	0.471858
10	(Outros Fatores De Risco, Diabetes, Cardiopatia)	(Obito)	0.013497	0.461439
11	(Outros Fatores De Risco, Genero_Masculino, Cardiopatia)	(Obito)	0.016191	0.452934
12	(Outros Fatores De Risco, idade_avancada)	(Obito)	0.051494	0.451905
13	(Outros Fatores De Risco, Genero_Masculino, Diabetes)	(Obito)	0.014059	0.448278
14	(Diabetes, Obesidade)	(Obito)	0.010046	0.438714
15	(Outros Fatores De Risco, Cardiopatia)	(Obito)	0.029515	0.426361
16	(Outros Fatores De Risco, Diabetes)	(Obito)	0.026379	0.421434
17	(Obesidade, Cardiopatia)	(Obito)	0.013476	0.414796
18	(Outros Fatores De Risco, Genero_Masculino)	(Obito)	0.039734	0.385152

5. Criação de Modelos de Machine Learning

Os modelos foram criados usando usando funções que recebem o número da tentativa para saber qual versão do tratamento de dados utilizar, evitando também a repetição de código. As funções exibem um “relatório” da execução do modelo e informações que foram consideradas relevantes.

A matriz de confusão gráfica foi feita com um módulo disponível pelo link https://github.com/DTrimarchi10/confusion_matrix. Uma função também foi definida para o uso desse módulo, facilitando a chamada do gráfico. Essa função é usada então por todos os modelos e foi definida pelo código a seguir:

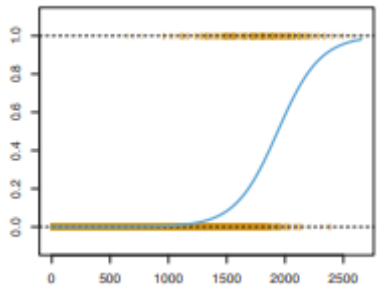
```
import cf_matrix as cfm # Módulo disponível em https://github.com/DTrimarchi10/confusion_matrix

#função utilizando o módulo que será usada para visualizar a matrix de confusão para todos os modelos
def make_confusion_matrix(cf_matrix):
    labels = ['True Neg','False Pos','False Neg','True Pos']
    categories = ['Zero', 'One']
    cfm.make_confusion_matrix(cf_matrix,
                          group_names=labels,
                          categories=categories,
                          cmap='binary')
```

Foram utilizados para este trabalho os algoritmos Regressão Logística, Árvores de Decisão, Florestas Aleatórias e Classificação Bayesiana, por se tratarem de algoritmos de classificação. Para todos os algoritmos, foram utilizadas as implementações da biblioteca Scikitlearn, do Python, sendo que todas trabalham com variáveis categóricas.

Regressão Logística

Regressão Logística é uma técnica que usa função logística para a solução de problemas. A característica dessa função é que o valor de Y será sempre entre 0 e 1, independente do valor de X, produzindo sempre uma curva com formato de “S”. Em um problema de classificação binária, o valor de Y pode ser facilmente interpretado como probabilidade da classe ser 0 ou 1, dependendo de qual dos dois valores está mais próximo. (JAMES et al, 2013)

A seguir, um exemplo de curva para a função logística.



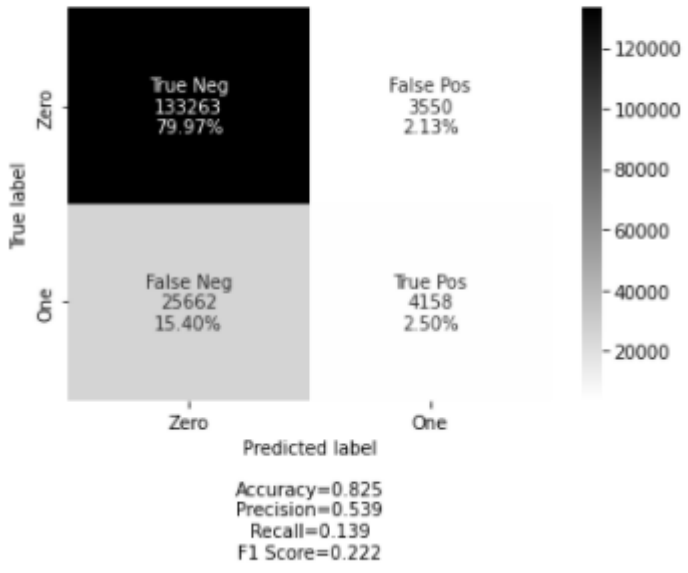
A primeira tentativa de gerar um modelo de Machine Learning com o algoritmo de Regressão Logística utiliza a primeira versão de tratamento e processamento dos dados deste estudo. A matriz de confusão do gerada foi a seguinte:

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.84	0.97	0.90	136813
1	0.54	0.14	0.22	29820
accuracy			0.82	166633
macro avg	0.69	0.56	0.56	166633
weighted avg	0.79	0.82	0.78	166633

Matriz de Confusão do Modelo:

```
[[133263   3550]
 [ 25662   4158]]
```



A acurácia está alta, mas esta medida não é muito relevante sozinha, pois seu valor está muito próximo ao valor da precisão da classe zero, que é a classe com maior número de ocorrências nos registros.

A precisão mostra que o modelo tem uma capacidade muito mais alta de prever um caso de paciente recuperado do que um caso de óbito. A precisão da classe 1 (óbito) mais especificamente, diz respeito a razão entre o número de casos classificados corretamente como óbito e o número total de casos classificados como óbito, estejam estes classificados correta ou erroneamente. Este valor se aproxima de 50% pois os dados referentes ao grupo "True Positive" e os referentes ao grupo "False Positive" representam uma parcela semelhante dos dados: pouco mais de 2%.

A enorme diferença entre o recall, ou revocação, para as classes 0 e 1 mostra novamente que o modelo tem mais capacidade de prever o caso de recuperação do que o caso de óbito, ou ainda, que o algoritmo classifica a maioria dos casos como recuperados, já que o grupo de False Negative também é bem alto em comparação aos outros.

A respeito do F1 Score, o baixo valor apresentado leva a interpretar que a acurácia provavelmente não é relevante e que os valores VP, VN, FP, FN podem apresentar distorções.

Para a tentativa 2 foram adicionadas faixas etárias aos dados ao invés da coluna binária sobre o indivíduo ter idade avançada ou não. Essa mudança no tratamento dos dados fez com que a precisão na classificação de óbitos subisse 2%, o que ainda é uma mudança muito pequena. A acurácia ainda não tem relevância, pois continua muito próxima da precisão da classe 0 (zero). Outras métricas que também subiram um pouco em relação a tentativa anterior foram a revocação e o f-score para a classe óbito, mas nada que represente uma grande melhora no modelo.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.84	0.97	0.90	136780
1	0.56	0.18	0.27	29853
accuracy			0.83	166633
macro avg	0.70	0.57	0.59	166633
weighted avg	0.79	0.83	0.79	166633

Matriz de Confusão do Modelo:

```
[[132621   4159]
 [ 24519   5334]]
```

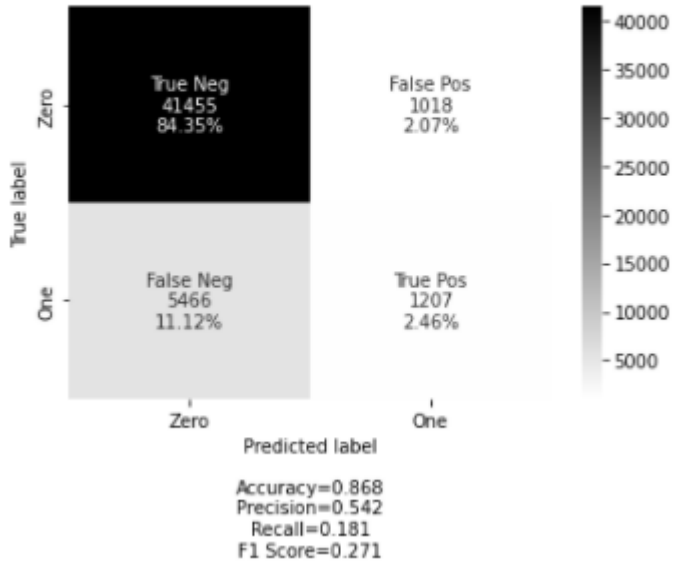
Com a seleção do período mais específico, segundo semestre do ano de 2020, a precisão da classe de recuperados subiu para 88%, mas a que se desejava melhorar era a da classe de óbitos, não houve mudança.que acabou voltando para 54%.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.88	0.98	0.93	42473
1	0.54	0.18	0.27	6673
accuracy			0.87	49146
macro avg	0.71	0.58	0.60	49146
weighted avg	0.84	0.87	0.84	49146

Matriz de Confusão do Modelo:

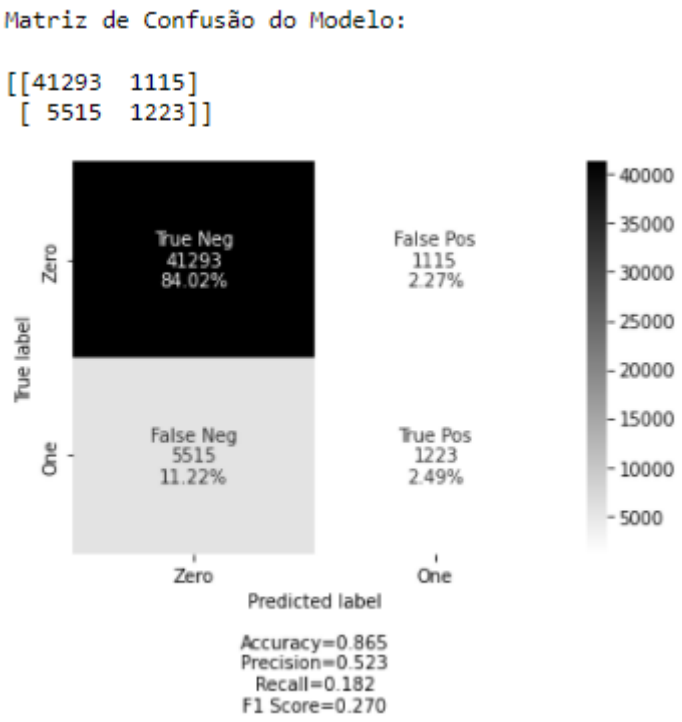
```
[[41455  1018]
 [ 5466  1207]]
```



Na tentativa 4, com uma coluna para quantidade de comorbidades a precisão da classe de óbitos caiu ainda mais, para 52%.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.88	0.97	0.93	42408
1	0.52	0.18	0.27	6738
accuracy			0.87	49146
macro avg	0.70	0.58	0.60	49146
weighted avg	0.83	0.87	0.84	49146



Os modelos até o momento não apresentaram melhoria relevante de um para outro. Acredita-se que o desbalanceamento da classe esteja atrapalhando a classificação, fazendo com que um modelo aprenda mais sobre casos de recuperados do que casos que vieram a óbito. Desta forma, os modelos tendem a prever a maioria dos casos como recuperados.

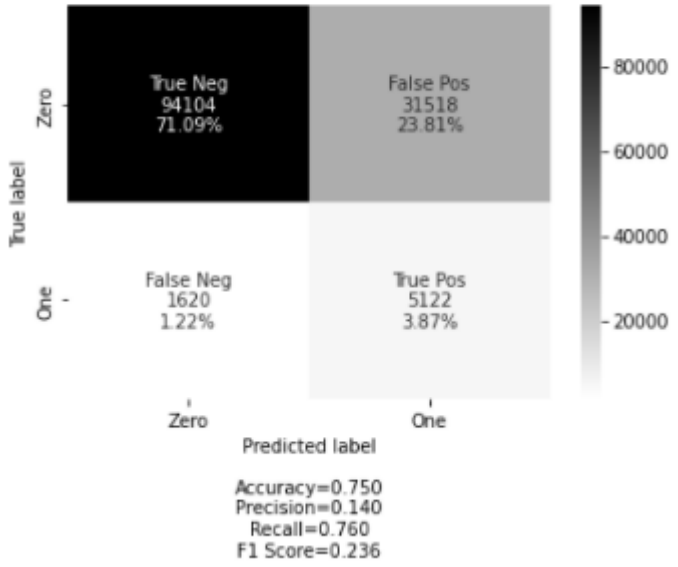
A tentativa 5 então faz uso de uma técnica de balanceamento dos dados conhecida como undersampling. A técnica consiste em remover registros da amostra de treino até que a quantidade para cada classe de óbito (0 e 1) sejam iguais. Os modelos usando undersampling vão acessar arquivos de treino e teste já definidos na fase de tratamento e processamento dos dados, sendo que os registros removidos dos dados de treino não foram descartados, mas sim colocados nos dados de teste. Os resultados para esta tentativa foram os seguintes.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.98	0.75	0.85	125622
1	0.14	0.76	0.24	6742
accuracy			0.75	132364
macro avg	0.56	0.75	0.54	132364
weighted avg	0.94	0.75	0.82	132364

Matriz de Confusão do Modelo:

```
[[94104 31518]
 [ 1620  5122]]
```



Infelizmente um comportamento inesperado ocorreu: aumentou a quantidade de Falsos Positivos. O modelo está prevendo muito mais óbitos, porém sem acertar. Se for analisada a precisão do modelo, para a classe de recuperados aumentou absurdamente para 98%, já para a classe de óbitos caiu ainda mais, para 14%. O modelo está prevendo a maior parte da amostra de teste como recuperados e não está sendo capaz de identificar os óbitos. O balanceamento com undersampling não foi útil.

Para a tentativa 6 foi usada a técnica de oversampling com SMOTE, que seleciona pares da classe minoritária e cria registros sintéticos “entre” esses pares, até que essa classe tenha a mesma quantidade de registros da classe majoritária.

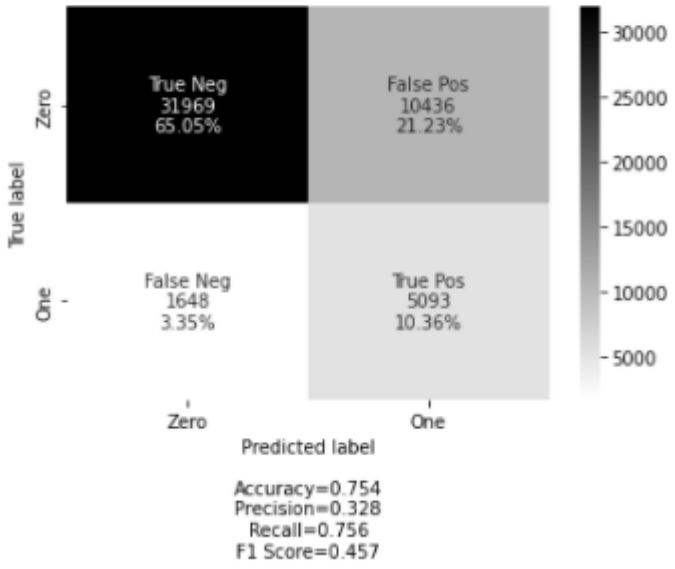
O resultado do modelo com smote foi melhor do que o anterior com undersampling, ainda assim foi pior do que todos os outros até então. O modelo ainda não tem a capacidade de prever a classe óbito.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.95	0.75	0.84	42405
1	0.33	0.76	0.46	6741
accuracy			0.75	49146
macro avg	0.64	0.75	0.65	49146
weighted avg	0.87	0.75	0.79	49146

Matriz de Confusão do Modelo:

```
[[31969 10436]
 [ 1648  5093]]
```



Um outro modelo tentou trabalhar com hiperparâmetros, porém o GridSearchCV sugeriu o simples mesmo, pesos iguais para as classes e no máximo 100 iterações. O resultado foi semelhante aos iniciais. Não trouxe melhorias.

Árvores de Decisão

Árvores de Decisão é outro algoritmo de Machine Learning que também é usado na solução de problemas de aprendizado supervisionado para classificação, onde o aprendizado se dá basicamente através de uma hierarquia de condições que levam a uma decisão.

Em uma árvore de decisão, cada nó representa um questionamento a ser feito sobre um registro, cada resposta direciona o fluxo a outra pergunta. Os nós ao final da árvore são chamados de nós-folha e contêm a resposta para a classificação. Árvores de Decisão possuem risco de overfitting. (MULLER; GUIDO, 2017)

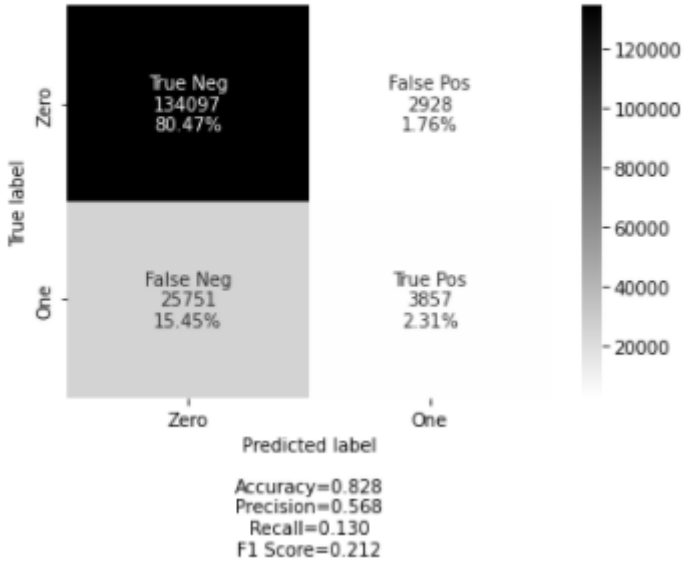
A primeira tentativa foi com o arquivo de tratamentos iniciais, que trouxe uma precisão de 57% para a classe de óbitos. Apesar de estar longe do desejável, foi o melhor resultado para a precisão desta classe até o momento. O modelo ainda traz índices de revocação distantes e f-score baixo.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.84	0.98	0.90	137025
1	0.57	0.13	0.21	29608
accuracy			0.83	166633
macro avg	0.70	0.55	0.56	166633
weighted avg	0.79	0.83	0.78	166633

Matriz de Confusão do Modelo:

```
[[134097   2928]
 [ 25751   3857]]
```



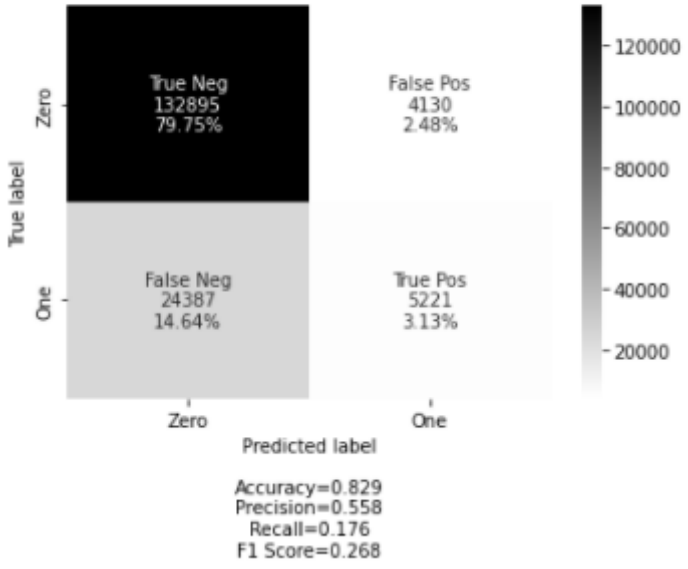
A segunda tentativa, com as faixas etárias, não traz melhoria. Pelo contrário, caiu em 1% a precisão da classe de óbitos.

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.84	0.97	0.90	137025
1	0.56	0.18	0.27	29608
accuracy			0.83	166633
macro avg	0.70	0.57	0.59	166633
weighted avg	0.79	0.83	0.79	166633

Matriz de Confusão do Modelo:

```
[[132895   4130]
 [ 24387   5221]]
```



Tentativa 3 e 4, com a seleção do semestre específico e quantidade de comorbidades respectivamente, também não trouxeram melhorias. As tentativas 5 e 6, undersampling e oversampling SMOTE respectivamente, se mostraram um pouco melhores do que as mesmas tentativas com a Regressão Logística, porém ainda piores do que a maioria dos resultados até então. Também foi feita uma tentativa com hiperparâmetros, onde de {'criterion':('gini', 'entropy'), 'min_samples_leaf':[1, 2, 3], 'max_depth': np.arange(3, 15)} o GridSearchCV escolheu {'criterion': 'gini',

'max_depth': 11, 'min_samples_leaf': 2}, mas também não trouxe nada de significativo e novo ao estudo.

Florestas Aleatórias

A técnica de Árvores Aleatórias utiliza o algoritmo de Árvores de Decisão gerando várias árvores e considerando as respostas mais encontradas pela maioria das árvores. Essas árvores são mais “simples” do que usar apenas uma árvore, pois utilizam subconjuntos dos preditores e são geralmente usadas sem hiperparâmetros. Ainda assim costumam encontrar resultados melhores do que usar apenas uma árvore de decisão, também costuma resolver o problema de overfitting da árvore única. Florestas Aleatórias costumam apresentar boa performance mesmo com muitos preditores. (JAMES et al, 2013)

O primeiro modelo de Florestas Aleatórias teve a precisão das classes recuperados e óbitos de 84% e 57%, respectivamente. No segundo modelo, com faixas etárias, mudou para 85% e 56%. Sem novidades, em ambas as tentativas 3 e 4 estas precisões foram para 88% e 54%. Nas tentativas relacionadas ao undersampling e oversampling os resultados também foram semelhantes aos dos modelos anteriores. Não se trabalha com hiperparâmetros em florestas aleatórias, a ideia do algoritmo é combinar os resultados de árvores simples de decisão. Foi definido apenas a quantidade de árvores no modelo de Florestas Aleatórias como 200, para todos os modelos gerados com o algoritmo.

Classificação Bayesiana

A Classificação Bayesiana, também conhecida como algoritmo Naive Bayes, é uma técnica para criação de modelos de classificação para aprendizado supervisionado. O algoritmo aprende sobre os parâmetros olhando para cada característica individualmente, obtendo as estatísticas para cada classe a partir das características. Ou seja, considera a probabilidade de cada classe, dado os valores dos parâmetros de cada registro. Desta forma o algoritmo tem um funcionamento simples e um treinamento rápido, porém nem sempre é o que tem melhores resultados, pois gera um modelo de performance generalizada. (MULLER; GUIDO, 2017)

Para este algoritmo a precisão para a classe óbito começou em 53% e foi caindo a cada tentativa, com exceção do SMOTE que, assim como nos outros algoritmos, foi melhor do que o undersampling, entretanto pior do que as demais tentativas. A Classificação Bayesiana também usa conceitos simples, aplicada sem hiperparâmetros.

6. Interpretação dos Resultados

Neste estudo foram analisadas informações disponibilizadas no portal de dados abertos do Estado de São Paulo sobre casos confirmados de Covid-19. O conjunto de dados fornecido tinha pouquíssimos dados faltantes, porém grande parte dos dados foi informada como “ignorado” no momento do registro, o que pode ser ou não alguma falta de compromisso dos responsáveis pela coleta. Muitas idades também foram informadas de forma errada. As idades acabaram sendo excluídas já que as maiores do que 89 apareceram no boxplot como outliers.

Com a análise exploratória dos dados já foi possível perceber que os casos que chegam a óbito são a minoria entre os que testam positivo para Covid-19 e que apesar de ter mais mulheres do que homens entre os casos confirmados, os homens ainda têm uma ocorrência maior de óbitos.

Sobre doenças pré-existentes, o presente estudo mostrou uma frequência maior de duas doenças na amostra de casos confirmados de Covid-19. São elas, diabetes, acometendo 39,9% dos indivíduos na amostra e cardiopatia, acometendo 39,5%. As taxas de óbito para os indivíduos com estas doenças foram, respectivamente, 19,1% e 17%. As doenças mais frequentes corroboram em parte com Piola et al (2021), que encontrou doenças cardiovasculares como das mais frequentes entre casos confirmados de Covid-19.

A respeito da “idade avançada”, considerada um fator de risco para indivíduos com Covid-19, alguns estudos estabelecem uma idade para dizer que indivíduos com idade igual ou superior a esta correm risco maior de óbito do que os demais. O estudo de Piola et al (2021) cita 60 anos como a “idade de risco”, já o trabalho de Oliveira et al (2021) aponta risco maior para indivíduos com mais de 80 anos, porém, durante a análise exploratória do presente estudo, observou-se que há uma correlação positiva entre a probabilidade de óbito e a idade dos indivíduos. Ou seja, dizer que há uma idade para ser considerada de risco pode gerar uma falsa interpretação dos riscos da doença, já que o risco de óbito para indivíduos mais novos de uma suposta idade estabelecida são muito semelhantes aos riscos da mesma. Claro que se for necessário utilizar uma variável binária, como feita em alguns momentos neste trabalho, é preciso definir o valor da idade de risco. Entretanto é de grande impor-

tância que seja divulgada a correlação alta e positiva para a probabilidade de óbito e a idade dos indivíduos, mostrando que na verdade a tendência é que a cada idade de um adulto a probabilidade de óbito se torna um pouco maior.

Sobre as informações, nem doença pré-existente, nem a idade avançada por si só consegue definir se uma pessoa virá a óbito ou não, nem mesmo apresentar uma probabilidade tão alta que pudesse ser considerada como indicador definitivo. Note que na amostra do trabalho apenas uma comorbidade (doença neurológica) apresentou taxa de óbito de mais de 50%. Acredita-se então que o resultado de óbito seja por uma combinação de fatores.

A análise em conjunto das comorbidades mostrou que algumas doenças pré-existentes, apesar de raras na população, podem ser as que trazem um risco maior de óbito para o indivíduo. Porém ao utilizar Regras de Associação para fatores com no mínimo 1% de suporte, percebe-se que essas comorbidades menos comuns não aparecem muito nas regras mais relevantes com óbito como consequente. A relevância das regras foi definida apartir dos itemsets de suporte 1% ou superior, lift de no mínimo 1,1% e óbito como consequente o que definiu 36 regras. Filtrar as regras com confiança acima da média trouxe exatamente metade (18 regras), o que mostra que não são tão mais relevantes assim do que as descartadas. O que mais se vê nas regras são as comorbidades ou condições mais comuns, exceto pela doença neurológica, que apareceu devido sua grande taxa de óbito para os indivíduos que apresentam esse fator de risco. As condições ou doenças pré-existentes que aparecem nas regras são: doença neurológica, idade avançada, gênero masculino, cardiopatia, diabetes, obesidade, pneumopatia e “outros fatores de risco”.

Das 18 regras citadas, a que apresenta a confiança mais alta é a combinação doença neurológica e idade avançada. A regra tem um suporte de apenas 1% nos dados da amostra, mas uma confiança que aponta a probabilidade de 56% de chance de óbito para o indivíduo com esta combinação de fatores. Em seguida, a segunda regra conta com os fatores gênero masculino, idade avançada, cardiopatia e “outros fatores de risco”. A regra em terceiro lugar quanto à confiança é muito semelhante à segunda, porém ao invés de cardiopatia ela conta com diabetes no itemset. A quarta regra, em ordem de confiança, conta apenas com a doença neurológica, com a confiança já conhecida de mais de 50% de desfechos de óbito.

A utilização de regras de associação na análise exploratória se mostrou útil e de grande valor para este tipo de problema. Talvez trouxe mais entendimento sobre o tema do que os próprios modelos de Machine Learning que serão vistos mais a frente. Observe por exemplo a sexta regra com maior confiança, uma combinação de obesidade e idade avançada. A regra apresenta 49,39% de desfecho de óbito para os indivíduos neste contexto, sendo que nenhum dos 2 fatores combinados teriam essa taxa de óbito sozinhos. Note também que nem são doenças. Vale ressaltar que mesmo que a obesidade for a única comorbidade, ela é capaz de piorar o quadro da doença Covid-19, como mostrado por Cândido et al (2021).

Já analisando os modelos de Machine Learning, os resultados mostraram de forma geral alta precisão na predição da classe de recuperados, porém a precisão para a classe de óbitos ficava bem aquém do desejado. Os modelos não estavam conseguindo aprender tanto sobre os casos de óbito quanto aprendiam sobre os recuperados provavelmente pela grande diferença de quantidade das duas classes nos dados. Acabavam predizendo a maior parte da amostra de teste como recuperados. O problema disso é que em geral quando os dados são desbalanceados o que se busca é exatamente conseguir identificar e predizer os registros da classe minoritária, principalmente no que diz respeito a estudos na área da saúde.

Para lidar com os problemas de dados desbalanceados foram utilizadas técnicas de resampling, sendo uma de undersampling e uma de oversampling com o algoritmo SMOTE. Entretanto, não funcionou como o esperado e a classe minoritária óbito teve uma queda ainda maior no valor de precisão. O oversampling com o SMOTE trouxe uma melhora em relação ao undersampling, mas ainda assim foi pior do que todos os demais modelos que não utilizaram resampling. A precisão foi extremamente abaixo do desejado.

Entre os modelos sem resampling,o modelo de maior valor de precisão para a classe de recuperados foi obtido do algoritmo de Classificação Bayesiana, utilizando a versão do tratamento dos dados com faixas etárias, período específico e coluna de quantidade de comorbidades (tentativa número 4 da Classificação Bayesiana). O valor da precisão de recuperados foi de 90%, enquanto para a classe de óbitos foi de 46%. Já o maior valor de precisão para óbitos, 57%, foi gerado com 2 modelos. O primeiro foi um modelo com Árvores de Decisão com a primeira versão de tratamen-

tos. Este modelo obteve 84% de precisão para a classe de recuperados. O segundo modelo foi Florestas Aleatórias com o mesmo resultado para ambas as precisões e a mesma versão de tratamento de dados.

De forma geral, os modelos não trazem novidades relevantes e acabam não se distanciando muito do primeiro, cujos comentários foram sobre a acurácia não confiável, diferença entre os valores de revocação, dificuldade de predição da classe minoritária e o baixo valor f-score, que indica possíveis distorções nos valores da matriz de confusão.

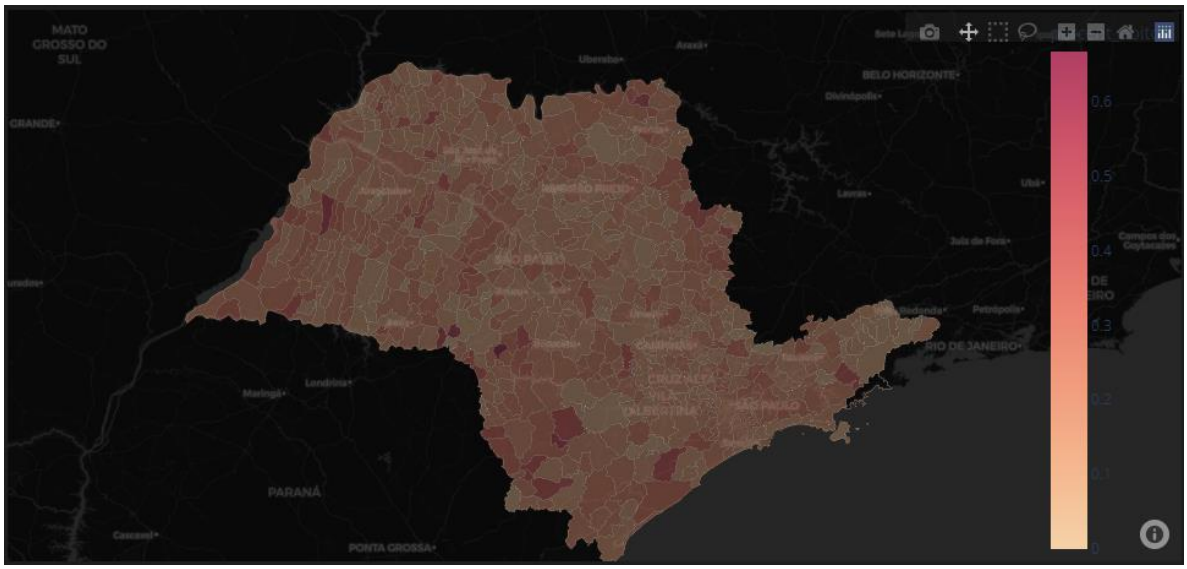
Se este trabalho fosse apenas sobre predição de óbitos valeria a pena dizer que com mais informações sobre os indivíduos poderiam ser obtidos melhores resultados. Por exemplo, se os dados informam a data de início dos sintomas, poderiam ser informados também quais os sintomas a pessoa estava sentindo. Outros fatores não informados na base de dados também podem ter influência para o desfecho de óbito, como por exemplo se o indivíduo é fumante, se é sedentário ou qual a variante do vírus ele tinha, se possível. Outro fator que pode influenciar no desfecho de óbito é o IDH (Índice de Desenvolvimento Humano) do município, como visto por Almeida et al (2021), ainda que os fatores de risco não variem muito entre regiões, como visto por Oliveira et al (2021) estudando os casos no Rio de Janeiro. Quando se trata de modelos de Machine Learning, o “sucesso depende da disponibilidade de dados para a etapa de aprendizado de modelos preditivos” (SANTOS et al, 2019, p. 14).

Contudo, este trabalho teve o objetivo de mostrar as comorbidades no desfecho de óbito dos indivíduos com Covid-19, com a intensão de gerar mais informação sobre o risco de óbito no caso de pessoas expostas a fatores de risco. Acredita-se que o objetivo foi concluído, mesmo que mais pela análise exploratória do que pelos modelos de Machine Learning.

7. Apresentação dos Resultados

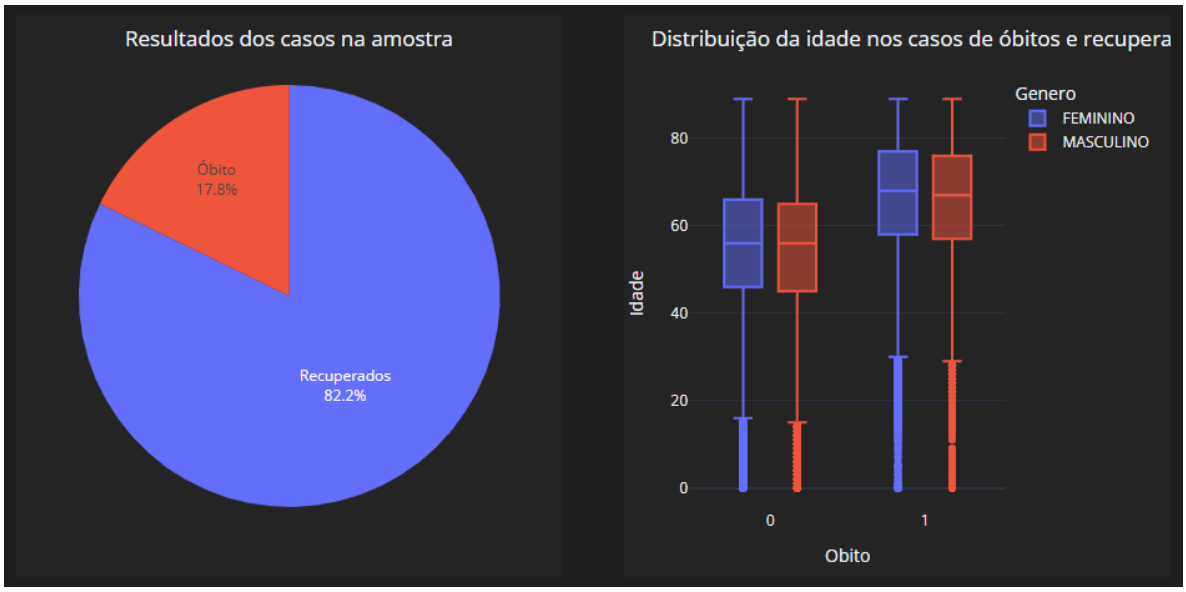
Os resultados mais relevantes deste estudo foram obtidos da análise exploratória dos dados. Os gráficos de possibilitaram esses resultados foram repetidos no dashboard, porém no dashboard eles representam a amostra que foi utilizada para geração dos modelos de ML, não uma análise geral da população, como foi durante a análise exploratória. Como os dados foram tratados tentando obter bons resultados dos modelos, o objetivo destes gráficos não foi descrever a situação geral do estado de São Paulo, mas especificamente da amostra utilizada para o estudo.

O mapa do estado mostra a proporção de óbitos nos municípios, através da intensidade da cor.

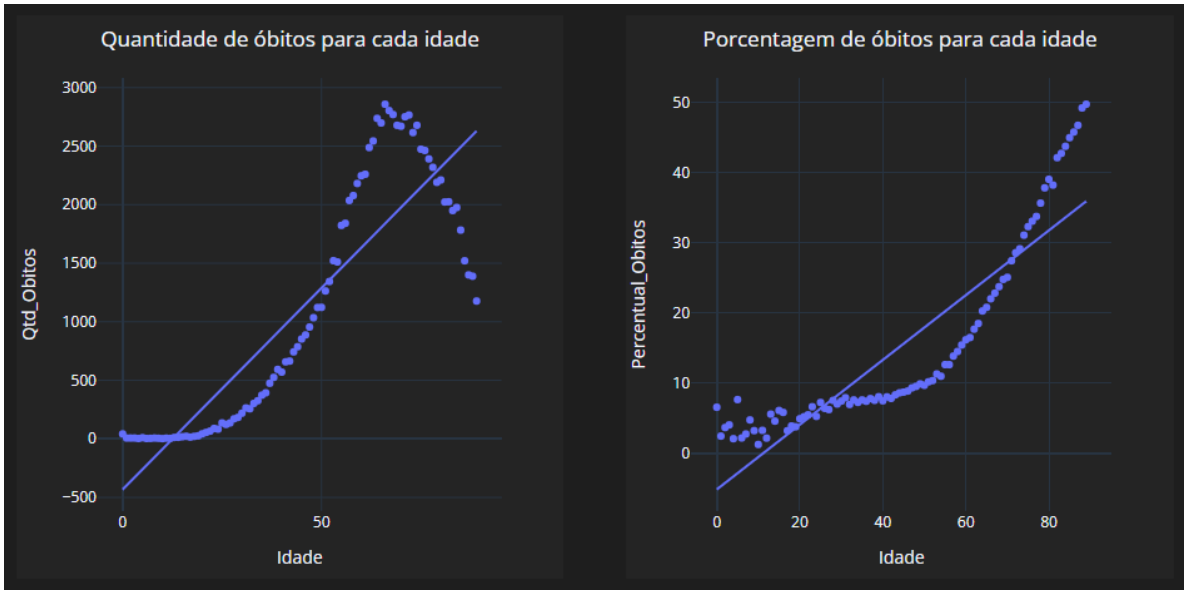


O gráfico de pizza mostra uma porcentagem de óbito mais alta do que a inicial, antes de algumas exclusões. Isto pois a amostra foi reduzida removendo registros de casos recuperados, grande parte deles que tiveram ignorado em todas as informações de fatores de risco. Acreditou-se que esses registros não tinham muita informação a acrescentar para os modelos preditivos e a redução da amostra também facilitou a execução dos modelos.

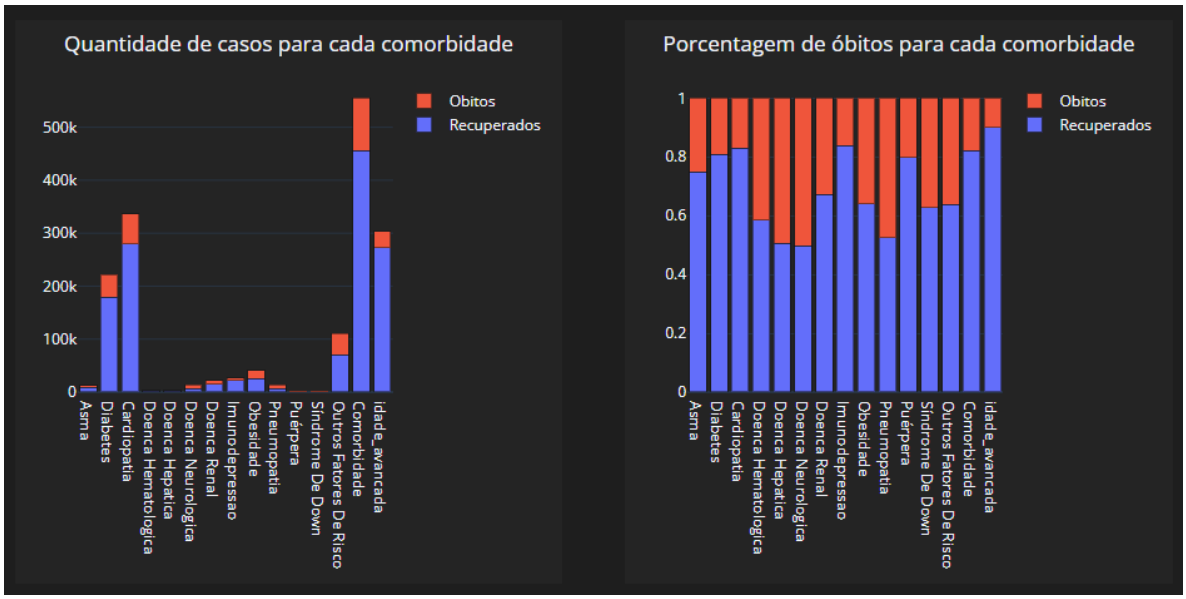
O gráfico de boxplot para idade separando por gênero os casos de óbito e recuperados também foi comentado na parte de interpretação da análise exploratória. O comentado foi que as mulheres na amostra são um pouco mais velhas do que os homens, tanto as que se recuperam como as que chegam a óbito. Entretanto a proporção de óbito para homens é maior do que a proporção para as mulheres, chegando a considerar o gênero masculino um fator influente ao agravamento dos casos de Covid-19.



Um dos resultados mais relevantes mostrados por este estudo é a correlação positiva entre idade e a porcentagem de óbitos. A análise aponta a tendência de que para adultos em geral a probabilidade de desfecho de óbito se torna cada vez maior a cada idade tomada como exemplo.



Outro ponto de grande importância visto com este estudo são as comorbidades que as vezes não são muito comuns, mas acabam apresentando grande risco aos indivíduos que aprensentam a condição. Esse resultado pode ser visto com a análise em conjunto das comorbidades.



Todos os gráficos do painel foram gerados com a versão do primeiro tratamento de dados, já que as demais versões eram tentativas de melhorar os modelos de ML.

Também estão disponíveis um arquivo html único com o resultado do dashboard em python e o próprio dashboard.py, que poderá ser executado através do Prompt de Comando. Ao chamar o arquivo usando o comando “python” pelo cmd, o ip fornecido pode ser usado para acessar o painel pelo navegador.

Explicando este projeto de acordo com a organização proposta por Vasandani (2019), temos os passos a seguir.

Etapa 1: Identificação do problema

Com este projeto procura-se produzir conteúdo sobre a pandemia de Covid-19, com o objetivo de mitigar o problema de *fake news*, combatendo-o através de gerar informações verdadeiras a respeito do tema.

Etapa 2: Resultados/previsões pretendidos

Pretende-se alcançar informações sobre o impacto de doenças pré-existentes e outros fatores de risco nos casos de Covid-19, principalmente sobre o que leva ao óbito.

Etapa 3: Fontes de dados

Todos os dados foram obtidos da internet, sendo que os principais foram fornecidos pelo portal de dados abertos do Estado de São Paulo. O volume de dados inicialmente foi consideravelmente grande, contando com dados registrados até o dia do download.

Etapa 4: Escolha dos modelo

As características dos dados obtidos são em maioria categóricos e binários. Se tratando de um problema de classificação com dados supervisionados, optou-se por utilizar os algoritmos de Regressão Logística, Árvores de Decisão, Florestas Aleatórias e Classificação Bayesiana. Regras de Associação também foram utilizadas durante a análise exploratória, o que se mostrou muito útil.

Etapa 5: Métricas de avaliação do modelo

As métricas utilizadas para avaliação e interpretação dos modelos foram Acurácia, Precisão, Revocação e F1 Score.

Etapa 6: Preparação de dados

O projeto contou com uma etapa de tratamento e processamento dos dados contendo remoção de registros com dados faltantes, remoção de outliers, alteração de informações como “ignorado” para “não” mantendo as informações binárias, criação de colunas e uma extensa análise exploratória que levava a mais tratamentos.

8. Links

Link para o vídeo: https://youtu.be/i4qbO5WTd1Q

Link para o repositório: https://github.com/thalitasylvia/TCC

REFERÊNCIAS

ALMEIDA, Gabriel Berg; FORLATEZA, Carlos M.C.B.; GUIMARÃE, Raul Borges; FERREIRA, Claudia Pio; PRONUNCIATE, Micheli. Fatores de vulnerabilidade à Covid-19 em cidades do interior do estado de São Paulo. **The Brazilian Journal of Infectious Diseases**. v. 25, 2021.

APUKE, Oberiri Destiny; OMAR, Bahiyah. Fake news and COVID-19: modelling the predictors of fake news sharing among social media users. **Telematics and Informatics**. v.56, 2021.

BAKER, Israa; MARZOUQA, Nizar ; YAGHI, Bashar Nafe'; ADAWI, Samer Osama; YOUSEF, Shahd; SABOOH, Tayseer Nedal; SALHAB. Nataly Mazen; KHRISHI, Hiba Mahmoud; QABAJA, Yahya; RIAD, Abanoub; KATEEB, Elham; ATTIA, Sameh. The Impact of Information Sources on COVID-19-Related Knowledge, Attitudes, and Practices (KAP) among University Students: A Nationwide Cross-Sectional Study. **International Journal of Environmental Research and Public Health**. v. 18, 2021.

CÂNDIDO, José Auricélio Bernardo, MOREIRA, Maria Rosilene Cândido, ALEXANDRE, Severino Ferreira, PARENTE, Natália Campos, CAVALCANTE, Naara Regia Pinheiro. Obesidade Em Paciente Com Prognóstico De Gravidade Para Covid-19. **Research, Society and Development**, Web. v. 10, n. 10, p. e459101019121, 2021.

JAMES, Gareth; WITTEN, Daniela; HASTIE, Trevor; TIBSHIRANI, Robert. **An Introduction to Statistical Learning** with Applications in R. Springer Science+Business Media New York, 2013.

MULLER, Andreas C.; GUIDO, Sarah. **Introduction to Machine Learning with Python**: A Guide for Data Scientists. Sebastopol. O'Reilly Media, Inc., 2017.

OLIVEIRA, Marcella Cini; ELEUTERIO, Tatiana de Araujo; CORRÊA, Allan Bruno de Andrade; SILVA, Lucas Dalsenter Romano; RODRIGUES, Renata Coelho; OLIVEIRA, Bruna Andrade; MARTINS, Marlos Melo; RAYMUNDO, Carlos Eduardo;

MEDRONHO, Roberto de Andrade. Factors associated with death in confirmed cases of COVID-19 in the state of Rio de Janeiro. **BMC Infect Dis**. Published online 2021.

Ministério da Saúde. **Atendimento e Fatores de Risco**. Disponível em: https://www.gov.br/saude/pt-br/coronavirus/atendimento-tratamento-e-fatores-de-risco. Acesso em 2 de jan. 2022

PIOLA, Bianca Lima; TREVISOL, Maico; PASCOTTO, Claudicéia Risso; LUCIO, Leia Carolina; BRIZOLA, Fernando Mazetto; WENDT, Guilherme Welter; FERRETO, Lirane Elize Defante. Frequência E Fatores Associados Em Casos Confirmados E Descartados E óbitos Da COVID-19 Em Um Hospital Secundário. **Revista De Saúde Pública Do Paraná**. Internet. v. 4, n. 4, p. 61-76, 2021.

SANTOS, Hellen Geremias; NASCIMENTO, Carla Ferreira; IZBICKI, Rafael; DUARTE, Yeda Aparecida de Oliveira; FILHO, Alexandre Dias Porto Chiavegatto. Machine Learning para análises preditivas em saúde: exemplo de aplicação para predizer óbito em idosos de São Paulo, Brasil. **Cad. Saúde Pública (Online)**, 2019.

TAN, Minying; CHUA, Alton Y.K. Students' Motivations for Not Sharing Rumours during the COVID-19 Pandemic in Singapore. **Proceedings of the Association for Information Science and Technology**. v.51, 2021.

VASANDANI, Jasmine. **A Data Science Workflow Canvas to Kickstart Your Projects**. Disponível em: https://towardsdatascience.com/a-data-science-workflow-canvas-to-kickstart-your-projects-db62556be4d0. Acesso em 3 de jan. 2022

VIOLA, Carmine; TOMA, Pierluigi; MANTA, Francesco; BENVENUTO, Marco. The more you know, the better you act? Institutional communication in Covid-19 crisis management. **Technological Forecasting and Social Change**. v. 170, 2021.

WHO. **Coronavirus disease (COVID-19)**. Disponível em https://www.who.int/health-topics/coronavirus#tab=tab_1. Acesso em 5 de jan. 2022

WHO. **Tracking SARS-CoV-2 variants**. Disponível em https://www.who.int/en/activities/tracking-SARS-CoV-2-variants/. Acesso em: 5 de jan. 2022

YING, Weijun; CHENG, Cecilia. Public Emotional and Coping Responses to the COVID-19 Infodemic: A Review and Recommendations. **Front Psychiatry**. v. 12, 2021.

APÊNDICE

Scripts Python

Os scripts python referentes ao notebook desenvolvido para este trabalho estão disponíveis no GitHub.

Obs.: Se os relatórios de classificação no notebook a seguir forem comparados aos apresentados neste relatório ou nos slides, podem divergir 1 ou 2% em alguns valores, pois por um problema técnico foi necessário executar os códigos novamente. Ressalto que os códigos foram os mesmos.