UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CAMPUS CURITIBA

ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS E SUAS APLICAÇÕES

GUILHERME GRACIANO TREMEL

ANÁLISE EXPLORATÓRIA DOS ATENDIMENTOS DO HOSPITAL PEQUENO PRÍNCIPE

CURITIBA

2021

GUILHERME GRACIANO TREMEL

ANÁLISE EXPLORATÓRIA DOS ATENDIMENTOS DO HOSPITAL PEQUENO PRÍNCIPE

Exploratory Analysis from Pequeno Príncipe Hospital Care

Trabalho apresentado como requisito parcial à obtenção do título de Especialista em Ciência de Dados e Suas Aplicações, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Leandro Batista de Almeida.

CURITIBA

2021

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
UTFPR - CAMPUS CURITIBA
DIRETORIA-GERAL - CAMPUS CURITIBA
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO - CAMPUS CURITIBA
DEPARTAMENTO DE APOIO DAS ESPECIALIZAÇÕES LATO-SENSU DOS CURSOS DE INFORMÁTICA - CAMPUS CURITIBA
CURSO DE ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS E SUAS APLICAÇÕES

TERMO DE APROVAÇÃO

ANÁLISE EXPLORATÓRIA DOS ATENDIMENTOS DO HOSPITAL PEQUENO PRÍNCIPE

por

Guilherme Graciano Tremel

Este Trabalho de Conclusão de Curso foi apresentado às 20h00 do dia 02 de agosto de 2021 por videoconferência como requisito parcial à obtenção do grau de Especialista em Ciência de Dados e suas Aplicações na Universidade Tecnológica Federal do Paraná - UTFPR - Campus Curitiba. O aluno foi arguido pela Banca de Avaliação abaixo assinados. Após deliberação, a Banca de Avaliação considerou o trabalho aprovado.

_____

Prof. Dr. Leandro Batista de Almeida (Presidente/Orientador – DAINF-CT/ UTFPR-CT)

_____

Prof. Msc. Christian Carlos de Souza Mendes (Avaliador1 – DAELN-CT/ UTFPR-CT)

_____

Profa. Dra. Rita Cristina Galarraga Berardi (Avaliadora 2 – DAINF-CT/ UTFPR-CT)

O Termo de Aprovação assinado encontra-se no sistema SEI- Processo nº 23064.031953/2021-11

Dedico este trabalho à minha esposa Kauane e
ao meu filho Bernardo.

AGRADECIMENTOS

Gostaria de agradecer primeiramente a minha família que sempre foi grande incentivadora na conquista dos meus sonhos. Meus pais, que me ensinaram a importância de se construir conhecimento para seguir em frente.

A minha esposa que esteve presente em todos os momentos, e foi grande auxiliadora para que eu conseguisse chegar até o fim.

Meu filho que fez com que eu valorizasse ainda mais essa fase da minha vida, para que um dia ele tenha orgulho de quem sou.

Agradeço também a todos os colegas que me ajudaram no decorrer da caminhada, aos professores que compartilharam de suas experiências.

E a Deus, que conduziu minha vida até aqui.

Conhecimento não é aquilo que você sabe, mas o que você faz com aquilo que você sabe.

(Aldous Huxley)

RESUMO

Monografia realizada com o intuito de desenvolver um estudo de análise exploratória de dados, com ênfase na visualização de dados, com base em informações relativas aos atendimentos do Hospital Pequeno Príncipe, localizado em Curitiba/PR. A análise é baseada em um *dataset* gerado por trabalhadores internos do hospital e disponibilizado para uso educacional, que contém cerca de 190 mil registros, do ano de 2012 até 2021. Esta monografia abordará o uso de análise exploratória utilizando aprofundamento na programação Python, para a tentativa de obtenção de resultados satisfatórios, avaliando se os mesmos são relevantes, na possibilidade de que sejam usados para estudo posterior. Este estudo aborda a perspectiva e possibilidade de que em qualquer área o estudo dos dados pode fornecer resultados promissores.

Palavras-chave: Atendimento Hospitalar. Análise Exploratória. Visualização de Dados. Avaliação de resultados.

ABSTRACT

Monograph made with the objective of developing a data exploratory analysis study, with emphasis on data visualization, using informations related to the Hospital Pequeno Principe care, located in Curitiba/PR. This analysis is based on a dataset generated by internal workers of the hospital and made available for educational usage, containing about 190 thousand records, from year 2012 to 2021. This monograph will approach the use of exploratory analysis deepening the utilization of Python programming, trying to obtain satisfactory results, evaluating whether they are relevant, in the possibility that they will be used for further study. This study addresses the perspective and possibility that in any area the study of data can provide promising results.

Keywords: Hospital care. Exploratory Analysis. Data Visualization. Results Evaluation.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1. INTRODUÇÃO

Nossa vida está a todo momento gerando dados. Mesmo quando não estamos conectados a *Internet*, estamos gerando conteúdos que são coletados a todo tempo. Está constante rede de informações, aliada ao crescimento da rede de ciência da informação faz com que a todo instante a vida seja resumida a estudos gráficos, análises didáticas e processamentos virtuais do que vivemos dia após dia. Grande parte desse estudo está a procura de uma correlação ou significado oculto, que gere, ou não, um resultado.

Segundo o matemático americano Peter Norvig, ex-diretor de tecnologia da informação da NASA, o processo de coleta de dados demora muito mais tempo do que chegar a um resultado, porque, a maioria da informação pode parecer dispensável. O verdadeiro diferencial está em olhar com outros olhos o que muitos consideram lixo. Dentro destas informações está o que pode “revelar o mundo”. Visto isto, a ideia inicial deste estudo surgiu do crescimento abrangente do uso da análise de um banco de dados para auxílio na área da saúde, sendo utilizado informações dos atendimentos hospitalares do Hospital Pequeno Príncipe.

Observando a possibilidade de efetivamente se antecipar ao surgimento de novas demandas de atendimento do hospital, agravamento de doenças, intensificação de campanhas de vacinação, por exemplo, baseados na recorrência de atendimentos feitos no pronto socorro da instituição.

Utilizando-se de um dataset com 190 mil linhas, contendo dados dos pacientes que foram atendidos entre os anos de 2012 a 2021. Exames realizados, tempo de internação, pressão aferida, temperatura e CID do atendimento são algumas das informações detalhadas no arquivo e que serão utilizadas na análise exploratória para tentar obter algum resultado que seja relevante.

2. OBJETIVO GERAL

O trabalho segue as diretrizes de estudo do curso, e por meio do uso da programação Python (utilizando Aplicação Anaconda), visa extrair dados estruturados que futuramente possam ser usados para estudo e até mesmo resolução de problemas analíticos do Hospital Pequeno Príncipe, o qual surgiu no ano de 1919 e desde então vem fazendo um trabalho de excelência no atendimento de crianças e adolescentes de todo o Brasil.

Inicialmente, obteve- se as informações brutas armazenadas no banco de dados da Instituição desde o ano de 2012 para que fosse possível gerar estáticas e resultados. Entre estas informações estão: pressão, temperatura, dias de vida do paciente, exames entre outros. Assim, este projeto tem como objetivo contribuir para uma leitura mais acessível e dinâmica dos dados processados de forma a contribuir para o avanço no uso da tecnologia em busca da otimização do atendimento, ou seja, em qual forma o Hospital poderia aplicar na prática a leitura destes resultados obtidos.

2.1 OBJETIVOS ESPECÍFICOS

Os objetivos específicos são:

- Aprofundamento sobre linguagem de programação Python
- Manipulação, limpeza e processamento de dados utilizando a biblioteca Pandas
- Apresentação de resultados coletados pelos dados fornecidos pelo hospital

2.2 ORGANIZAÇÃO DO TRABALHO

Este projeto está dividido em cinco capítulos:

No Capítulo um é feito a introdução do tema.

No Capítulo dois é apresentado o Objetivo geral desta monografia, acompanhada do objetivo específico e organização do trabalho.

No Capítulo três, é apresentado um resumo da base teórica utilizada durante o curso, e literatura consultada para elaboração da estrutura didática da monografia, contextualizando a aplicação dos códigos e resultados obtidos.

No Capítulo quatro é possível fazer uma leitura do método proposto, assim como as ferramentas utilizadas para obtenção dos resultados, como elas foram utilizadas, e quais foram os dados mais relevantes.

No Capítulo cinco as considerações finais dos resultados obtidos pelos dados e adequações necessárias para aplicação, além das técnicas aplicadas.

3. FUNDAMENTAÇÃO TEÓRICA

3.1 DADOS HOSPITALARES

Bilhões de pessoas se utilizam de hospitais, seja para uma consulta de rotina, um exame, um internamento ou algo mais grave. Todos esses dados são acumulados em banco de dados para que se tenha um histórico desse paciente, facilitando as próximas consultas desse indivíduo.

Em alguns casos nessas consultas o médico pode avaliar esse histórico para tentar enxergar algum padrão ou alguma variável que possa ser a causadora dessa ida ao médico. Mesmo assim, a grande maioria dos dados ficam guardadas sem terem algum uso mais benéfico, apenas ‘acumulando poeira’.

> “O uso de big data na área da saúde trará importantes ganhos em termos de dinheiro, tempo e vidas e precisa ser ativamente defendido por cientistas de dados e epidemiologistas.” (CHIAVEGATTO FILHO, 2015, P.325)

Este trabalho tem por objetivo justamente dar um destino a alguns desses dados, em particular do Hospital Pequeno Príncipe, situado em Curitiba-PR, para obter uma leitura relevante que possa apontar alguma futura ação do hospital. Por exemplo, em caso de alta demanda em determinado setor, será possível se antecipar ao caos, gerando assim uma alocação maior de funcionários/médicos para suprir essa carga.

Ainda segundo Chiavegatto Filho (2015), existem dois pontos que merecem atenção: a medicina de precisão e o prontuário do paciente. Considerando a medicina de precisão, usando ciência de dados pode-se ter um diagnóstico mais específico sobre o método mais eficaz para determinado paciente, como também qual método para se evitar devido à condição do paciente se enquadrar com outros diagnósticos semelhantes. Isso seria possível aumentando o escopo das pesquisas, coletando mais dados e procurando relação entre os diferentes dados. Exemplificando, pode-se procurar entender se a reação de um medicamento em

certos pacientes pode estar atrelado ao sexo, idade, peso, entre outros atributos que podem ter sido descartados anteriormente.

Com relação ao prontuário do paciente, Chiavegatto Filho (2015) menciona que os dois grandes problemas para o avanço nessa área são prontuários em papel, portanto de difícil acesso, e não existir uma ligação entre os bancos de dados dos centros de saúde. Atualmente cada unidade de saúde tem sua própria base de dados não-interligada com outras unidades, o que ocasiona um atraso na efetivação do atendimento além da duplicação de exames, pois o mesmo procedimento pode ter sido realizado recentemente em outra unidade de saúde. Unificando em uma única base regional ou nacional, os médicos poderiam ter um histórico dos vários exames e diagnósticos realizados por aquele paciente, acelerando o atendimento e aumentando a eficácia do mesmo.

Além desses problemas, um outro ponto considerado se refere à quanta informação pode ser disponibilizada ao público, tendo em vista que dados dos pacientes são sigilosos.

3.1.1 Anonimização

De acordo com a Lei Geral de Proteção de Dados Pessoais, Nº 13.709 da Constituição Federal Brasileira, a anonimização é definida como “(...) dado relativo a titular que não possa ser identificado, considerando a utilização de meios técnicos razoáveis e disponíveis na ocasião de seu tratamento” (BRASIL, 2019). Ou seja, o dado perde a associação direta ou indireta, a um individuo, possibilitando assim sua utilização sem prejuízo algum. O uso desses dados compartilhados são vedados, exclusivamente, para uso com objetivo de obtenção de vantagem econômica.

Quando se trata do banco de dados analisado neste trabalho, a desvinculação pessoal é essencial para a análise neutra dos comportamentos, e aprendizados em massa. Esse tratamento desse ser compatível com as finalidades para que sejam utilizados.

É importante ressaltar a importância da contratação por parte da Instituição Pequeno Príncipe de profissionais ou empresas especializadas nestas metodologias que gerem relatórios do impacto da proteção de dados, visando a conformidade com as regras, gerando diminuição dos riscos de direitos e liberdades dos titulares destas informações.

3.2 ANÁLISE EXPLORATÓRIA

Análise exploratória é uma abordagem da Ciência de Dados para resumir as características de um conjunto de dados. Ciência de Dados, segundo (Cielen, Maysman e Ali, 2016) é o processo de extrair conhecimento usando métodos e análises de uma série de dados e fazer previsões em cima dessas análises. Já Fawcett (2016) acredita que o objetivo principal da Ciência de Dados é o aprimoramento da tomada de decisão. São usados tabelas ou gráficos para realizar a demonstração desses resultados, de uma forma mais intuitiva.

Segundo Tukey (1977), a análise exploratória de dados nunca é todo o processo, não é possível chegar ao fim da análise apenas nessa etapa, porém nenhum outro processo se encaixa melhor na primeira etapa.

Existem algumas fases que são realizados na análise exploratória e que são imprescindíveis para que haja um resultado final satisfatório. Obtenção dos dados; Processamento dos dados; Limpeza dos dados; Análise Exploratória.

- Obtenção dos Dados: Pode ser criado à mão porém comumente gerado ou extraído de outras fontes externas, onde o dado é considerado cru, ou seja, sem nenhum tratamento prévio. Como (Cielen, Maysman e Ali, 2016) mencionam, esses dados podem ter variados tipos: Estruturados, Não-Estruturados, Linguagem Natural, Linguagem de Máquina, Gráficos, Vídeo, Áudio e Imagens, além de streaming. Uma boa porcentagem dos datasets são em formato estruturado, que consiste em linhas de registro versus colunas de parâmetros.

- Análise dos Dados: Consiste na fase de carregar esses dados, entender de que tipo são e o que eles representam. Segundo (Rahm e Do, 2000) essa fase também é chamada de *data profiling* (perfil dos dados), pois foca na análise dos atributos e suas instâncias. Também como mencionam (Rahm e Do, 2000), essa fase incluí analisar os dados para começar a enxergar os possíveis problemas para a próxima etapa de limpeza. Pode ser considerado uma parte inicial de Limpeza dos Dados. De acordo com (Galhardas, Florescu, Shasha e Simon, 1999), existem vários quesitos onde é preciso se atentar, como dados de diferentes origens podem ser descritos de formas diferentes. Também podem haver erros nos dados, além de inconsistências.
- Limpeza dos Dados: Como menciona Batista (2003), também chamada de Pré-processamento dos Dados, é uma das fases que demandam um maior tempo, pois é preciso visualizar os dados nas colunas, entender os dados contidos, limpar (ou não) valores brancos ou nulos, tentar encontrar valores que não se encaixam, por vezes removendo uma linha inteira (um registro) pois há muitas informações faltantes ou incorretas. Segundo Rezig (2021), é difícil conseguir usar um processo automático para limpeza, normalmente é preciso haver um trabalho manual em cada dataset, pois cada um pode ter suas peculiaridades. A limpeza de dados é uma fase onde é preciso ter um bom conhecimento dos dados, além de uma fase onerosa, porém muito importante, pois uma boa análise exploratória depende de uma boa base de dados. Segundo Faceli (2011), apesar de algoritmos de aprendizado de máquina serem frequentemente adotados para extrair conhecimento de conjunto de dados, seu desempenho é geralmente afetado pelo estado dos dados.

> "Muitos pesquisadores têm citado que dados coletados diretamente de bancos de dados são de má qualidade, ou seja, possuem informações incorretas e imprecisas, além de uma grande quantidade de valores desconhecidos." (BATISTA, 2003, P. 40).

- Análise Exploratória – Visualização: Esta fase consiste na utilização de gráficos e outros métodos para apresentar informações do dataset usado. Segundo Zhang (2017), a principal utilidade desta etapa é para comunicar essas informações de uma forma que seja facilmente entendida. Mesmo dados complexos podem ser traduzidos quando vistos em forma de gráfico. Comparações dentre os dados também são realizadas mais intuitivamente, possibilitando extrair um conhecimento maior dessas informações, onde o cientista de dados ou o leitor obtenha um entendimento maior sobre aqueles dados. Visualizando os gráficos e modelos julgados úteis, descobrindo novos padrões anteriormente desconhecidos, e apresentando-os com mais detalhes fazem parte desta etapa. Há dois tipos de gráficos, de acordo com (Chen, Härdle e Unwin, 2008): gráficos de apresentação e gráficos exploratórios. Gráficos de apresentação servem para agrupar toda a informação necessária, mostrando o resultado final com detalhes; já gráficos de exploração servem como o caminho para esses resultados, as etapas necessárias antes da concretização do gráfico de apresentação. Também existe a chamada *data mining* (mineração de dados) segundo (Rahm e Do, 2000), onde é extraído o conhecimento a partir de grande quantidade de dados. Para isso é realizado várias consultas aos dados e subsequentes extrações de informações, padrões e tendências que normalmente são desconhecidos, conforme é explicado por Thuraisingham (1999). “Frequentemente, mineração de dados tem sido considerada e classificada como uma mistura de pesquisas em estatística, inteligência artificial e bancos de dados.”(CÔRTES, PORCARO e LIFSCHITZ, 2002, P.2).

3.3 LINGUAGEM DE PROGRAMAÇÃO PARA CIÊNCIA DE DADOS

Há várias linguagens de programação no mercado que podem ser usadas para a Ciência de Dados e vários fatores que podem alterar essa escolha, como bibliotecas, requisito no trabalho ou apenas preferência pessoal. Python e R podem

ser consideradas as principais do ramo, como menciona King J (2015), porém existem outras opções utilizáveis como Scala e Julia. Saindo do escopo da linguagem em si, também existem as opções do SQL e Excel.

É possível aprender e utilizar todas, porém cada uma tem suas especialidades e peculiaridades, sendo melhor talvez focar em uma ou duas para extrair o máximo de conhecimento possível, aumentando a habilidade naquela linguagem para um nível satisfatório.

3.3.1 Linguagem de Programação R

A linguagem de programação R é mais voltada para computação estatística e matemática, é também um software livre assim como Python. Segundo Chiavegatto Filho (2015), esse ponto aliado ao fato de serem de graça e com uma base grande programadores e cientistas as tornam bastante atrativas. Também possui uma grande biblioteca, e segundo Grolemund (2017), usar esses pacotes são a chave para o sucesso para se especializar no R.

3.3.2 Linguagem de Programação Scala

Conforme Alexander (2013) descreve, Scala é uma linguagem de programação mais avançada, voltada ao desenvolvedor sênior, porém ainda podendo ser bem utilizada por alguém com menos experiência, mesmo que não a utilize com todo seu potencial. Esta linguagem é voltada para a legibilidade, podendo ser usadas bibliotecas próprias do Scala mas também do Java, considerando que Scala roda em JVM.

3.3.3 Linguagem de Programação Julia

A sintaxe do Julia é parecida com linguagens de programação R e Python, conforme descrito por (McNicholas e Tait, 2019). Foi desenhado especificamente para uso com computação numérica, portanto pode ter um desempenho parecido com linguagens como C. Com uma biblioteca, é possível integrar um software escrito em Julia com outras linguagens de uma maneira rápida e fácil.

3.3.4 Linguagem de Programação Python

Segundo Lutz (2013), o uso de certas linguagens de programação podem decorrer de uma preferência pessoal, porém existem alguns fatores que ajudam a influenciar na escolha do Python, como: qualidade do software, visto que Python foca na legibilidade; na produtividade do desenvolvedor, considerando que o código Python geralmente é menor se comparado às outras linguagens; Grande biblioteca disponível para as tarefas, tendo diversos pacotes que podem ser de grande ajuda, agregando bibliotecas padrão e as geradas por desenvolvedores; Portabilidade e Integração também são outros fatores que a destacam no cenário da programação.

O nome Python tem origem no show Monty Python, um dos shows favoritos do desenvolvedor da linguagem, Guido van Rossum. A linguagem de programação Python está sendo utilizada extensivamente nos recentes anos devido à sua natureza, que pode ser voltada para a ciência de dados, trabalhando com manipulação de dados, entre outras tarefas. Segundo (Cielen, Meysman e Ali, 2016), Python é uma ótima linguagem por toda a ampla biblioteca disponível e pelo suporte de vários softwares. Nesta monografia, se utilizará a linguagem de programação Python. Esta é uma linguagem de aprendizado mais fácil, pois sua tipagem é mais leve, não necessitando declarar o tipo de uma variável. Outras linguagens utilizam chaves para delimitar blocos de código, Python usa indentação, como menciona Grus (2016). O código Python é conciso e efetivo portanto é

considerado de fácil entendimento e manutenção. Essa linguagem pode ser utilizada para resolver problemas tanto de web, como desktop e mobile. Outra função interessante da linguagem é o modo como lida com a memória, que é alocada apenas quando cria o objeto, e quando o ciclo de vida deste objeto acaba, a memória é liberada.

Como Lutz (2013) menciona, o problema mais significativo dessa linguagem é a velocidade de execução, se comparado a algumas linguagens de mais baixo nível. Mesmo assim, vale notar que Python já foi otimizado diversas vezes e com um compilador C embutido no interpretador do Python é possível rodar aplicações na velocidade do C.

Essas otimizações são possíveis graças a um grupo de pessoas focadas em continuar melhorando a linguagem para todos, além de uma comunidade grande para suporte, conforme menciona Lutz (2013). Uma mudança em Python pode ser realizada por um PEP (*Python Enhancement Project*), e o PSF (*Python Software Foundation*) é o responsável por isso.

3.3 BIBLIOTECAS

Bibliotecas ou pacotes são uma ferramenta importante para Ciência de Dados, normalmente são delas os métodos usados para que se alcance o objetivo proposto. Bibliotecas facilitam a vida dos desenvolvedores, uma vez que é possível poupar o trabalho de programar uma função para determinado processo, usando uma função existente dentro de uma biblioteca. Segundo Grolemund (2017), pacotes são a unidade fundamental, que incluem funções reusáveis, a documentação de como utilizar esses pacotes e dados de exemplo.

Um dos pontos usados na escolha da linguagem de programação é justamente as bibliotecas, pois o tipo de bibliotecas disponíveis, quão ativa é a comunidade e o suporte podem influenciar esta escolha.

Existem bibliotecas para recursos matemáticos, para análise de dados, voltadas para machine learning, para gráficos, para big data.

Considerando o Python, segundo (Cielen, Meysman e Ali, 2016), os pacotes podem ser divididos em 3 setores de Aprendizado de Máquina: Dados que cabem na memória (ex: Pandas); Otimização de código (ex. Blaze); e Big Data (ex: PySpark).

Considerando manipulação de Dados, Pandas é uma biblioteca Python que fornece ferramentas de análise de dados e estruturas de dados de um modo simplificado. Ela separa a estrutura de dados em Series e Dataframe, onde Series é uma estrutura unidimensional (por exemplo uma lista de valores) e Dataframe é uma estrutura bidimensional (por exemplo uma planilha). Segundo (Cielen, Meysman e Ali, 2016), foi o pacote Pandas que introduziu Dataframes ao Python, um tipo de tabela dentro da memória.

Existem várias funções dentro do Pandas que facilitam a análise dos dados, além da manipulação conforme desejado. É possível criar esses dataframes de acordo com as necessidades, usando diversos métodos para ler arquivos prontos que foram gerados e utilizando parâmetros que facilitam o carregamento.

Para recursos matemáticos, podemos citar NumPy (Python): Numpy é uma biblioteca Python para uso de computação científica, realizando cálculos em arrays e funções de álgebra linear, conforme (Cielen, Meysman e Ali, 2016). O Numpy fornece um grande conjunto de funções e operações de biblioteca que ajudam os programadores a executar cálculos numéricos.

Como uma opção para demonstrar gráficos, podemos mencionar o matplotlib da linguagem Python e o ggplot2 da linguagem R. Matplotlib serve para plotagens na maioria em 2D, porém com algumas funções 3D. É um pacote que funciona muito bem para simples gráficos de barras, linhas e de dispersão, segundo Grus (2016).

É importante lembrar que para um gráfico ser apresentado corretamente, é preciso que os dados estejam corretos. Porém como menciona Wickham (2010), os dados são independentes dos gráficos produzidos; gráficos podem e devem ser

utilizado por vários datasets variados, sem ser preciso modificá-los. Contando que o mapeamento das variáveis escolhidas seja satisfatório, o gráfico produzido também será satisfatório.

4 MÉTODO PROPOSTO E FERRAMENTAS UTILIZADAS

Para a realização deste trabalho foi utilizado a aplicação Anaconda, vastamente utilizada pela comunidade de Data Science, por ser um gerenciador de pacotes. Com ênfase na aplicação chamada Jupyter Notebook, um aplicativo web para compilação do código de várias linguagens de programação, neste caso, utilizando Python.

4.1 FERRAMENTAS

Foram utilizadas algumas ferramentas combinadas ao código Python para se alcançar o resultado desejado nesse trabalho. Aplicação Anaconda é um software de distribuição das linguagens de programação Python, simplificando o gerenciamento e implantação de pacotes. Esse software inclui pacotes de ciência de dados para Windows, Linux e macOS e dentro dele o Jupyter Notebook, uma aplicação web onde é possível criar código e compilá-lo de uma forma intuitiva.

Utilizando o Jupyter Notebook é possível criar códigos Python e utilizar os pacotes para data science existentes no Anaconda. Estes pacotes que se interligam com o código para produzirem um efeito mais robusto, com pouco trabalho para quem usa além de conhecer os métodos e funções.

Dentre os utilizados podemos citar o pacote Numpy, para recursos matemáticos; Matplotlib, para criar visualizações; Seaborn, baseado em Matplotlib e também serve para visualizações; Shapefile, para poder ler os arquivos em shapefile; Descartes, Geopandas, Point, Polygon e Geodataframe, para manipular dados geométricos.

4.2 COLETA DE DADOS

Os dados se originaram do banco de dados do Hospital Pequeno Príncipe, localizado em Curitiba – PR. O hospital é especializado em atendimento a crianças e adolescentes, sendo de grande referência para o país. Os funcionários do hospital do departamento de TI disponibilizaram essa base de dados, com cerca de 192 mil linhas, para que pudéssemos usá-lo de forma a tentar encontrar resultados, pontos de interesse, e qualquer assunto relacionado que pudesse ser utilizado pelo hospital para ter um maior conhecimento de seus dados e seus pacientes.

4.3 PROCESSAMENTO DOS DADOS

Primeiramente recebendo a base de dados, começa a análise crua, vendo o número de registros, quantas colunas existem nessa base, qual o tipo dessa coluna, que dados existem ali. Nessa etapa já é possível observar que algumas colunas possuem mais campos em brancos que outros, como as colunas dos exames.

Depois dessa primeira avaliação essa base de dados é carregada para dentro do jupyter notebook para que seja possível começar o trabalho em cima dela.

É preciso também importar algumas bibliotecas que serão utilizadas conforme o necessário descritos no apêndice A.

Nota-se que para podermos realizar a importação da base é preciso o uso da biblioteca Pandas, que contém a função de ler o dataset (neste caso, pd.read_csv).

Com o dataset carregado para dentro de um dataframe (df), pode-se usar a funçao de head(), que retorna os primeiros cinco registros, para se ter uma ideia de como foram importados os dados do arquivo, geralmente para ter uma noção se o dataframe foi carregado corretamente.

4.4 LIMPEZA DOS DADOS

Para a limpeza dos dados pode-se utilizar algumas funções que ajudam a entender melhor onde pode haver problema de dados, já que nesta etapa tenta-se eliminar ou minimizar ao máximo o impacto desses dados incorretos ou faltantes, seja arrumando esses dados (se utilizando de um senso comum; se atentar que senso comum pode gerar falhas), inserindo branco ou nulo, ou removendo a linha completamente (quando há muitos dados faltantes/incorretos ou esses dados faltantes/incorretos são de uma coluna chave).

Utilizando *df.info(), função* para visualizar o tipo de cada coluna percebida pelo dataframe, e quantos objetos estão com informação (não-nulo).

4.3.1 Temperatura

Começando pela temperatura que é um bom indicativo e amplamente utilizado e que nos primeiros registros já é possível ver inconsistências.

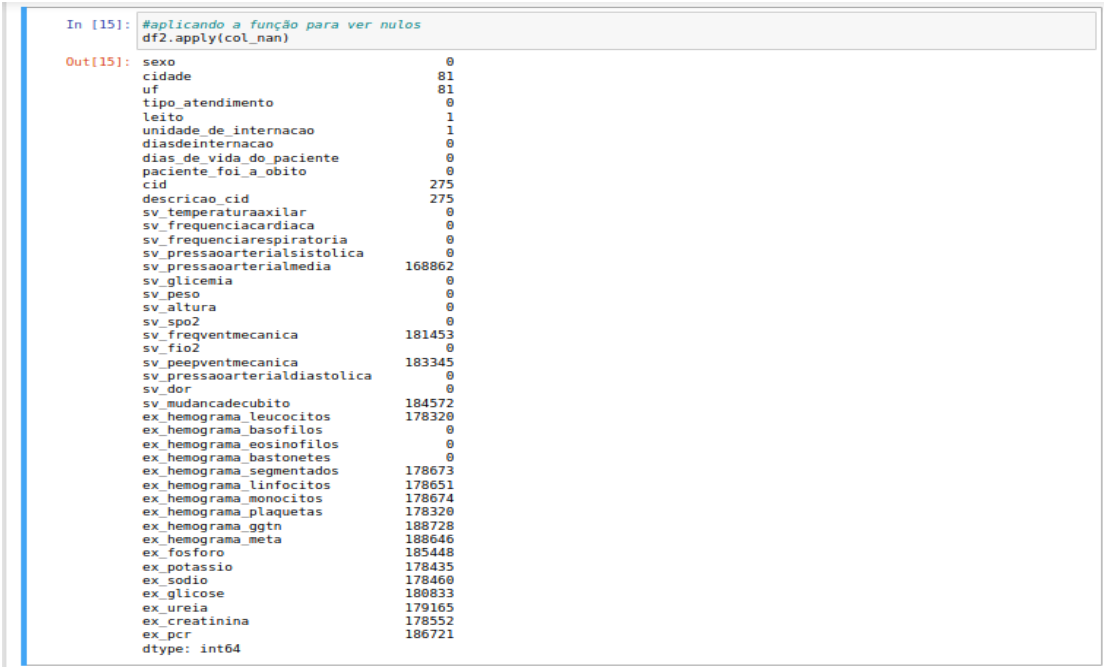
Utilizando a função *unique()* para visualizar quantos dados únicos existem dentro da coluna, e é possível perceber que não há um padrão claro seguido pelos médicos responsáveis por inserir os dados. Há temperaturas com vírgula; sem vírgula; dois, três ou quatro números. Para deixar mais padronizado, é possível recorrer a algumas funções do pandas, descritas no apêndice B.

Depois desse tratamento, os valores que sobraram estão em um padrão, ex. 37,6, o que facilita na hora de entender os resultados e realizar algumas plotagens de gráficos, além de poder utilizar mais campos. Campos com temperatura 0 apenas foram ignorados pois havia um grande número de 0 (~60 mil), portanto seria uma grande perda para futuras investigações.

4.3.2 Campos principais com poucos nulos

Para perceber de uma forma melhor a quantidade de registros nulos existentes em cada coluna, criei uma função para poder ser usada de forma geral em todas as colunas, acelerando o tempo e a visualização. A função está descrita no apêndice C.

Figura 1 – Função para visualização de nulos



```
In [15]: #aplicando a função para ver nulos
         df2.apply(col_nan)

Out[15]: sexo                              0
         cidade                           81
         uf                               81
         tipo_atendimento                  0
         leito                             1
         unidade_de_internacao             1
         diasdeinternacao                  0
         dias_de_vida_do_paciente          0
         paciente_foi_a_obito              0
         cid                             275
         descricao_cid                   275
         sv_temperaturaaxilar              0
         sv_frequenciacardiaca             0
         sv_frequenciarespiratoria         0
         sv_pressaoarterialsistolica       0
         sv_pressaoarterialmedia      168862
         sv_glicemia                       0
         sv_peso                           0
         sv_altura                         0
         sv_spo2                           0
         sv_freqventmecanica          181453
         sv_fio2                           0
         sv_peepventmecanica          183345
         sv_pressaoarterialdiastolica      0
         sv_dor                            0
         sv_mudancadecubito           184572
         ex_hemograma_leucocitos      178320
         ex_hemograma_basofilos            0
         ex_hemograma_eosinofilos          0
         ex_hemograma_bastonetes           0
         ex_hemograma_segmentados     178673
         ex_hemograma_linfocitos      178651
         ex_hemograma_monocitos       178674
         ex_hemograma_plaquetas       178320
         ex_hemograma_ggtn            188728
         ex_hemograma_meta            188646
         ex_fosforo                   185448
         ex_potassio                  178435
         ex_sodio                     178460
         ex_glicose                   180833
         ex_ureia                     179165
         ex_creatinina                178552
         ex_pcr                       186721
         dtype: int64
```

Fonte: Autor (2021)

Na figura 1 é possível observar os diferentes campos que compõe o dataset: campos gerais como sexo, cidade, uf; alguns relacionados ao atendimento na ocasião, como tipo_atendimento, leito, unidade_de_internacao, diasdeinternacao, dias_de_vida_do_paciente, paciente_foi_a_obito, cid e descricao_cid; e também os diferentes tipos de exames realizados por esses pacientes. Olhando a quantidade de nulos dentre os exames podemos perceber que existem exames realizados mais frequentemente, e exames realizados em casos mais específicos.

Observando o resultado da função foi possível perceber com facilidade que os campos cidade e uf possuíam 82 nulos, os campos leito e unidade_de_internacao possuíam 1 nulo, e cid e descricao_cid possuíam 276 nulos.

Como são campos importantes e o número de registrados nulos são proporcionalmente baixos, a opção escolhida seria remover a linha inteira para deixar o dataset mais consistente, utilizando funções descritas no apêndice D.

Depois de feito essa remoção foi feita uma checagem em alguns outros campos, e um campo que chamou a atenção foi ‘paciente_foi_a_obito’, que deveria ter apenas valores S(sim) ou N(não). Porém foi detectado que havia um terceiro valor utilizado, O(?).

Como o senso comum não é possível decidir em qual valor o ‘O’ se encaixa, e como são apenas 2 linhas com esse valor, seria mais produtivo remover essas linhas. Como não está sendo removido nulo, e sim um campo específico, é preciso alterar os parâmetros da função, também descrito no apêndice D.

4.5 ANÁLISE EXPLORATÓRIA

Nesta fase o objetivo é imaginar os possíveis cenários que seriam interessantes de se analisar com a base de dados, agora que esta base está mais limpa e portanto com um grau de confiança maior. Considerando isto, os gráficos e funções abaixo descrevem diferentes análises realizadas a partir dos dados, onde apareceram indagações e que podem ou não serem usadas futuramente para outros estudos mais aprofundados.

4.5.1 Atendimentos de pacientes do sexo Feminino x Masculino

Para essa análise se utilizou uma função de *size*, para descobrir quantas ocorrências de F (Feminino) e M (Masculino) existem.

Figura 2 - Agrupamento de pacientes por sexo

```
df1.groupby(['sexo']).size()
```

```
sexo
F     74348
M    114345
dtype: int64
```

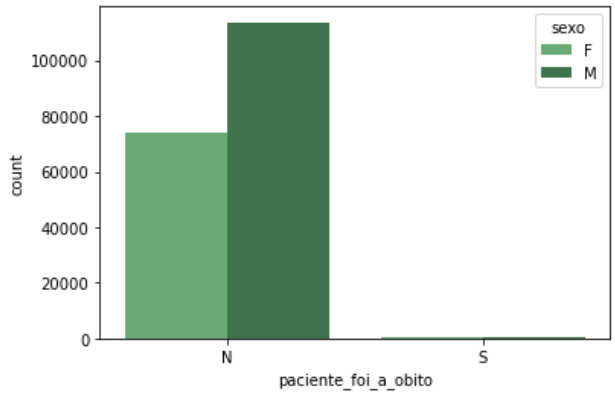
Fonte: Autor (2021)

Porém é possível fazer uma visualização desses dados de uma forma mais clara, agrupando também por número de óbitos para se ter uma relação de atendimentos por sexo versus quantidade de óbitos por sexo. Para se obter uma visualização mais clara foi utilizado o gráfico de countplot, pois é possível entender de uma maneira mais clara a quantidade de pacientes em cada variável, para óbitos e não-óbitos, divididos em feminino e masculino.

Figura 3 – Número de óbitos de pacientes por sexo

```
sns.countplot(x='paciente_foi_a_obito', hue='sexo', data=df3, palette="Greens_d")
```

```
<AxesSubplot:xlabel='paciente_foi_a_obito', ylabel='count'>
```



Fonte: Autor (2021)

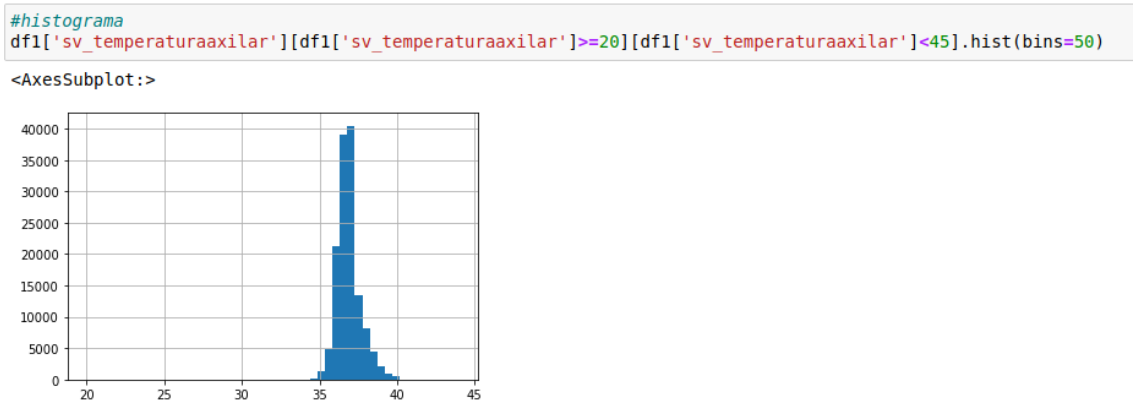
É possível perceber que há uma grande diferença entre atendimentos por sexo, onde pacientes do sexo masculino tem uma quantidade expressiva a mais de

registros. Também nesse gráfico é possível perceber pela proporcionalidade o número de óbitos, sinalizado pelo ‘S’ no eixo X. Apesar de não ser um número pequeno, é de bom grado saber que os atendimentos onde não há óbito passam em muito os atendimentos onde houve morte do paciente.

4.5.2 Temperatura média aferida

Para essa informação utilizamos de um histograma, pois pode-se aumentar a granularidade do gráfico aumentando a quantidade de ‘caixas’ com valores distintos. Fica aparente que a maior parte dos casos registrados e que possuem temperatura aferida tem esta temperatura marcada em 37 e 38 graus, o que indica um bom resultado. Vale notar que uma parte dos pacientes atendidos não tiveram sua temperatura aferida, talvez devido à natureza do atendimento que receberam, portanto não é possível ter certeza de que a média teria esse resultado caso todos os pacientes fossem obrigados a ter sua temperatura medida.

Figura 4 – Temperatura dos Pacientes



Fonte: Autor (2021)

4.5.3 Obesidade em crianças e adolescentes – IMC dos pacientes

Outra análise interessante que foi abordada nesta monografia foi a relação da obesidade entre os pacientes, e qual o Índice de Massa Corporal (ou IMC) dessas crianças. Para isso foi utilizado o cálculo geral para conseguir esse índice, que seria a massa dividida pela altura ao quadrado.

$IMC = Peso/Altura^2$

Com esse IMC temos o resultado do índice de cada paciente que teve seu peso e sua altura examinada.

Para a divisão entre cada grupo de IMC foi feito um cálculo médio independente da idade, portanto se o paciente possuísse um IMC menor que 13.5 ele seria considerado abaixo do peso; Entre 13.6 e 19.1 seria considerado um paciente com IMC normal; Maior que 19.1 e menor que 22 é categorizado como Sobre-peso; E se o paciente obtivesse um IMC maior que 22 seria considerado Obeso. Novamente é bom ressaltar que a quantidade de pacientes que realizaram exame de pesagem e altura é menor que o total de pacientes da base de dados; além disso, alguns desses dados estavam impossibilitados de serem usados mesmo após a padronização, portanto tiveram que ser descartados.

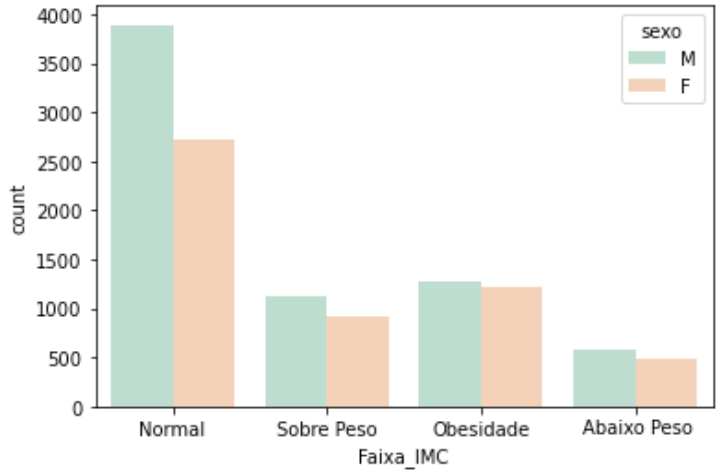
Podemos aferir do gráfico abaixo de countplot, utilizado pois destaca os números com clareza de acordo com cada faixa, que a maior parte dos pacientes observados estão na faixa Normal, o que é excelente; é possível perceber que há um número maior de pacientes do sexo masculino em comparação ao sexo feminino. Nessa faixa e em todas as outras faixas segue-se essa tendência.

O segundo lugar nesse gráfico é para os pacientes obesos, seguido de perto pelos pacientes com sobre peso. Abaixo do peso fica em último, portanto podemos supor que desnutrição talvez não seja um fato de destaque entre a população que foi atendida no pequeno príncipe. Possivelmente se utilizando de outros dados se poderia tentar entender se há algum outro fator do paciente para que haja esse nível de IMC.

Figura 5 – Faixas de IMC por Sexo

```
sns.countplot(x='Faixa_IMC', hue='sexo', data=df4, palette="Pastel2")
```

```
<AxesSubplot:xlabel='Faixa_IMC', ylabel='count'>
```

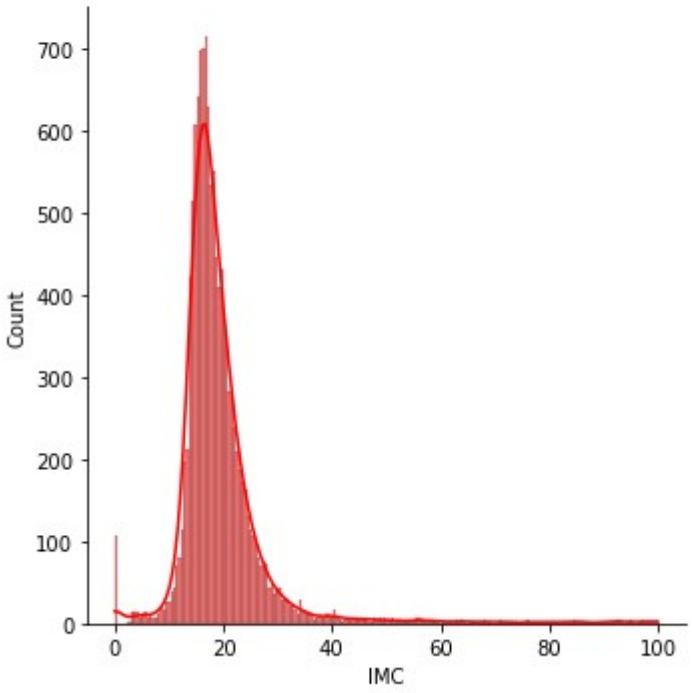


Fonte: Autor (2021)

Com esse gráfico de displot abaixo, utilizado para mostrar os dados em formato de curva e ter uma ideia da incidência de acordo com o IMC, podemos perceber que a maior parte dos pacientes registrados tem em torno de um IMC entre 14-18. Mas também podemos perceber que realmente existem poucos casos de Magreza, porém existem alguns IMCs com um nível muito abaixo, o que é preocupante.

Figura 6 – Gráfico de displot do IMC

```
sns.displot(df4['IMC'], kde=True, color="r")
<seaborn.axisgrid.FacetGrid at 0x7efc85da8520>
```

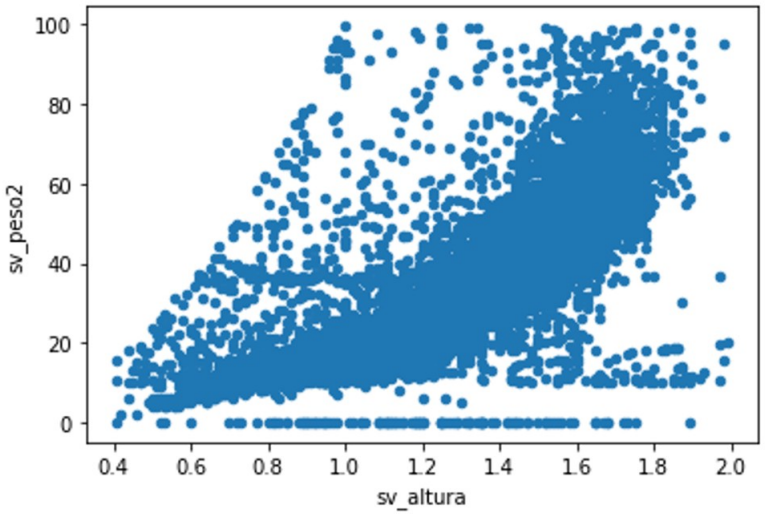


Fonte: Autor (2021)

Para o gráfico de scatterplot abaixo entre altura e peso, com a finalidade de apresentar a associação dos dados entre todos os pesos e alturas, podemos perceber que a grande parte dos registros segue uma mesma linha entre altura e o peso, mas também podemos perceber alguns problemas de dados onde o peso continua no zero mesmo com a altura aumentando, e também alguns com um mesmo muito acima do esperado.

Figura 7 – Gráfico de scatterplot do Peso x Altura

```
ax = df4.plot.scatter(x='sv_altura',y='sv_peso2')
```



Fonte: Autor (2021)

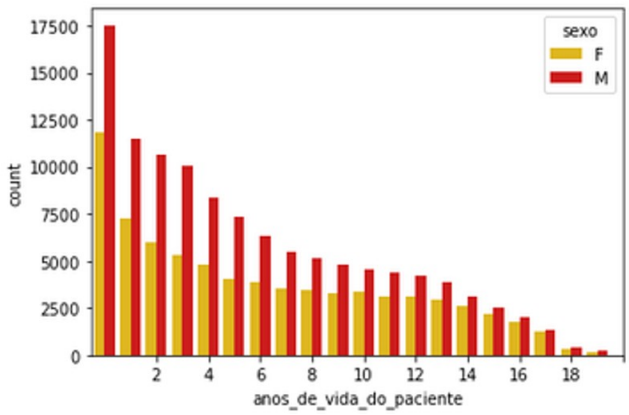
4.5.4 Idade de maior incidência de atendimentos

Para essa análise foi pego o campo ‘dias de vida do paciente’ e transformado em anos, para se ter uma representação mais simples. Após isso é realizado o plot do gráfico abaixo, neste caso um countplot pois demonstra as quantidades de cada parâmetro de uma forma clara, onde podemos ver claramente que o maior número de atendimentos ocorre nos primeiros anos de vida, e então vai ocorrendo uma diminuição de casos. Como o hospital Pequeno Príncipe é referência em tratamento de crianças e adolescentes, é possível imaginar que o número alto de atendimentos em crianças menores de 2 anos se deve ao reconhecimento do hospital, além de talvez uma inexperiência por parte dos parentes desse paciente, que preferem levar a criança em um hospital de renome à levá-lo em um postinho ou um hospital mais perto de sua residência.

Figura 8 - Média de vida do paciente atendido por sexo

```
ax, fig = plt.subplots()
sns.countplot(x='anos_de_vida_do_paciente', hue='sexo', data=df6, palette="hot_r")
plt.xticks(range(2, 22,2))
```

```
([<matplotlib.axis.XTick at 0x7efc2291edc0>,
  <matplotlib.axis.XTick at 0x7efc2291ed90>,
  <matplotlib.axis.XTick at 0x7efc2291e430>,
  <matplotlib.axis.XTick at 0x7efc22843970>,
  <matplotlib.axis.XTick at 0x7efc22843e80>,
  <matplotlib.axis.XTick at 0x7efc2284e3d0>,
  <matplotlib.axis.XTick at 0x7efc2284e8e0>,
  <matplotlib.axis.XTick at 0x7efc2284edf0>,
  <matplotlib.axis.XTick at 0x7efc2284e550>,
  <matplotlib.axis.XTick at 0x7efc228d7370>],
 [Text(0, 0, '0'),
  Text(1, 0, '1'),
  Text(2, 0, '2'),
  Text(3, 0, '3'),
  Text(4, 0, '4'),
  Text(5, 0, '5'),
  Text(6, 0, '6'),
  Text(7, 0, '7'),
  Text(8, 0, '8'),
  Text(9, 0, '9')])
```



Fonte: Autor (2021)

4.5.5 Dias de vida do paciente X Dias de internação

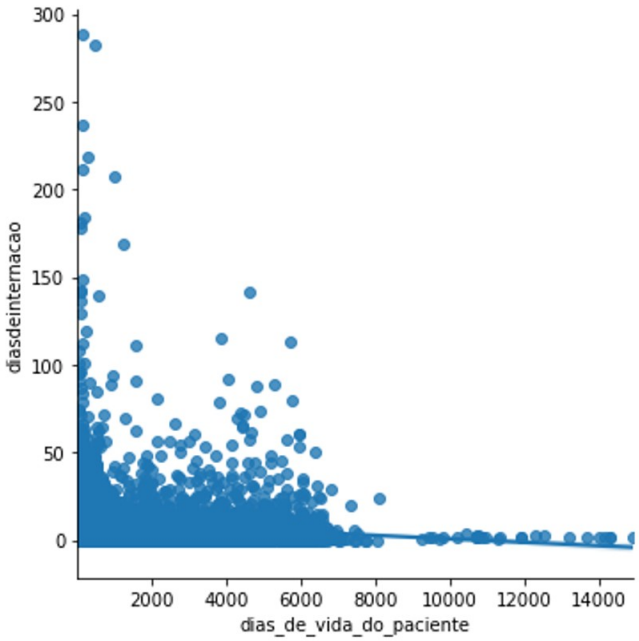
Com essa análise, foi possível entender se existia alguma relação entre os dias de vida do paciente com a quantidade de dias que esse paciente precisaria ficar internado.

Foi utilizado o lmplot abaixo pois é um gráfico de dispersão com um resultado interessante pois concentra as maiores incidências dos casos. Fica evidente que a grande maioria dos pacientes fica abaixo dos 50 dias internados, e esse número cai ainda mais quando se trata de pacientes mais velhos. Por outro lado, os dias de internação mais altos se concentram nos primeiros dias ou meses de vida do paciente, com alguns casos no começo da adolescência.

Figura 9 – Dias de vida do paciente por dias de internação

```
sns.lmplot(x="dias_de_vida_do_paciente", y="diasdeinternacao", data=df5)
```

```
<seaborn.axisgrid.FacetGrid at 0x7efc564d3790>
```



Fonte: Autor (2021)

4.5.6 Emergência – Tipos (SUS e Convênio) – maiores ocorrências

Esse gráfico foi realizado com o intuito de observar visualmente quais as maiores ocorrências de atendimento de emergência, sejam elas provenientes do SUS ou de Convênios.

Foi utilizado um countplot na horizontal para especificar os registros, e se baseando na legenda com a quantidade de registros.

Fica claro visualizando o gráfico abaixo que a maior parte dos atendimentos de emergência se referem à epilepsia e suas variadas formas.

Outra ocorrência que repete também é a de traumatismo intracraniano, além de bronquiolite e pneumonia.

Apesar de epilepsia poder ter várias causas, podemos relacionar o traumatismo intracraniano do paciente com as brincadeiras realizadas pelas crianças que talvez não enxerguem o perigo e o descuido dos parentes.

Podemos também relacionar os casos de bronquiolite e pneumonia ao tempo natural de Curitiba, que geralmente possui uma temperatura mais abaixo, além de chuvas mais regulares.

Figura 10 – Tipos de ocorrências na categoria ‘Emergência’

Fonte: Autor (2021)

4.5.7 Maior ocorrências de atendimento ao todo

As ocorrências totais de atendimento seguem o mesmo fluxo do gráfico anterior, apenas excluindo o filtro total de urgências.

Podemos perceber que a principal ocorrência se deve ao fato de complicações de fimose, com uma grande diferença para os outros atendimentos, com quase 10mil registros a mais. Em segundo lugar visualiza-se enfermidade na pele e terceiro lugar complicações com as amígdalas.

Vale ressaltar que leucemia linfoblástica, uma forma de doença mais grave, também possui um registro entre as 20 maiores ocorrências.

Figura 11 – Maiores ocorrências de atendimentos

Fonte: Autor (2021)

4.5.8 Correlação entre exames realizados

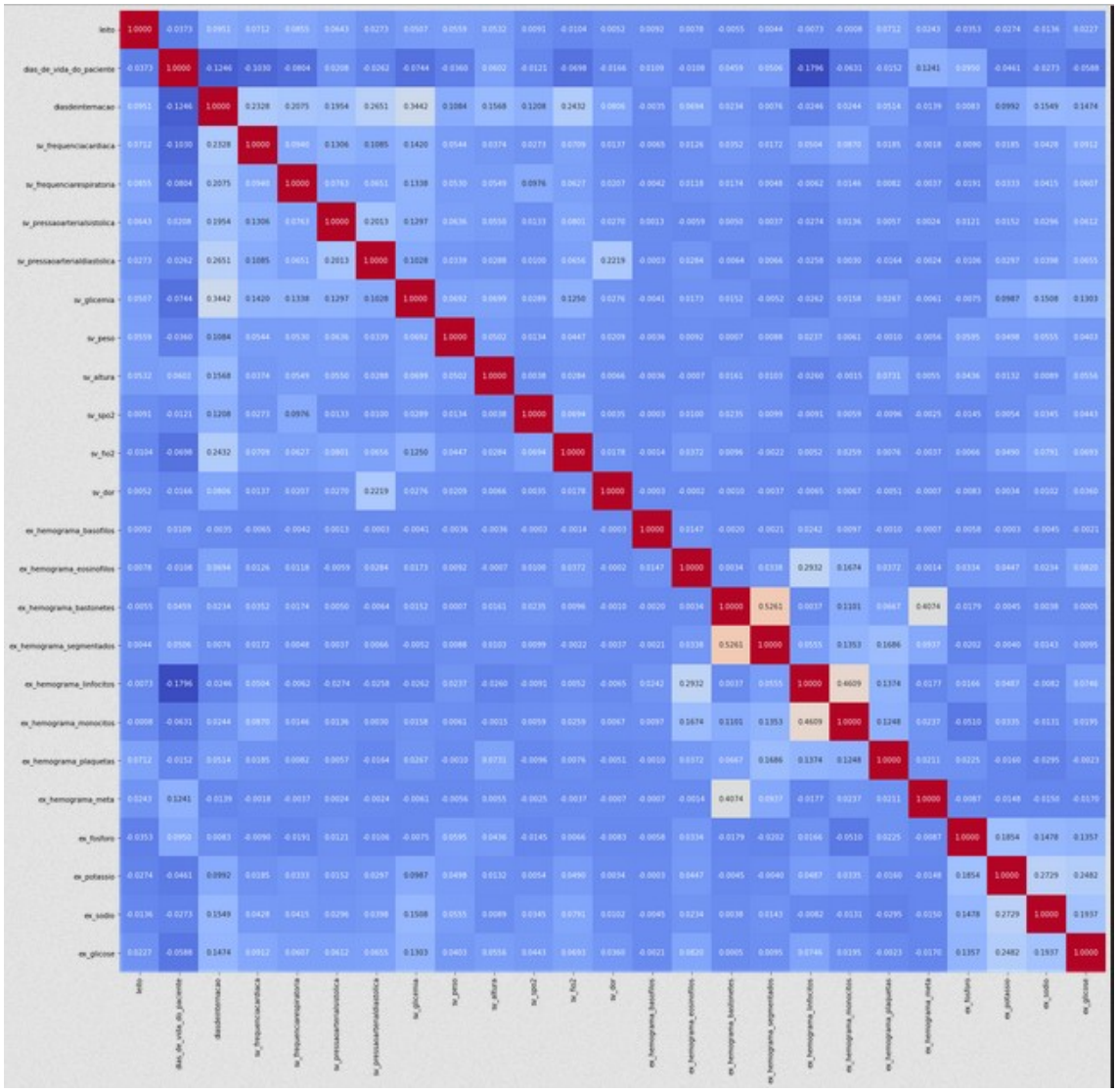
Esta correlação foi realizada para se averiguar se os exames feitos pelos pacientes teriam algum tipo de relação entre si. É importante registrar que existe um registro pequeno de pacientes que precisaram se submeter a esses tipos de exames, além disso, dentre estes alguns exames foram mais comuns que outros, portanto é relevante lembrar que essas informações podem causar um desvio na correlação, talvez alterando ou mascarando uma possível correlação maior caso os dados estivessem mais completos, mas é possível afirmar também que isso também

pode apontar uma correlação correta entre as variáveis, considerando que um exame possa estar atrelado a outro exame.

Foi usado o gráfico de correlação pois é um importante instrumento para avaliar a relação de todos os diversos atributos, que talvez não sejam tão facilmente visualizados e entendidos manualmente.

Pelos exames abaixo, não é possível identificar uma grande correlação entre as variáveis de exame, a maior sendo exame de hemogramas-bastonete e exame hemograma-segmentado (com 0,52 de correlação), seguido de exame de hemograma de monócitos e linfócitos (0,46) e exame de hemogramas-bastonete com exame hemograma-meta (0,40). Dentre os outros é possível apontar a correlação entre dias de internação e dor (0,31) e exame de hemograma de linfócitos e eosinófilos (0,29). Para os leigos, é difícil enxergar uma consequência dessa correlação, exceto talvez a correlação entre dias de internação e dor, onde podemos inferir que se a pessoa está com um nível de dor, existe a possibilidade da extensão da internação, porém pelo 0,31 não é possível ter certeza.

Figura 12 – Correlação dos exames realizados



Fonte: Autor (2021)

Há vários exames que podem ser realizados pelo hospital em um paciente, além de outros dados. Abaixo estão esses dados com uma breve descrição para cada variável:

-Leito: Número do quarto onde o paciente foi internado;

-Dias de vida do paciente: Dias de vida do paciente no dia do atendimento;

-Dias de internação: Quantidade de dias em que o paciente ficou internado;

-Frequência cardíaca: Batimentos cardíacos do paciente;

-Frequência respiratória: Taxa de respiração do paciente;

-Pressão arterial sistólica: É a pressão máxima, marca a contração do músculo cardíaco;

-Pressão arterial diastólica: Marca o repouso do músculo;

-Glicemia: Taxa de glicose no sangue;

-Peso: Peso do paciente;

-Altura: Altura do paciente;

-SpO2: Saturação periférica de oxigênio;

-FiO2: Fração inspirada de oxigênio;

-Dor: de 0 a 10, nível de dor percebido pelo paciente;

-Hemograma basófilos: Basófilos são células pertencentes ao sistema imunológico;

-Hemograma eosinófilos: Eosinófilos são glóbulos brancos do sangue que desempenham uma resposta a reações alérgicas, asma e infecção por parasitas;

-Hemograma bastonetes: Bastonetes são glóbulos brancos, neutrófilos imaturos, ainda não desenvolvidos;

-Hemograma segmentados: Os neutrófilos segmentados são células responsáveis pela defesa do organismo e são encontrados em maior quantidade no sangue quando comparado aos outros neutrófilos;

-Hemograma linfócitos: Linfócitos são células responsáveis pela defesa do corpo. Eles pertencem ao grupo dos leucócitos, também chamados de glóbulos brancos;

-Hemograma monócitos: Os monócitos são células responsáveis pela defesa do corpo. Eles pertencem ao grupo dos leucócitos;

-Hemograma plaquetas: As plaquetas são células sanguíneas incolores que ajudam na formação do coágulo sanguíneo;

-Hemograma meta: Exame para detectar quantidade de metamielócitos no sangue;

-Fósforo: Com o exame é possível identificar e acompanhar patologias nos rins ou no trato gastrointestinal;

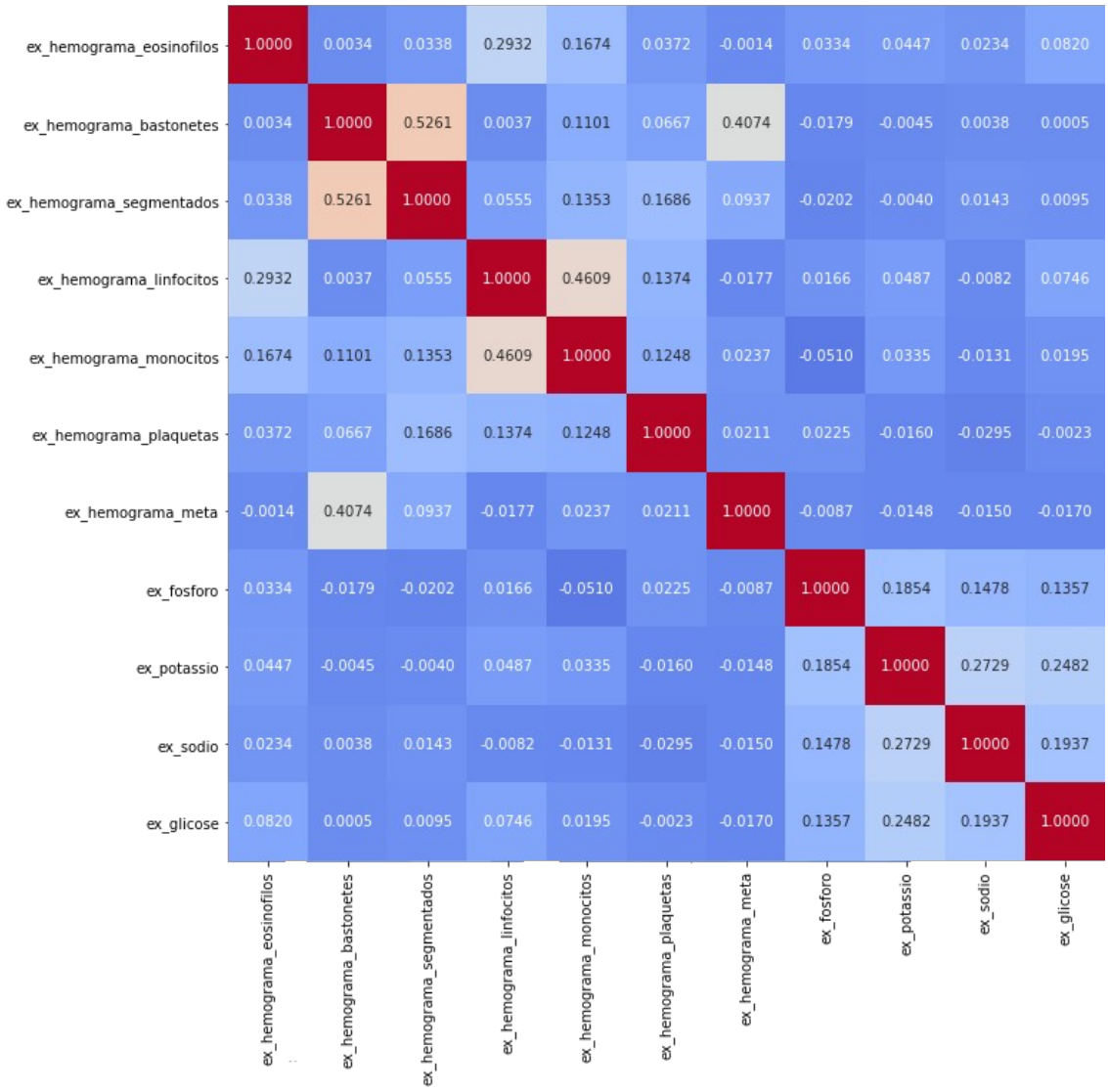
-Potássio: Medir o nível de potássio no sangue para identificar irregularidades;

-Sódio: É usado para detectar concentrações anormais de sódio, denominadas hiponatremia e hipernatremia;

-Glicose: O teste de glicemia detecta a hipo e a hiperglicemia, ou seja, quando há pouco ou muito açúcar em circulação.

Focando mais onde há uma alteração um pouco maior entre as diferentes variáveis:

Figura 13 – Correlação dos exames realizados - Foco



Fonte: Autor (2021)

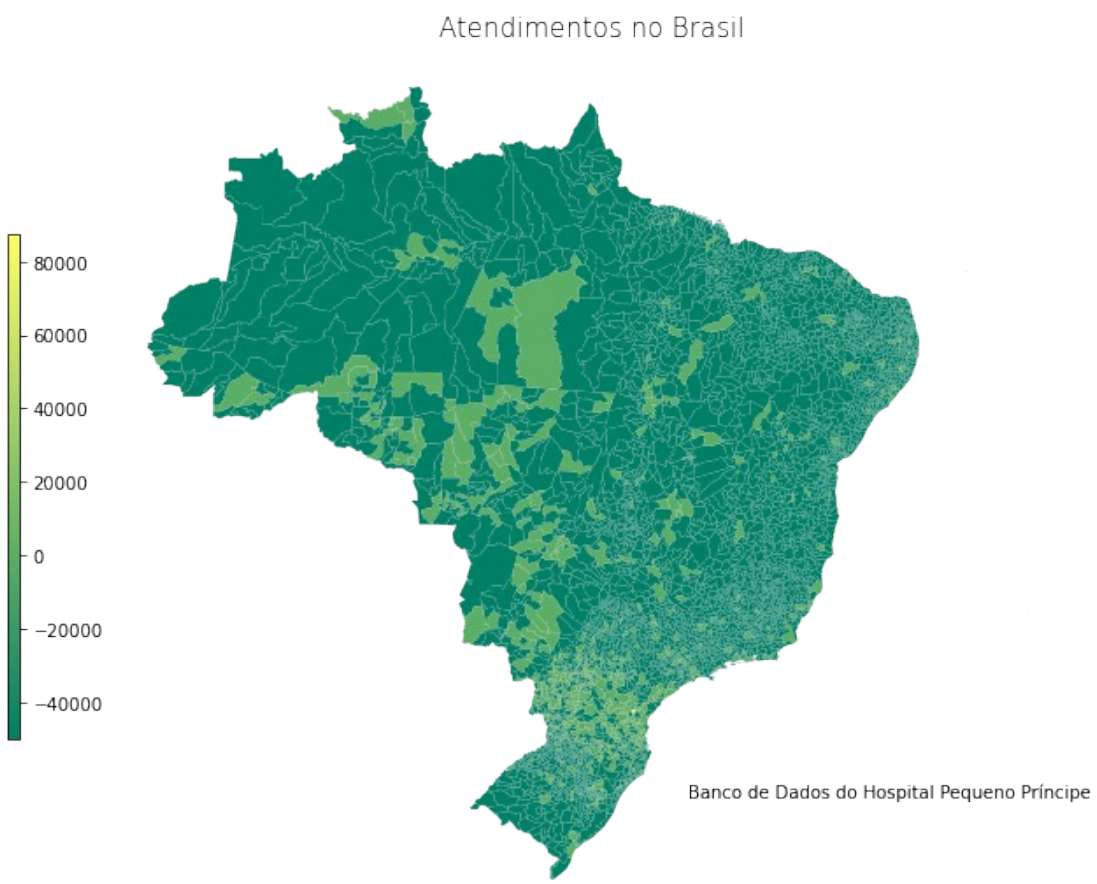
4.5.9 Concentração de pacientes no Brasil

Com o objetivo de visualizar a abrangência dos atendimentos do hospital Pequeno Príncipe pelo território nacional, foi realizado um mapa com as quantidades de ocorrências por todo o Brasil. É importante se atentar ao fato de que devido a não ter existido atendimentos em todos os municípios do Brasil, foi preciso realizar um

acréscimo de um valor negativo aos registros anteriormente com 0. Isso possibilita a criação de um cenário condizente com o mapa do território brasileiro.

Obviamente a maior parte dos atendimentos se concentra no estado do Paraná e mais especificamente em Curitiba, sede do hospital. Interessante notar que o centro-oeste e o norte possuem uma grande área de atendimentos também, mesmo com a distância necessária para percorrer, o que indica que o hospital tem um grande renome, e possivelmente referência para tratamento de certas doenças.

Figura 14 – Atendimentos no Brasil



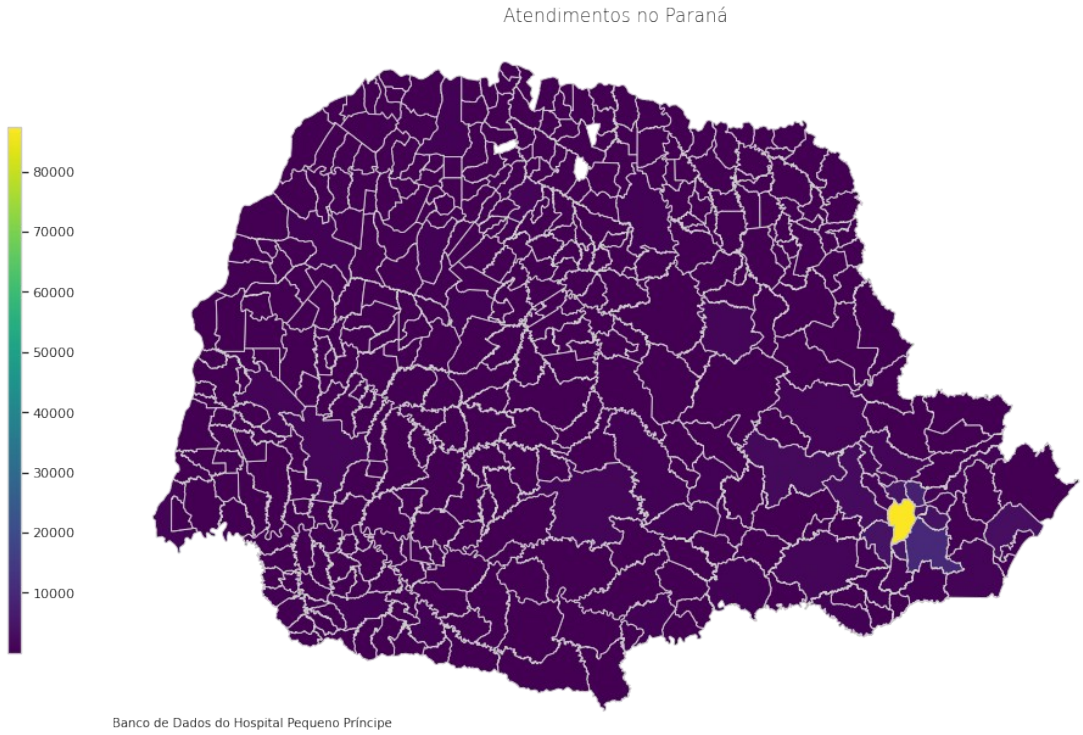
Fonte: Autor (2021)

4.5.10 Concentração de pacientes no Estado do Paraná

Concentrando a visualização apenas no mapa do Paraná, é possível entender melhor a extensão em que o hospital chega para atendimentos.

Um empecilho para essa primeira visualização é que Curitiba sendo a sede do hospital e sendo uma capital, possui um número muito significativo em relação às outras cidades, o que inviabiliza uma leitura mais dinâmica desse mapa. O que é possível aferir desse primeiro gráfico é que existem muito poucas cidades onde não houve pacientes tratados ou atendidos pelo Pequeno Príncipe.

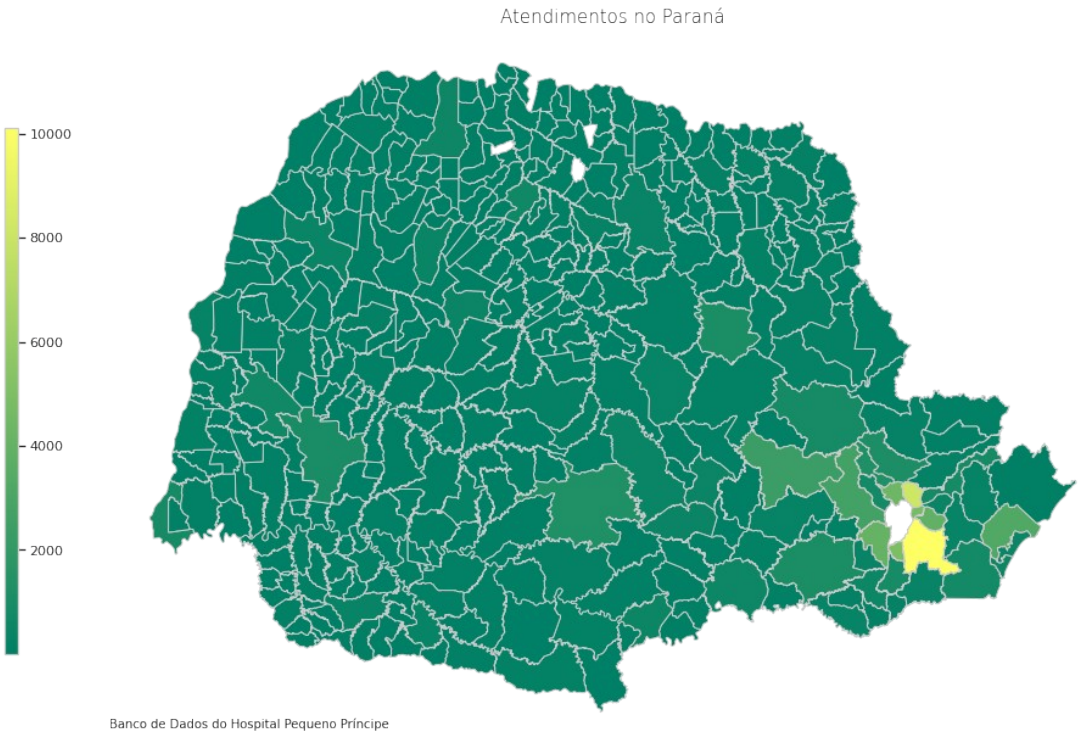
Figura 15 – Atendimentos no Paraná



Fonte: Autor (2021)

Uma alternativa para se realizar uma leitura mais fácil do mapa de atendimentos no Paraná é remover a capital e sede do hospital, Curitiba, deixando assim uma disparidade menor, e então perceber com clareza que a região metropolitana de Curitiba é a segunda maior utilizadora do hospital. Nesse novo gráfico, também é possível perceber com mais facilidade a quantidade de atendimentos nos diversos municípios que integram o estado do Paraná.

Figura 16 – Atendimentos no Paraná - Foco



Fonte: Autor (2021)

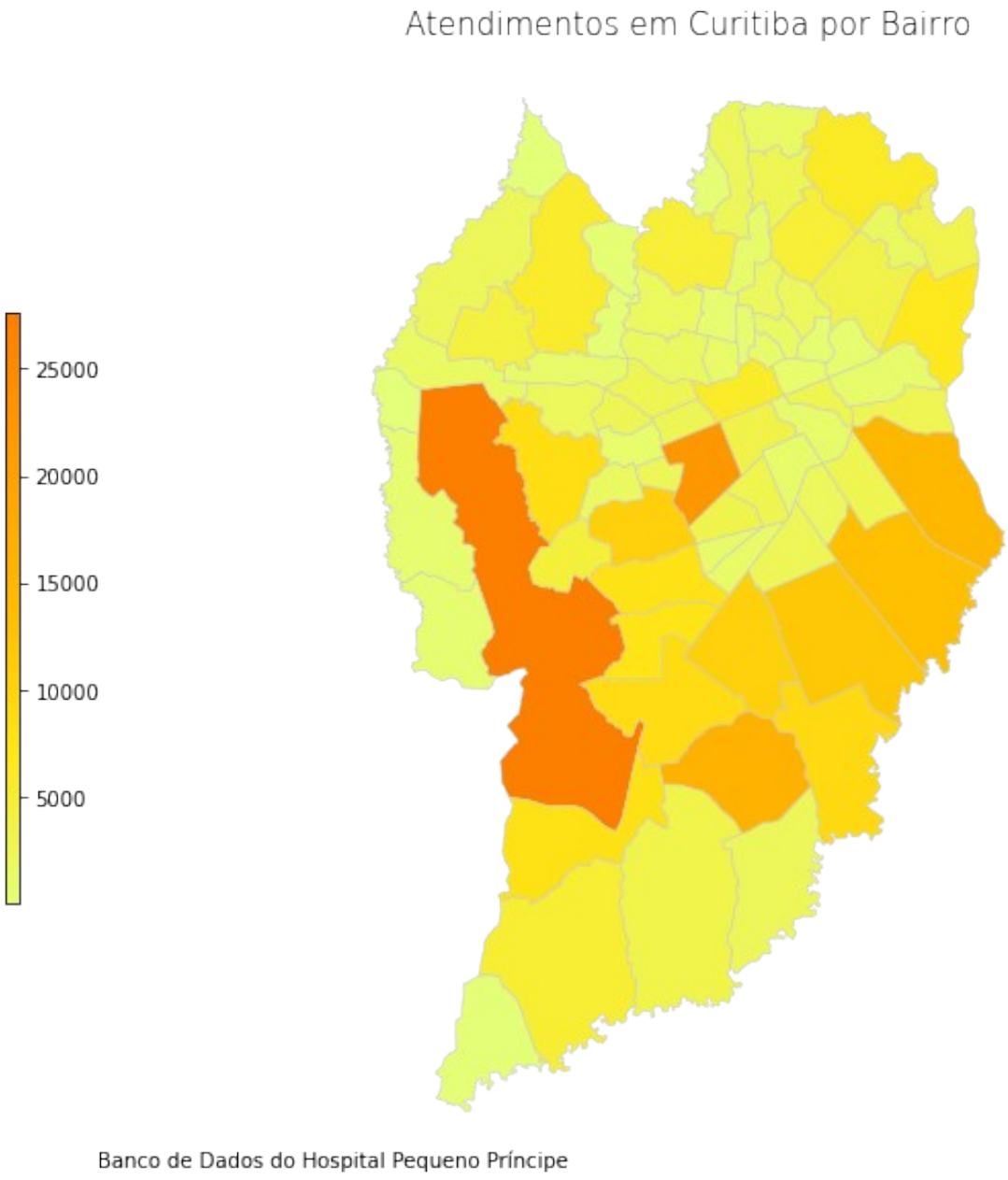
4.5.11 Concentração de pacientes na Cidade de Curitiba

Finalmente é realizado um gráfico para contornar e atribuir aos diversos bairros de Curitiba os atendimentos realizados pelo hospital Pequeno Príncipe. É possível perceber com relativa facilidade que os bairros que mais se beneficiaram do hospital foram Cidade Industrial e Água Verde.

Utilizando o mapa, é possível deduzir o motivo dos dois terem o maior número de ocorrências. Cidade Industrial tem uma área gigantesca se comparada aos outros bairros de Curitiba, portanto é possível concluir ser esse o motivo do alto número de atendimentos; já no Água Verde fica a sede do hospital Pequeno Príncipe, portanto também podemos entender ser esse o motivo.

Importante ressaltar também que existe uma grande quantidade de atendimentos para a faixa sul da cidade se comparado com a faixa norte. Apenas com esses dados numéricos fica trabalhoso encontrar o motivo, e talvez com outras bases de dados poderia se encontrar um motivo mais concreto, porém pode-se tentar relacionar isso ao número de crianças e adolescentes nesses diferentes bairros, ou também à ocorrência de atendimentos dessas crianças e adolescentes às outras unidades de saúde de Curitiba.

Figura 17 – Atendimentos nos bairros de Curitiba



Fonte: Autor (2021)

5 RESULTADOS

Os resultados obtidos com este trabalho foram alcançados na forma de uma visualização dos vários atributos retirados do dataset. Utilizando diferentes filtros dentre esses dados foi possível obter diversas oportunidades para demonstrar esses dados de uma forma fluida. Com os diferentes gráficos gerados a partir desses dados é possível ter uma noção maior de como esses dados estão estruturados, além de que uma pessoa leiga teria uma maior facilidade para entender como essas informações estão distribuídas.

Foi possível também utilizar a linguagem de programação Python para fazer o processamento desse dataset, além de toda a limpeza e a visualização em si, o que ocasionou um aperfeiçoamento na programação Python, além de uma especialização dos pacotes utilizados neste trabalho.

Utilizando várias bibliotecas, incluindo a biblioteca Pandas, foi possível realizar uma manipulação desses dados, além da limpeza, arrumando ou removendo dados incorretos, para se ter uma eficácia e confiança maior na hora da criação dos dados.

6 CONSIDERAÇÕES FINAIS

Com este trabalho foi apresentado dados de atendimentos do hospital Pequeno Príncipe em forma de variados gráficos. Gráficos que foram realizados para exemplificar e realizar algumas análises em cima desses dados de uma forma mais prática.

Utilizando-se desse material, o leitor pode ter uma base prévia de como estão agrupados as diversas informações do hospital, sendo possível utilizá-los para uma análise mais profunda e a partir disso tomar uma ação, caso o leitor tenha esse poder.

Uma sugestão de trabalho futuro pode ser o próprio aprofundamento dessa análise exploratória, procurando encontrar mais detalhes e relacionamentos para os possíveis questionamentos encontrados nessa monografia. Com um dataset mais robusto, por exemplo com as datas dos atendimentos, poderia ser realizado um mapeamento dos atendimentos através dos anos, encontrando ou não algum aumento em casos específicos.

Outra sugestão de trabalho futuro poderia ser a realização de um algoritmo de predição, para tentar entender se os exames realizados por um paciente poderiam indicar algum quadro de alguma doença.

7 REFERÊNCIAS BIBLIOGRÁFICAS

Grus, J. **Data Science do Zero: Primeiras Regras com o Python.** Alta Books, 2019.

FAWCETT, T.; PROVOST, F. **Data Science para negócios.** 1ª ed. Alta Books: 2016.

FACELI, Katti; LORENA, Ana Carolina; GAMA, João; CARVALHO, André Carlos Ponce de Leon Ferreira de. **Inteligência artificial:** uma abordagem de aprendizado de máquina. [S.l: s.n.], 2011

Lutz, Mark. **Learning Python.** 5a ed. O'Reilly, 2013

Wickham, Hadley e Grolemund, Garret. **R for Data Science:** Import, Tidy, Transform, Visualize and Model Data. Canada: O'Reilly, 2017.

Alexander, Alvin. **Scala Cookbook.** 1a ed. O'Reilly, 2013.

Rezig, E. **Data Cleaning in the Era of Data Science:** Challenges and Opportunities, 2021.

Galhardas H. Florescu D. Shasha D. and Simon. E. **An extensible framework for data cleaning.** In Proc. ACM SIGMOD Int. Conf. on Management of Data, 1999.

Côrtes, S. C., Porcaro R.M., Lifschitz, S. **Mineração de Dados –** Funcionalidades, Técnicas e Abordagens, 2002.

Thuraisingham, B. **Data Mining.** CRC Press, 1999.

Chiavegatto Filho, A. D. **Uso de big data em saúde no Brasil:** perspectivas para um futuro próximo. 2015.

King J, Magoulas R. **Data science salary survey:** tools, trends, what pays (and what doesn't) for data professionals. Sebastopol: O'Reilly, 2014.

Batista, Gustavo Enrique de Almeida Prado Alves. **Pré-processamento de Dados em Aprendizado de Máquina Supervisionado.** 2003.

CIELEN, Davy; MEYSMAN, Arno; ALI, Mohamed. **Introducing data science:** big data, machine learning, and more, using Python tools. Manning Publications Co., 2016.

Tukey, John W. **Exploratory Data Analysis.** Addison-Wesley, 1977.

Wickham, Hadley. **A Layered Grammar of Graphics.** American Statistical Association, Institute of Mathematical Statistics, and Interface Foundation of North America Journal of Computational and Graphical Statistics, 2010.

Rahm E. and Do. H.H. **Data cleaning:** problems and current approaches. IEEE Data Engineering Bulletin, 2000.

Zhang, A. **Data analytics:** Practical guide to leveraging the power of Algorithms, data science, data mining, statistics, big data, and predictive analysis to improve business, work, and life. 2017.

Chun-houh Chen, Wolfgang Karl Härdle and Antony R. Unwin. **Handbook of Data Visualization.** 2008.

BRASIL. **Lei nr. 13.709, de 14 de agosto de 2018.** Disponível em http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm?fbclid=IwAR1P2O4UDFglxERYjVaw_NdpN5bTa9GuWi4QlPCDpkr8PAtYPB5pqXwAxto – Acesso em: 27 de junho às 16:00.

REVISTA VEJA. **O berço do Big Data.** 15 de maio de 2013 p. 71-81. Disponível em: https://lucianabicalho.files.wordpress.com/2013/08/veja-big-data.pdf - Acesso em: 13 de junho às 12:38.

McNicholas, Paul D. and Tait, Peter A. **Data Science with Julia.** CRC Press, 2019.

APÊNDICE A – Imports

```
import os
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import descartes
import geopandas as gpd
from shapely.geometry import Point, Polygon
from geopandas import GeoDataFrame

df = pd.read_csv("TCC/Lista_pacientes_exames_sinais_vitais3.csv")
df.head()

df.info()
```

APÊNDICE B – Limpeza - Temperatura

```
#remover virgula
df1['sv_temperaturaaxilar'] = df1['sv_temperaturaaxilar'].str.replace(',','')

#Acrescentar '0' a números com caracteres menores que três
df1['sv_temperaturaaxilar'] = df1['sv_temperaturaaxilar'].str.ljust(3, '0')

#Inserindo '0' em campos nulos e alterando o tipo para integer
df1['sv_temperaturaaxilar'] = df1['sv_temperaturaaxilar'].fillna(0).astype(int)

#Dividir os números por 10 para ter a temperatura no padrão
df1['sv_temperaturaaxilar']= df1['sv_temperaturaaxilar']/10

#Remover apenas temperaturas abaixo de 20 e acima de 45, o que significa dados incorretos, pois o corpo humano não alcança essas temperaturas.
#Deixar números com 0 pois existe um número grande, assim não impactando outras análises.
df1 = df1.drop(df1[df1['sv_temperaturaaxilar'] >= 45].index)
df1 = df1.drop(df1[df1['sv_temperaturaaxilar'].between(0.1, 20., inclusive=False)].index)

#Após checagem dos números de nulos em cada coluna, removido os que tem poucas linhas como cidade, leito e cid, assim ficando mais consistente
df1 = df1.dropna(subset=['cidade'])
df1 = df1.dropna(subset=['leito'])
df1 = df1.dropna(subset=['cid'])

#Removendo duas linhas com valor incorreto
df1 = df1.drop(df1[df1['paciente_foi_a_obito'] == 'O'].index)
```

APÊNDICE C – Função para visualizar nulos em cada coluna

```
def col_nan(coluna):
    return coluna.isna().sum()

#aplicando a função para ver nulos
df.apply(col_nan)
```

APÊNDICE D - IMC

```
df4 = df1[['sexo','sv_altura','sv_peso','dias_de_vida_do_paciente']]
df4['sv_altura'] = df4['sv_altura'].fillna(0).astype(int)

#Dividir os números por 100 para ter a altura no padrão, em metros
df4['sv_altura']= df4['sv_altura']/100
df4['sv_altura'].value_counts()
#Excluir mais de 2 metros e menos de 40cm, o que deve ser erro
df4 = df4.drop(df4[df4['sv_altura'] >= 2].index)
df4 = df4.drop(df4[df4['sv_altura'].between(0.0, 0.4, inclusive=True)].index)
df4['anos_de_vida_do_paciente']= df4['dias_de_vida_do_paciente']/365
df4['anos_de_vida_do_paciente'] =
df4['anos_de_vida_do_paciente'].map("{:.2f}".format)

#Inserindo 0 nos números nulls
#se o peso estiver mais de 100, dividir por 10
df4['sv_peso'] = df4['sv_peso'].fillna(0).astype(int)
df4.loc[df4['sv_peso'] >= 100, 'sv_peso2'] = df4['sv_peso']/10
df4.loc[df4['sv_peso'] < 100, 'sv_peso2'] = df4['sv_peso']

#Excluir mais de 100kg, possivelmente dado mal-computado
df4 = df4.drop(df4[df4['sv_peso2'] >= 100].index)

#Fazendo cálculo do imc
df4['IMC'] = df4['sv_peso2'] / (df4['sv_altura'] * df4['sv_altura'])

df4 = df4.drop(df4[df4['IMC'] >= 100].index)

#Atribuindo especificações de acordo com o imc, apesar de ser usado um
imc geral
mask_normal = (df4['IMC'] >= 13.6) & (df4['IMC'] < 19.1)
mask_sobre_peso = (df4['IMC'] >= 19.1) & (df4['IMC'] < 22)

df4.loc[df4['IMC'] < 13.5, 'Faixa_IMC'] = 'Magreza'
df4.loc[mask_normal, 'Faixa_IMC'] = 'Normal'
df4.loc[mask_sobre_peso, 'Faixa_IMC'] = 'Sobre Peso'
df4.loc[df4['IMC'] > 22, 'Faixa_IMC'] = 'Obesidade'
df4 = df4.dropna(subset=['Faixa_IMC'])
```

APÊNDICE E - Correlação entre Exames

```
df5 = df1[['leito','dias_de_vida_do_paciente','diasdeinternacao','sv_frequenciacardiaca','sv_frequenciarespiratoria','sv_pressaoarterialsistolica','sv_pressaoarterialdiastolica',
'sv_glicemia','sv_peso','sv_altura','sv_spo2','sv_fio2','sv_dor','ex_hemograma_basofilos',
'ex_hemograma_eosinofilos','ex_hemograma_bastonetes','ex_hemograma_segmentados',
'ex_hemograma_linfocitos','ex_hemograma_monocitos','ex_hemograma_plaquetas','ex_hemograma_ggtn','ex_hemograma_meta',
'ex_fosforo','ex_potassio','ex_sodio','ex_glicose','ex_ureia','ex_creatinina','ex_pcr']]
#Tirando um certo número de NaNs para correlação ter mais itens presentes, tendo mais precisão
df5 = df5.dropna(subset=['ex_hemograma_basofilos'])

df5 = df5.replace(',','',regex=True)
df5['sv_frequenciacardiaca'] = df5['sv_frequenciacardiaca'].str.rjust(3, '0')
df5['sv_frequenciarespiratoria'] = df5['sv_frequenciarespiratoria'].str.rjust(3, '0')
df5['sv_pressaoarterialsistolica'] = df5['sv_pressaoarterialsistolica'].str.rjust(3, '0')
df5['sv_pressaoarterialdiastolica'] = df5['sv_pressaoarterialdiastolica'].str.rjust(3, '0')
df5['sv_glicemia'] = df5['sv_glicemia'].str.rjust(3, '0')
df5['sv_peso'] = df5['sv_peso'].str.rjust(3, '0')
df5['sv_altura'] = df5['sv_altura'].str.rjust(3, '0')
df5['sv_spo2'] = df5['sv_spo2'].str.rjust(3, '0')
df5['sv_fio2'] = df5['sv_fio2'].str.rjust(3, '0')
df5['sv_dor'] = df5['sv_dor'].str.rjust(3, '0')
df5['ex_hemograma_basofilos'] = df5['ex_hemograma_basofilos'].str.rjust(3, '0')
df5['ex_hemograma_eosinofilos'] = df5['ex_hemograma_eosinofilos'].str.rjust(3, '0')
df5['ex_hemograma_bastonetes'] = df5['ex_hemograma_bastonetes'].str.rjust(3, '0')
df5['ex_hemograma_segmentados']=df5['ex_hemograma_segmentados'].str.rjust(3, '0')
df5['ex_hemograma_linfocitos']=df5['ex_hemograma_linfocitos'].str.rjust(3, '0')
df5['ex_hemograma_monocitos']=df5['ex_hemograma_monocitos'].str.rjust(3, '0')
```

```
        df5['ex_hemograma_plaquetas']=df5['ex_hemograma_plaquetas'].str.rjust(3,
'0')
        df5['ex_hemograma_ggtn']=df5['ex_hemograma_ggtn'].str.rjust(3, '0')
        df5['ex_hemograma_meta']=df5['ex_hemograma_meta'].str.rjust(3, '0')
        df5['ex_fosforo']=df5['ex_fosforo'].str.rjust(3, '0')
        df5['ex_potassio']=df5['ex_potassio'].str.rjust(3, '0')
        df5['ex_sodio']=df5['ex_sodio'].str.rjust(3, '0')
        df5['ex_glicose']=df5['ex_glicose'].str.rjust(3, '0')
        df5['ex_ureia']=df5['ex_ureia'].str.rjust(3, '0')
        df5['ex_creatinina']=df5['ex_creatinina'].str.rjust(3, '0')
        df5['ex_pcr']=df5['ex_pcr'].str.rjust(3, '0')
        ##
        df5 = df5.drop(df5[df5['sv_frequenciacardiaca'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_frequenciarespiratoria'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_pressaoarterialsistolica'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_pressaoarterialdiastolica'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_glicemia'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_peso'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_altura'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_spo2'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_fio2'] == '00*'].index)
        df5 = df5.drop(df5[df5['sv_dor'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_basofilos'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_eosinofilos'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_bastonetes'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_segmentados'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_linfocitos'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_monocitos'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_plaquetas'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_ggtn'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_hemograma_meta'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_fosforo'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_potassio'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_sodio'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_glicose'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_ureia'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_creatinina'] == '00*'].index)
        df5 = df5.drop(df5[df5['ex_pcr'] == '00*'].index)
        ##
        df5['sv_frequenciacardiaca']                                    =
df5['sv_frequenciacardiaca'].fillna(0).astype(int)
        df5['sv_frequenciarespiratoria']                                =
df5['sv_frequenciarespiratoria'].fillna(0).astype(int)
        df5['sv_pressaoarterialsistolica']                              =
df5['sv_pressaoarterialsistolica'].fillna(0).astype(int)
```

```
        df5['sv_pressaoarterialdiastolica'] =
df5['sv_pressaoarterialdiastolica'].fillna(0).astype(int)
        df5['sv_glicemia'] = df5['sv_glicemia'].fillna(0).astype(int)
        df5['sv_peso'] = df5['sv_peso'].fillna(0).astype(int)
        df5['sv_altura'] = df5['sv_altura'].fillna(0).astype(int)
        df5['sv_spo2'] = df5['sv_spo2'].fillna(0).astype(int)
        df5['sv_fio2'] = df5['sv_fio2'].fillna(0).astype(int)
        df5['sv_dor'] = df5['sv_dor'].fillna(0).astype(int)
        df5['ex_hemograma_basofilos'] =
df5['ex_hemograma_basofilos'].fillna(0).astype(int)
        df5['ex_hemograma_eosinofilos'] =
df5['ex_hemograma_eosinofilos'].fillna(0).astype(int)
        df5['ex_hemograma_bastonetes'] =
df5['ex_hemograma_bastonetes'].fillna(0).astype(int)
        df5['ex_hemograma_segmentados']=df5['ex_hemograma_segmentados'].filln
a(0).astype(int)
        df5['ex_hemograma_linfocitos']=df5['ex_hemograma_linfocitos'].fillna(0).asty
pe(int)
        df5['ex_hemograma_monocitos']=df5['ex_hemograma_monocitos'].fillna(0).a
stype(int)
        df5['ex_hemograma_plaquetas']=df5['ex_hemograma_plaquetas'].fillna(0).ast
ype(int)
        df5['ex_hemograma_meta']=df5['ex_hemograma_meta'].fillna(0).astype(int)
        df5['ex_fosforo']=df5['ex_fosforo'].fillna(0).astype(int)
        df5['ex_potassio']=df5['ex_potassio'].fillna(0).astype(int)
        df5['ex_sodio']=df5['ex_sodio'].fillna(0).astype(int)
        df5['ex_glicose']=df5['ex_glicose'].fillna(0).astype(int)
        df5['ex_ureia']=df5['ex_ureia'].fillna(0).astype(int)
        df5['ex_creatinina']=df5['ex_creatinina'].fillna(0).astype(int)
        df5['ex_pcr']=df5['ex_pcr'].fillna(0).astype(int)
```

APÊNDICE F - Anos de vida paciente

```
df6=df1
df6['anos_de_vida_do_paciente']= df6['dias_de_vida_do_paciente']/365
df6['anos_de_vida_do_paciente'] =
df6['anos_de_vida_do_paciente'].astype(int)

df6 = df6.drop(df6[df6['anos_de_vida_do_paciente'] >= 20].index)
df6 = df6.drop(df6[df6['anos_de_vida_do_paciente'] < 0].index)
```

APÊNDICE G - Ocorrências por Urgência SUS/Convênio

```
df7=df1
df7['unidade_de_internacao']=df7['unidade_de_internacao'].str.replace(' ','')

df7['mask_unidade'] = df7['unidade_de_internacao'].str.contains("URGENCIA")

df7.loc[df7['mask_unidade'] == True, 'atend_urgencia'] = 'Urgencia'
df7 = df7.drop(df7[df7['mask_unidade'] == False].index)

df8 = pd.DataFrame(df7['descricao_cid'].value_counts().reset_index().values, columns=["descricao_cid", "ocorrencias"])
df8_1 = df8.head(20)
```

APÊNDICE H - Ocorrências Totais

```
df08 = pd.DataFrame(df7['descricao_cid'].value_counts().reset_index().values, columns=["descricao_cid", "ocorrencias"])
df08_1 = df08.head(20)

#Alterar o df para ascendente, coincidindo com o gráfico
df08_1=df08_1.sort_values(by='ocorrencias', ascending=True)
#plot
countplt, ax = plt.subplots(figsize = (10,13))
sns.countplot(y='descricao_cid', hue='ocorrencias',data=df08_1, palette="rocket")
```

APÊNDICE I - Mapa Brasil

```
brasil = gpd.read_file("TCC/mapa_brasil/T_LM_MUNICIPIOS_2010Polygon.shp")

df10 = df1[['cidade','uf']]
contagem_brasil = df10.groupby(['cidade']).size().to_frame('total').reset_index()

merge_br = pd.merge(left=brasil,right=contagem_brasil,how='left',left_on='NM_MUNICIP', right_on ='cidade')
#Para poder mostrar o mapa completo (com cidades onde não houve atendimento)
merge_br['total'] = merge_br['total'].fillna(-50000).astype(int)

#Criando o heat map
titulo= 'Atendimentos no Brasil'
col='total'
source='Banco de Dados do Hospital Pequeno Príncipe'
vmin=merge_br[col].min()
vmax=merge_br[col].max()
cmap='summer'

#Criando figura e ax para matplotlib
fig, ax = plt.subplots(1,figsize=(10,10))

#Removendo axis e plotando
ax.axis('off')
merge_br.plot(column=col,ax=ax,edgecolor='0.8',linewidth=0.05,cmap=cmap)

#Título
ax.set_title(titulo, fontdict={'fontsize':'15','fontweight':'3'})

#Criando anotação
ax.annotate(source, xy=(0.61, .15),xycoords='figure fraction', horizontalalignment='left', verticalalignment='bottom',fontsize=10)

#Criando gráfico de cor na legenda
sm = plt.cm.ScalarMappable(norm=plt.Normalize(vmin=vmin,vmax=vmax),cmap=cmap)
```

```
#Esvaziando data array
sm._A = []

#Inserindo gráfico de cor
cbaxes = fig.add_axes([0.05,0.30,0.01,0.4])
cbar = fig.colorbar(sm,cax=cbaxes)
```

APÊNDICE J - Mapa Paraná

```
parana =  gpd.read_file("TCC/mapa_parana/41MUE250GC_SIR.shp")
df9 = df1[['cidade','uf']]
df9 = df9.drop(df9[df9['uf'] != 'PR'].index)
contagem_cidade = df9.groupby(['cidade']).size().to_frame('total').reset_index()

#geopandas -parana
#count - contagem_cidade
merge = pd.merge(left=parana,right=contagem_cidade,how='left',left_on='NM_MUNICIP', right_on ='cidade')

#Criando o heat map
titulo= 'Atendimentos no Paraná'
col='total'
source='Banco de Dados do Hospital Pequeno Príncipe'
vmin=merge[col].min()
vmax=merge[col].max()
cmap='viridis'

#Criando figura e ax para matplotlib
fig, ax = plt.subplots(1,figsize=(15,15))

#Removendo axis e plotando
ax.axis('off')
merge.plot(column=col,ax=ax,edgecolor='0.8',linewidth=1,cmap=cmap)

#Título
ax.set_title(titulo, fontdict={'fontsize':'15','fontweight':'3'})

#Criando anotação
ax.annotate(source, xy=(0.1, .15),xycoords='figure fraction', horizontalalignment='left', verticalalignment='bottom',fontsize=10)

#Criando gráfico de cor na legenda
sm = plt.cm.ScalarMappable(norm=plt.Normalize(vmin=vmin,vmax=vmax),cmap=cmap)

#Esvaziando data array
sm._A = []
```

```
#Inserindo gráfico de cor
cbaxes = fig.add_axes([0.05,0.30,0.01,0.4])
cbar = fig.colorbar(sm,cax=cbaxes)
```

APÊNDICE K – Mapa Curitiba

```
rua_curitiba =  gpd.read_file("TCC/mapa_rua_curitiba/EIXO_RUA.shp")
rua_bairro = gpd.read_file("TCC/mapa_curitiba/DIVISA_DE_BAIRROS.shp")

#Removendo cep em branco/ excluindo o que não for dentro de Curitiba
df13 = df13.dropna(subset=['cd_cep'])
df13 = df13.drop(df13[df13['nm_cidade'] != 'CURITIBA'].index)
df13 = df13.reset_index(drop=True)

#alterando formatação do cep
df13['cd_cep']=df13['cd_cep'].astype(int)
df13['cd_cep']=df13['cd_cep'].astype(dtype=pd.StringDtype())
rua_curitiba['CEP_E']=rua_curitiba['CEP_E'].astype(dtype=pd.StringDtype())
rua_ctba = rua_curitiba[['CEP_E','BAIRRO_E']]

merge_curitiba                                                    =
pd.merge(left=df13,right=rua_ctba,how='left',left_on='cd_cep', right_on ='CEP_E')
merge_curitiba2= merge_curitiba.drop_duplicates(subset='cd_paciente')

contagem_curitiba                                                 =
merge_curitiba2.groupby(['BAIRRO_E']).size().to_frame('total').reset_index()

merge_bairro                                                      =
pd.merge(left=contagem_curitiba,right=rua_bairro,how='left',left_on='BAIRRO_E',
right_on ='NOME')

#Para transformar dataFrame em GeoDataFrame
merge_bairro= GeoDataFrame(merge_bairro)

#Criando o heat map
titulo= 'Atendimentos em Curitiba por Bairro'
col='total'
source='Banco de Dados do Hospital Pequeno Príncipe'
vmin=merge_bairro[col].min()
vmax=merge_bairro[col].max()
cmap = 'Wistia'

#Criando figura e ax para matplotlib
fig, ax = plt.subplots(1,figsize=(10,10))

#Removendo axis e plotando
ax.axis('off')
```

```
        merge_bairro.plot(column=col,ax=ax,edgecolor='0.8',linewidth=0.5,cmap=cm
ap)

        #Título
        ax.set_title(titulo, fontdict={'fontsize':'15','fontweight':'3'})

        #Criando anotação
        ax.annotate(source,          xy=(0.1,          .07),xycoords='figure          fraction',
horizontalalignment='left', verticalalignment='bottom',fontsize=10)

        #Criando gráfico de cor na legenda
        sm                                                                                  =
plt.cm.ScalarMappable(norm=plt.Normalize(vmin=vmin,vmax=vmax),cmap=cmap)

        #Esvaziando data array
        sm._A = []


        #Inserindo gráfico de cor
        cbaxes = fig.add_axes([0.05,0.30,0.01,0.4])
        cbar = fig.colorbar(sm,cax=cbaxes)
```