ADRIANO DOMINGOS GOULART

DESENVOLVIMENTO DE APLICAÇÕES COM USO DE CIÊNCIA DE DADOS:

ESTÁGIO AGÊNCIA ZETTA

LAVRAS – MG

2021

ADRIANO DOMINGOS GOULART

DESENVOLVIMENTO DE APLICAÇÕES COM USO DE CIÊNCIA DE DADOS:

ESTÁGIO AGÊNCIA ZETTA

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Ciência da Computação, para a obtenção do título de Bacharel.

Prof. Dr. Erick Galani Maziero

Orientador

LAVRAS – MG

2021

ADRIANO DOMINGOS GOULART

DESENVOLVIMENTO DE APLICAÇÕES COM USO DE CIÊNCIA DE DADOS: ESTÁGIO AGÊNCIA ZETTA

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Ciência da Computação, para a obtenção do título de Bacharel.

APROVADA em 18 de Novembro de 2021.

Prof. DSc. Erick Galani Maziero	UFLA
Prof. DSc. Paula Christina Figueira Cardoso	UFLA
Everton Leonardo de Almeida	ZETTA

Prof. Dr. Erick Galani Maziero
Orientador

LAVRAS – MG
2021

RESUMO

O presente documento descreve as atividades desenvolvidas durante o estágio supervisionado na área de Ciência de Dados na Agência de Inovação Zetta. Durante o estágio, o aluno teve a oportunidade de utilizar e expandir os conhecimentos adquiridos durante a graduação em Ciência da Computação. O estágio proporcionou a evolução tanto na área da Estatística quanto em análise, visualização e manipulação de dados possibilitando ao estudante dar início a sua carreira profissional. No decorrer deste documento, serão apresentadas as ferramentas, metodologias e tecnologias aprendidas no estágio. A principal ferramenta utilizada foi a linguagem de programação Python com suas bibliotecas e ferramentas auxiliares como Pandas, Pandas Profiling, Matplotlib, NetworkX e Jupyter. Ao final desta monografia, são realizadas considerações sobre as contribuições de diversas disciplinas do curso para as atividades do estudante.

Palavras-chave: Python, Jupyter, Pandas, Pandas Profiling, Matplotlib, NetworkX, PostgreSQL, Git, Ciência de Dados.

ABSTRACT

This document describes the activities developed during the supervised internship in the Data Science area at Zetta Inovation Agency. During the internship, the student had the opportunity to use and expand the knowledge acquired during the graduation in Computer Science. The internship provided the evolution both in the area of Statistics and in data analysis, visualization and manipulation, enabling the student to start his professional career. Throughout this document, the tools, methodologies and technologies learned in the internship will be presented. The main tool used was the Python programming language with its libraries and auxiliary tools such as Pandas, Pandas Profiling, Matplotlib, NetworkX and Jupyter. At the end of this monograph, considerations are made about the contributions of the various disciplines of the course to the student's activities.

Keywords: Python, Pandas, Matplotlib, Seaborn, Data Science.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Este capítulo tem o objetivo de realizar uma breve apresentação do assunto tratado neste trabalho. Durante esta apresentação, será abordada uma introdução da Agência Zetta, o objetivo do projeto desenvolvido durante o estágio, a atuação do estagiário dentro do projeto e, ao final, uma visão geral dos próximos capítulos deste documento.

1.1 Contextualização

A Tecnologia da Informação mudou o mundo e continua a influenciar cada dia mais a humanidade. Com o início da Era Tecnológica no século XX a tecnologia se tornou presente em muitas novas áreas, desde entretenimento a exploração espacial. A partir do século XXI com novos grandes avanços tecnológicos e aprimoramento de tecnologias desenvolvidas no século passado, como principalmente a Internet, iniciava-se uma nova fase, a dos dados.

Na icônica frase do matemático londrino Clive Humby *"Data is the new oil"* (The Guardian, 2013), ou em português “Os dados são o novo petróleo”, se expressa uma nova realidade, em que os dados gerados pela humanidade tem grande valor, não apenas científico mas também monetário. A partir daí a Ciência de Dados entra com o propósito de transformar dados em informação para que seu valor seja maximizado.

O grande crescimento da Ciência de Dados em conjunto com o Aprendizado de Máquina, a possibilidade de integração com as diversas áreas do conhecimento, além da busca por conhecimento e aprimoramento pessoal e desafios, incentivaram o estagiário a iniciar um estágio supervisionado na primeira equipe de Ciência de Dados da Agência Zetta. Estágio esse que proporcionou ao aluno conhecimento e experiências interpessoal e profissional.

1.1.1 A Agência Zetta

A Zetta, ou Agência UFLA de inovação, geotecnologia e sistemas inteligentes atua principalmente em desenvolvimento de software, buscando soluções que promovam a sustentabilidade e contribuam para a formação humana e intelectual, missão primeira da Universidade Federal de Lavras.

Segundo (EMBRAPII, 2021) "A Zetta/UFLA é credenciada na área de Agricultura Digital, com ênfase nas competências de inovação tecnológica, geotecnologias, ciência de dados e implementação de sistemas inteligentes. Como Unidade Embrapii, poderá executar projetos de Pesquisa, Desenvolvimento e Inovação junto às empresas do Agronegócio Brasileiro, contribuindo ainda mais para o fortalecimento da indústria, gerando maior competitividade nacional e internacional."

A Zetta nasceu em Abril de 2020 a partir do antigo LEMAF - Laboratório de Estudos e Projetos em Manejo Florestal, herdando suas experiências multidisciplinares além do conhecimento acumulado do desenvolvimento de centenas de soluções para instituições públicas e privadas.

A Zetta, a partir de sua criação, deu início a construção de uma nova equipe para abranger um novo nicho do conhecimento, a equipe de Ciência de Dados. A equipe começou com quatro pessoas e um professor, com os desafios de montar uma infraestrutura própria com dados atualizados e criar os processos necessários para o desenvolvimento de projetos com eficiência e qualidade.

1.1.2 Objetivos

Este trabalho tem o objetivo de descrever a trajetória do estagiário na Zetta, no período de Janeiro/2021 a Maio/2021. Os projetos desenvolvidos no período foram para um órgão público de fiscalização que será referenciado com a sigla OPF. O estagiário atuou nos seguintes projetos: Estatísticas Descritivas de Dados do OPF e Tipologias Relacionadas à Lavagem de Dinheiro.

1.1.2.1 O projeto: Estatísticas Descritivas de Dados do OPF

O projeto desenvolvido teve a finalidade de gerar estatísticas descritivas em diversas tabelas da base de dados relacionais do OPF. As estatística tiveram como objetivo explicitar e tornar mais claro aspectos importantes dos dados com a finalidade de obter um melhor conhecimento das potencialidades e limitações dos dados à disposição da equipe, assim facilitando futuras demandas tecnológicas. Como o volume de dados é muito grande, esse tipo de análise de forma manual é impraticável, logo todas as estatísticas foram executadas de forma automatizada a partir de bibliotecas da linguagem Python.

1.1.2.2 O projeto: Tipologias Relacionadas à Lavagem de Dinheiro

O objetivo do projeto foi estudar a viabilidade de identificação e automatização de duas tipologias relacionadas à lavagem de dinheiro: *"Smurfing"* e "Comunicação entre Investigados com Localidades Próximas". Estas tipologias refletem padrões que indicam comportamentos associados a crimes, estes analisados diariamente pela equipe do OPF. Obtendo-se sucesso com as detecções dos padrões, a versão inicial do código será utilizada posteriormente na implementação no projeto de um sistema de inteligência com análise, visualização de dados e detecção de padrões que tem como finalidade facilitar as investigações.

1.2 Estrutura do trabalho

Na estrutura deste trabalho, o Capítulo 2 apresenta o referencial teórico, visando a apresentação das tecnologias e metodologias aprendidas. o Capítulo 3 descreve as atividades desenvolvidas pelo estagiário. Por fim, Capítulo 4 apresenta as considerações finais deste relatório de estágio.

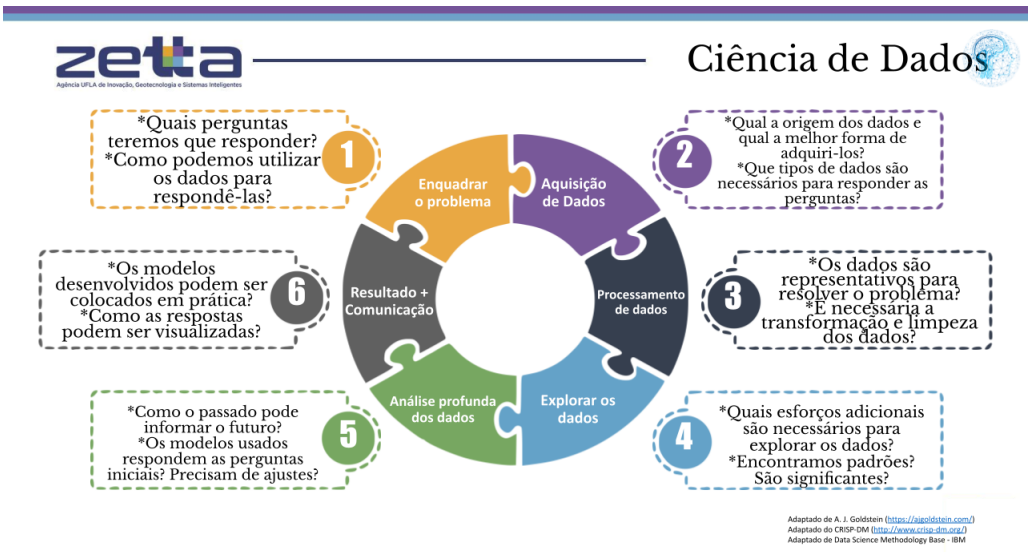
2 REFERENCIAL TEÓRICO

Este capítulo apresenta as principais bibliotecas, metodologias e tecnologias de desenvolvimento utilizadas.

2.1 Metodologia de Desenvolvimento

No intuito de reunir e organizar as atividades da equipe em um mesmo projeto, é necessário adotar formas de organizar, dividir e comunicar o andamento de cada atividade do projeto. Desta forma, nesta seção será apresentada a principal metodologia de desenvolvimento adotada no estágio.

Figura 2.1 – Ciclo da Ciência de Dados Zetta



Fonte: Zetta, 2021

A Figura 2.1 é a representação do ciclo de Ciência de Dados utilizado na Zetta. Ele foi construído a partir da adaptação de diversas fontes e abaixo será abordada cada parte.

1- Enquadrar o Problema:

- **Quais perguntas teremos que responder?**
  Etapa com a finalidade de dividir o problema abordado em partes, assim facilitando o entendimento e definindo o escopo do projeto.

- **Como podemos utilizar os dados para respondê-las?**
  Visualização global dos dados disponíveis e de dados necessários a serem obtidos de acordo com as perguntas do projeto.

2- Aquisição de Dados:

- **Qual a origem dos dados e qual a melhor forma de adquiri-los?**
  Mapeamento para saber se os dados já estão disponíveis ou se são passíveis de obtenção, quais as melhores e mais eficientes fontes para obtê-los.

- **Que tipo de dados são necessários para responder as perguntas?**
  Verificação para saber se os dados serão tabulares, espaciais, estruturados, não estruturados dentre outras formas.

3- Processamento de Dados:

- **Os dados são representativos para resolver o problema?**
  Verificação para estimar se as informações necessárias para resolver o problema estão contidas nos dados.

- **É necessária a transformação e limpeza dos dados?**
  Verificação se os dados já estão prontos ou se é necessário algum tipo de pré-processamento.

4- Explorar os Dados:

- **Quais esforços adicionais são necessários para explorar os dados?**
  Análise para saber se basta utilizar estatísticas nos dados, algoritmos simples ou se cabe a utilização de Aprendizagem de Máquina.

- **Encontramos padrões? São significantes?**
  Exploração para descobrir se os dados apresentam padrões interessantes de serem utilizados no projeto.

5- Análise Profunda dos Dados:

- **Como o passado pode informar o futuro?**
  A partir dos dados é feita a verificação e testes para observar a possibilidade de algum tipo de predição.
- **Os modelos usados respondem às perguntas iniciais? Precisam de ajustes?**
  Testes para saber a qualidade dos algoritmos desenvolvidos.

6- Resultado + Comunicação:

- **Os modelos desenvolvidos podem ser colocados em prática?**
  A partir de todo o estudo e testes nos algoritmos descobre-se a viabilidade dos mesmos.
- **Como as respostas podem ser visualizadas?**
  Trabalha-se na melhor forma de visualizar os resultados do projeto, mantendo uma visualização simples, direta e de fácil entendimento.

2.2 Bibliotecas e Tecnologias utilizadas

Existem uma infinidade de projetos construídos em Python. Com isso também existem várias bibliotecas e *frameworks* que auxiliam e facilitam o desenvolvimento de novos projetos. As comunidades de desenvolvimento em Python já desenvolveram diversas bibliotecas que estão disponíveis para qualquer usuário da linguagem. O principal repositório que agrega muitas delas é o *Python Package Index* (PyPI) em que os autores distribuem gratuitamente seus softwares (PyPI,

2021). Desta forma, esta seção descreve as principais bibliotecas e tecnologias aprendidas e utilizadas no desenvolvimento dos projetos durante o estágio.

2.2.1 Python

Python é uma linguagem de programação interpretada, interativa e orientada a objetos que incorpora módulos, exceções, tipagem dinâmica, tipos de dados dinâmicos de alto nível e classes. Ela foi criada por Guido van Rossum e teve sua primeira versão disponibilizada no ano de 1991 (Docs Python, 2021).

A linguagem é de código aberto, ou seja, qualquer pessoa tem acesso ao código fonte e pode modificá-lo. Essa dentre outras características estimulam a linguagem a ser amplamente utilizada e com uma grande comunidade de desenvolvedores empenhados em ajudar o projeto desenvolvendo novas ferramentas compatíveis com a linguagem. A comunidade de desenvolvedores faz com que ela seja facilmente utilizada em diversas áreas como Ciência de Dados, aplicações WEB, desenvolvimento *Back-End*, criação de jogos, segurança da informação dentre outras.

Em Junho de 2021, Python alcançou a segunda colocação no Índice TIOBE que é um indicador da popularidade das linguagens de programação, ficando atrás da primeira colocada C por apenas 0.7% (TIOBE Index, 2021). Somente o repositório de software da comunidade PyPI conta com mais de 313 mil projetos e mais de 519 mil usuários (PyPI, 2021).

A Figura 2.2 exemplifica um trecho de código Python de uma repetição iterando de 0 a 2 utilizando a função *range()*, com condicionais que verificam se a variável x que recebe o valor do iterador é igual a 0, a 1 ou diferente de 0 e 1. Para cada condicional usa-se a função de impressão *print()* para imprimir uma mensagem.

Figura 2.2 – Exemplo de código Python

```
>>> for i in range(3):
...     x = i
...     if x == 0:
...             print('x é igual a 0')
...     elif x == 1:
...             print('x é igual a 1')
...     else:
...             print('x não é igual a 0 nem igual a 1')
...
x é igual a 0
x é igual a 1
x não é igual a 0 nem igual a 1
>>>
```

Fonte: Do Autor, 2021

2.2.2 Jupyter

O Jupyter ou Projeto Jupyter é um projeto de código aberto sem fins lucrativos, nascido do Projeto IPython em 2014, à medida que evoluiu para dar suporte à Ciência de Dados interativa e à Computação Científica em todas as linguagens de programação (Project Jupyter, 2021).

Dentro do projeto existe o Jupyter Notebook que é um ambiente de computação interativo que permite aos usuários criar documentos que incluem: Código, *Widgets* interativos, Gráficos, Texto, Equações, Imagens e Vídeos (Jupyter nbviewer, 2021).

Esses documentos fornecem um registro completo e independente de programas que pode ser convertido em vários formatos. O Notebook Jupyter combina três componentes: A Aplicação WEB para os documentos, *Kernels* e os Documentos (Jupyter nbviewer, 2021).

2.2.2.1 Aplicação WEB

A aplicação WEB tem como principal objetivo permitir o usuário (Jupyter nbviewer, 2021):

- Editar o código no navegador, com destaque de sintaxe.

- Executar o código no navegador, com os resultados das execuções anexados ao código que os gerou.

- Ver os resultados das execuções com vários tipos de representações como HTML, LaTeX, PNG, SVG, PDF, etc.

- Criar e usar *widgets* JavaScript interativos, que vinculam controles e visualizações da interface do usuário a cálculos do *kernel*.

- Escrever textos usando a linguagem de Markdown.

- Incluir equações matemáticas usando a sintaxe de LaTeX no Markdown, que são renderizadas no navegador.

2.2.2.2 Kernels

Por meio do *kernel* do Jupyter e da arquitetura de mensagens, o Jupyter Notebook permite executar código em uma variedade de linguagens de programação diferentes. Para cada documento que o usuário abre, a aplicação WEB inicia um *kernel* que executa o código para esse documento. Cada *kernel* é capaz de executar código em uma única linguagem de programação, porém, existem *kernels* disponíveis nas seguintes linguagens: Python, Julia, R, Ruby, Haskell, Scala, node.js e Go (Jupyter nbviewer, 2021).

2.2.2.3 Documentos

Os documentos contêm as entradas e saídas da execução do código da linguagem de programação, e também o texto que acompanha o código porém não é executável. A saída gerada pela execução de código, incluindo HTML, imagens, vídeos e gráficos, é incorporada ao documento, o que o torna um registro único e conciso (Jupyter nbviewer, 2021).

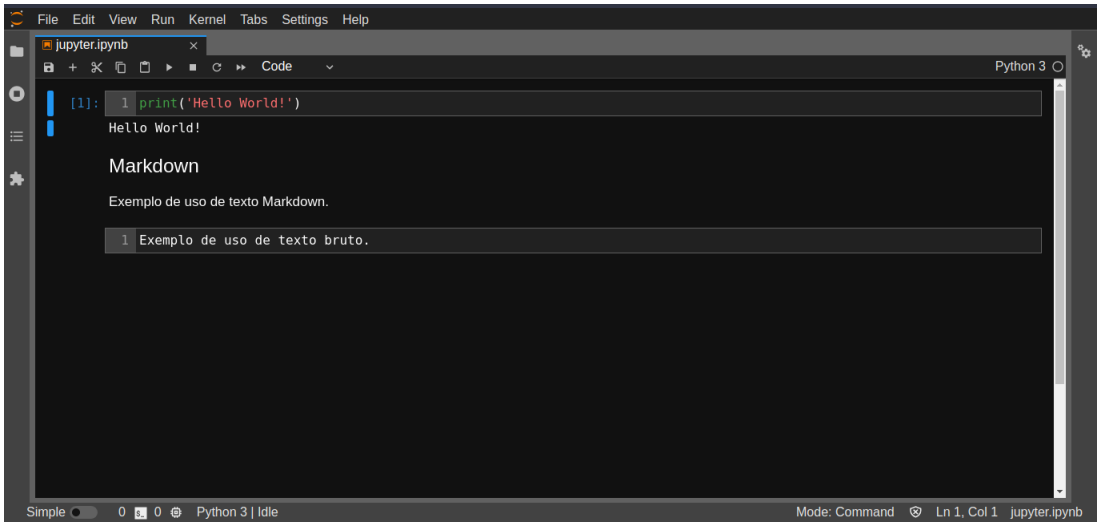
Quando executados e salvos, gera-se um arquivo com extensão .ipynb no sistema de arquivos local. Os documentos consistem em uma sequência linear de células que podem ser divididas em três tipos (Jupyter nbviewer, 2021):

- Células de código: entrada e saída de código que é executado no *kernel*.
- Células Markdown: texto com equações LaTeX incorporadas.
- Células brutas: texto não formatado que é incluído, sem modificação.

Por estes e outros fatores a aplicação se mostra excelente para o uso relacionado a Ciência de Dados, uma vez que em um único arquivo é possível agregar informações em diversos formatos, de maneira fácil e rápida.

A Figura 2.3 exemplifica um documento Jupyter com uma célula de código Python, uma de texto em Markdown e uma em formato bruto, todas já executadas.

Figura 2.3 – Exemplo de Documento Jupyter



Fonte: Do Autor, 2021

2.2.3 Pandas

O Pandas é uma ferramenta de código aberto, de alto nível para análise de dados. No ano de 2008, o desenvolvimento do Pandas começou na empresa AQR

Capital Management, ao final de 2009, o projeto passou a ser de código aberto e começou a ser apoiado por uma comunidade de indivíduos com ideias semelhantes em todo o mundo, que contribuem para ajudar a tornar os Pandas cada vez melhor (Pandas, 2021).

Como uma biblioteca de análise de dados, o Pandas é capaz de processar e visualizar dados de diversas formas. Para isso é necessário o uso de estrutura de dados, sendo as duas principais chamadas de *Series* e *DataFrame*.

A *Series* é uma matriz rotulada unidimensional capaz de conter qualquer tipo de dados como inteiros, *strings*, números de ponto flutuante, objetos Python, etc. Os rótulos dos eixos são chamados coletivamente de índice (Series, 2021).

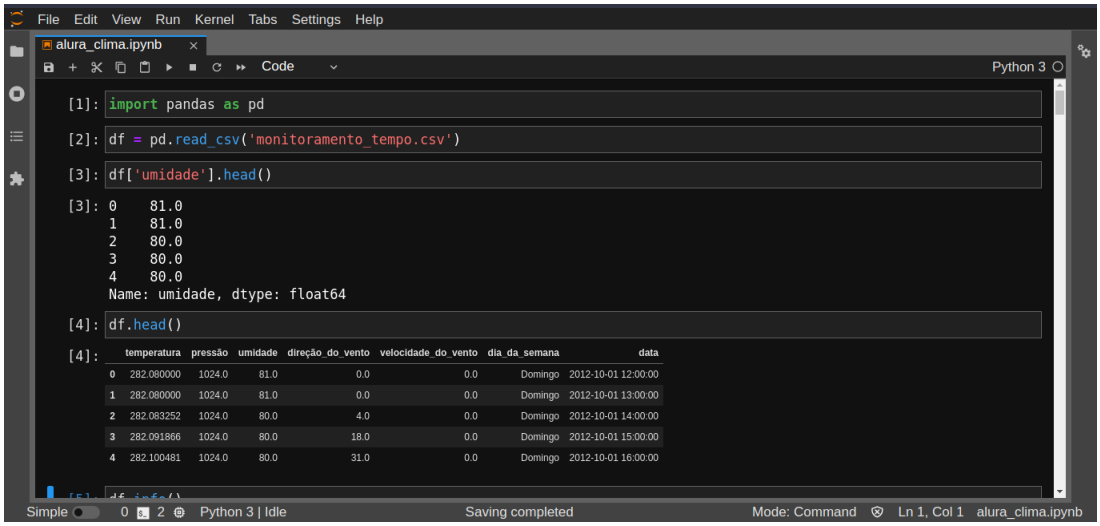
O *DataFrame* é uma estrutura de dados rotulada bidimensional com colunas de tipos potencialmente diferentes. Sua estrutura se assemelha a uma planilha, tabela SQL, ou um dicionário de objetos *Series*. Geralmente é o objeto Pandas mais utilizado (DataFrame, 2021).

Algumas das principais funcionalidades da biblioteca são (Pandas, 2021):

- Objeto tabular rápido e eficiente para manipulação de dados com indexação integrada.
- Ferramentas para ler e gravar dados entre estruturas de dados em memória e diferentes formatos: CSV, arquivos de texto, Microsoft Excel, bancos de dados SQL e HDF5.
- Remodelagem e rotação de conjuntos de dados.
- Fatiamento de dados, indexação avançada e divisão em subconjuntos de grandes conjuntos de dados.
- Inserção e exclusão de estruturas de dados para mutabilidade de tamanho.
- Agregar ou transformar dados com operações de “agrupar por” eficientes, permitindo operações dividir-aplicar-combinar em conjuntos de dados.

- Mesclagem e junção de conjuntos de dados em alto desempenho.
- Funcionalidade de série temporal: geração de intervalo de datas, conversão de frequência, médias móveis, conversão de datas, etc.

Figura 2.4 – Exemplo de uso do Pandas



Fonte: Do Autor, 2021

A Figura 2.4 exemplifica o uso da biblioteca Pandas. Na primeira célula é feita a importação, na segunda é feito o carregamento dos dados "monitoramento_tempo.csv", na terceira exibe-se as cinco primeiras linhas da coluna "umidade" representada por uma *Series* e na quarta célula exibe-se as cinco primeiras linhas dos dados representados por um *DataFrame*.

2.2.4 Matplotlib

Matplotlib é uma biblioteca de código aberto para criar gráficos 2D de vetores em Python. Foi criada em 2008 pelo neurobiólogo americano John D. Hunter, quando durante os seus trabalhos com dados de eletroencefalograma (EEG), usando a ferramenta MATLAB encontrou limitações perante estruturas de dados complexas, a partir daí John decidiu migrar para Python. Depois disso, diante da

dificuldade em encontrar um pacote de plotagem 2D ele decidiu criar sua própria biblioteca inspirada nas capacidades de plotagem do MATLAB (HUNTER, 2007).

Embora Matplotlib seja escrita principalmente em Python puro, ela faz uso intenso da biblioteca NumPy e de outros códigos de extensão para atingir um bom desempenho, mesmo para matrizes grandes. A biblioteca, conceitualmente, pode ser dividida em três partes (Matplotlib, 2021):

1. Interface Pylab que é o conjunto de funções fornecidas pelo módulo Pylab que permitem ao usuário criar gráficos com código bastante semelhante ao código de geração de figura MATLAB.

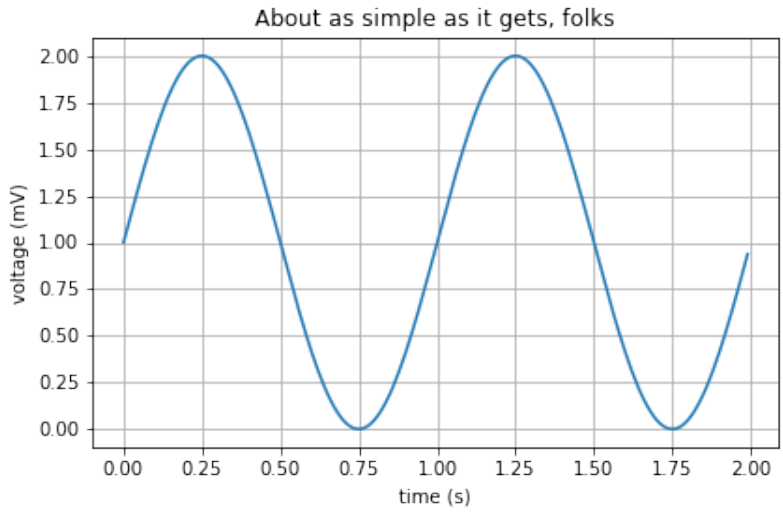
2. O *frontend* Matplotlib ou API Matplotlib que é o conjunto de classes que fazem o trabalho pesado, criando e gerenciando figuras, texto, linhas, gráficos. É uma interface abstrata que não sabe nada sobre saída.

3. Os *back-ends* são dispositivos de desenho, ou renderizadores, que transformam a representação do *front-end* em saída para impressão ou exibição como por exemplo gráficos em formato PostScript, SVG, PNG, etc.

A biblioteca é capaz de gerar uma imensa variedade de gráficos, por exemplo: Gráfico de Linhas, Histogramas, Gráficos 3D, Gráfico de Barras, Gráfico de Pizza e Gráfico de Pontos. A comunidade de desenvolvimento que contribui para o projeto da biblioteca segue constantemente criando novas formas de visualização, implementando correções e novas opções de customização dos gráficos. Devido a estes e outros fatores a biblioteca é uma das principais ferramentas de visualização para a linguagem Python.

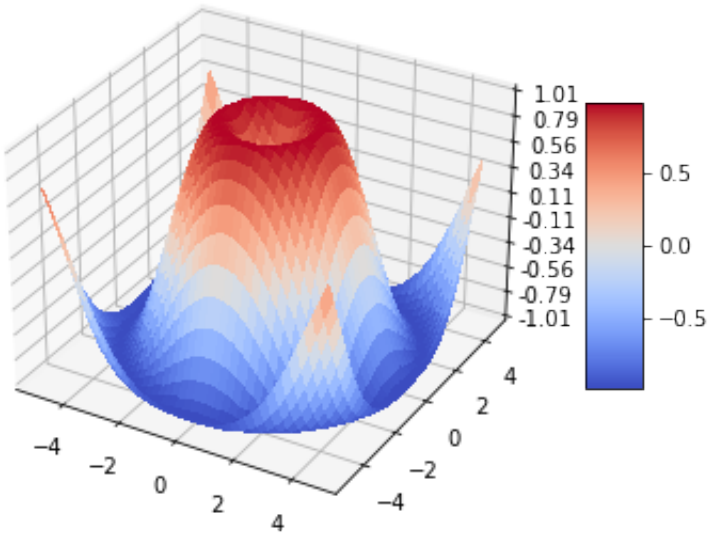
As Figuras 2.5 e 2.6 exemplificam gráficos gerados com o uso da biblioteca Matplotlib.

Figura 2.5 – Exemplo de Gráfico de Linhas



Fonte: matplotlib.org, 2021

Figura 2.6 – Exemplo de Gráfico 3D



Fonte: matplotlib.org, 2021

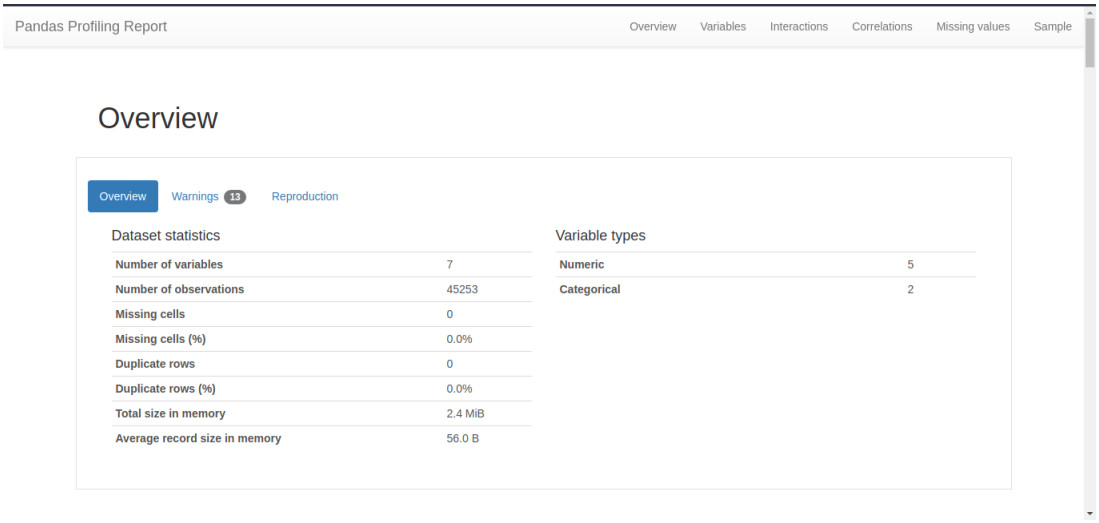
2.2.5 Pandas Profiling

Pandas Profiling é uma biblioteca de código aberto com a função de gerar relatórios automáticos a partir de um *DataFrame* do Pandas. Sua primeira versão foi lançada no GitHub em Janeiro de 2016 pelo belga Jos Polfliet.

A biblioteca é muito utilizada para análise exploratória de dados, pois com poucas linhas de código é possível obter várias estatísticas e informações relevantes sobre os dados abordados. Para cada coluna do *DataFrame* analisado são calculadas as seguintes operações (Pandas Profiling, 2021):

- Inferência de tipo: detecta os tipos de colunas em um *DataFrame*.
- Fundamentos: tipo, valores únicos, valores ausentes.
- Estatísticas de quantis como valor mínimo, Q1, mediana, Q3, máximo, intervalo, intervalo interquartil.
- Estatísticas descritivas como média, moda, desvio padrão, soma, desvio absoluto mediano, coeficiente de variação, curtose, assimetria.
- Valores mais frequentes.
- Histogramas.
- Destaque de correlações de variáveis altamente correlacionadas, matrizes de Spearman, Pearson e Kendall.
- Matriz de valores ausentes, contagem, mapa de calor e dendrograma de valores ausentes.
- Linhas duplicadas e listagem das linhas duplicadas com maior ocorrência.
- A análise de texto aprende sobre categorias (maiúsculas, espaço), *scripts* (Latim, Cirílico) e blocos (ASCII) de texto.

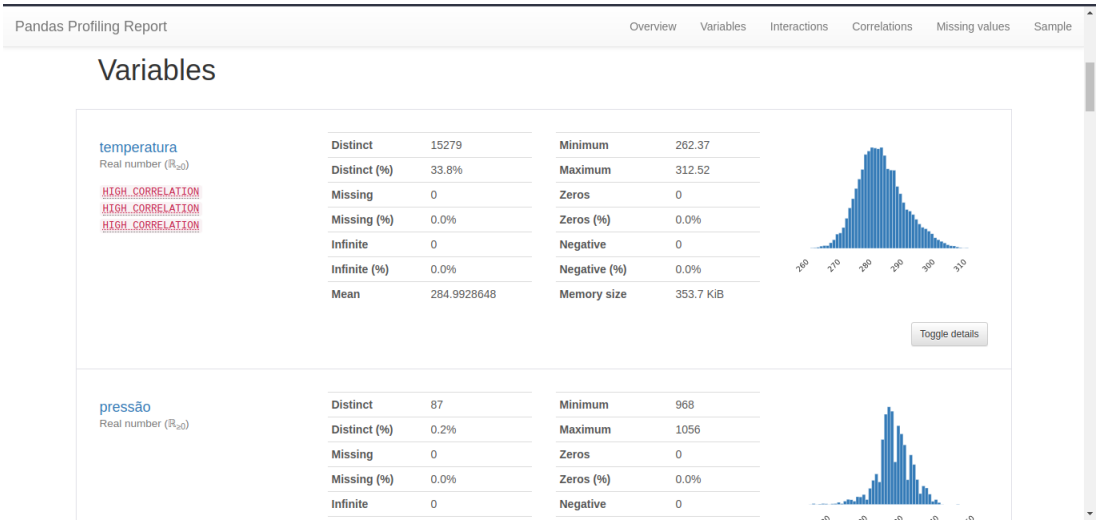
Figura 2.7 – Exemplo de página gerada pelo Pandas Profiling



Fonte: Do Autor, 2021

A Figura 2.7 exemplifica o resultado do uso da biblioteca Pandas Profiling. A imagem é referente a página de Visão Geral, que expõe estatísticas das colunas do *DataFrame* como um todo.

Figura 2.8 – Exemplo de página gerada pelo Pandas Profiling



Fonte: Do Autor, 2021

A Figura 2.8 exemplifica o resultado do uso da biblioteca Pandas Profiling. A imagem é referente a página de Variáveis, que expõe estatísticas de todas as colunas do *DataFrame*. A imagem mostra em específico as colunas “temperatura” e “pressão”.

2.2.6 NetworkX

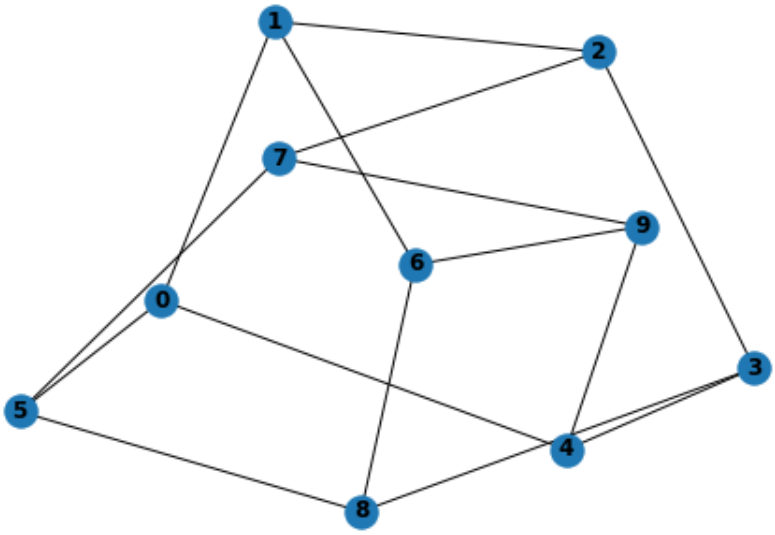
O NetworkX é uma biblioteca em Python, de código aberto, para a criação, manipulação e estudo da estrutura, dinâmica e funções de redes complexas, originalmente escrito por Aric Hagberg, Dan Schult e Pieter Swart, e desenvolvido com a ajuda de muitos outros, tendo sua primeira versão lançada em 2014 no GitHub (HAGBERG; SCHULT; SWART, 2008).

As principais características do NetworkX são (NetworkX, 2021):

1. Criação de estruturas de dados para grafos, digrafos e multigrafos.
2. Possibilidade de utilização de muitos algoritmos de grafo padrão.
3. Estrutura de rede e medidas de análise.
4. Geradores para grafos clássicos, grafos aleatórios e redes sintéticas.
5. Os nós podem ser criados a partir de várias fontes, por exemplo: texto, imagens, registros XML.
6. As arestas podem conter dados arbitrários como pesos ou séries temporais.
7. Bem testado com mais de 90% de cobertura de código.

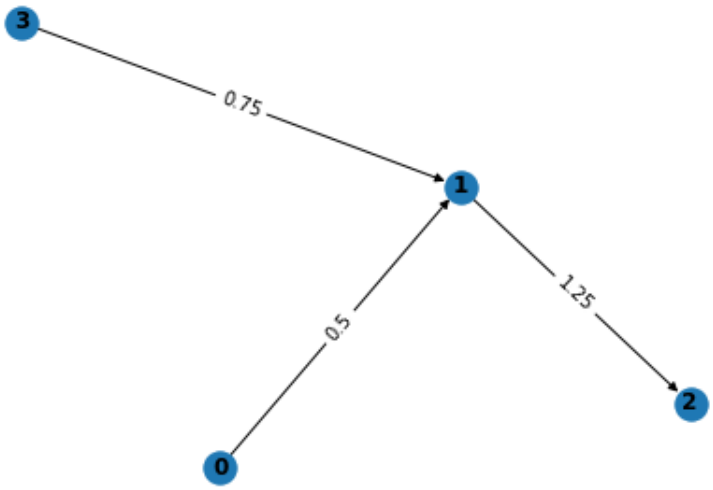
As figuras exemplificam o resultado do uso da biblioteca NetworkX. A Figura 2.9 é um ao grafo gerado aleatoriamente pela função *petersen_graph* e a Figura 2.10 é um digrafo com peso nas arestas.

Figura 2.9 – Exemplo de grafo gerado pelo NetworkX



Fonte: Do Autor, 2021

Figura 2.10 – Exemplo de digrafo gerado pelo NetworkX



Fonte: Do Autor, 2021

2.2.7 PostgreSQL

O sistema de gerenciamento de banco de dados objeto-relacional de código aberto agora conhecido como PostgreSQL é derivado do pacote POSTGRES

escrito na Universidade da Califórnia em Berkeley. Com mais de duas décadas de desenvolvimento, o PostgreSQL é agora o banco de dados de código aberto mais avançado disponível em qualquer lugar (PostgreSQL Org, 2021).

O projeto POSTGRES, liderado pelo Professor Michael Stonebraker, foi patrocinado pela Agência de Projetos de Pesquisa Avançada de Defesa (DARPA), o Gabinete de Pesquisa do Exército (ARO), a Fundação Nacional de Ciência (NSF) e ESL, Inc. A implementação do POSTGRES começou em 1986 (PostgreSQL Org, 2021).

O PostgreSQL é um banco de dados relacional cujo armazenamento é feito em tabelas, assim a única limitação de espaço é a capacidade de armazenamento do hardware. Ele suporta uma grande parte do padrão SQL além de oferecer muitos recursos modernos. Alguns dos recursos da ferramenta são (PostgreSQL Org, 2021):

- Consultas complexas
- Chaves estrangeiras
- Gatilhos
- Visualizações atualizáveis
- Integridade transacional
- Controle de simultaneidade multiversão
- Tipos de dados
- Funções
- Operadores
- Funções agregadas
- Métodos de índice

- Linguagens procedurais

Figura 2.11 – Exemplo de consulta do PostgreSQL

```
SELECT
   first_name,
   last_name,
   email
FROM
   customer;
```

	first_name character varying (45)	last_name character varying (45)	email character varying (50)
1	Jared	Ely	jared.ely@sakilacustomer.org
2	Mary	Smith	mary.smith@sakilacustomer.org
3	Patricia	Johnson	patricia.johnson@sakilacustomer.org
4	Linda	Williams	linda.williams@sakilacustomer.org
5	Barbara	Jones	barbara.jones@sakilacustomer.org
6	Elizabeth	Brown	elizabeth.brown@sakilacustomer.org
7	Jennifer	Davis	jennifer.davis@sakilacustomer.org
8	Maria	Miller	maria.miller@sakilacustomer.org
9	Susan	Wilson	susan.wilson@sakilacustomer.org
10	Margaret	Moore	margaret.moore@sakilacustomer.org
11	Dorothy	Taylor	dorothy.taylor@sakilacustomer.org

Fonte: postgresqltutorial.com, 2021

A Figura 2.11 exemplifica a consulta "SELECT" de uma tabela no PostgreSQL. Nesse exemplo são selecionadas da tabela *"customer"* as colunas *"first_name"*, *"last_name"* e *"email"*.

2.2.8 Tecnologia de controle de versão: Git

O controle de versão é um sistema que registra alterações em um arquivo ou conjunto de arquivos ao longo do tempo para que seja possível recuperar versões específicas posteriormente. Um Sistema de Controle de Versão (VCS) permite reverter os arquivos selecionados para um estado anterior, reverter todo o projeto para um estado anterior, comparar as alterações ao longo do tempo, ver quem modificou pela última vez algo que pode estar causando um problema, quem

introduziu um problema e quando, e muito mais. Usar um VCS geralmente significa que, se algum problema ocorrer ou acontecer perda de arquivos, será possível recuperar facilmente (CHACON; STRAUB, 2014).

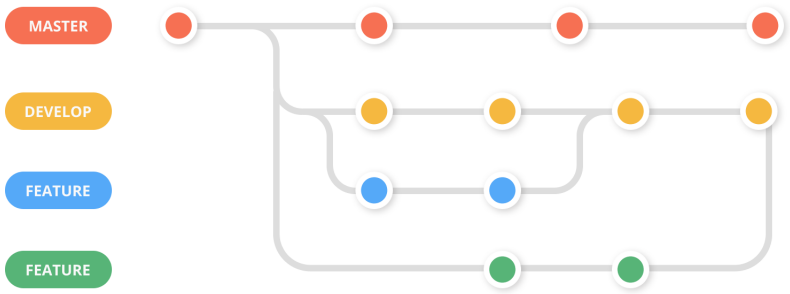
O Git nasceu em 2005, a partir da comunidade de desenvolvimento do Linux (e em particular Linus Torvalds, o criador do Linux) que buscava um VCS com atributos como: velocidade, design simples, suporte para desenvolvimento não linear (milhares de ramificações paralelas), totalmente distribuído e capaz de lidar com grandes projetos como o *kernel* do Linux de forma eficiente (velocidade e tamanho dos dados). A partir daí desenvolveu-se uma ferramenta de código aberto, incrivelmente rápida, muito eficiente com grandes projetos e incrível para desenvolvimento não linear (CHACON; STRAUB, 2014).

O funcionamento geral básico do Git pode ser entendido como (Atlassian, 2021):

1. Cria-se um repositório com uma ferramenta de hospedagem de Git (GitLab, GitHub, Bitbucket, etc).
2. Copia-se o repositório na máquina local.
3. Adiciona-se o arquivo ao repositório local e salva-se (*commit*) as alterações.
4. Adiciona (*push*) as alterações na sua ramificação principal.
5. Faz-se uma alteração no arquivo com uma ferramenta de hospedagem de Git e salva-se (*commit*).
6. Obtém-se (*pull*) as alterações para a máquina local.
7. Cria-se uma versão (*branch*), faz-se uma alteração, salva-se (*commit*) a alteração.
8. Abre-se uma proposta de alterações na ramificação principal (*pull request*).

9. Mescla-se (*merge*) a ramificação atual com a principal.

Figura 2.12 – Exemplo de *branches* Git



Fonte: https://zepel.io/blog/how-to-create-a-new-branch-in-github/, 2021

A Figura 2.12 ilustra quatro *branches* de um *software* em desenvolvimento. Primeiro temos a *branch master*, esta é dividida em uma *branch develop* e uma *feature*. A *branch develop* em seu estado inicial gera uma *branch feature*, depois de duas modificações é feito a mesclagem (*merge*) na *branch develop*. A *branch feature* criada a partir da *master* depois de duas modificações é feito a mesclagem (*merge*) na *branch develop*. Posteriormente as *branches* poderão ser mescladas (*merge*) na *master* que passará a ter todas as modificações das *branches* criadas a partir dela.

Neste capítulo, foram apresentadas as principais tecnologias utilizadas durante o desenvolvimento de dois projetos nos quais o estagiário atuou. Desde a linguagem de programação Python, ambiente de desenvolvimento Jupyter, bibliotecas de desenvolvimento como Pandas, Matplotlib, Pandas Profiling, NetworkX, sistema de gerenciamento de banco de dados PostgreSQL e tecnologia de controle de versão Git. Todas em conjunto foram de grande utilidade para os projetos desenvolvidos durante o estágio.

3 ATIVIDADES DESENVOLVIDAS

Neste capítulo, serão descritas as atividades desenvolvidas durante o estágio, lições aprendidas e dificuldades encontradas. As atividades são divididas em dois projetos, sendo eles o de Estatísticas Descritivas de Dados da OPF e Tipologias Relacionadas à Lavagem de Dinheiro.

3.1 Estatísticas Descritivas de Dados do OPF

O objetivo deste projeto é, a partir de estatísticas, extrair informações, conhecimento, padrões e *insights* que servem de base para tomada de decisões. As figuras usadas nos exemplos foram modificadas devido a sigilosidade dos dados.

O OPF possui um significativo acervo de dados, alimentado por um fluxo contínuo, oriundo de elevada diversidade de fontes, formatos e formas de transmissão. Com isso, existe uma grande diversidade de dados, que precisam ser captados, transmitidos, validados, tratados, armazenados e processados pelos seus diversos sistemas em utilização.

As estatísticas descritivas são parte da análise exploratória de dados, que é o processo de analisar os dados. Este processo inclui o exame dos componentes e da estrutura do conjunto de dados, as distribuições de variáveis individuais e os relacionamentos entre duas ou mais variáveis. "A análise exploratória de dados fornece um extenso repertório de métodos para um estudo detalhado dos dados, antes de adaptá-los. Nessa abordagem, a finalidade é obter dos dados a maior quantidade possível de informação, que indique modelos plausíveis a serem utilizados numa fase posterior, a análise confirmatória de dados ou inferência estatística." (MEDRI, 2011)

3.1.1 Tecnologias utilizadas

A equipe de Ciência de Dados trabalhou com as seguintes tecnologias no projeto:

- SGBD PostgreSQL para o banco de dados;
- Python como linguagem de programação;
- Pandas Profiling para execução de estatísticas;
- Matplotlib para gerar gráficos.

3.1.2 Base de dados Amostrais

Para o armazenamento das informações utilizou-se o PostgreSQL. A estrutura do banco de dados foi baseada na do próprio OPF, dividido por *schemas* conforme o nome das bases de dados fornecidos.

Por se tratar de dados sigilosos houve ênfase em relação à segurança das informações. Cada pessoa da equipe tem um usuário e senha individual para utilização da base de dados, que é feita somente através de VPN.

3.1.3 Métricas e Análises

Para cada variável (coluna) de cada tabela do banco de dados foram feitas as seguintes análises:

- Mínimo e Máximo
- Média
- Mediana
- Quartil
- *Outliers*
- Desvio Padrão
- Assimetria
- Curtose

- Correlações de Pearson, Spearman, Kendall e Phi K

3.1.4 Mínimo e Máximo

Os valores mínimo e máximo de um conjunto de dados, apesar de representarem números extremamente fáceis de determinar, aparecem no cálculo de outras estatísticas descritivas.

O valor mínimo é o número que é menor ou igual a todos os outros valores do conjunto de dados. Já o valor máximo é o número que é maior ou igual a todos os outros valores do conjunto de dados.

Esses números, para análises em Ciência de Dados, são frequentemente utilizados para calcular intervalos e identificação de *outliers* que serão abordados posteriormente.

A partir do cálculo do mínimo e máximo de cada coluna das tabelas, utilizando o Pandas Profiling foram encontrados alguns possíveis erros nos dados. Em uma tabela referente a salário, através do valor mínimo pode-se observar um valor negativo, o que provavelmente é um dado errado, uma vez que esses valores deveriam representar a remuneração e não descontos. Já em uma tabela referente a transações financeiras, existe uma coluna referente a valor transacionado com um valor máximo de centenas de milhões, o que pode ser um erro uma vez que é um valor muito alto para uma única transação.

3.1.5 Média

A média aritmética é uma das medidas mais intuitivas de tendência central. Supondo que uma variável de tamanho n consista nos valores $x_1, x_2, ..., x_n$. A média aritmética destes dados é definida como a soma de todos os elementos do conjunto, dividido pelo número de elementos que compõem esse conjunto, como demonstrado na fórmula: $\frac{1}{n}\sum_{i=1}^{n} x_i$.

O cálculo da média de cada coluna numérica da base de dados foi feito com a biblioteca Pandas Profiling. Como esta medida pode ser influenciada por *outliers* em algumas colunas, a média se mostra distante dos valores esperados.

Por exemplo, em um conjunto de dados referente a renda de pessoas físicas e jurídicas, na coluna referente a valor de renda a média se mostra na casa de milhões, o que provavelmente ocorreu por influência das rendas de grandes empresas, que por poderem ser valores na casa de bilhões acabam distorcendo a média.

3.1.6 Mediana

A mediana é o valor da variável que divide os dados ordenados em duas partes de igual frequência.

Para calcular a mediana dos valores de n, os valores dos dados deverão ser ordenados do menor para o maior. Assim, se n for ímpar, a mediana da amostra é exatamente o valor no meio. Já, se n for par, a mediana da amostra é a média dos dois valores do meio.

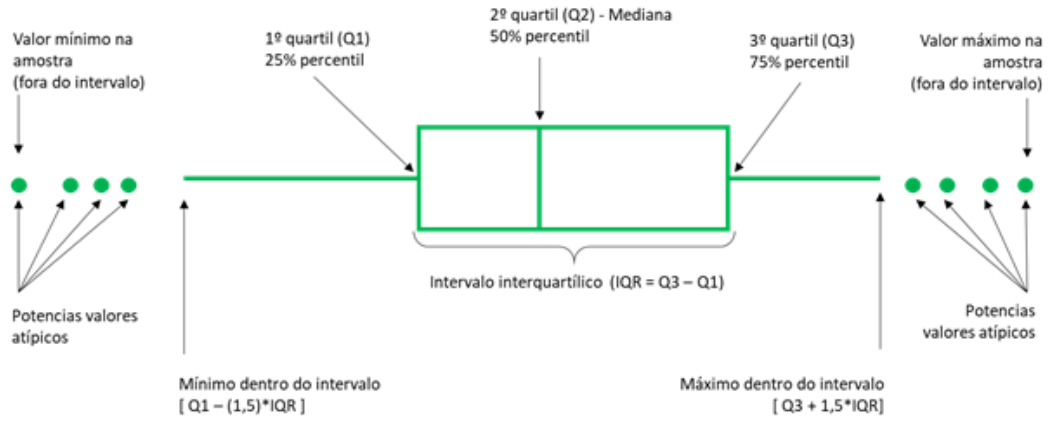
Por exemplo a Figura **??** é um gráfico gerado em uma tabela com valores de remuneração, onde a linha vermelha identifica o valor da mediana. Informações foram omitidas devido à confidencialidade dos dados.

3.1.7 Quartil

Considerado como um dos tipos de medidas separatrizes, os quartis têm como objetivo dividir um conjunto de dados em 4 partes de igual frequência. O primeiro quartil (Q1) está na marca de 25%, o segundo quartil (Q2) está na marca de 50% e o terceiro quartil (Q3) está na marca de 75%. Os cálculos para Q1 e Q3 são semelhantes ao cálculo da mediana. Já Q2 é igual ao valor mediano (MYATT; JOHNSON, 2006).

Na Figura 3.1, está representada a localização dos quartis em um conjunto de dados. Os quartis são muito usuais na Ciência de Dados, pois pode-se dividir bases de dados complexas e grandes em partes iguais, e a partir disso, explorar e entender os conjuntos menores de dados separadamente, sua visualização é feita principalmente em diagramas de caixa, por serem simples e diretos.

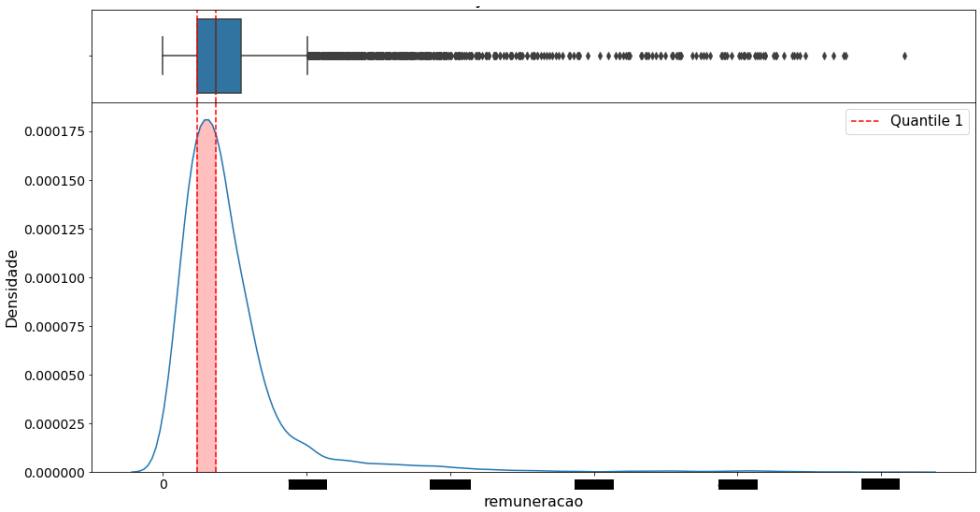
Figura 3.1 – Diagrama de Caixa



Fonte: https://medium.com/@claudio.siervi/interpretando-o-diagrama-de-caixa-boxplot-1876b7c099af, 2021

A Figura 3.2 exemplifica, no topo, o diagrama de caixa gerado com a biblioteca Matplotlib para a coluna valores de remuneração, destacando-se o quartil Q1 entre as duas linhas vermelhas. Informações foram omitidas devido à confidencialidade dos dados.

Figura 3.2 – Distribuição de valores de remuneração.



Fonte: Do Autor, 2021

3.1.8 *Outliers*

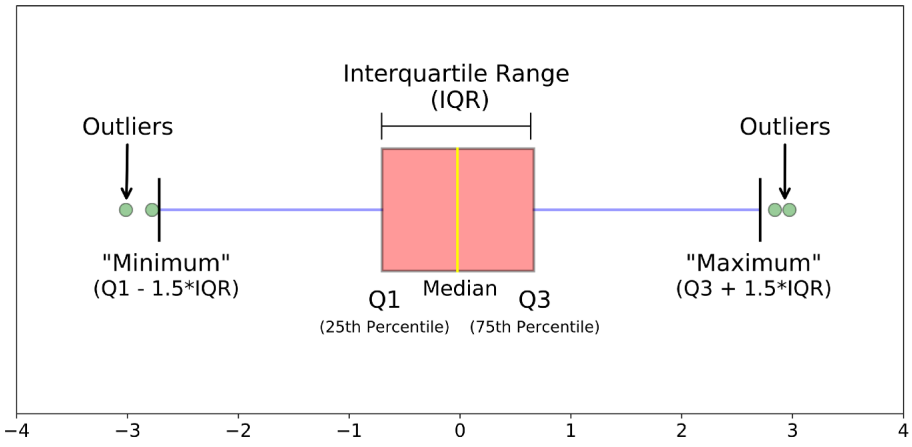
Os *outliers* são valores considerados atípicos em uma distribuição de dados, ou seja, são valores que estão distantes dos outros valores em um conjunto de dados. Após a identificação dos *outliers*, é sempre recomendado verificar se há valores discrepantes, que são valores extremos que podem ser erros de medição e registro ou podem ser relatórios precisos de eventos raros.

Ao contrário da análise de dados comum, em que os *outliers*, às vezes, são informativos e, às vezes, um empecilho, na detecção de anomalias, os pontos de interesse são os *outliers*, e a maior massa de dados serve principalmente para definir o "normal"contra o qual as anomalias são medidas (BRUCE; BRUCE, 2017).

O gráfico da Figura 3.3 é um exemplo da identificação de *outliers* representados em um diagrama de caixa.

Na análise de outra tabela com valores de remuneração, a tabela apresenta 218 *outliers* que podem ser visualizados como os losangos escuros no diagrama de caixa da Figura 3.4. Esse grande número pode causar distorção em outras métricas,

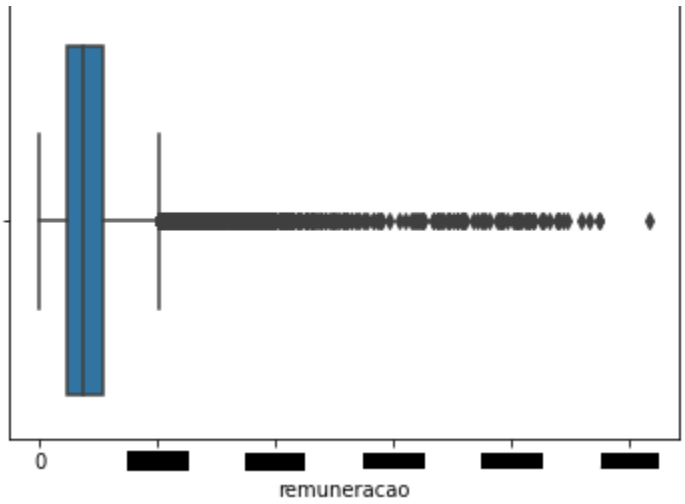
Figura 3.3 – Outliers em um Diagrama de Caixa



Fonte: https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51, 2021

e pode sinalizar valores errados na base de dados. Informações foram omitidas devido à confidencialidade dos dados.

Figura 3.4 – Outliers na coluna remuneração

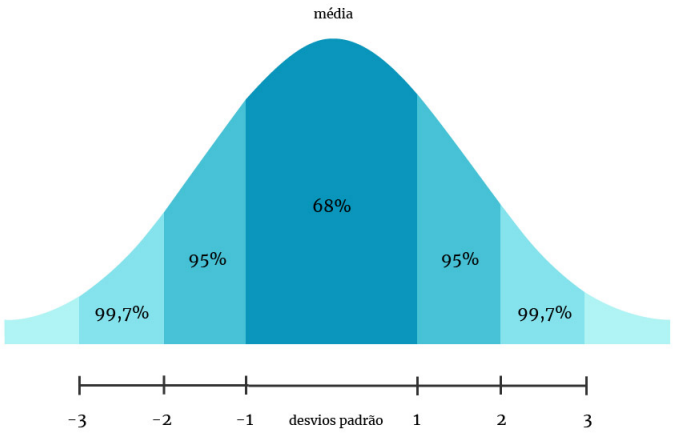


Fonte: Do Autor, 2021

3.1.9 Desvio Padrão

O desvio padrão mede o quanto as observações variam ou como as mesmas estão dispersas em torno da média aritmética do conjunto de dados, como pode ser observado na 3.5.

Figura 3.5 – Gráfico do desvio padrão em relação a média



Fonte: http://henriquefreitas.com.br/, 2021

Um desvio padrão com valor baixo, indica que os valores estão altamente concentrados em torno da média. Já um desvio padrão com valor alto, indica menor concentração das observações, ou valores, em torno da média. Os desvios padrão, podem ser calculados para uma série de variáveis medidas nas escalas de intervalo ou razão.

Nas análises automatizadas na base de dados do OPF foram encontrados dados com desvio padrão extremamente alto, o que pode ser explicado por valores muito distantes da média.

Em uma tabela que contém dados bancários de pessoas físicas e jurídicas na coluna de valor da renda em um determinado período, possui um desvio padrão com valor de bilhões. Já em outra tabela que contém informações de transações bancárias, a coluna de valor transação que agrega valores transacionados, possui desvio padrão de centenas de milhares. Esses números muito altos servem como alerta para se efetuar uma checagem mais detalhada dos dados.

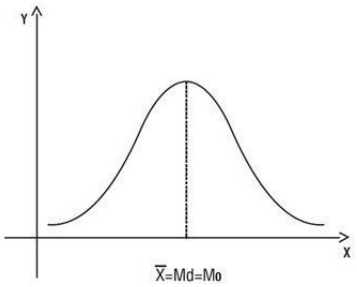
3.1.10 Assimetria

As medidas de assimetria possibilitam analisar uma distribuição de dados em relação a sua média, mediana e moda.

Embora as médias de posição e de variação, anteriormente demonstradas, possibilitam descrever estatisticamente um conjunto de dados, é necessário verificar como está se comportando de forma geral essa distribuição. Isso é possível através da distribuição de frequência e de histograma. Sendo que as distribuições possam tomar praticamente qualquer forma, a maioria que se encontra, na prática, é discreta por alguns tipos (MEDRI, 2011).

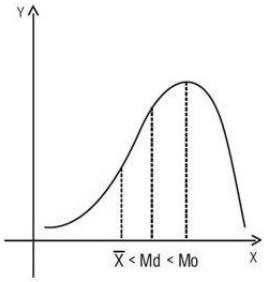
Baseando-se em uma curva normal, se traçado um eixo vertical no meio da curva, e as duas partes forem iguais, significa que os dados são simétricos 3.6. Um valor negativo indica assimetria para a esquerda 3.7 e um valor positivo indica assimetria para a direita 3.8.

Figura 3.6 – Distribuição Simétrica



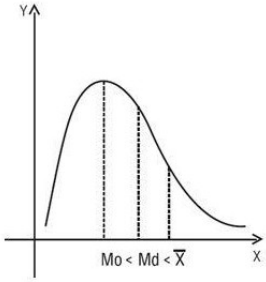
Fonte: https://ptdocz.com/doc/1884191/apresentação-4, 2021

Figura 3.7 – Distribuição Assimétrica à Esquerda



Fonte: https://ptdocz.com/doc/1884191/apresentação-4, 2021

Figura 3.8 – Distribuição Assimétrica à Direita



Fonte: https://ptdocz.com/doc/1884191/apresentação-4, 2021

Com o uso da biblioteca Pandas Profiling foi possível identificar na base de dados do OPF colunas com valores de assimetria. Por exemplo, em uma tabela que contém informações de transações bancárias com dados de valores de transação possui valor de assimetria na casa de centenas que representa assimetria à direita.

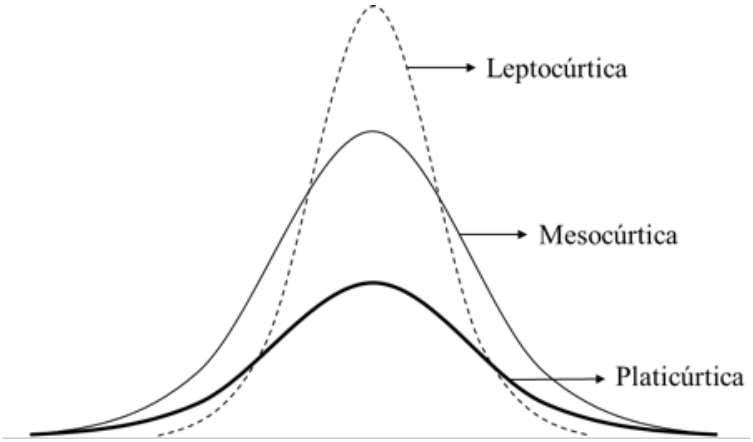
3.1.11 Curtose

O coeficiente de curtose está associado ao grau de achatamento da distribuição dos dados, ou seja, é calculado em relação à distribuição normal. Um valor positivo geralmente indica que a distribuição tem um pico mais acentuado do que a distribuição normal. Um valor negativo indica que a distribuição tem um pico mais plano do que a distribuição normal.

A curva normal, que é a base referencial, recebe o nome de Mesocúrtica. Já, uma distribuição que apresenta uma curva de frequência mais achatada do que a normal é denominada de Leptocúrtica, e a que apresenta uma curva de frequência mais aberta, recebe o nome de Platicúrtica (MEDRI, 2011).

A Figura 3.9 representa as três curvas descritas acima.

Figura 3.9 – Curvas de Curtose



Fonte: https://revistas.tec.ac.cr/index.php/eagronegocios/article/view/4456/4046, 2021

Os relatórios automatizados apontaram colunas com valores elevados de Curtose. Por exemplo, em uma tabela que contém informações de transações bancárias com dados de valores de transação possui valor da curtose na casa de dezenas de milhares que representa uma distribuição dos dados muito distante da distribuição normal, caracterizando uma curva Leptocúrtica.

3.1.12 Correlações de Pearson, Spearman, Kendall e Phi K

Para proporcionar maior solidez nas tomadas de decisões, é útil identificar a existência de correlação linear entre duas variáveis ou entre mais de duas variáveis e, se apropriado, quantificar a correlação.

A análise exploratória de dados, em muitos projetos de modelagem, usa-se o estudo da correlação entre preditores, e entre preditores e uma variável alvo. As variáveis X e Y (cada uma com dados medidos) são tidas como positivamente correlacionadas se valores altos de X acompanharem os valores altos de Y e os valores baixos de X acompanharem os valores baixos de Y. Se os valores altos de X acompanharem os valores baixos de Y, e vice-versa, as variáveis são negativamente correlacionadas (BRUCE; BRUCE, 2017).

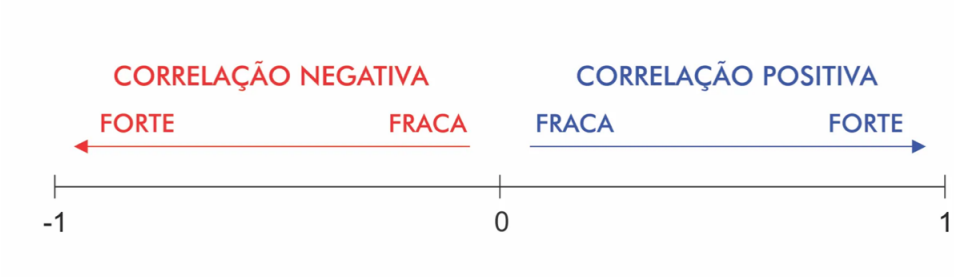
Para o projeto, foram utilizados os coeficientes de correlação de Pearson, Spearman, Kendall e Phi K com visualização feita através da biblioteca Pandas Profiling, utilizando-se matrizes *Heatmap*. As correlações são descritas com mais detalhes nas subseções abaixo.

3.1.12.1 Pearson

O coeficiente de correlação de Pearson é uma medida da variância compartilhada entre duas variáveis. O coeficiente de correlação varia de -1 a 1. O sinal indica direção positiva ou negativa do relacionamento e o valor sugere a força da relação entre as variáveis, como demonstrado na Figura 3.10. Uma correlação perfeita (-1 ou 1) indica que o escore de uma variável pode ser determinado exatamente ao se saber o escore da outra. No outro oposto, uma correlação de valor zero indica que não há relação linear entre as variáveis (FILHO; JÚNIOR, 2009).

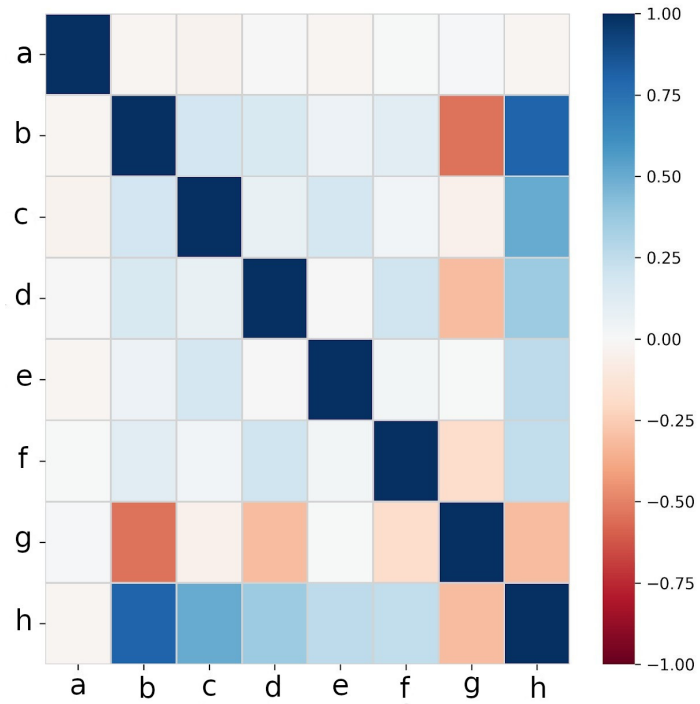
Para valores positivos, indica uma associação positiva, à medida que o valor de uma variável aumenta, o mesmo acontece com o valor da outra variável. Já com valores menores que 0, tem-se uma associação negativa, isto é, à medida que o valor de uma variável aumenta o valor da outra diminui.

Figura 3.10 – Força da correlação entre variáveis na Correlação de Pearson



Fonte: https://operdata.com.br/blog/coeficientes-de-correlacao/, 2021

Figura 3.11 – Correlação de Pearson entre colunas.



Fonte: Do Autor, 2021

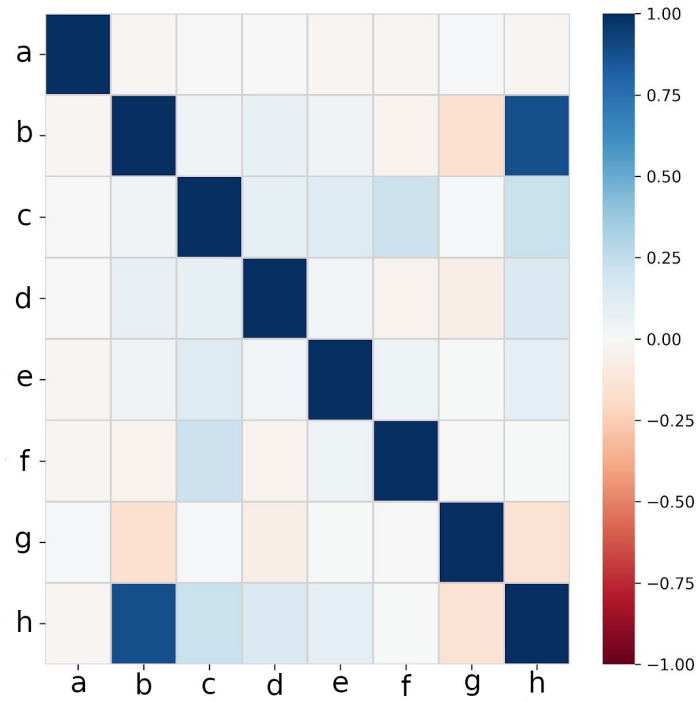
Na matriz da Figura 3.11 o valor de correlação entre as colunas **b** e **h** é de 0,8041 o que demonstra uma proporção entre os valores. Quanto maior **b**, maior **h**. Informações foram omitidas devido à confidencialidade dos dados.

3.1.12.2 Spearman

O coeficiente de Spearman varia entre -1 e 1. Quanto mais próximo estiver destes extremos, maior será a associação entre as variáveis. O sinal negativo da correlação significa que as variáveis variam em sentido contrário, isto é, as categorias mais elevadas de uma variável estão associadas a categorias mais baixas da outra variável (MADALENA MALVA, 2007).

Ao contrário do coeficiente de Pearson, o coeficiente de Spearman não exige a suposição de que a relação entre as variáveis seja linear, nem requer que as mesmas sejam quantitativas – pode inclusive ser utilizado para verificar relação entre variáveis medidas no nível ordinal (Oliveira B., 2019).

Figura 3.12 – Correlação de Spearman entre colunas.



Fonte: Do Autor, 2021

Na matriz acima na Figura 3.12 o valor de correlação entre as colunas **b** e **h** é de 0,8826 o que demonstra uma relação direta positiva entre os valores.

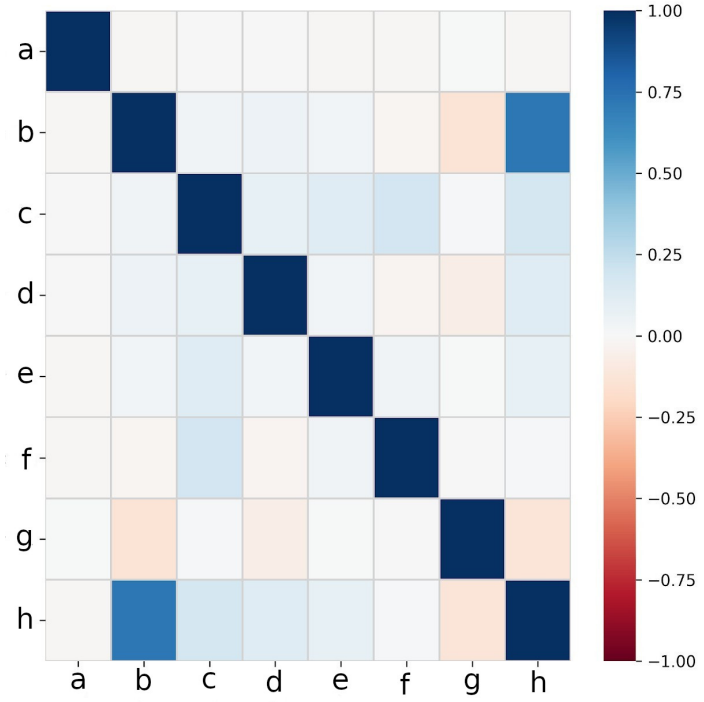
Quanto maior **b**, maior **h**. Informações foram omitidas devido à confidencialidade dos dados.

3.1.12.3 Kendall

O coeficiente de correlação tau-b de Kendall, denominado pela letra grega tau (τ), é outra correlação não paramétrica, que permite associação para variáveis ordinais. Uma vantagem sobre o coeficiente de Spearman é a possibilidade de ser generalizado para um coeficiente de correlação parcial (CAPP; NIENOV, 2020).

Na matriz da Figura 3.13 o valor de correlação entre as colunas **b** e **h** é de 0,7225 o que demonstra uma proporção entre os valores. Quanto maior **b**, maior **h**. Informações foram omitidas devido à confidencialidade dos dados.

Figura 3.13 – Correlação de Kendall entre colunas

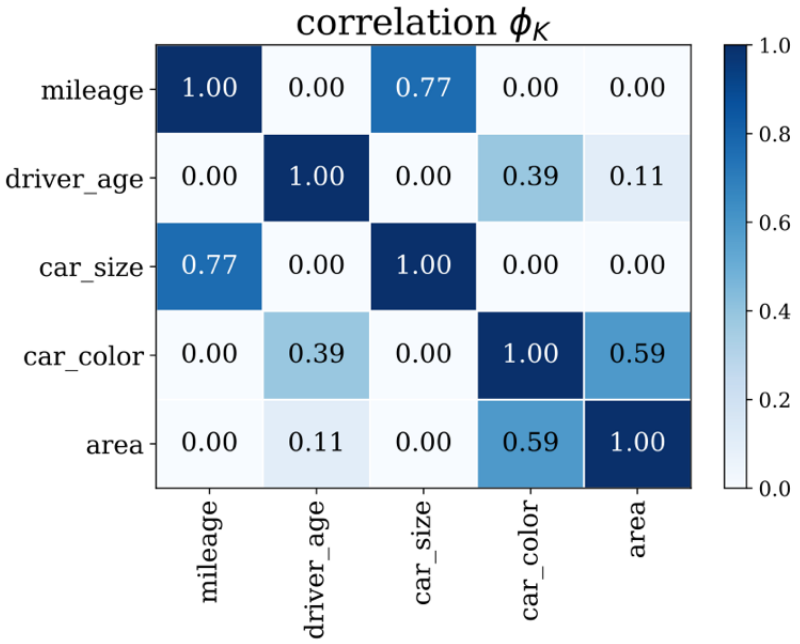


Fonte: Do Autor, 2021

3.1.12.4 Phi K

A correlação φK segue um tratamento uniforme para variáveis intervalares, ordinais e categóricas. Nas análises mais modernas, ao estudar as dependências entre um conjunto de variáveis com tipos mistos, onde algumas variáveis são categóricas, a correlação φK se mostra particularmente útil. Os valores extraídos na correlação estão limitados no intervalo [0,1], com 0 para nenhuma associação e +1 para associação completa, como pode ser observado na Figura 3.14 (M. et al., 2019).

Figura 3.14 – Exemplo de coeficiente de correlação calculado para um conjunto de dados, mostrando a matriz de correlação φK.



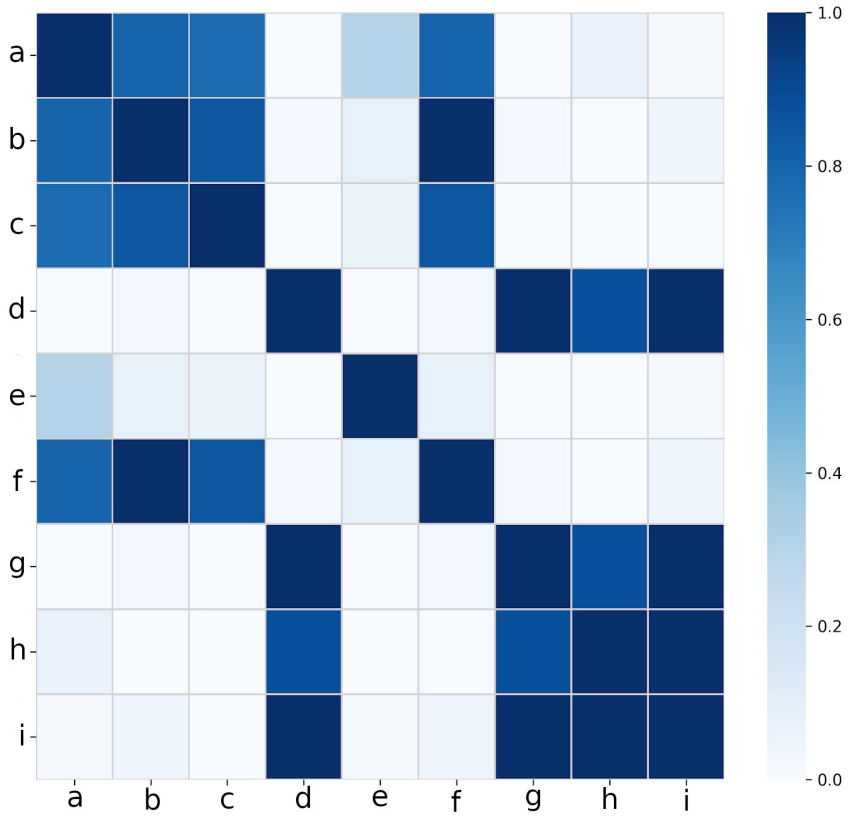
Fonte: https://arxiv.org/pdf/1811.11440.pdf, 2021

A interpretação é semelhante ao coeficiente de correlação de Pearson e é equivalente no caso de uma distribuição normal de entrada bivariada. Ao contrário de Pearson, que descreve a dependência linear média entre duas variáveis, o co-

eficiente de correlação φK também captura relações não lineares e é extensível a mais de duas variáveis (M. et al., 2019).

A correlação Phi K representada na Figura 3.15, foi aplicada através da biblioteca Pandas Profiling. A correlação que se mostra mais interessante é a da coluna **h**. Ela tem uma correlação de 0,88 ou 88% com a coluna **g**. Informações foram omitidas devido à confidencialidade dos dados.

Figura 3.15 – Correlação Phi K entre colunas da tabela.



Fonte: Do Autor, 2021

3.1.13 Resultados

A partir do resultado da análise exploratória da amostra de dados repassada identificaram-se variáveis de grande valor para o desenvolvimento de propostas

para resolver problemas e responder perguntas estratégicas a investigações, além de auxiliar o desenvolvimento de algoritmos e sistemas futuros.

Foram também encontrados potenciais erros e fragilidades dos conjuntos de dados amostrais que podem identificar problemas em outras bases completas, com isso é possível analisar com um direcionamento prévio as mesmas.

3.2 Tipologias Relacionadas à Lavagem de Dinheiro

Durante reuniões com o OPF foi identificada a necessidade do desenvolvimento e implementação de algoritmos que automatizam a extração de tipologias em dados de quebra de sigilo bancário e telefônico. Estas tipologias refletem padrões que indicam comportamentos associados a crimes, de casos em que a equipe do OPF analisa diariamente.

No projeto foi proposto o estudo das possibilidades de desenvolvimento e implementação de algoritmos que automatizam a extração de duas tipologias, consideradas como prioritárias durante as reuniões de descoberta com o OPF. As duas tipologias abordadas são: "Comunicação entre investigados com localidades próximas"e "Smurfing".

3.2.1 Tecnologias utilizadas

A equipe de Ciência de Dados trabalhou com as seguintes tecnologias no projeto:

- SGBD PostgreSQL para o banco de dados;
- Python como linguagem de programação;
- Pandas para manipulação dos dados;
- NetworkX para algoritmos e gerar grafos;
- Matplotlib para gerar gráficos.

3.2.2 Comunicação entre investigados com localidades próximas

Os algoritmos apresentados nesta seção foram desenvolvidos para possibilitar a fácil visualização da comunicação entre indivíduos investigados por ligações telefônicas. A primeira tabela utilizada possui informações telefônicas e cadastrais dos investigados e de indivíduos que se relacionaram com os mesmos. Outra tabela possui informações referentes às torres de telefonia, por exemplo as suas localizações. Com o resultado da união das duas tabelas tem-se uma nova com os dados das chamadas e das torres utilizadas na conexão.

A partir dessa nova tabela tem-se colunas referentes a longitude e latitude da primeira torre que o indivíduo de origem e de destino da ligação se conectaram. Com essas colunas é calculada a distância em quilômetros entre a origem e o destino de cada ligação, que então é armazenada em uma nova coluna de distância. A partir dessa tabela foram desenvolvidas três funções, essas são descritas abaixo.

3.2.2.1 Função 1: *plot_row_map*

O objetivo desta função é visualizar a distância entre duas torres que se conectaram. Elas representam o local onde o terminal do indivíduo originador da ligação e do indivíduo destino se conectaram. Além da tabela que foi gerada como descrito acima, usa-se como base o mapa das unidades federativas brasileiras obtido diretamente do IBGE.

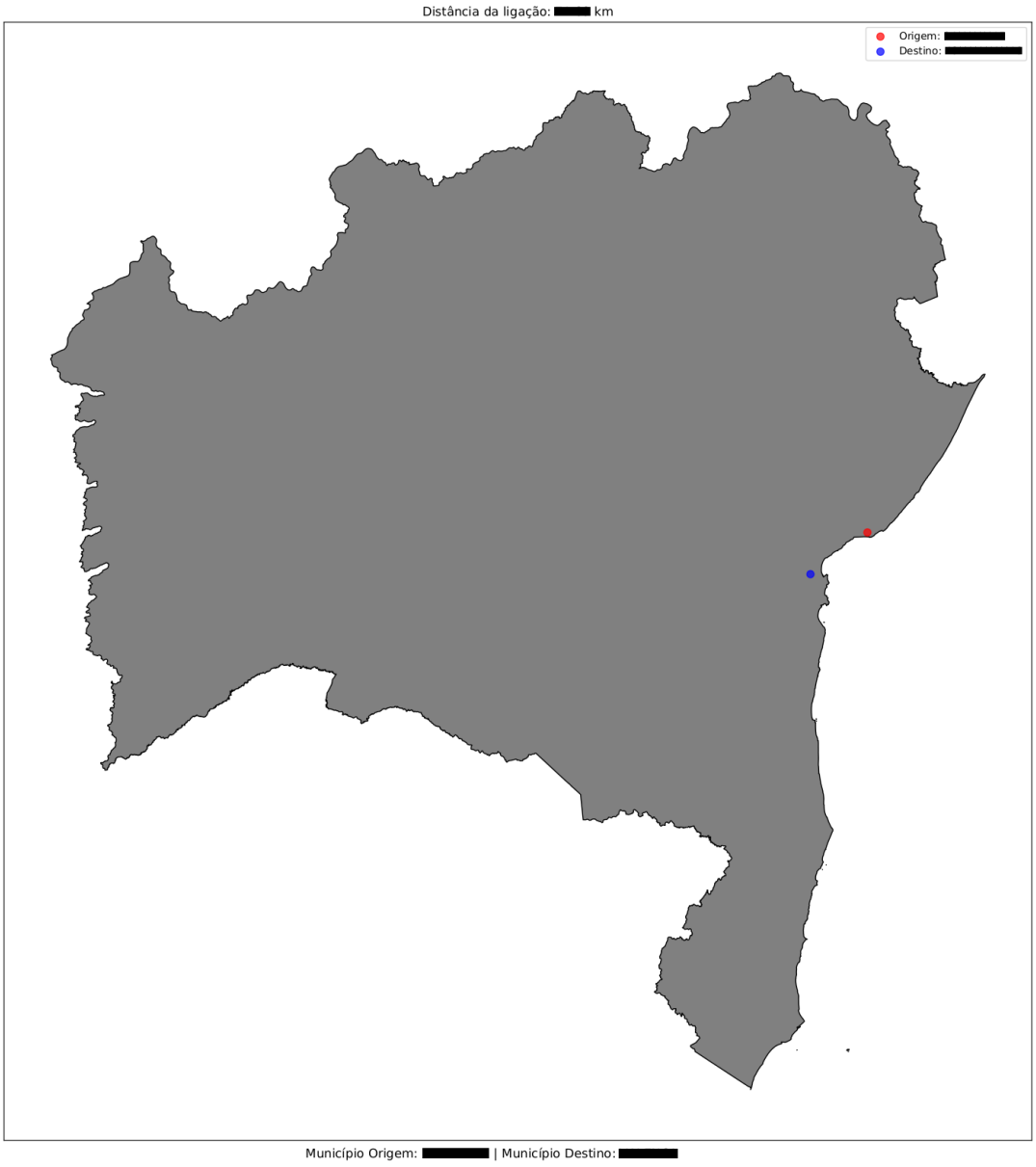
A função para cada linha da tabela, ou seja cada ligação efetuada, gera um mapa com um ponto vermelho representando a origem da ligação e um ponto azul o destino, com seus respectivos CPFs/CNPJs. Além disso, exibe-se o município de origem, de destino e a distância da chamada.

A função *plot_row_map* possui 3 argumentos obrigatórios:

- peos: *DataFrame* com uma coluna relativa aos pontos da torre de origem da ligação.
- peds: *DataFrame* com uma coluna relativa aos pontos da torre de destino da ligação.
- ufs: *DataFrame* com o mapa das unidades federativas brasileiras.

Exemplo: A figura 3.16 na próxima página é o resultado da execução da função *plot_row_map*, exibindo um mapa da ligação de número 150 da base de dados, de origem da ligação (ponto vermelho) e o destino (ponto azul), ambos no mesmo estado.

Figura 3.16 – Exemplo de imagem gerada pela função *plot_row_map*.



3.2.2.2 Função 2: *calls_from_one_cpf_cpnj*

O objetivo desta função é visualizar a distância entre as posições das torres que o indivíduo origem de todas as ligações se conectou e as torres que todos os destinos se conectaram, ou seja, todos os pontos de origem de ligações de um indivíduo e todos os pontos de destino para um ou mais indivíduos. A função permite usar filtros para direcionar os resultados de acordo com a necessidade do investigador.

Como filtros da função pode-se usar:

- start_date - data de início, são analisadas apenas chamadas posteriores a data definida.
- end_date - data de fim, são analisadas apenas chamadas anteriores a data definida.
- max_distance - distância máxima em linha reta em quilômetros entre as torres, são analisadas apenas chamadas com distância menores ou iguais ao valor definido.

A função recebe um CPF/CNPJ que é origem de ligações e gera o mapa de todas as ligações relacionadas. No mapa os pontos vermelhos representam os locais de origem das ligações e os pontos azuis representam os destinos. Também é retornado um *DataFrame* com as colunas:

- data - data de origem da ligação.
- origem - CPF/CNPJ origem da ligação.
- mun_ori - município da localização de origem da ligação.
- uf_ori - UF da localização de origem da ligação.
- destino - CPF/CNPJ destino da ligação.

- mun_des - município da localização de destino da ligação.
- uf_des - UF da localização de destino da ligação.
- distancia - distância entre origem e destino da ligação.

A função *calls_from_one_cpf_cpnj* possui 4 argumentos obrigatório e 3 opcionais:

- cpf_cnpj_orig (obrigatório): O CPF/CNPJ que será usado como origem das ligações.
- peos (obrigatório): *DataFrame* com coluna relativa aos pontos da torre de origem da ligação.
- peds (obrigatório): *DataFrame* com coluna relativa aos pontos da torre de destino da ligação.
- ufs (obrigatório): *DataFrame* com o mapa das unidades federativas brasileiras.
- start_date (opcional): Data de início (no formato A-m-d) para filtrar.
- end_date (opcional): Data de fim (no formato A-m-d) para filtrar.
- max_distance (opcional): distância máxima em quilômetros entre as torres para filtrar.

Exemplo: Mapa e Tabela 3.1 gerados pela filtragem das ligações de origem do CPF cpf1, ocorridas entre os dias dd/mm/aaaa e dd/mm/aaaa, com distância máxima de 100km. Os pontos vermelhos representam as origens e os azuis os destinos. Informações foram omitidas devido à confidencialidade dos dados.

Figura 3.17 – Exemplo de imagem gerada pela função *calls_from_one_cpf_cpnj*.

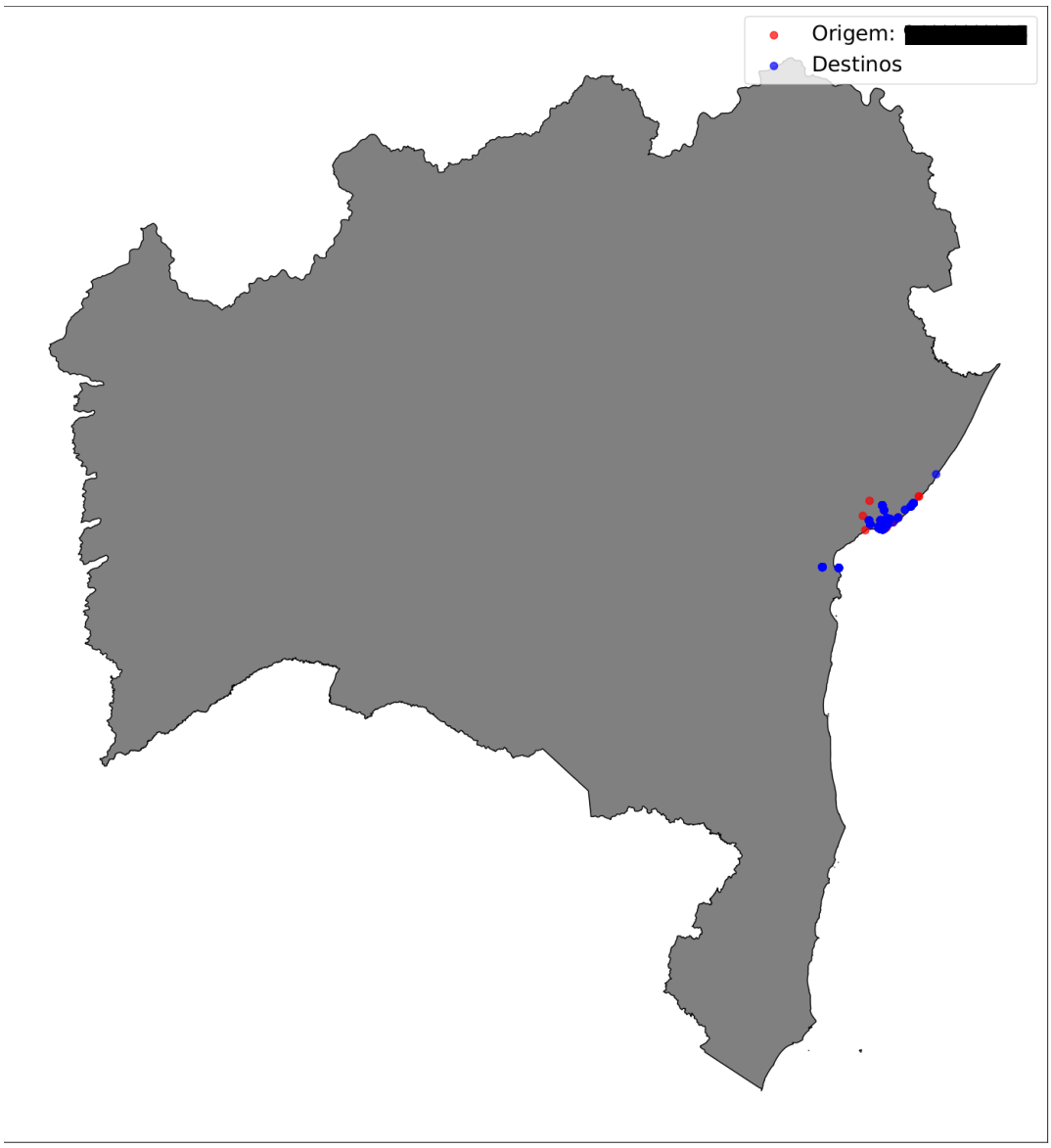


Tabela 3.1 – Tabela gerada pela função *calls_from_one_cpf_cpnj* com as informações de cada ligação exibida no mapa.

data	origem	mun_ori	uf_ori	destino	mun_des	uf_des	distancia
2014-01-01	cpf1	municipio1	UF1	cpf2	municipio2	UF2	d1
2014-01-01	cpf1	municipio2	UF2	cnpj4	municipio2	UF3	d2
2014-01-01	cpf1	municipio2	UF3	cnpj4	municipio2	UF3	d2
2014-02-01	cpf1	municipio2	UF4	cpf3	municipio2	UF2	d3
2014-02-01	cpf1	municipio2	UF5	cpf3	municipio2	UF5	d3

3.2.3 Smurfing

Smurfing é o nome dado a uma das possíveis formas de fraudes financeiras, a qual de maneira geral, se tenta movimentar uma quantidade financeira relevante sem que esta operação chame atenção de órgãos fiscalizadores. Esta é uma técnica utilizada para mascarar pagamentos ou ganhos ilícitos de várias possíveis situações. A definição de *Smurfing* pode ser ampla assim como a quantidade de suas formas práticas, mas em geral, este é também um crime financeiro que pode ser classificado como crime de lavagem de dinheiro (CHADHA et al., 2018).

Os algoritmos apresentados nesta seção têm como base de construção a definição gerada por um documento disponibilizado pelo OPF, que define a prática de *smurfing* como

"**Identificação de transações repetitivas oriundas de diferentes depósitos em dinheiro ou transferências que individualmente não representam valor relevante.**"

3.2.3.1 Função 1: *detect_smurf*

Os dados utilizados nesta função vem de uma tabela que possui informações de transações bancárias de indivíduos investigados que passaram pelo processo legal de quebra de sigilo bancário.

As principais métricas para identificar os possíveis casos de *Smurfing* são:

- A quantidade de transações entre os indivíduos (operações de débito ou crédito);
- Coeficiente de variação dos valores transacionados por um alvo.

O coeficiente de variação é determinado pela equação (3.1). Este coeficiente permite visualizar a taxa de variação de valores transacionados em um grupo de transações de um investigado.

$$CV = \frac{s}{\bar{x}} \ , \tag{3.1}$$

em que $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ e $s = \sqrt{\frac{1}{(n-1)}\sum_{i=1}^{n}(x_i - \bar{x})^2}$.

A função agrega valores com frequência determinada pelo usuário (diária, semanal, mensal ou anual) através do CPF e tipo de operação (débito ou crédito) e ordena os resultados.

A função *detect_smurfing* possui 1 argumento obrigatório e 8 opcionais:

- agg_type (obrigatório): Forma de agregar os dados. Diária, semanal, mensal ou anual.
- cpf_cnpj (opcional) : Lista de CPFs/CNPJs a que se deseja analisar as transações.
- operation (opcional): Lista de operações que se deseja analisar. As opções são “D” (débito), “C” (crédito) ou “*”(bloqueada).
- start_date (opcional): Data de início (no formato A-m-d) para filtrar.
- end_date (opcional): Data de fim (no formato A-m-d) para filtrar.
- coef_var_max (opcional): Coeficiente de variação máximo. Limite superior do coeficiente.
- min_value_transaction (opcional): Limite inferior do valor das transações.

- max_value_transaction (opcional): Limite superior do valor das transações.
- min_number_transactions (opcional): Quantidade mínima de transações.

O resultado da função é uma tabela representada através de um *DataFrame* com as seguintes colunas:

- data_lanc - data de lançamento das transações.
- natureza_lanc - natureza do lançamento (débito, crédito ou *).
- total_mov - total movimentado pelo indivíduo no período.
- coef_var_transacao - coeficiente de variação das transações.
- qtde_transacoes - quantidade total de transações do indivíduo no período.

Exemplo: Deseja-se filtrar apenas operações de débito ocorridas entre dd/mm/aaaa e dd/mm/aaaa cujos valores estão entre v1 reais e v2 reais. Além disso, deseja-se que seja realizada a agregação mensal das informações. O resultado é dado pela Tabela 3.2, que mostra as primeiras 5 linhas da tabela resultante, onde na primeira linha um investigado movimentou 19 vezes o valor de v1 em operações de débito em mês de aaaa, através de 19 transações, com o coeficiente de variação 0, ou seja, todas com mesmo valor. Informações foram omitidas devido à confidencialidade dos dados.

Tabela 3.2 – Exemplo da função *detect_smurf*.

data_lanc	natureza_lanc	total_mov	coef_var_transacao	qtde_transacoes
aaaa-mm	D	v1*19	0.0	19
aaaa-mm	D	v1*3	0.0	3
aaaa-mm	D	v1*2	0.0	2
aaaa-mm	D	v1*2	0.0	2

3.2.3.2 Função 2: *plot_graph*

O objetivo desta função é facilitar a visualização gráfica das relações financeiras entre os investigados.

A partir do resultado da função *detect_smurfing* é possível gerar o grafo com todas as conexões entre os CPFs/CNPJs escolhidos de cada linha. Escolhe-se CPFs/CNPJs do resultado da função *detect_smurfing* os indivíduos que tiveram relação com estes aparecerão no grafo. A função também gera uma tabela com a quantidade de transações realizadas no período. Em casos de CPFs/CNPJs não identificados eles são representados por NI.

A função recebe como argumento obrigatório uma lista de CPFs/CNPJs que estão contidos no resultado de uma execução da função *detect_smurfing*. Os CPFs/CNPJs são exibidos em cada nó, sendo os vermelhos os que foram passados na lista e nas arestas a quantidade de transações entre os nós.

A função exibe o grafo e uma tabela com as colunas:

- cpf/cnpj_orig - CPF ou CNPJ de origem da transação.
- cpf/cnpj_dest - CPF ou CNPJ de destino da transação.
- qtde - quantidade de transações entre os pares.

Figura 3.18 – Exemplo de grafo gerado pela função *plot_graph*

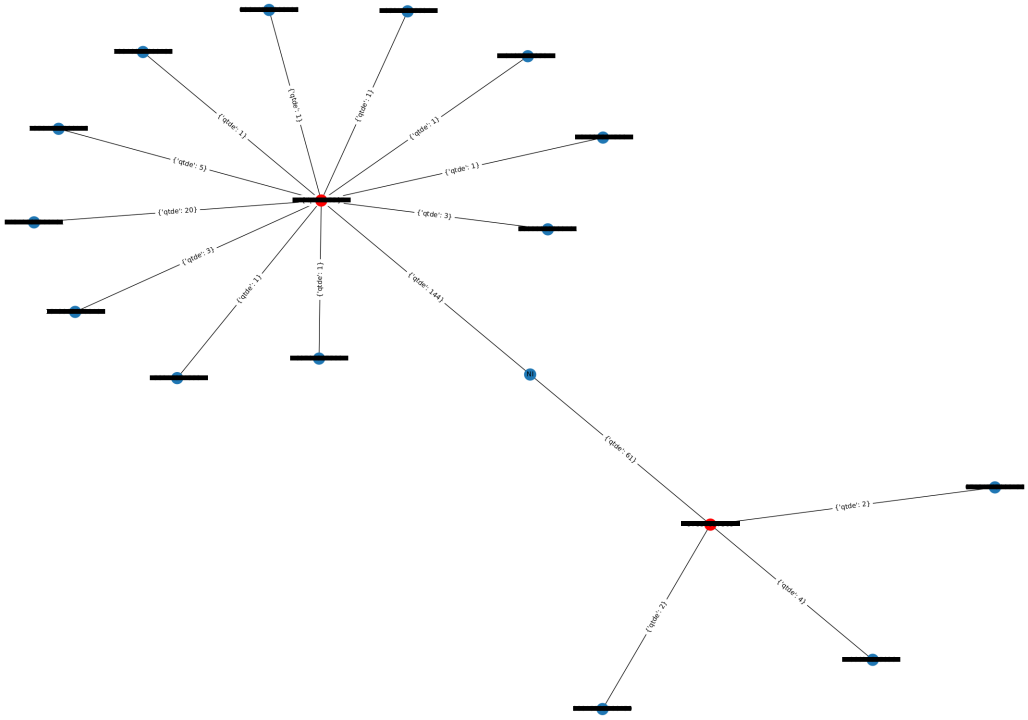


Tabela 3.3 – Exemplo de tabela resultado da função *plot_graph*. Informações foram omitidas devido à confidencialidade dos dados.

cpf/cnpj_orig	cpf/cnpj_dest	qtde
cpf1	NI	144
cpf2	NI	61
cpf1	cpf1	55
cpf1	cnpj1	20
cpf1	cnpj2	5

3.2.4 Resultados

As funções desenvolvidas se mostraram eficientes na detecção de características das tipologias, sendo ferramentas interessantes para gerar informações e visualizações gráficas dos dados. Posteriormente eles poderão ser implantados em um sistema do OPF, dedicado à abordagem e detecção de tipologias.

4 CONSIDERAÇÕES FINAIS

O estágio apresentou um papel fundamental na formação do aluno, possibilitou o desenvolvimento de conhecimentos além das salas de aula através de experiência prática, além de promover a interação e aprendizado com pessoas mais experientes. Tudo isso somado ao conhecimento teórico e prático abordados no curso de Ciência da Computação capacitam o aluno a ingressar no mercado de trabalho.

Durante todo o estágio o aluno pode aplicar os conhecimentos adquiridos nas disciplinas da graduação de forma prática. Conceitos básicos de lógica e programação são aprendidos na disciplina de Introdução aos Algoritmos, posteriormente complementados e agregados em Estruturas de Dados que fornece ferramentas fundamentais para manipulações de dados, possibilitando um grande salto de conhecimento de programação. A disciplina de Estatística permite o entendimento de métricas e métodos que ajudam a entender informações e padrões contidas nos dados e algoritmos.

Em Introdução a Sistemas de Banco de Dados desenvolve-se uma noção inicial de como projetar, criar e utilizar um banco de dados e posteriormente em Sistemas Gerenciadores de Banco de Dados aprende-se o funcionamento dos SGBDs, como seus processos funcionam e como utilizá-los de forma eficiente essas duas disciplinas em conjunto consolidam um conhecimento fundamental para entender e possibilitar o armazenamento de dados.

A disciplina de Algoritmos em Grafos apresentou conceitos fundamentais para o desenvolvimento do estágio e com certeza agrega conhecimento para o futuro profissional, por apresentar algoritmos e formas de abordar problemas de forma otimizada em diversas áreas. Assim como em Complexidade e Projetos de Algoritmos o aluno pode aprender formas eficientes de usar e projetar algoritmos, analisando sua complexidade.

A disciplina de Redes de Computadores se mostrou de grande importância, uma vez que o estágio foi totalmente remoto, nela foram aprendidos conceitos de acesso remoto via SSH, Rede Privada Virtual (VPN) e conceitos de segurança de rede que auxiliaram no uso dos servidores remotos da Zetta.

Como na matriz curricular do curso infelizmente não existe uma matéria específica de Ciência de Dados o estágio se mostra uma experiência ainda mais engrandecedora, por permitir assimilar e utilizar o conhecimento aprendido durante todo o curso, de forma prática em um ambiente empresarial e agregar conhecimentos que não são ensinados diretamente. Através dele o aluno pode se desenvolver na próspera área de Ciência de Dados, utilizando ferramentas e aprendendo tecnologias que o tornaram apto a ingressar no mercado de trabalho, assim, o estágio foi a porta de entrada na jornada profissional.

REFERÊNCIAS

Atlassian. **Git basics**. 2021. Disponível em: <https://www.atlassian.com/git>. Acesso em: 08 jul. 2021.

BRUCE, P.; BRUCE, A. **Practical Statistics for Data Scientists**. [S.l.]: O'Reilly Media, 2017.

CAPP, E.; NIENOV, O. H. **Bioestatística Quantitativa Aplicada**. [S.l.]: UFRGS, 2020.

CHACON, S.; STRAUB, B. **Pro Git**. [S.l.]: Apress, 2014.

CHADHA et al. Handling smurfing through big data. **Big Data Analytics**, 2018.

DataFrame. **Pandas DataFrame**. 2021. Disponível em: <https://pandas.pydata.org/docs/user_guide/dsintro.html#dataframe>. Acesso em: 02 jul. 2021.

Docs Python. **Why was Python created in the first place?** 2021. Disponível em: <https://docs.python.org/3/faq/general.html#what-is-python>. Acesso em: 30 jun. 2021.

EMBRAPII. **UNIDADE EMBRAPII DE AGRICULTURA DIGITAL – ZETTA/UFLA – UNIVERSIDADE FEDERAL DE LAVRAS**. 2021. Disponível em: <https://embrapii.org.br/unidades/unidade-embrapii-de-agricultura-digital-zetta-ufla-universidade-federal-de-lavras/>. Acesso em: 22 jun. 2021.

FILHO, D. B. F.; JÚNIOR, J. A. da S. Desvendando os mistérios do coeficiente de corelação de pearson. **Revista Política Hoje, Vol.18, n.1**, 2009.

HAGBERG, A. A.; SCHULT, D. A.; SWART, P. J. Exploring network structure, dynamics, and function using networkx. In: VAROQUAUX, G.; VAUGHT, T.; MILLMAN, J. (Ed.). **Proceedings of the 7th Python in Science Conference**. Pasadena, CA USA: [s.n.], 2008. p. 11 – 15.

HUNTER, J. D. Matplotlib: A 2d graphics environment. **Computing in Science & Engineering**, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.

Jupyter nbviewer. **What is the Jupyter Notebook?** 2021. Disponível em: <https://nbviewer.jupyter.org/github/jupyter/notebook/blob/master/docs/source/examples/Notebook/What\%20is\%20the\%20Jupyter\%20Notebook.ipynb#>. Acesso em: 01 jul. 2021.

M., B. et al. A new correlation coefficient between categorical, ordinal and interval variables with pearson characteristics. **KPMG Advisory N.V.**, 2019.

MADALENA MALVA. **Coeficiente de Correlação Ró de Spearman**. 2007. Disponível em: <http://www.estgv.ipv.pt/PaginasPessoais/malva/TratamentoEstatistico\%20de\%20dados/Coeficiente\%20de\%20Correla\%C3\%A7\%C3\%A3o\%20R\%C3\%B3\%20de\%20Spearman.pdf>. Acesso em: 28 jul. 2021.

Matplotlib. **Matplotlib**. 2021. Disponível em: <https://matplotlib.org/stable/users/history.html>. Acesso em: 05 jul. 2021.

MEDRI, W. Análise exploratória de dados. **Centro de Ciências Exatas – CCE/UEL**, 2011.

MYATT, G. J.; JOHNSON, W. P. **Making Sense of Data: A Practical Guide to Exploratory Data Analysis and Data Mining**. [S.l.]: Wiley-Interscience, 2006.

NetworkX. **NetworkX**. 2021. Disponível em: <https://networkx.org/>. Acesso em: 06 jul. 2021.

Oliveira B. **Coeficientes de correlação**. 2019. Disponível em: <https://operdata.com.br/blog/coeficientes-de-correlacao/>. Acesso em: 29 jul. 2021.

Pandas. **History of development**. 2021. Disponível em: <https://pandas.pydata.org/about/index.html>. Acesso em: 02 jul. 2021.

Pandas Profiling. **Pandas Profiling**. 2021. Disponível em: <https://pandas-profiling.github.io/pandas-profiling/docs/master/rtd/pages/introduction.html>. Acesso em: 06 jul. 2021.

PostgreSQL Org. **PostgreSQL Documentation**. 2021. Disponível em: <https://www.postgresql.org/docs/13/index.html>. Acesso em: 08 jul. 2021.

Project Jupyter. **About Project Jupyter**. 2021. Disponível em: <https://jupyter.org/about>. Acesso em: 01 jul. 2021.

PyPI. **The Python Package Index (PyPI) is a repository of software for the Python programming language.** 2021. Disponível em: <https://pypi.org/>. Acesso em: 29 jun. 2021.

Series. **Pandas Series**. 2021. Disponível em: <https://pandas.pydata.org/docs/user_guide/dsintro.html#series>. Acesso em: 02 jul. 2021.

The Guardian. **Tech giants may be huge, but nothing matches big data**. 2013. Disponível em: <https://www.theguardian.com/technology/2013/aug/23/tech-giants-data>. Acesso em: 25 out. 2021.

TIOBE Index. **TIOBE Index for June 2021**. 2021. Disponível em: <https://www.tiobe.com/tiobe-index/>. Acesso em: 30 jun. 2021.