

**UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"**

**FACULDADE DE CIÊNCIAS - CAMPUS BAURU**

**DEPARTAMENTO DE COMPUTAÇÃO**

**BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**JOÃO VITOR LOPES DA SILVA**

**MINERAÇÃO DE DADOS APLICADO À GESTÃO DE ENERGIA  
ELÉTRICA**

**BAURU**

**Setembro/2019**

JOÃO VITOR LOPES DA SILVA

## **MINERAÇÃO DE DADOS APLICADO À GESTÃO DE ENERGIA ELÉTRICA**

Trabalho de Conclusão de Curso do Curso  
de Ciência da Computação da Universidade  
Estadual Paulista “Júlio de Mesquita Filho”,  
Faculdade de Ciências, Campus Bauru.  
Orientador: Prof. Dr. Kléber Rocha de Oliveira

BAURU  
Setembro/2019

João Vitor Lopes da Silva      Mineração de Dados Aplicado à Gestão de Energia Elétrica/ João Vitor Lopes da Silva. – Bauru, Setembro/2019-      47 p. : il. (algumas color.) ; 30 cm.  
Orientador: Prof. Dr. Kléber Rocha de Oliveira  
Trabalho de Conclusão de Curso – Universidade Estadual Paulista “Júlio de Mesquita Filho”  
Faculdade de Ciências  
Ciência da Computação, Setembro/2019.  
1. Tags 2. Para 3. A 4. Ficha 5. Catalográfica

João Vitor Lopes da Silva

## **Mineração de Dados Aplicado à Gestão de Energia Elétrica**

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

---

**Prof. Dr. Kléber Rocha de Oliveira**

Orientador

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Faculdade de Ciências

Departamento de Ciência da Computação

---

**Prof. Dr. Kelton Augusto Pontara da Costa**

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Faculdade de Ciências

Departamento de Ciência da Computação

---

**Prof<sup>a</sup>. Dr<sup>a</sup>. Simone das Graças Domingues Prado**

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Faculdade de Ciências

Departamento de Ciência da Computação

Bauru, \_\_\_\_\_ de \_\_\_\_\_ de \_\_\_\_\_.

# Agradecimentos

Agradeço à minha família por acreditarem em mim e por todo apoio e incentivo para que minha graduação fosse possível.

Agradeço aos professores com quem tive contato ao longo do curso, por todo conhecimento técnico e experiências de vida compartilhados.

Aos amigos e colegas feitos em Bauru, com quem pude crescer como pessoa e participar de histórias que jamais esquecerei.

*"A leitura do mundo precede a leitura da palavra."*  
(Paulo Freire)

# Resumo

A energia elétrica é um recurso importante das sociedades atuais e a continuidade de sua distribuição é um dos indicadores mais relevantes na gestão de energia. Este projeto teve como objetivo analisar uma base de dados da distribuidora CPFL Paulista em busca de características que possam contribuir para a melhoria do serviço prestado aos consumidores, através de técnicas de aprendizagem de máquina.

**Palavras-chave:** Ciência de dados, energia elétrica, aprendizado de máquina não supervisionado.

# Abstract

Electricity is an important resource of today's societies and the continuity of its distribution is one of the most relevant indicators in energy management. This project aimed to analyze a database of distributor CPFL Paulista in search of characteristics that can contribute to the improvement of the service provided to consumers through machine learning techniques.

**Keywords:** Data science, electricity, unsupervised machine learning.



# Lista de figuras

Figura 1 – Ciclo de vida do dado. . . . .	15
Figura 2 – Tarefas que compõem a Ciência de Dados e produtos derivados de cada uma delas. . . . .	16
Figura 3 – Exemplo de execução do K-Means com $k = 3$ . . . . .	19
Figura 4 – Formação de cotovelo para detecção do melhor número de agrupamentos para o K-Means. . . . .	20
Figura 5 – Exemplo do ambiente Colaboratory Jupyter Notebook. . . . .	21
Figura 6 – Diagrama da arquitetura do Scrapy. . . . .	22
Figura 7 – Representação da estrutura de um ndarray Numpy. . . . .	24
Figura 8 – Uso do método shape de um objeto ndarray. . . . .	24
Figura 9 – Fluxograma do Processo de Apuração e Avaliação dos Indicadores de Continuidade de Distribuição Elétrica. . . . .	26
Figura 10 – Amostra da base de dados. . . . .	29
Figura 11 – Estrutura de projeto Scrapy. . . . .	30
Figura 12 – Estrutura do HTML da base de dados. . . . .	30
Figura 13 – Implementação do ParseColumnsPipeline. . . . .	32
Figura 14 – Implementação do CleanFieldsPipeline. . . . .	32
Figura 15 – Implementação do CSVFormatPipeline. . . . .	33
Figura 16 – Implementação do CSVWriterPipeline. . . . .	33
Figura 17 – Identificação do número adequado de agrupamentos para o K-Means com dados não normalizados. . . . .	36
Figura 18 – Identificação do número adequado de agrupamentos para o K-Means com dados normalizados. . . . .	36
Figura 19 – K-Means para $k = 3$ com dados não normalizados. . . . .	37
Figura 20 – Análise do melhor número de componentes. . . . .	38
Figura 21 – Variação explicada por cada componente. . . . .	39
Figura 22 – Visualização parcial da base de dados redefinida em termos dos 7 componentes principais. . . . .	39
Figura 23 – Componentes mais significativos para explicar a variabilidade. . . . .	40
Figura 24 – Identificação de alguns conjuntos considerados <i>outliers</i> . . . . .	41
Figura 25 – K-Means para $k = 3$ com dados normalizados. . . . .	42
Figura 26 – K-Means para $k = 20$ com dados normalizados. . . . .	43

# Lista de tabelas

Tabela 1 – Número de observações por agrupamento. . . . .	42
Tabela 2 – Número de observações por agrupamento quando $k = 20$ . . . . .	44
Tabela 3 – Coeficiente de Silhueta para o estimador K-Means. . . . .	44

# Lista de abreviaturas e siglas

HTTP	<i>Hypertext Transfer Protocol</i>
ANEEL	Agência Nacional de Energia Elétrica
CSV	<i>Comma-Separated Values</i>
CPFL	Companhia Paulista de Força e Luz
DEC	Duração Equivalente de Interrupção Por Unidade Consumidora
FEC	Frequência Equivalente de Interrupção Por Unidade Consumidora
PRODIST	Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional
HTML	<i>Hypertext Markup Language</i>
CSS	<i>Cascading Style Sheets</i>
XML	<i>Extensible Markup Language</i>
PCA	<i>Princial Components Analysis</i>

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>13</b>
<b>1.1</b>	<b>Objetivos</b>	<b>13</b>
1.1.1	Objetivo Geral	13
1.1.2	Objetivos Específicos	13
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>15</b>
<b>2.1</b>	<b>Ciência de Dados</b>	<b>15</b>
<b>2.2</b>	<b>Web Scraping</b>	<b>16</b>
<b>2.3</b>	<b>Preprocessamento e Transformação dos Dados</b>	<b>16</b>
<b>2.4</b>	<b>Mineração de Dados</b>	<b>17</b>
<b>2.5</b>	<b>Aprendizado de Máquina</b>	<b>17</b>
2.5.1	Aprendizado de Máquina Supervisionado	17
2.5.2	Aprendizado de Máquina Não Supervisionado	17
2.5.3	Análise de Componentes Principais	18
2.5.4	K-Means	18
2.5.5	Avaliação de Performance de Agrupamento	19
<b>3</b>	<b>METODOLOGIA</b>	<b>21</b>
<b>3.1</b>	<b>Ferramentas</b>	<b>21</b>
3.1.1	Jupyter Notebook	21
3.1.2	Scrapy	22
3.1.3	Scikit-learn	23
3.1.4	NumPy	23
3.1.5	Pandas	24
3.1.6	SciPy	25
3.1.7	Matplotlib	25
<b>4</b>	<b>DESENVOLVIMENTO</b>	<b>26</b>
<b>4.1</b>	<b>Base de dados</b>	<b>26</b>
<b>4.2</b>	<b>Aquisição de dados</b>	<b>29</b>
4.2.1	Extração de Dados	29
4.2.2	Formatação de Dados	31
4.2.2.1	Separa e Mapeia Colunas	31
4.2.2.2	Limpeza de Dados	31
4.2.2.3	Formatador CSV	32
4.2.2.4	Escritor de Arquivo CSV	32

<b>4.3</b>	<b>Treinamento</b> . . . . .	<b>33</b>
4.3.1	Análise de Componentes Principais . . . . .	34
4.3.2	K-Means . . . . .	35
<b>5</b>	<b>RESULTADOS</b> . . . . .	<b>38</b>
<b>5.1</b>	<b>PCA</b> . . . . .	<b>38</b>
<b>5.2</b>	<b>K-Means</b> . . . . .	<b>41</b>
5.2.1	K-Means para 3 Agrupamentos . . . . .	41
5.2.2	K-Means para 20 agrupamentos . . . . .	42
<b>5.3</b>	<b>Avaliação dos Estimadores</b> . . . . .	<b>42</b>
<b>6</b>	<b>CONCLUSÃO</b> . . . . .	<b>45</b>
<b>6.1</b>	<b>Trabalhos Futuros</b> . . . . .	<b>45</b>
	<b>REFERÊNCIAS</b> . . . . .	<b>46</b>

# 1 Introdução

A disponibilização de energia elétrica está diretamente atrelada à vida das pessoas na sociedade contemporânea, seja em termos de qualidade de vida e conforto doméstico como em termos de produção e indústria ([MEHL, 2012](#)).

Dada esta importância, faz-se relevante o estudo sobre a qualidade deste insumo. A Agência Nacional de Energia Elétrica é o órgão responsável pela regulação dos padrões de qualidade à que as empresas do setor devem responder. Estes padrões podem ser de naturezas distintas como geração, transmissão, distribuição e comercialização de energia.

Pelo fato de estar ligada ao consumidor final mais evidentemente, a distribuição é um dos principais itens da gestão de energia elétrica ([MARTINHO, 2009](#)).

Conforme a Agência Nacional de Energia Elétrica recebe indicadores de continuidade de distribuição coletados pelas concessionárias, bases de dados surgem e podem ser empregadas em diversas técnicas computacionais, como as da ciência de dados.

Ciência de dados é um termo muito recorrente na área da computação. O volume de dados armazenados pelos sistemas de computador só tende a crescer, visto que a cada dia surgem novos dispositivos de armazenamento, transmissão e processamento de informações ([FERREIRA et al., 2008](#)).

Dispondo de uma base de dados viável, torna-se possível a aplicação de técnicas de ciência de dados para buscar maior entendimento e soluções dos problemas encontrados na gestão de energia elétrica.

## 1.1 Objetivos

Abaixo estão descritos os objetivos geral e específicos.

### 1.1.1 Objetivo Geral

O objetivo geral deste trabalho foi elaborar um projeto de ciência de dados, usando bases de dados de indicadores de continuidade da distribuição de energia elétrica, para identificar padrões que possam apoiar a gestão de energia.

### 1.1.2 Objetivos Específicos

- a) Estudar conceitos de mineração de dados e aprendizagem de máquina.

- b) Analisar ferramentas para o desenvolvimento do algoritmo de aprendizagem de máquina.
- c) Preparar o conjunto de dados para aplicação do algoritmo.
- d) Definir os estimadores e implementar os algoritmos.
- e) Treinar os estimadores.
- f) Avaliar resultados.
- g) Criar visualizações gráficas para os resultados obtidos.

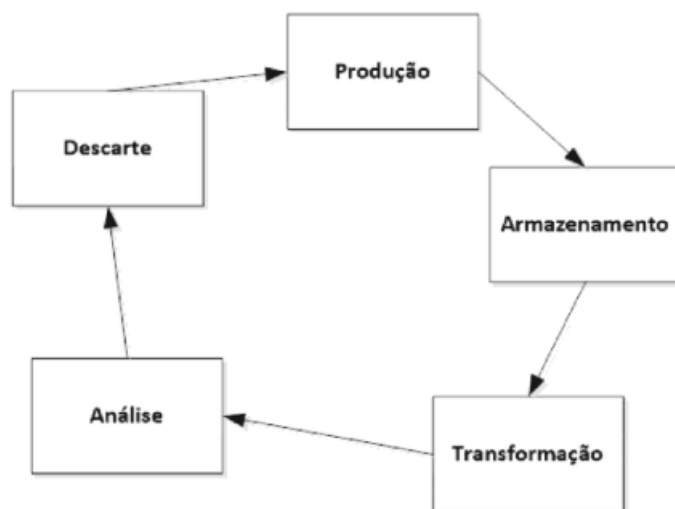
## 2 Fundamentação Teórica

### 2.1 Ciência de Dados

Ciência de dados consiste na aplicação de métodos quantitativos e qualitativos sobre grandes quantidades de dados, para resolução de problemas e predição de resultados (WALLER; FAWCETT, 2013).

Ela trata de estudar o dado em todo o seu ciclo de vida: da produção ao descarte, através de processos, modelos e tecnologias (AMARAL, 2016). O ciclo de vida do dado está ilustrado na figura 1, a seguir.

Figura 1 – Ciclo de vida do dado.



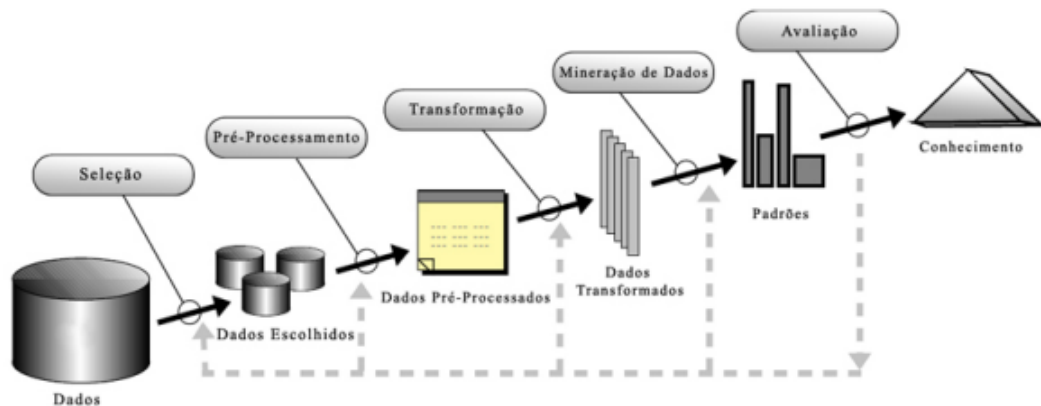
Fonte: (AMARAL, 2016)

A Ciência de Dados pode ser dividida numa série de tarefas, cujo objetivo final é a descoberta de conhecimento. As tarefas são: seleção, pré-processamento, transformação, mineração dos dados e avaliação de resultados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). A figura 2 a seguir, mostra cada etapa do processo de Ciência de Dados e o produto derivado de cada uma delas.

Nas seções a seguir, serão definidos alguns conceitos e técnicas relacionados às diferentes fases do processo de Ciência de Dados.



Figura 2 – Tarefas que compõem a Ciência de Dados e produtos derivados de cada uma delas.



Fonte: (CAMILO; SILVA, 2009)

## 2.2 Web Scraping

Uma das técnicas que pode ser usada para a etapa de seleção é a raspagem (*scraping*) de dados na web. Raspagem de dados é a atividade de coletar dados automaticamente de páginas na internet. Isso é alcançado por meio de um programa que se comunica à um servidor, recupera dados (geralmente em formato HTML e outros tipos de arquivos que compõem páginas web) e analisa este documento em busca de um conjunto de informações específico (MITCHELL, 2018).

## 2.3 Preprocessamento e Transformação dos Dados

Preprocessamento e transformação de dados consistem na preparação destes para a etapa de mineração. Como subdivisões da etapa de preprocessamento temos a limpeza e integração de dados (CAMILO; SILVA, 2009).

Na limpeza, são definidas estratégias para lidar com problemas como características faltantes nos dados, dados inconsistentes, dados duplicados, etc. Exemplos de estratégias possíveis para atacar o problema de característica faltante em um dado podem ser a completa remoção dele ou a atribuição de um valor médio.

Muitas vezes os dados são obtidos de fontes diferentes, exigindo a integração deles em um formato comum preestabelecido.

Diferentes algoritmos de mineração usam diferentes formatos de dados. Há algoritmos que trabalham apenas com valores numéricos, enquanto outros trabalham com valores categóricos. Assim, na fase de transformação de dados são empregadas técnicas para transformar dados categóricos em numéricos ou o contrário, normalizar dados, onde as características devem respeitar a mesma escala, entre outras (CAMILO; SILVA, 2009).

## 2.4 Mineração de Dados

Mineração de dados pode ser compreendida como uma tarefa multidisciplinar, onde dado um conjunto de dados, sob limitações aceitáveis de eficiência computacional, são aplicadas técnicas estatísticas, de reconhecimento de padrões, de visualização, etc. a fim de testar hipóteses, encontrar relacionamentos inesperados e de resumir os dados para melhor compreensão deles (CAMILO; SILVA, 2009).

A maioria dos métodos de mineração de dados são baseados em técnicas testadas de estatística ou de aprendizagem de máquina: classificação, regressão, agrupamento, etc (FAYYAD; PIATETSKY-SHAPIO; SMYTH, 1996).

## 2.5 Aprendizado de Máquina

Aprendizagem de máquina é uma área da Inteligência Artificial que tem como objetivo o desenvolvimento de técnicas computacionais sobre o aprendizado e a construção de sistemas que podem adquirir conhecimento automaticamente. O sistema que pode adquirir conhecimento de forma automática é aquele que toma decisões adequadas, porque conhece soluções de problemas similares (MONARD; BARANAUSKAS, 2003).

Os algoritmos de aprendizado de máquina podem ser divididos em dois tipos: supervisionados e não-supervisionados.

### 2.5.1 Aprendizado de Máquina Supervisionado

Os algoritmos supervisionados de aprendizado de máquina são aqueles onde os dados possuem rótulos que identificam suas classes. Classe é a meta que se deseja aprender e fazer previsões a respeito (MONARD; BARANAUSKAS, 2003).

O conjunto de dados é então dividido em conjunto de treinamento e de teste.

Com o conjunto de treinamento, o algoritmo aprende sobre os relacionamentos entre as características do dado e seu rótulo. Depois, este aprendizado é usado para fazer previsões sobre as classes dos dados desconhecidos até o momento, no conjunto de teste.

Se o rótulo pertencer à um conjunto de valores discretos, temos um algoritmo de classificação. Se for parte de um conjunto de valores contínuos, o algoritmo resolve problemas de regressão (MONARD; BARANAUSKAS, 2003).

### 2.5.2 Aprendizado de Máquina Não Supervisionado

Para algoritmos de aprendizado de máquina não supervisionados, que serão o foco desta monografia, são fornecidos dados que não possuem nenhum rótulo. Portanto, neste tipo de análise não há interesse em fazer previsões.

O objetivo destes algoritmos é o descobrimento de fatos interessantes sobre o conjunto de dados, como formas de visualizá-los, presença de subgrupos (*clusters*) ou a identificação das características que mais os definem.

Por não possuir dados rotulados que nos informem sobre sua verdadeira natureza, as técnicas que envolvem algoritmos não supervisionados necessitam de atenção especial na interpretação de resultados. Diferentes abordagens em relação aos dados podem levar à diferentes conclusões (JAMES et al., 2013).

Alguns algoritmos, como o K-Means por exemplo, requerem que seja definido previamente o número de agrupamentos que buscamos encontrar, mesmo que isto seja um dos fatos que deseja-se descobrir acerca do problema.

Dados descritos por distribuição normal podem apresentar resultados diferentes dos mesmos dados não normalizados.

A presença de valores atípicos é um fator que pode distorcer um *cluster*, uma vez que mesmo que ocorra agrupamentos de observações, um certo conjunto delas pode ter características diferentes entre si e de todas as outras (JAMES et al., 2013).

Para evitar conclusões incorretas, adota-se uma abordagem exploratória, onde são testadas diversas variações dos formatos dos dados e de parâmetros que influenciam nos agrupamentos. Então, conclusões podem ser formuladas com base nos resultados, atuando como ponto de partida para estudos mais específicos (JAMES et al., 2013).

### 2.5.3 Análise de Componentes Principais

Descobrir a dimensionalidade intrínseca de uma base de dados é uma tarefa fundamental na Ciência de Dados (MINKA, 2001).

Neste algoritmo de aprendizado não supervisionado, busca-se descobrir quais combinações de características dos dados são mais representativas. Isso pode ser útil para criar visualizações gráficas de problemas com grande número de dimensões.

O algoritmo busca minimizar o número de dimensões e manter o máximo da variação dos dados quanto possível. As variáveis para aplicação deste método devem estar em mesma escala, para que uma não se sobressaia sobre as outras (JAMES et al., 2013).

### 2.5.4 K-Means

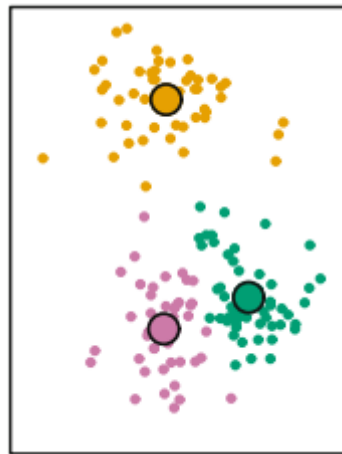
No algoritmo K-Means, buscamos dividir todas as observações em um número inteiro  $k$  de grupos, idealmente não sobrepostos (JAMES et al., 2013). O algoritmo procura separar os dados minimizando o somatório das  $k$  variâncias intra agrupamento  $W$ , conforme a equação

## 2.1.

$$\text{minimizar}_{C_1, \dots, C_k} \left\{ \sum_{k=1}^k W(C_k) \right\} \quad (2.1)$$

Na figura 3 podemos observar uma representação gráfica dos *clusters* finais definidos pelo K-Means com parâmetro  $k = 3$ , aplicado numa base de dados simulada.

Figura 3 – Exemplo de execução do K-Means com  $k = 3$



Fonte: (JAMES et al., 2013)

Existem diversas técnicas para determinar o número ideal de grupos. Uma delas é o método visual chamado Método do Cotovelo (*Elbow Method*). Nesta abordagem, o número de grupos  $k$  é iniciado em 1 e é aumentado em 1 a cada passo.

Em cada passo é calculado o custo de treinamento. Entre certos números de grupos, este custo pode cair drasticamente. Ao atingir um número de grupos onde a taxa de decrescimento do custo seja menor, temos a formação do cotovelo (KODINARIYA; MAKWANA, 2013).

Na figura 4 a seguir, observa-se a formação de cotovelo quando  $k = 2$ .

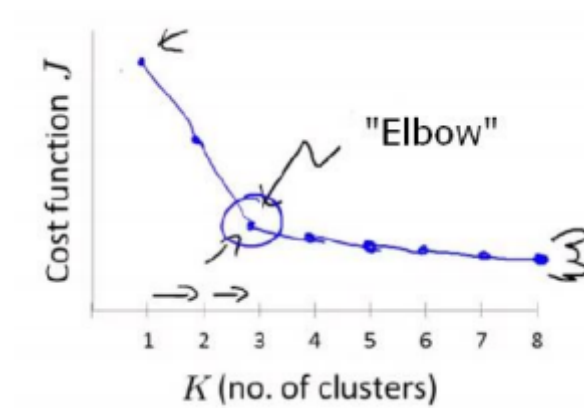
A formação do cotovelo nos informa um meio termo entre um baixo número de agrupamentos, onde pode existir dados juntos que deveriam estar separados, e um alto número de agrupamentos, que podem possuir dados separados que deveriam estar juntos.

### 2.5.5 Avaliação de Performance de Agrupamento

Uma das métricas utilizadas para avaliar a performance de um estimador é o Coeficiente de Silhueta. Nesta avaliação, é analisado o quanto os dados de um determinado *cluster* estão agrupados entre si e quão separados os *clusters* estão uns dos outros (ROUSSEEUW, 1987).

O coeficiente é calculado pela fórmula a seguir, onde  $a$  representa a distância média entre uma observação e as demais dentro do mesmo agrupamento e  $b$  representa a distância

Figura 4 – Formação de cotovelo para detecção do melhor número de agrupamentos para o K-Means.



Fonte: (KODINARIYA; MAKWANA, 2013)

média entre uma observação e todas as observações do agrupamento mais próximo:

$$s = \frac{b - a}{\max\{a, b\}} \quad (2.2)$$

## 3 Metodologia

Neste capítulo serão apresentadas as principais ferramentas utilizadas para o desenvolvimento do projeto.

### 3.1 Ferramentas

#### 3.1.1 Jupyter Notebook

O ambiente escolhido para o desenvolvimento foi o Colaboratory da empresa Google. Ele consiste em um Notebook Jupyter gratuito e que pode ser acessado diretamente de um navegador web. Todo o processamento é feito na nuvem e os códigos podem ser facilmente compartilhados e salvos.

A escolha desse ambiente se deu pelos Notebooks Jupyter serem uma tecnologia projetada para suportar o fluxo de trabalho científico. Um Notebook Jupyter é uma interface web, open source organizada em células. Estas podem conter uma ou mais linhas de código, podem ser executadas individualmente e o resultado da execução de cada uma é salvo diretamente abaixo dela. Outra vantagem é poder integrar trechos de texto plano, imagens, tabelas e gráficos para fins informativos do projeto (KLUYVER et al., 2016).

Na Figura 5, temos uma demonstração do ambiente.

Figura 5 – Exemplo do ambiente Colaboratory Jupyter Notebook.

Carregando base de dados.

```
[2] import pandas as pd
dados_raw = pd.read_csv("dec_fec_junho_2018.csv")
dados_raw.head()
```

	CONJUNTO	CODIGO	NDECONSUMIDORES	DECAPURADO	DECLIMITE	FECAPURADO	FECLIMITE
0	AGUDOS1	13861	15.100	15.73	7.0	7.12	6.0
1	ALTAIR1	13921	7.077	16.49	11.0	11.34	8.0
2	ALTINÓPOLIS1	13956	10.365	20.07	10.0	9.83	7.0
3	AMERICANA1	13823	10.046	4.88	6.0	2.85	5.0
4	AMERICANA2-CAMPOVERDE	13822	19.262	1.92	5.0	1.15	5.0

Processamento dos dados.

1 cell hidden

Treinamento.

1 cell hidden

Fonte: Elaborado pelo autor.

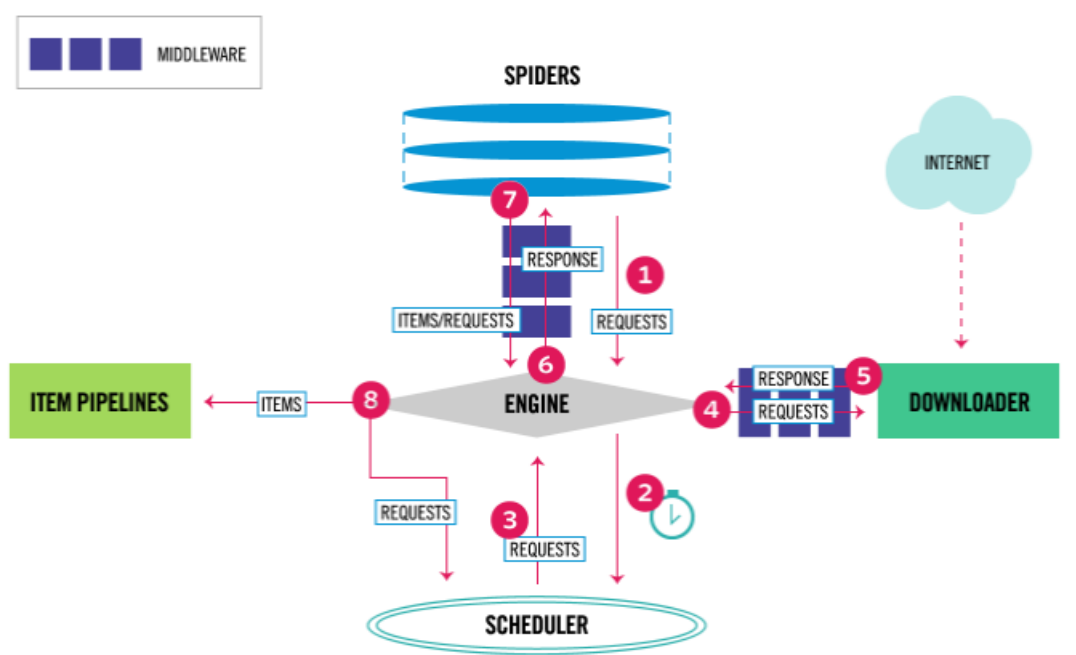
Dentre as diversas linguagens de programação disponíveis para se trabalhar dentro de Notebooks Jupyter, a escolhida para este projeto será o Python versão 3, por familiaridade do autor, por possuir diversas bibliotecas para aprendizado de máquina e uma sintaxe de fácil leitura. Dessa forma, a reprodução e entendimento dos códigos do projeto tornam-se muito práticos.

### 3.1.2 Scrapy

O Scrapy é um framework open-source, escrito em Python para web crawling e web scraping. Ele consiste num conjunto de aplicações integradas capazes de fazer o gerenciamento de requisições HTTP, baixar páginas web e expor classes chamadas *spiders*, responsáveis por conter a lógica de analisar respostas HTTP e extrair os dados desejados da página (MYERS; MCGUFFEE, 2015).

Na figura 6 temos um diagrama representando o fluxo de dados na arquitetura do Scrapy.

Figura 6 – Diagrama da arquitetura do Scrapy.



Fonte: (SCRAPY, 2019)

Para esta monografia, o Scrapy será utilizado para extrair os dados de treinamento da página da Agência Nacional de Energia Elétrica (ANEEL) e exportá-los como um arquivo no formato CSV.

### 3.1.3 Scikit-learn

Scikit-learn é um módulo Python que integra muitos algoritmos de aprendizado de máquina para problemas supervisionados e não supervisionados. O foco da biblioteca é a facilidade de uso, por sua interação em alto nível, performance e documentação bem detalhada com referências de classes e tutoriais.

Os objetos centrais deste módulo são os estimadores, que possuem métodos em comum como treinar e prever. O primeiro recebe a base de dados como parâmetro e aprende com ela. O segundo, mais aplicado à problemas de aprendizado supervisionado, usa o conjunto de testes para estimar valores.

No módulo também estão presentes funcionalidades de pré-processamento de dados como um método que os normaliza, uma vez que existem algoritmos de aprendizado de máquina sensíveis à valores cuja ordem de grandeza é diferente dos demais. Há também métodos para avaliar quão bem um determinado treinamento foi feito ([PEDREGOSA et al., 2011](#)).

Por ser construído no ecossistema científico do Python, a biblioteca Scikit-learn pode ser incorporada em diversos domínios, além da análise e exploração estatística de dados. Outros módulos deste ecossistema científico são o NumPy, Pandas e SciPy.

### 3.1.4 NumPy

Um dos componentes principais do ecossistema científico Python é o pacote NumPy. Este pacote é importante porque ele tem estruturas de dados que servem como base para muitas outras bibliotecas ([PEDREGOSA et al., 2011](#)).

Nele estão definidos:

- Um objeto equivalente à um vetor multidimensional chamado ndarray, rápido e eficiente.
- Funções de álgebra linear, transformada de Fourier e geração de números aleatórios.
- Ferramentas de leitura e escrita de bases de dados formatadas como vetores em disco.
- Ferramentas para integração de códigos Python com outras linguagens como C, C++, Fortran.

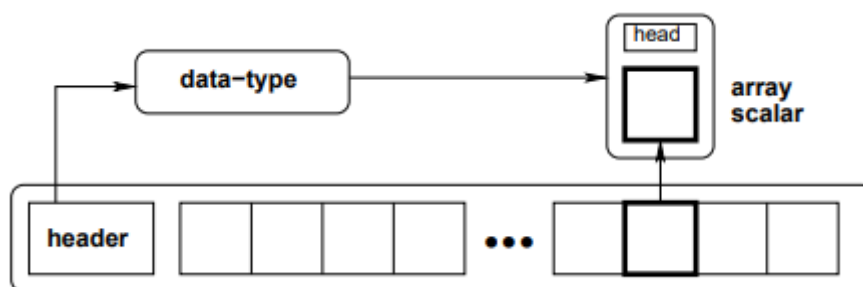
Além da alta velocidade de processamento de vetores, o NumPy é capaz de otimizar o armazenamento de dados numéricos, facilitando a passagem desses dados entre diferentes algoritmos ([MCKINNEY, 2012](#)).

A Figura 7 mostra a estrutura de um ndarray.

Um método dos objetos de tipo ndarray muito utilizado quando usamos NumPy para análise de dados é o shape, ou formato, que nos informa o número de dimensões em uma base



Figura 7 – Representação da estrutura de um ndarray Numpy.



Fonte: (OLIPHANT, 2006)

de dados e o quanto o índice pode variar em cada uma delas. Na figura 8, temos um exemplo de uso do método `shape` que nos diz que a base de dados contém 225 observações, cada uma com 5 características.

Figura 8 – Uso do método `shape` de um objeto ndarray.

```

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
dados_escalados = scaler.fit_transform(dados_raw)
dados_escalados.shape

```

(225, 5)

Fonte: Elaborado pelo autor.

### 3.1.5 Pandas

Pandas é um módulo Python que contém diversas ferramentas para a manipulação de dados estruturados. Ele combina a performance otimizada do NumPy com a flexibilidade presente na representação dos dados na forma de planilhas e bancos de dados relacionais (MCKINNEY, 2012).

Em desenvolvimento desde 2008, a biblioteca Pandas tem como objetivo unir as ferramentas de análise de dados em Python, uma linguagem de propósito geral, com as funcionalidades de linguagens de bancos de dados.

É comum que as bases de dados de problemas reais tenham um formato tabular, onde cada linha representa uma observação diferente e as colunas representam características variadas dessas observações. Poderíamos usar o `ndarray` do NumPy para manter esta base. No entanto, problemas podem surgir pelo fato de muitas funcionalidades do NumPy exigirem que todos os dados dentro do vetor sejam do mesmo tipo. Outro problema decorrente do uso direto do `ndarray` é que haveria uma perda na facilidade de uso.

Para contornar esses problemas, o Pandas expõe sua estrutura de dados principal, *DataFrame*, como uma tabela que é uma coleção de colunas independentes, cada uma representada por um *ndarray* (MCKINNEY, 2011).

Algumas das funcionalidades presentes no Pandas são a capacidade de lidar com valores faltantes na base de dados, juntar diferentes bases de dados e agrupar dados conforme um critério definido.

### 3.1.6 SciPy

O módulo SciPy é um conjunto de algoritmos para problemas específicos da computação científica (MCKINNEY, 2012). Alguns dos pacotes são:

- integrate: rotinas de integração numérica e de resolução de equações diferenciais.
- linalg: rotinas de álgebra linear e decomposição de matrizes.
- optimize: otimização de funções e algoritmos para cálculo de raízes.
- stats: testes estatísticos, distribuições normais de probabilidades.

### 3.1.7 Matplotlib

O pacote Matplotlib é responsável por criar visualizações gráficas 2D em diversas linguagens de programação. O foco está na facilidade de uso e em diversas funcionalidades disponíveis como símbolos, gráficos temporais, *antialiasing*, etc (BARRETT et al., 2005).

## 4 Desenvolvimento

Neste capítulo serão apresentados aspectos do desenvolvimento do projeto, com detalhes sobre a base de dados escolhida e como as ferramentas mencionadas no capítulo anterior foram empregadas. O desenvolvimento foi dividido em duas etapas: aquisição de dados e treinamento do algoritmo de aprendizado de máquina.

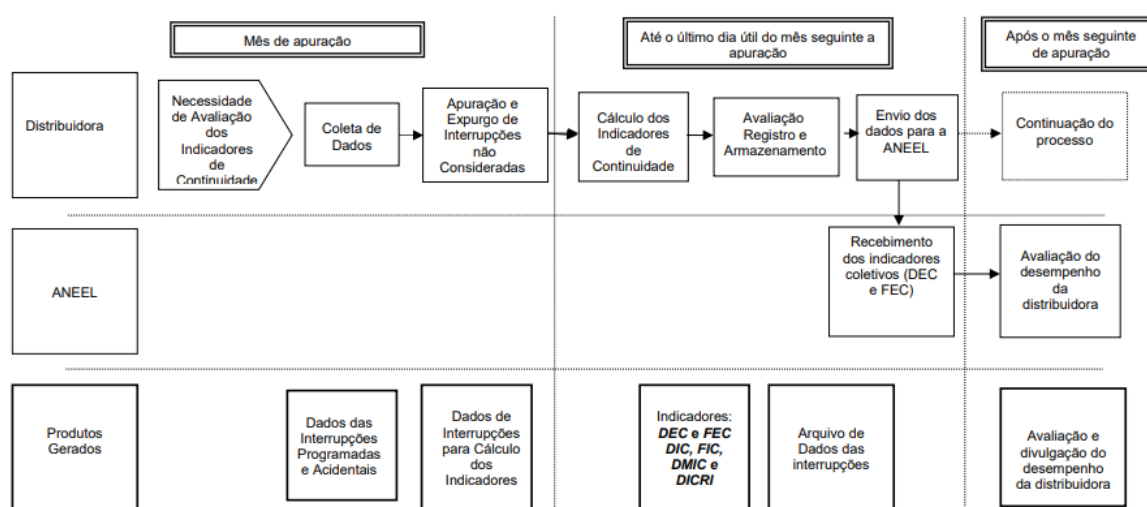
### 4.1 Base de dados

A base de dados utilizada neste projeto é disponibilizada pela Agência Nacional de Energia Elétrica. A ANEEL é o órgão responsável por regulamentar políticas quanto a exploração e utilização dos serviços de energia elétrica, bem como definir padrões de qualidade que devem ser atendidos pelas empresas do setor.

As modalidades de regulação podem ser de natureza técnica, onde são observados dados da geração, transmissão e distribuição de energia, financeira ou de pesquisa e desenvolvimento.

A base de dados escolhida diz respeito à dois indicadores da qualidade de distribuição de energia elétrica, DEC e FEC, da distribuidora CPFL Paulista no ano de 2018. Os indicadores são medidos pelas concessionárias e enviados à ANEEL, para que esta avalie a continuidade da energia elétrica oferecida à população ([AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA, 2018a](#)). Na figura 9 temos um fluxograma detalhando o processo de apuração dos indicadores de continuidade.

Figura 9 – Fluxograma do Processo de Apuração e Avaliação dos Indicadores de Continuidade de Distribuição Elétrica.



Fonte: ([AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA, 2018b](#))

Conforme determinado no manual regulatório Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional (PRODIST) (AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA, 2018b), DEC e FEC correspondem, respectivamente, à duração equivalente de interrupção por unidade consumidora e frequência equivalente de interrupção por unidade consumidora e são expressos por:

(a) Duração Equivalente de Interrupção por Unidade Consumidora (DEC)

$$DEC = \frac{\sum_{i=1}^{C_c} DIC_i}{C_c} \quad (4.1)$$

(b) Frequência Equivalente de Interrupção por Unidade Consumidora (FEC)

$$FEC = \frac{\sum_{i=1}^{C_c} FIC_i}{C_c} \quad (4.2)$$

onde:

DEC = duração equivalente de interrupção por unidade consumidora, expressa em horas e centésimos de hora;

FEC = frequência equivalente de interrupção por unidade consumidora, expressa em número de interrupções e centésimos do número de interrupções;

i = índice de unidades consumidoras atendidas;

C<sub>c</sub> = número total de unidades consumidoras faturadas do conjunto no período de apuração;

DIC(i) = Duração de Interrupção Individual por Unidade Consumidora, excluindo-se as centrais geradoras;

FIC(i) = Frequência de Interrupção Individual por Unidade Consumidora, excluindo-se as centrais geradoras.

Os dados da base correspondem à 225 conjuntos regionais do estado de São Paulo atendidos pela CPFL Paulista e para cada um deles existem 23 características diferentes que foram medidas. As características são:

- Número de consumidores: número de unidades consumidoras no conjunto.
- DECTOT: total de todos os DECs, ou seja, a duração total das interrupções.
- FECTOT: total de todos os FECs, ou seja, a quantidade total das interrupções.
- DECXP: DEC de interrupção de origem externa ao sistema de distribuição e programada.
- FECXP: FEC de interrupção de origem externa ao sistema de distribuição e programada.

- DECXN: DEC de interrupção de origem externa ao sistema de distribuição e não programada.
- FECXN: FEC de interrupção de origem externa ao sistema de distribuição e não programada.
- DECIP: DEC de interrupção de origem interna ao sistema de distribuição e programada.
- FECIP: FEC de interrupção de origem interna ao sistema de distribuição e programada.
- DECIND: DEC de interrupção de origem interna não programada e não expurgável.
- FECIND: FEC de interrupção de origem interna não programada e não expurgável.
- DECINE: DEC de interrupção de origem interna não programada e ocorrida em situação de emergência, não ocorrida em Dia Crítico.
- FECINE: FEC de interrupção de origem interna não programada e ocorrida em situação de emergência, não ocorrida em Dia Crítico
- DECINC: DEC de interrupção de origem interna ao sistema de distribuição, não programada e ocorrida Dia Crítico.
- FECINC: FEC de interrupção de origem interna ao sistema de distribuição, não programada e ocorrida Dia Crítico
- DECINO: DEC de interrupção de origem interna não programada, vinculadas a racionamento ou alívio de carga solicitado pelo ONS, não ocorrida em Dia Crítico.
- FECINO: FEC de interrupção de origem interna não programada, vinculadas a racionamento ou alívio de carga solicitado pelo ONS, não ocorrida em Dia Crítico.
- DECIPC: DEC de interrupção de origem interna ao sistema distribuição, programada e ocorrida em Dia Crítico.
- FECIPC: FEC de interrupção de origem interna ao sistema distribuição, programada e ocorrida em Dia Crítico
- DECXPC: DEC de interrupção de origem externa ao sistema distribuição, programada e ocorrida em Dia Crítico.
- FECXPC: FEC de interrupção de origem externa ao sistema distribuição, programada e ocorrida em Dia Crítico.
- DECXNC: DEC de interrupção de origem externa ao sistema distribuição, não programada e ocorrida em Dia Crítico.

- FECXNC: FEC de interrupção de origem externa ao sistema distribuição, não programada e ocorrida em Dia Crítico.

Na figura 10 podemos ver uma amostra da base de dados, como ela é apresentada no portal eletrônico da ANEEL.

Figura 10 – Amostra da base de dados.

CONJUNTO	CÓDIGO	Nº DE CONSUMIDORES	DECTOT	FECTOT	DECXP	FECXP	DECXN	FECXN	DECIP	FECIP	DECIND	FECIND
Agudos 1	13861	15.100	16,55	7,76	0,00	0,00	0,00	0,00	3,20	0,69	12,53	6,43
Altair 1	13921	7.077	23,23	14,50	0,00	0,00	0,00	0,00	1,53	0,35	14,96	10,99
Altinópolis 1	13956	10.365	22,26	10,41	0,00	0,00	0,00	0,00	4,52	1,43	15,55	8,40
Americana 1	13823	10.046	5,21	3,20	0,00	0,00	0,00	0,00	1,44	0,34	3,44	2,51
Americana 2- Campo Verde	13822	19.262	3,06	1,91	0,00	0,00	0,00	0,00	0,59	0,18	1,33	0,97
Americana 4- Jardim	13826	45.176	7,30	4,49	0,00	0,00	0,00	0,00	2,61	0,58	2,52	2,73
Americana 5-Ipe	13821	25.899	6,14	4,27	0,00	0,00	0,00	0,00	1,61	0,33	3,06	2,77
Americana 6- Cilios	13825	22.916	2,72	1,65	0,00	0,00	0,00	0,00	0,86	0,26	1,30	1,08

Fonte: (AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA, 2018a)

## 4.2 Aquisição de dados

Com a base de dados determinada, inicia-se a etapa de aquisição dos dados. Nesta fase, foi desenvolvido um programa para raspagem de dados utilizando a ferramenta Scrapy. Ele pode ser instalado pelo gerenciador de pacotes pip do Python. O produto final desta etapa é um arquivo em formato CSV, onde a primeira linha corresponde aos nomes das colunas da tabela e as linhas seguintes correspondem às observações. Este formato foi escolhido por ser adequado para carregamento no módulo Pandas do Python.

A figura 11 mostra a estrutura de diretórios do projeto Scrapy desenvolvido.

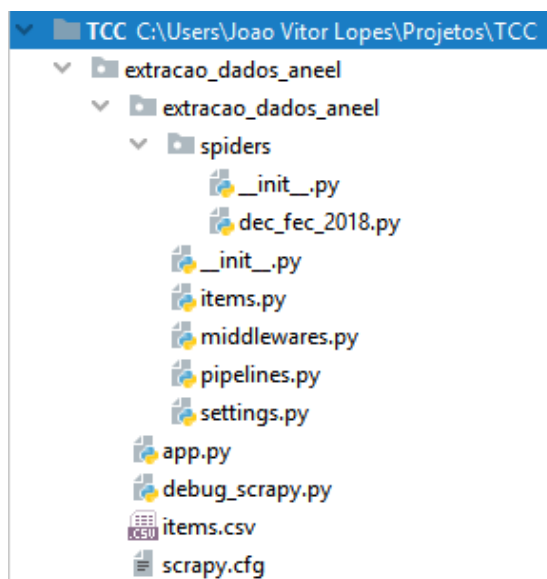
Dois conceitos do Scrapy serão utilizados nesta fase: Spider para a extração dos dados e Pipeline de item para a formatação deles.

### 4.2.1 Extração de Dados

Executada na fase de extração de dados, a Spider é o componente responsável por acessar páginas web e analisá-las em busca de informações relevantes. Para seu funcionamento, ela deve conter o método *start\_requests*, onde o endereço da página com as informações será acessado, e o método *parse* que contém a resposta do acesso e é onde ocorre a extração.

Para extrair dados de uma página HTML com Scrapy, devemos definir por qual tipo de dado estamos buscando. Na página online da ANEEL, os dados são apresentados estruturadamente numa tabela. Com o auxílio das ferramentas de desenvolvedor do navegador

Figura 11 – Estrutura de projeto Scrapy.



Fonte: Elaborado pelo autor.

Google Chrome, podemos observar como é a estrutura da tabela. A figura 12 ilustra esta estrutura.

Figura 12 – Estrutura do HTML da base de dados.



Fonte: Elaborado pelo autor.

Na imagem, os dados necessários correspondem à todas as tags HTML `<tr>`, dentro da tag HTML `<table>`, cuja classe é `"tab_indice"`. Desta forma, sabendo a hierarquia de tags que corresponde aos itens desejados, pode-se informar à Spider o que é relevante para o projeto.

Na implementação da Spider há uma chamada ao método `xpath`, que recebe como parâmetro uma string. O Xpath é uma tecnologia que modela um documento XML ou HTML

como uma árvore de nós (CLARK; DEROSE et al., 1999) que podem ou não, ter atributos (texto, identificadores CSS, etc).

Assim avançamos um nível da árvore correspondente com o uso do caractere / na string passada como parâmetro. Ao usar duas vezes este caractere, avançamos quantos níveis forem necessários até encontrar o nó ou atributo referenciado em sequência.

Um nó pode ser referenciado por seu nome e é possível usar seletores CSS para identificá-los mais precisamente.

Após este filtro, o método `extract` nos retorna todas as linhas da tabela na forma de um vetor de cadeias de caracteres. Em seguida, cada linha deste vetor é passada para o próximo componente do Scrapy.

## 4.2.2 Formatação de Dados

Na fase de formatação de dados, outro componente da arquitetura do Scrapy é executado: os Pipelines de Item. Um pipeline é uma classe que recebe um item extraído de uma página, faz determinado processamento nele e lança este item modificado para o próximo pipeline, se houver.

Neste projeto foram definidos quatro pipelines, detalhados a seguir.

### 4.2.2.1 Separa e Mapeia Colunas

Neste pipeline, a linha recebida é analisada em termos de quantidade de colunas. Essa quantidade é obtida por meio de um xpath que busca as tags HTML `<td>` (*table data*) dentro da linha. O formato de interesse de uma linha é aquele que apresenta 25 colunas, sendo que as demais são descartadas.

O método `extract` é novamente empregado, retornando um vetor onde cada posição representa uma coluna da linha. Iterando sobre este vetor, acessamos cada coluna removendo espaços em branco e tabulações e também substituindo o caractere vírgula pelo caractere ponto, para que não haja erros ao gerar o arquivo final CSV, onde a vírgula deve ser usada como separador entre dois campos.

Após, é feito um mapeamento das colunas em uma estrutura de dados do Python chamada dict. Um dict é um dicionário de chaves e valores, onde as chaves foram definidas como o index da coluna e os valores seu conteúdo.

A figura 13 a seguir, apresenta a implementação deste pipeline.

### 4.2.2.2 Limpeza de Dados

O próximo pipeline deste fluxo faz a limpeza do dado correspondente à coluna de número de consumidores. No dado original, há um ponto final separando a casa de milhar do



Figura 13 – Implementação do ParseColumnsPipeline.

```

class ParseMapColumnsPipeline(object):
    def process_item(self, item, spider):
        item_selector = Selector(text=item["linha"])
        colunas = item_selector.xpath("//td//text()").extract()

        # Ignorando título da tabela.
        if len(colunas) != 25:
            DropItem("Descartando linha que corresponde à cabeçalho ou campos que não serão usados\n{}".format(item))

        # Se a linha possui 25 colunas, ela se refere aos nomes das colunas ou dados em si.
        else:
            # Criando dict para cada observação.
            dicionario = dict()
            for index, c in enumerate(colunas):
                dicionario[str(index)] = c.strip().replace(',', '.').# Removendo espaços.
            return dicionario

```

Fonte: Elaborado pelo autor.

valor. Para que este ponto não fosse interpretado erroneamente como um separador decimal, é feita uma substituição de ponto por uma cadeia de caracteres vazia.

A figura 14 a seguir, apresenta a implementação deste pipeline.

Figura 14 – Implementação do CleanFieldsPipeline.

```

class CleanFieldsPipeline(object):
    def process_item(self, item, spider):
        # Removendo o caractere '.' do número de consumidores.
        item["2"] = item["2"].replace('.', '')
        return item

```

Fonte: Elaborado pelo autor.

#### 4.2.2.3 Formatador CSV

Neste pipeline ocorre a transformação dos valores do dicionário de volta em um vetor de cadeias de caracteres. Isso é feito porque existem recursos de um vetor Python que permitem a concatenação dos elementos do vetor em uma cadeia de caracteres, sendo possível definir um delimitador entre eles. Assim temos cada linha da tabela saindo deste pipeline totalmente formatada.

A figura 15 a seguir, apresenta a implementação deste pipeline.

#### 4.2.2.4 Escritor de Arquivo CSV

Neste último pipeline, cada linha que chega é escrita no final de um arquivo criado com a extensão CSV.

A figura 16 a seguir, apresenta a implementação deste pipeline.

Figura 15 – Implementação do CSVFormatPipeline.

```
class CSVFormatPipeline(object):
    def process_item(self, item, spider):
        colunas = list(item.values())
        linha_formato_csv = ','.join(colunas)
        return {"linha": linha_formato_csv}
```

Fonte: Elaborado pelo autor.

Figura 16 – Implementação do CSVWriterPipeline.

```
class CSVWriterPipeline(object):
    def open_spider(self, spider):
        self.file = open("items.csv", 'w', encoding="utf-8")

    def close_spider(self, spider):
        self.file.close()

    def process_item(self, item, spider):
        self.file.write("{}\n".format(item["linha"]))
        return item
```

Fonte: Elaborado pelo autor.

## 4.3 Treinamento

Após a aquisição e preprocessamento de dados, foi iniciada a fase de treinamento.

Neste trabalho, serão aplicados dois algoritmos de aprendizado não supervisionado diferentes: Análise de Componentes Principais (PCA) e K-Means. O algoritmo PCA será aplicado em estudos da dimensionalidade da base de dados e o K-Means fará o agrupamento de dados em um número de grupos predeterminado.

A seguir, temos o código responsável pelo carregamento da base de dados nos dois algoritmos, através da biblioteca *pandas*.

Código 4.1 – Carregamento da base de dados com *pandas*

```
import pandas as pd
# Lendo a base de dados para a variavel dados_raw.
dados_raw = pd.read_csv("dec_fec_2018.csv")
```

Outra tarefa comum à todos os algoritmos foi a separação das colunas relevantes para treinamento. As duas primeiras colunas da base de dados, CONJUNTO e CÓDIGO, são desconsideradas. O código que segue faz esta separação.

Código 4.2 – Separação das colunas relevantes.

```
colunas = list(dados_raw.columns)
# Colunas de indice 2 ate o indice final do vetor.
dados_para_escalar = dados_raw[colunas[2:]]
```

A tarefas seguintes incluem as chamadas dos métodos de treinamento dos algoritmos e dos métodos para visualização gráfica dos resultados.

### 4.3.1 Análise de Componentes Principais

Para este algoritmo foi necessário normalizar os dados com o método *scale* da biblioteca Sklearn, conforme o algoritmo:

Código 4.3 – Normalização de dados.

```
from sklearn.preprocessing import scale
dados_escalados = scale(dados_para_escalar)
```

Com os dados normalizados, duas estratégias foram estudadas. Uma delas busca avaliar o melhor número de componentes que definem o problema através da redução de dimensionalidade e a outra busca visualizar graficamente os dois componentes mais significativos.

Para analisar o melhor número de componentes que definem o problema, usou-se o algoritmo PCA sem informar qualquer quantidade de componentes. Neste cenário, o algoritmo decide que o número máximo de componentes é dado pelo mínimo entre o total de observações e o número de características delas.

Nesta base de dados, há 255 observações com 23 características cada, portanto, o número de componentes máximo foi 23. Com este dado, o estimador calcula a quantidade de variância acumulada para cada número de componentes variando entre zero e 23.

O código na sequência mostra o treinamento do algoritmo.

Código 4.4 – Executando PCA sem definição do número de componentes.

```
from sklearn.decomposition import PCA
# Instancia classe PCA sem definir numero de componentes.
pca_n = PCA().fit(dados_escalados)
```

O segundo treinamento do estimador foi feito com base no número ideal obtido do primeiro.

### 4.3.2 K-Means

No treinamento do algoritmo K-Means, primeiro foi necessário definir o número de *clusters* ideal. Para tal, foi utilizado o Método do Cotovelo.

O K-Means foi treinado para número de agrupamentos variando entre 1 e 100 com os dados originais e também normalizados. O número 100 foi escolhido arbitrariamente.

O custo de treinamento foi recuperado para cada treinamento e então um gráfico foi plotado, com objetivo de encontrar visualmente o ponto do cotovelo.

O algoritmo a seguir define uma função que recebe como parâmetro o número de agrupamentos desejados, treina os dados com K-Means e retorna uma lista composta pelo número de grupos e o custo associado ao treinamento (atributo *inertia\_*).

Código 4.5 – Função que calcula o erro associado ao treinamento do K-Means para  $k$  *clusters*.

```
from sklearn.cluster import KMeans
def kmeans(k, dados):
    modelo = KMeans(n_clusters=k)
    modelo.fit(dados)
    return [k, modelo.inertia_]
```

Com os resultados dos treinamentos para os dados normalizados e originais, os gráficos apresentados pelas figuras 17 e 18 foram impressos.

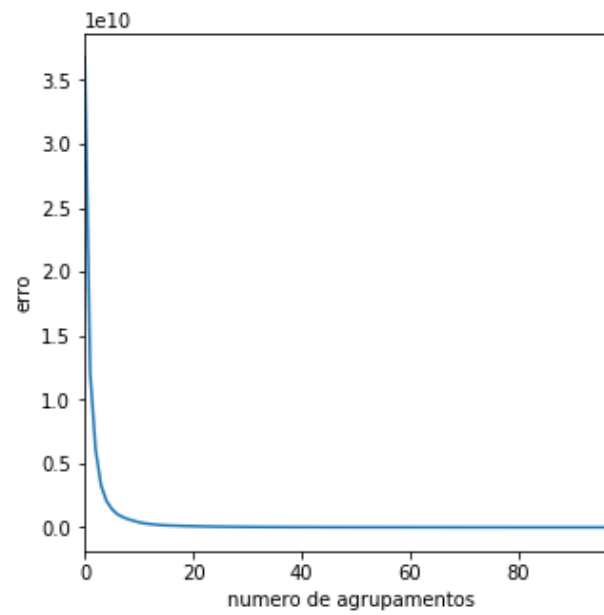
Percebe-se dois resultados muito distintos. Dados não normalizados indicaram que o número adequado de agrupamentos  $k$  é 3 enquanto os mesmos dados normalizados indicaram que o valor apropriado para  $k$  é aproximadamente 20.

Os resultados do K-Means quando não normalizamos a base de dados não são interessantes para este trabalho. O agrupamento foi feito inteiramente com base na variável número de consumidores, por esta possuir ordem de grandeza maior que a das outras.

Isso pode ser percebido no gráfico gerado pelo treinamento, conforme a figura 19.

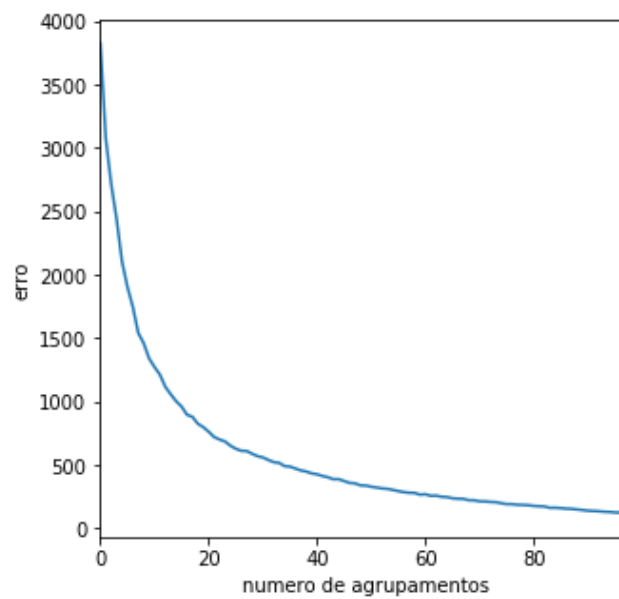
Embora o número ideal de grupos definido pelo Método do Cotovelo para dados normalizados tenha sido aproximadamente 20, o treinamento dos dados também foi realizado com um número menor de agrupamentos  $k = 3$ , para fins exploratórios.

Figura 17 – Identificação do número adequado de agrupamentos para o K-Means com dados não normalizados.



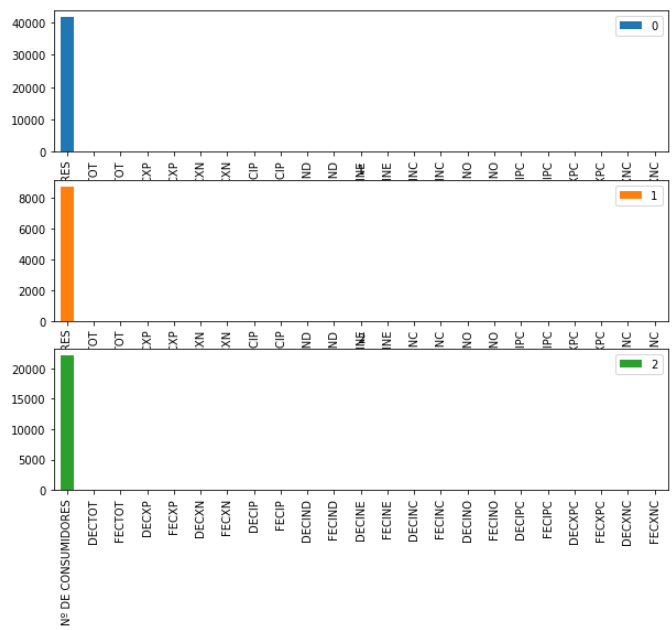
Fonte: Elaborado pelo autor.

Figura 18 – Identificação do número adequado de agrupamentos para o K-Means com dados normalizados.



Fonte: Elaborado pelo autor.

Figura 19 – K-Means para  $k = 3$  com dados não normalizados.



Fonte: Elaborado pelo autor.

## 5 Resultados

Neste capítulo serão analisados os resultados dos treinamentos de cada estimador utilizado na fase de desenvolvimento.

### 5.1 PCA

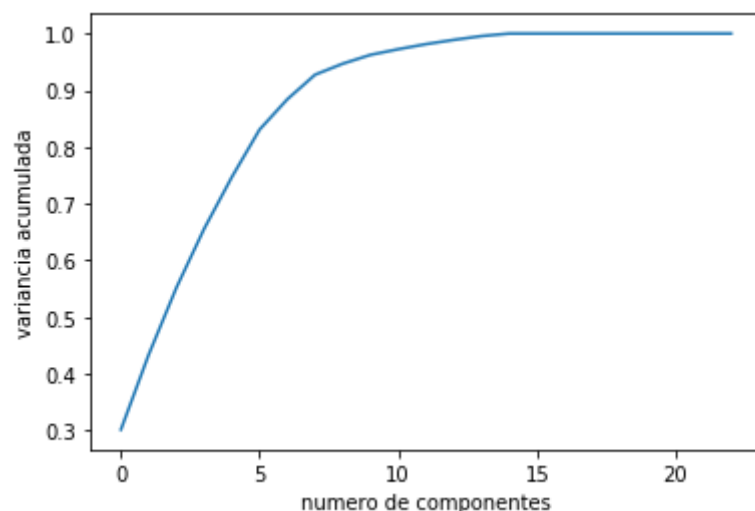
O resultado do método de treinamento *fit* do estimador PCA é uma variável com diversos atributos. O atributo *explained\_variance\_ratio\_* nos devolve um vetor de tamanho igual ao número de componentes, onde cada posição representa a quantidade de variação dos dados explicada por ele.

Na primeira execução do estimador, não foi definido um número de componentes. Por isso, o algoritmo foi executado para todos os valores inteiros entre 0 e 23.

Um gráfico da variância acumulada pelo número de componentes foi plotado com a biblioteca Matplotlib.

A figura 20 mostra este gráfico.

Figura 20 – Análise do melhor número de componentes.



Fonte: Elaborado pelo autor.

Verifica-se no gráfico que aproximadamente 90% da variação presente nos dados pode ser resumida em 7 componentes principais. Isto corresponde à uma diminuição de mais de um terço da dimensionalidade inicial do problema.

Uma vez determinado que 7 é um número de componentes que representa o problema sem perder muitas informações, o PCA foi executado novamente recebendo 7 como parâmetro.

Na figura 21 a seguir, que mostra a quantidade de variação obtida para cada um dos 7 componentes, podemos observar que há uma combinação de variáveis que por si só corresponde à 30% da variabilidade do conjunto todo.

Figura 21 – Variação explicada por cada componente.

```
array([0.30157916, 0.13204124, 0.11825075, 0.10379361, 0.09131434,
       0.08377087, 0.05340664])
```

Fonte: Elaborado pelo autor.

Assim, a base de dados foi redefinida em termos dos 7 componentes principais.

Código 5.1 – Redefinindo base de dados em termos dos 7 componentes principais.

```
# Unindo os conjuntos aos valores de principais componentes.
df_principal = pd.DataFrame(data=melhores_pca_7,
                             columns=['pc1', 'pc2', 'pc3', 'pc4',
                                      'pc5', 'pc6', 'pc7'])

df_final = pd.concat([df_principal, dados_raw[['CONJUNTO']]],
                     axis=1)
```

A figura 22 a seguir mostra o formato da base de dados redefinida.

Figura 22 – Visualização parcial da base de dados redefinida em termos dos 7 componentes principais.

	p1	p2	pc3	pc4	pc5	pc6	pc7	CONJUNTO
0	2.172784	0.589842	-0.644912	0.082274	-0.942317	-0.551558	-0.958559	Agudos 1
1	6.985909	-2.575659	2.300316	-1.508522	3.122319	-0.888337	-0.360362	Altair 1
2	4.964859	1.200955	1.645339	-1.424635	-0.640189	1.070273	-1.161340	Altinópolis 1
3	-1.565635	0.389560	-0.244720	-0.543875	0.163264	0.190893	-0.708446	Americana 1
4	-2.561165	-0.239952	-0.404781	-0.497320	0.810837	-0.299731	0.330227	Americana 2-Campo Verde

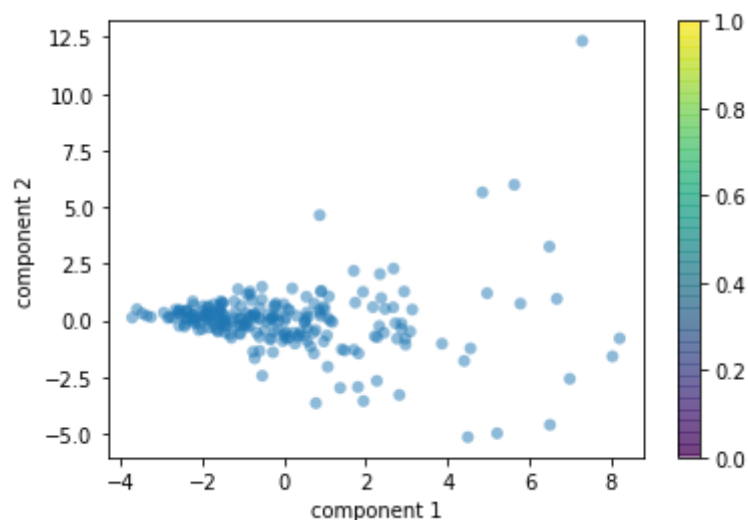
Fonte: Elaborado pelo autor.

Com base nos valores dos 7 componentes, foi elaborado o gráfico da figura 23 com as duas dimensões mais significativas da variabilidade.

Ao observar a imagem, deve-se lembrar que o grande aglomerado de observações pode não corresponder à dados realmente parecidos. Com a diminuição de dimensionalidade perde-se informações, como o quão distantes esses dados estariam se considerarmos as outras 5 dimensões não representadas, por exemplo.



Figura 23 – Componentes mais significativos para explicar a variabilidade.



Fonte: Elaborado pelo autor.

No entanto, observa-se no gráfico a presença de conjuntos considerados *outliers*, ou seja, que estão afastados do restante. O código a seguir, identifica pelo nome alguns destes conjuntos no gráfico.

Código 5.2 – Identificando *outliers* no gráfico.

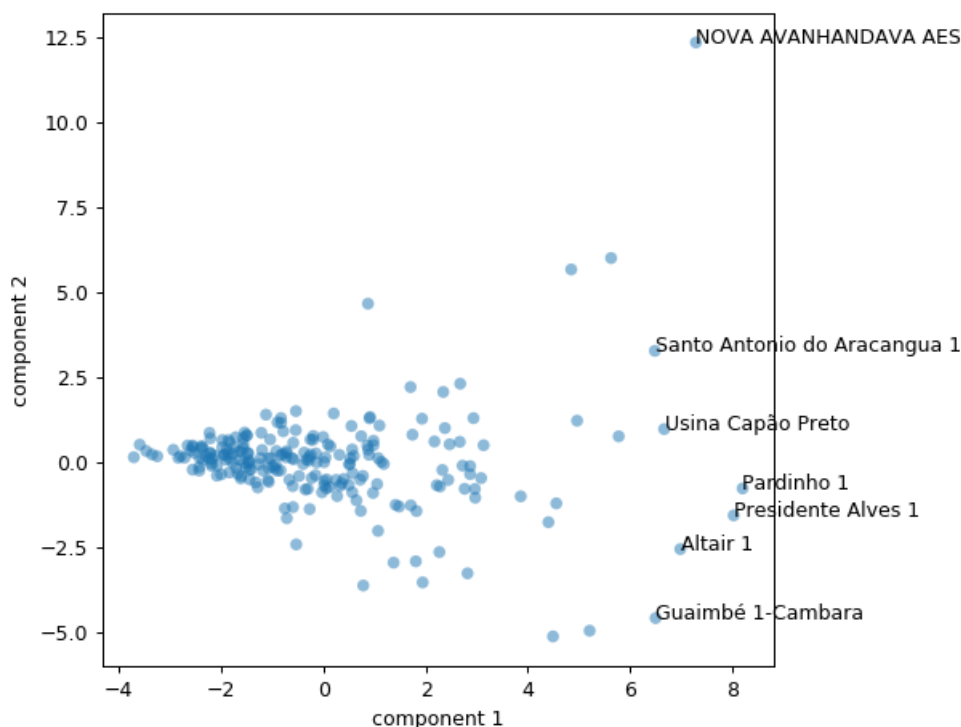
```
# Plot dos dois componentes principais.
plt.scatter(melhores_pca_7[:, 0], melhores_pca_7[:, 1],
            edgecolor='none', alpha=0.5,
            cmap=plt.cm.get_cmap('prism', 10))
plt.xlabel('component_1')
plt.ylabel('component_2')
plt.colorbar();

# Identificando outliers.
p1s = melhores_pca_7[:, 0] # Eixo x.
p2s = melhores_pca_7[:, 1] # Eixo y.
conjuntos = df_final["CONJUNTO"] # Nomes dos conjuntos.

for i, conjunto in enumerate(conjuntos):
    if (p1s[i] > 6) and (p2s[i] > 2.5 or p2s[i] < 2):
        plt.annotate(conjunto, (p1s[i], p2s[i]))
```

A figura 24 mostra os *outliers*.

Figura 24 – Identificação de alguns conjuntos considerados *outliers*.



Fonte: Elaborado pelo autor.

Nesta abordagem, o conjunto NOVA AVANHANDAVA AES é o que mais se destaca.

## 5.2 K-Means

### 5.2.1 K-Means para 3 Agrupamentos

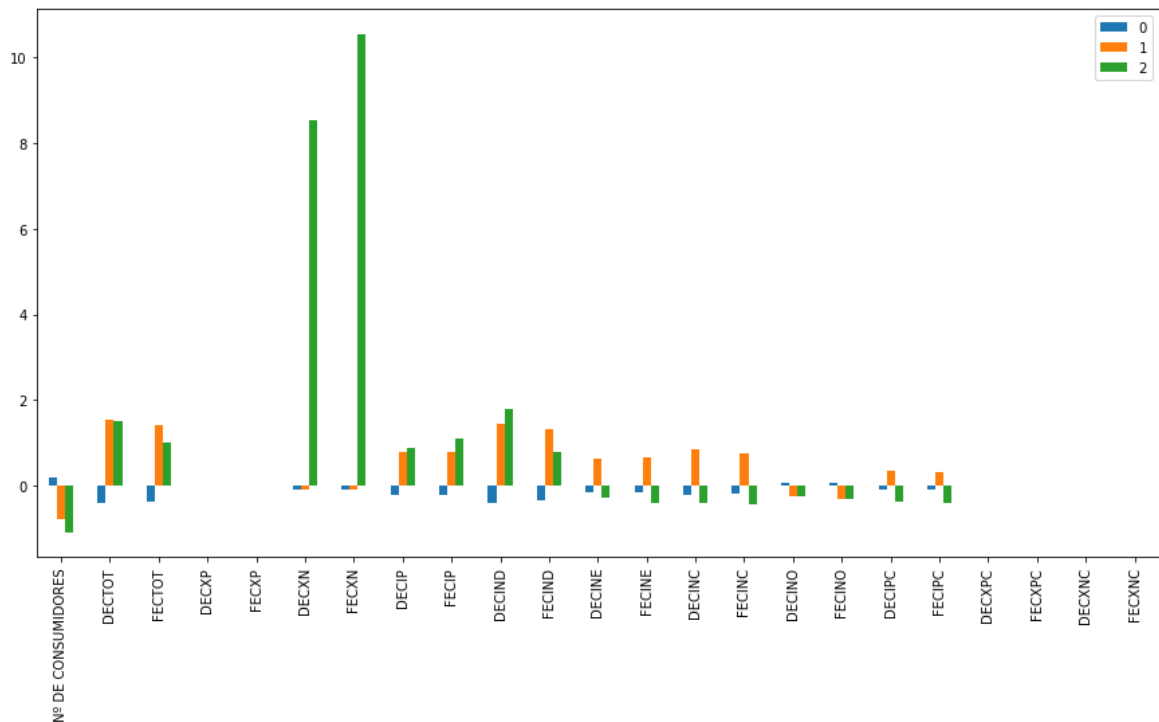
A figura 25 ilustra os agrupamentos formados no treinamento do K-Means para  $k = 3$ .

Quando buscamos identificar 3 grupos utilizando K-Means, podemos perceber visualmente algumas diferenças.

O grupo 2 apresenta valores muito grandes para as variáveis DECXN e FECXN quando comparado aos demais. O grupo 0, por outro lado, apresenta os melhores indicadores na maioria das variáveis e o maior número de consumidores.

Os números de observações em cada agrupamento se encontram na tabela 1.

A tabela indica que os conjuntos do grupo 2, identificados na base de dados por NOVA AVANHANDAVA AES e VALPARAISO CTEEP, foram capazes de, sozinhos, caracterizar um único agrupamento. Este resultado reforça as observações do estimador PCA, que indicou o conjunto NOVA AVANHANDAVA AES como um *outlier*.

Figura 25 – K-Means para  $k = 3$  com dados normalizados.

Fonte: Elaborado pelo autor.

Tabela 1 – Número de observações por agrupamento.

Grupo	Número de observações
0	177
1	46
2	2

Fonte: Elaborada pelo autor.

### 5.2.2 K-Means para 20 agrupamentos

Para o treinamento do K-Means com  $k = 20$ , temos grupos mais específicos. Na figura 26, podemos observar que o grupo 3, composto por 11 conjuntos, é o que apresenta maiores valores para o indicador DECHN.

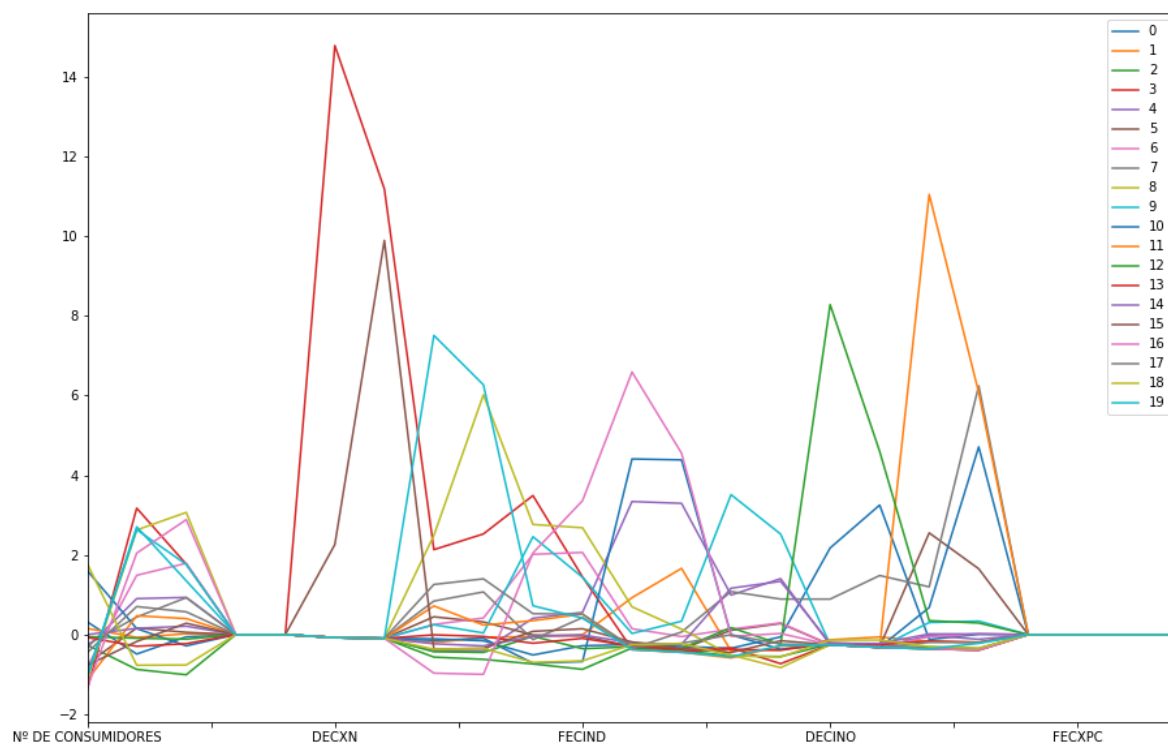
A tabela 2 a seguir mostra a quantidade de conjuntos agrupados em cada *cluster*.

## 5.3 Avaliação dos Estimadores

A métrica Coeficiente de Silhueta para avaliação de estimadores foi aplicada sobre o K-Means buscando comparar os treinamentos com diversos valores de agrupamentos. A tabela 3 a seguir, mostra estes resultados.

Nesta métrica, os valores podem variar entre -1 e 1 e quanto maior o coeficiente

Figura 26 – K-Means para  $k = 20$  com dados normalizados.



Fonte: Elaborado pelo autor.

calculado, mais definidos são os agrupamentos. Valores de coeficiente próximos de 0 são indicativos de sobreposição de *clusters* (ROUSSEEUW, 1987).

Tabela 2 – Número de observações por agrupamento quando  $k = 20$ 

<b>Grupo</b>	<b>Número de observações</b>
0	30
1	7
2	6
3	11
4	8
5	1
6	39
7	48
8	2
9	2
10	2
11	1
12	9
13	6
14	1
15	9
16	2
17	12
18	2
19	27

Fonte: Elaborada pelo autor.

Tabela 3 – Coeficiente de Silhueta para o estimador K-Means.

<b>Número de grupos</b>	<b>Coeficiente de Silhueta</b>
2	0.3890
3	0.3835
5	0.2843
10	0.1865
20	0.1896

Fonte: Elaborada pelo autor.

## 6 Conclusão

Nesta monografia, foram explorados diversos conceitos da Ciência de Dados na gestão de energia elétrica.

Foi possível a utilização de ferramentas do ambiente de desenvolvimento da linguagem Python, desde as fases de extração de dados, até o treinamento de algoritmos de aprendizado de máquina.

Pela natureza exploratória do aprendizado de máquina não supervisionado, o intuito dessa análise foi a busca de padrões que se repetem entre estimadores e a criação de visualizações gráficas que ajudem na interpretação dos dados.

Ambos os estimadores treinados no trabalho mostraram bons resultados no aspecto de identificação de *outliers*. Identificar valores atípicos é uma tarefa importante, porque pode criar prioridades, auxiliando organizações na administração de seus recursos.

Com 3 agrupamentos, observou-se que existe um conjunto de áreas administradas pela CPFL que apresentam alto número de consumidores e bons indicadores de continuidade, embora este grupo seja o que apresenta maiores índices de racionamento de energia elétrica.

Para 20 agrupamentos, observou-se uma separação muito voltada às diferentes características do dados.

O *outlier* mais relevante identificado em ambos os estimadores foi o conjunto NOVA AVANHANDAVA AES.

Diante destas análises, pode-se concluir que o presente estudo foi capaz de cumprir seu objetivo, trazendo considerações iniciais acerca da qualidade da distribuição de energia elétrica feita pela CPFL Paulista.

### 6.1 Trabalhos Futuros

Como sugestão de trabalhos futuros, pode-se considerar o treinamento de outros algoritmos de aprendizado não supervisionado ou supervisionado.

Pode-se também, unir conhecimentos específicos do domínio do problema aos treinamentos dos estimadores, para que sejam mais assertivos.

# Referências

- AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA. *Indicadores Coletivos de Continuidade (DEC e FEC)*. 2018. Disponível em : <<https://www.aneel.gov.br/indicadores-coletivos-de-continuidade>>. Acesso em: 25 ago. de 2019.
- AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA. *Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional – PRODIST*. [S.l.], 2018.
- AMARAL, F. *Introdução à Ciência de Dados: mineração de dados e big data*. [S.l.]: Alta Books Editora, 2016.
- BARRETT, P.; HUNTER, J.; MILLER, J. T.; HSU, J.-C.; GREENFIELD, P. matplotlib—a portable python plotting package. In: *Astronomical data analysis software and systems XIV*. [S.l.: s.n.], 2005. v. 347, p. 91.
- CAMILO, C. O.; SILVA, J. C. d. Mineração de dados: Conceitos, tarefas, métodos e ferramentas. *Universidade Federal de Goiás (UFG)*, p. 1–29, 2009.
- CLARK, J.; DEROSE, S. et al. *XML path language (XPath) version 1.0*. 1999.
- FAYYAD, U.; PIATETSKY-SHAPIO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37–37, 1996.
- FERREIRA, H. M. et al. Uso de ferramentas de aprendizado de máquina para prospecção de perdas comerciais em distribuição de energia elétrica. [sn], 2008.
- JAMES, G.; WITTEN, D.; HASTIE, T.; TIBSHIRANI, R. *An introduction to statistical learning*. [S.l.]: Springer, 2013. v. 112.
- KLUYVER, T.; RAGAN-KELLEY, B.; PÉREZ, F.; GRANGER, B.; BUSSONNIER, M.; FREDERIC, J.; KELLEY, K.; HAMRICK, J.; GROUT, J.; CORLAY, S.; IVANOV, P.; AVILA, D.; ABDALLA, S.; WILLING, C. Jupyter notebooks – a publishing format for reproducible computational workflows. In: LOIZIDES, F.; SCHMIDT, B. (Ed.). *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. [S.l.], 2016. p. 87 – 90.
- KODINARIYA, T. M.; MAKWANA, P. R. Review on determining number of cluster in k-means clustering. *International Journal*, v. 1, n. 6, p. 90–95, 2013.
- MARTINHO, E. *Distúrbios da energia elétrica*. [S.l.]: Saraiva Educação SA, 2009.
- MCKINNEY, W. pandas: a foundational python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, v. 14, 2011.
- MCKINNEY, W. *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. [S.l.]: "O'Reilly Media, Inc.", 2012.
- MEHL, E. L. Qualidade da energia elétrica. *UNIVERSIDADE FEDER-AL DO PARANÁ–UFPR*, 2012.
- MINKA, T. P. Automatic choice of dimensionality for pca. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2001. p. 598–604.

- MITCHELL, R. *Web Scraping with Python: Collecting More Data from the Modern Web*. [S.l.]: "O'Reilly Media, Inc.", 2018.
- MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas inteligentes-Fundamentos e aplicações*, v. 1, n. 1, p. 32, 2003.
- MYERS, D.; MCGUFFEE, J. W. Choosing scrapy. *Journal of Computing Sciences in Colleges*, Consortium for Computing Sciences in Colleges, v. 31, n. 1, p. 83–89, 2015.
- OLIPHANT, T. E. *A guide to NumPy*. [S.l.]: Trelgol Publishing USA, 2006. v. 1.
- PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.
- ROUSSEEUW, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, Elsevier, v. 20, p. 53–65, 1987.
- SCRAPY. *Architecture overview*. 2019. Disponível em : <<https://docs.scrapy.org/en/latest/topics/architecture.html>>. Acesso em: 18 set. de 2019.
- WALLER, M. A.; FAWCETT, S. E. Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management. *Journal of Business Logistics*, Wiley Online Library, v. 34, n. 2, p. 77–84, 2013.